

Locally Adaptive Optimization: Adaptive Seeding for Monotone Submodular Functions

Ashwinkumar Badanidiyuru
Google
ashwinkumarbv@gmail.com

Christos Papadimitriou
UC Berkeley
christos@cs.berkeley.edu

Aviad Rubinfeld
UC Berkeley
aviad@cs.berkeley.edu

Lior Seeman
Cornell University
lseeman@cs.cornell.edu

Yaron Singer
Harvard University
yaron@seas.harvard.edu

Abstract

The Adaptive Seeding problem is an algorithmic challenge motivated by influence maximization in social networks: One seeks to select among certain accessible nodes in a network, and then select, adaptively, among neighbors of those nodes as they become accessible in order to maximize a global objective function. More generally, adaptive seeding is a stochastic optimization framework where the choices in the first stage affect the realizations in the second stage, over which we aim to optimize.

Our main result is a $(1-1/e)^2$ -approximation for the adaptive seeding problem for any monotone submodular function. While adaptive policies are often approximated via non-adaptive policies, our algorithm is based on a novel method we call *locally-adaptive* policies. These policies combine a non-adaptive global structure, with local adaptive optimizations. This method enables the $(1-1/e)^2$ -approximation for general monotone submodular functions and circumvents some of the impossibilities associated with non-adaptive policies.

We also introduce a fundamental problem in submodular optimization that may be of independent interest: given a ground set of elements where every element appears with some small probability, find a set of expected size at most k that has the highest expected value over the realization of the elements. We show a surprising result: there are classes of monotone submodular functions (including coverage) that can be approximated almost optimally as the probability vanishes. For general monotone submodular functions we show via a reduction from PLANTED-CLIQUE that approximations for this problem are not likely to be obtainable. This optimization problem is an important tool for adaptive seeding via non-adaptive policies, and its hardness motivates the introduction of *locally-adaptive* policies we use in the main result.

1 Introduction

The surge of massive digital records of human interactions in recent years provides a new system-wide perspective on social networks. In addition to observing and predicting patterns of collective human behavior, in many cases the dynamics of the network can be engineered. One such example is when attempting to initiate a large cascade by seeding it at certain important nodes in the network to promote a product or social movement through word-of-mouth. The algorithmic challenge of selecting individuals who can serve as early adopters of a new idea, product, or technology in a manner that will trigger a large cascade in the social network is known as *influence maximization*. Since it was first posed by Domingos and Richardson [11, 34] and elegantly formulated and further developed by Kempe, Kleinberg, and Tárdo [22], a broad range of algorithmic methods have been developed for this canonical problem [4, 5, 8, 17, 26, 27, 28, 31].

In many applications of influence maximization, despite having full knowledge of the network, one may only have access to a small slice of the network. In marketing applications for example, companies often reward influential users who visit their online store, or who have engaged with them in other ways (subscribe to a mailing list, follow the brand, install an application etc.). If we think of users who arrive at a store or follow a brand as being randomly sampled from the network, observing high-degree users is a rare event. This is simply due to the heavy-tailed degree distributions of social networks. Since influence maximization techniques are based on selecting high degree users (not necessarily the *highest* degree), their application on such samples can become ineffective. In general, access to high degree individuals in social networks is often rare, which raises the following question.

Is it possible to design effective influence maximization strategies despite the rarity of influencers?

To tackle the problem of rare influencers, the adaptive seeding framework was recently developed in [36]. The framework is a stochastic optimization model which formalizes an intuitive approach: rather than spend the entire budget on the non-influential users, we can spend a fraction of the budget on the accessible users, wait for their friends to appear as a result, and optimize influence by using the remaining budget to seed influential friends. The idea is to leverage what's known as the *friendship paradox* [15], which suggests that although people are not likely to be influential, they are likely to know someone who is. In [24] it was shown that in well-established mathematical models of social networks there are asymptotic gaps between the degree of a random node and its neighbor. This structural property implies that dramatic improvements to influence maximization are indeed achievable by optimizing influence over friends. In recent work [20], along with scalable algorithms for this problem, it was shown through various experiments on the Facebook graph that dramatic improvements to naive application of influence maximization are indeed obtainable through adaptive seeding.

The adaptive seeding model. The adaptive seeding model is a *two-stage* stochastic optimization framework. We are given a set of nodes X , their set of neighbors $\mathcal{N}(X)$, each associated with a probability p_i , as well as a budget $k \in \mathbb{N}$ and a function $f : 2^{\mathcal{N}(X)} \rightarrow \mathbb{R}$. In the first stage, a set $S \subseteq X$ can be selected, which causes each one of its neighbors to materialize independently with probability p_i . In the second stage, the remainder of the budget can be used to optimize the function $f(\cdot)$ over the realized neighbors. This function quantifies the expected number of individuals in the network that will be influenced as a result of selecting a subset of early adopters. The goal is to select a subset $S \subseteq X$ of size at most k s.t. the function can be optimized in expectation over all possible realizations of its neighbors with the remaining budget $k - |S|$. Equivalently, our goal is to select $S \subseteq X$ s.t. in expectation over all possible realizations R_1, \dots, R_m of $\mathcal{N}(X)$ the value of a set of its neighbors T_i of size $k - |S|$ that appears in the realization R_i is optimal.

Stated in these terms the objective is:

$$\begin{aligned} \max_{S \subseteq X} \sum_{i=1}^m f(T_i) \cdot p(R_i) \\ T_i \subseteq R_i \cap \mathcal{N}(S) \quad \forall i \in [m] \\ |S| + |T_i| \leq k \quad \forall i \in [m] \end{aligned}$$

The adaptive seeding formulation models the challenge of recruiting neighbors who can become effective influencers using the budget, rather than trying to influence them to forward the information without incentives as in the standard influence maximization framework.¹ Although it seems quite plausible that the probabilities of attracting neighbors could depend on the rewards they receive, the simplification assuming unit costs is deliberate. This simplification is consistent with the celebrated Kempe-Kleinberg-Tárdos model [22], and can be extended to the case where nodes take on different costs [35].

While the motivation is influence maximization, adaptive seeding is a versatile framework of stochastic optimization (see related work section for further discussion on stochastic optimization). At its essence, it formalizes the following question.

Given a distribution on the consequences of the actions we take in the present, can we design algorithms that optimize events in the future?

The main question when considering stochastic optimization models is whether the same guarantees as standard optimization can be obtained [38]. It is not hard to show that for very simple objective functions such as $f(S) = |S|$, the two-stage optimization is NP-hard even in the non-stochastic case, i.e. when all probabilities are one. Thus, approximation is needed. Therefore, in the context of adaptive seeding the question is whether an objective that can be well approximated in standard optimization can also be well approximated in adaptive seeding.

Submodularity. A function $f : 2^N \rightarrow \mathbb{R}_+$ is submodular if $f(S \cup T) \leq f(S) + f(T) - f(S \cap T)$. Equivalently, a function is submodular if it has a natural diminishing returns property: for any $S \subseteq T \subseteq N$ and $a \in N \setminus T$ a function is submodular if $f_S(a) \geq f_T(a)$, where $f_A(B) = f(A \cup B) - f(A)$ for any $A, B \subseteq N$. Unless otherwise stated, we will assume the function is normalized ($f(\emptyset) = 0$) and *monotone* ($S \subseteq T$ implies $f(S) \leq f(T)$). In standard optimization submodularity is a certificate for desirable approximation guarantees (1-1/e approximation for maximizing such functions under a matroid constraint [41]), and a slightly broader class known as *fractionally subadditive* functions already exhibits strong information-theoretic lower bounds [29].

Adaptive Seeding of submodular functions. The effectiveness of adaptive seeding depends crucially on our ability to optimize general classes of submodular functions. Say that a function can be *adaptively seeded* if the adaptive seeding problem can be approximated within a constant factor for this objective. The main result in [36] shows that any function in a class known as the *triggering model* — a special class of monotone *submodular* functions defined in the seminal work of Kempe, Kleinberg, and Tárdos [22] — can be adaptively seeded. But far more general models of submodular functions are used to describe influence in social networks [22, 31], and the techniques

¹Having probabilities on the nodes in $\mathcal{N}(X)$ and not the edges implies that the neighbors are not more likely to appear if they have more parents in X selected by the algorithm. This corresponds to what's known in microeconomics as a standard Bayesian utility model with no externalities. For submodular functions, this problem can be shown to be equivalent to the one where probabilities appear on the edges and not the nodes. This corresponds to a model where nodes in the second stage are influenced by nodes in the first stage through the independent cascade model [22].

of [36] cannot be applied to these (see related work section for further discussion on submodular function, their connection with the influence maximization problem and the limitation of previous techniques for adaptive seeding). Thus the main question in this work is:

Can any submodular function be adaptively seeded?

Naturally, we will seek algorithms that obtain the best approximation ratio possible. The main challenge would be to obtain an algorithm that achieves a $(1 - 1/e)^2$ approximation ratio for general monotone submodular functions. This bound is a natural goal for this problem, as we discuss below.

1.1 Warm up: the non-stochastic case

We begin by considering the non-stochastic version of the problem. That is, the version in which every node in the set of neighbors appears with probability one. Here we are given a set X and its neighbors $\mathcal{N}(X)$, there is a monotone submodular function defined on $\mathcal{N}(X)$, and the goal is to select $t \leq k$ elements in X connected to a set of size at most $k - t$ in $\mathcal{N}(X)$ for which the submodular function has the largest value. A trivial solution would be to run a greedy algorithm with budget of $k/2$ on parent-child pairs. That is, at every stage select the node in $\mathcal{N}(X)$ whose marginal contribution given the nodes already selected is the largest, and add one of its parents (if none have been added in previous rounds) to ensure the solution is feasible. Since we need at most $k/2$ parents to select $k/2$ children the solution is feasible, and submodularity guarantees the solution's value is at least half of the value as if we were to run the algorithm with a budget of k , which is an upper bound on the optimal solution. It is well known that the greedy algorithm is a $1 - 1/e$ -approximation to the optimal solution, and hence this trivial algorithm would be a $(1 - 1/e)/2$ -approximation.

Optimal approximation via ϵ -blocks. A natural extension to the above approach would be to pair parent nodes (i.e. nodes in X) with *subsets* of their children. We call such pairs *blocks*, and the *density* of a block is simply the ratio between its marginal contribution and its size: for a set of children $T \subseteq \mathcal{N}(X)$, the marginal density of a block (x, B) with respect to T is

$$f_T(B)/(1 + |B|).$$

Ideally, for each parent we would add the subset of children which makes for the densest block, as one can then show that an algorithm which iteratively adds the densest block results in the optimal $1 - 1/e$ approximation. However, even for coverage functions finding the densest block implies solving an NP-hard problem. Instead of densest blocks we can consider using ϵ -blocks: a node in $x \in X$ and a subset of its neighbors of size at most $1/\epsilon$. Note that for any constant $\epsilon > 0$, finding the densest ϵ -block can be done in polynomial time by brute forcing all subsets of $1/\epsilon$ neighbors of every node $x \in X$ and computing their value. Importantly, one can show that for any $\epsilon > 0$ the densest ϵ -block B is a $(1 - \epsilon)$ approximation to the densest block O if $|O|$ is larger than $1/\epsilon$:

$$\frac{f_T(B)}{1 + |B|} \geq \left(\frac{1}{1 + 1/\epsilon} \right) \frac{1/\epsilon}{|O|} f_T(O) = \left(\frac{1}{1 + \epsilon} \right) \frac{f_T(O)}{|O|} \geq (1 - \epsilon) \frac{f_T(O)}{1 + |O|}$$

where the first inequality is by submodularity and taking the $1/\epsilon$ elements of O with highest value.

The algorithm is now simple: until exhausting the budget, add the densest ϵ -block to the solution. For any $\epsilon > 0$, this algorithm is a $(1 - 1/e - \epsilon)$ approximation, and the idea of the analysis is natural; at every stage this algorithm selects an ϵ -block that is a $(1 - \epsilon)$ approximation to the densest block, and by applying a standard inductive argument one can show that this results in a $(1 - 1/e - \epsilon)$ approximation. Although it only holds in the dummy non-stochastic version of the problem, this approach encapsulates the core idea in this paper.

1.2 Synopsis

In the stochastic case the optimal solution is an *adaptive* policy: one which selects a subset from X , and *after* the realization of its neighbors, selects an optimal solution with its remaining budget. Since adaptive policies are notorious in stochastic optimization for their difficulty, the standard approach is to design non-adaptive policies which approximate adaptive policies well. Informally, a *non-adaptive* policy is a policy which selects the subset S and a set of its neighbors, *a priori* to their realization. We cannot hope to obtain an approximation better than $1 - 1/e$ for the optimal non-adaptive policy unless $P=NP$ [14], as the non-stochastic case is a special case. As we show later, the ratio between non-adaptive policies and adaptive ones can be as bad $1 - 1/e$, and we therefore naturally seek algorithms whose approximation ratio is $(1 - 1/e)^2$. We do not know whether $(1 - 1/e)^2$ is the optimal approximation ratio, and it is one of the main open questions in this paper.

Similar to the exposition in the warmup above, finding a $(1 - 1/e)$ approximation to the optimal non-adaptive policy (and hence a $(1 - 1/e)^2$ approximation to the optimal adaptive policy) reduces to finding ϵ -blocks with arbitrarily good precision in *the stochastic case*. The key challenge in computing ϵ -blocks here reduces to the following fundamental problem (which may be of independent interest) we call *submodular-optimization-with-small-probabilities (SOSP)*: given a ground set of elements where every element appears with some small probability, find a set of *expected* size at most k that has the highest expected value over the realization of the elements. While for some classes of submodular functions near-optimal solutions for this problem can be obtained, arbitrarily good approximations for this problem are not likely to be obtainable in general. In other words, non-adaptive policies do not suffice for getting a $(1 - 1/e)^2$ -approximation algorithm.

Our main result builds on an alternative strategy for defining ϵ -blocks. Instead of non-adaptive policies we employ what we call *locally-adaptive policies*. Intuitively, a locally adaptive policy consists of a set of ϵ -blocks, where within each block the policy can make *adaptive decisions*. The adaptivity within a block lets us find the optimal ϵ -block and enables the $(1 - 1/e)^2$ guarantee.

1.3 Results

- **Tight Adaptivity gap.** In Section 2 we show that non-adaptive policies approximate adaptive policies within almost a factor of $1 - 1/e$. We then show that this gap is tight.
- **Algorithm for SOSP.** In section 2.1 we introduce this problem and show how it can be solved almost optimally for matroid-rank-sum functions (which include coverage functions) by convex programming as the probabilities vanish. We then show how to use this problem to design a $(1 - 1/e)^2$ -approximation algorithm for this class of submodular functions.
- **Hardness of SOSP.** We show that for general monotone submodular functions, the problem is hard to approximate to arbitrary precision by a reduction from the PLANTED-CLIQUE problem. Thus, for general submodular functions computing the optimal ϵ -block is also hard.
- **Our main result: A $(1 - 1/e)^2$ -approximation algorithm through locally-adaptive policies.** In Section 3 we describe our main algorithm designed for any monotone submodular function using value oracles. The algorithm finds a locally-adaptive policy whose value is guaranteed to be at least about $1 - 1/e$ of the value of the optimum locally-adaptive policy. Naturally, it remains to prove that the best locally-adaptive solution is $1 - 1/e$ away from the true optimum. We establish this by showing that *any non-adaptive policy can be approximated arbitrarily well by a locally-adaptive policy*, and utilizing the bounds we have for such policies. The idea of locally-adaptive policies is new and may be of independent interest.
- **Adaptivity gap for locally-adaptive policies.** In section 3.3 we exhibit a gap (≈ 0.853) between the optimal locally-adaptive policy and the optimal adaptive policy.

1.4 Related work

Adaptive Seeding. The Adaptive Seeding model was introduced by Seeman and Singer in [36]. They use a concave relaxation to achieve a constant bound approximation for the adaptive seeding problem with influence functions in the Triggering model. Unfortunately, their techniques do not extend to general submodular functions. In this work we introduce new non-adaptive and adaptive techniques for this problem, and achieve a better approximation bound for the Triggering model. Most importantly, our results hold for the entire class of submodular functions. In a follow-up paper [35] the adaptive seeding problem is studied under *knapsack constraints*. While the techniques used in that paper are applicable here, they give a different approximation bound than what we achieve here, which as shown in that paper is actually the appropriate bound for that case. However, as we show, in the cardinality case we can get better approximation bounds.

Submodular Functions. Monotone submodular set functions maximization is an extensively studied problem. Nemhauser et al. [32] show a simple greedy algorithm that achieves a $(1 - 1/e)$ -approximation for maximizing monotone submodular functions under cardinality constraints. Feige [14] shows that this is the best possible unless $P=NP$. A continuous version of the greedy algorithm was introduced by Calinescu et al. [6] to solve a *multilinear* extension of the problem which combined with the *pipage rounding* [1] techniques is shown to give a $(1 - 1/e)$ -approximation under matroid constraint for a special class of submodular function. Vondrak [41] shows that similar ideas can be used to give the same approximation for all monotone submodular functions. Checkuri, Vondrak, and Zenklusen [7] extend this framework and introduce *Contention Resolution Schemes* to show how to account for multiple matroid and knapsack constraints and also extend to non monotone functions. In this paper we use the contention resolution and a bound from [6] to bound the adaptivity gap between non-adaptive and adaptive policies.

Submodular functions have been a crucial tool in the influence maximization literature. Kempe, Kleinberg, and Tardos [22] show that a class of influence models called *Triggering model* are all submodular functions and thus can be approximated by a greedy algorithm as discussed above. Mossel and Roch [31] proved a conjecture posted in [22] and show that far more general influence models can be expressed by a submodular function. These models have been studied in numerous papers both theoretically and empirically [4, 5, 8, 17, 26, 27, 28], and thus are the main focus of this paper as well.

Stochastic optimization. The adaptive seeding model is a stochastic optimization framework (see [37] for a survey). There has been extensive work on stochastic optimization problems in the context of approximation algorithm varying from two-stage with recourse minimization problems ([19],[21], [33], [38, 39],[40]) and of maximizing an objective under a budget constraint ([10], [18],[23]). The adaptive seeding model is different from these problems as it combines a two stage model with a maximization under budget problem. Moreover, the constraint structure in adaptive seeding is such that the first stage decisions determine the available actions in the second stage (not just through the shared budget constraint). Another variant of multi-stage stochastic submodular maximization was studied by Asadpour et al. [3] and Golovin and Krause [16]. Despite the similar name, the adaptive seeding model is substantially different from these models, both in motivation and in the fact that the second stage problem is strongly dependent on choices made in the first stage. In these models the algorithm has access to the entire network but has the freedom to choose one node at a time (or group of nodes) and observe the realized value of these nodes. Yang et al. [43] and Chen et al. [9] also study “multi-level” models of the influence maximization problem, the latter partially inspired by the adaptive seeding model. However, their motivation, model and benchmark are substantially different from the adaptive seeding model, and apart from some special cases, their models do not have any constant factor approximations.

2 Non-adaptive policies

A non-adaptive policy is a pair of sets $(S, T) \subseteq X \times \mathcal{N}(X)$ where S represents the set selected in the first stage and $T \subseteq \mathcal{N}(S)$ is the set selected in the second stage. The natural definition for feasibility would be that the policy selects at most k nodes, though it is easy to construct examples that show that such strict policies have an unbounded approximation ratio. We therefore consider *relaxed* non-adaptive policies whose guarantee is to select at most k nodes *in expectation*, where the expectation is over the randomization in the model, i.e. the probabilities of nodes arriving in the second stage. In the rest of the paper we drop the *relaxed* prefix and just call such policies non-adaptive policies. We define the value of a non-adaptive policy (S, T) to be $F(T) = \sum_{i \in [m]} p(R_i) f(T \cap R_i)$ and its cost as $|S| + \mathcal{C}(T)$, where $\mathcal{C}(T) = \sum_{i \in T} p_i$. Finding the optimal non-adaptive policy requires solving the optimization problem:

$$\text{OPT}_{NA} = \max_{S, T} \{F(T) : |S| + \mathcal{C}(T) \leq k, S \subseteq X, T \subseteq \mathcal{N}(S)\}.$$

The crucial difference between these policies and adaptive ones is that non-adaptive policies fix a set T a priori to seeing the realization, whereas adaptive policies have the luxury of selecting a different set T_i for every realization R_i . Note that these policies are not a relaxation of adaptive policies nor are they a special case, since on the one hand they fix one set T but on the other hand are only restricted to the budget in expectation. Thus, it is not immediately clear they are useful for approximating adaptive policies.

We first show that the adaptivity gap (i.e. the ratio between the value of the optimal adaptive policy and that of the non-adaptive policy) is exactly $1 - 1/e$.

A tight $1 - 1/e$ adaptivity gap. Consider an instance with a single node in X connected to $n = 1/\delta^2$ nodes, each appearing with probability δ , for some small $\delta > 0$. The function is:

$$f(T) = \begin{cases} 1 & \text{if } T \neq \emptyset \\ 0 & \text{otherwise.} \end{cases}$$

For a budget of 2, an optimal adaptive policy seeds the single node in X , waits for the realization of its neighbors, and seeds whichever node realizes. The optimal non-adaptive policy here seeds the single node in X and spends the rest of its budget on $1/\delta$ nodes in $\mathcal{N}(X)$, which has an expected utility of $1 - (1 - \delta)^{1/\delta} \approx (1 - 1/e)$. The adaptivity gap is therefore at least $1 - 1/e$.

We complement the above example by showing that the adaptivity gap is at most $1 - 1/e$. The proof utilizes an interesting connection between the optimal adaptive policy and the *concave closure* of the underlying submodular function [6]. Using this connection and properties of the concave closure from [6] we can bound the error of a *fractional* non-adaptive policy that we then round to get our result.

Lemma 2.1. *For any budget k , $\text{OPT}_{NA} \geq (1 - 1/e - 2/k)\text{OPT}_A$.*

Proof. Let S be the set chosen by the optimal adaptive policy and let T_i be the set chosen by this policy in realization R_i . For a set $T \subseteq \mathcal{N}(S)$ let α_T be the total probability that T is chosen by the adaptive policy in the second stage. That is, $\alpha_T = \sum_{i \in \{i | T=T_i\}} p(R_i)$. For $i \in \mathcal{N}(S)$ let \mathbf{q}_i be the probability that i is seeded over all realizations in the second stage. That is, $\mathbf{q}_i = \sum_{\{T | i \in T\}} \alpha_T$. Consider the function:

$$f^+(\mathbf{q}) = \max_{\tilde{\beta}} \left\{ \sum_{T \subseteq \mathcal{N}(S)} \beta_T f(T) \mid \sum_T \beta_T = 1; \beta_T \geq 0; \sum_T \beta_T 1_T = \mathbf{q} \right\}.$$

Obviously, $\sum_T \alpha_T = 1$ and $\sum_T \alpha_T \mathbf{1}_T = \mathbf{q}$. Thus $\vec{\alpha}$ is a valid $\vec{\beta}$ for which $f^+(\mathbf{q})$ optimizes over. Since $\text{OPT}_A = \sum_T \alpha_T f(T)$ this mean that $f^+(\mathbf{q}) \geq \text{OPT}_A$.

Consider instead the process in which at the second stage each element $y_i \in \mathcal{N}(S)$ is chosen independently with probability \mathbf{q}_i and call the value of this process $F(\mathbf{q})$. By a consequence of [6, Lemma 5] we know that $F(\mathbf{q}) \geq (1 - 1/e)f^+(\mathbf{q}) \geq (1 - 1/e)\text{OPT}_A$. Now note that $\mathbf{q}_i \leq \mathbf{p}_i$ as an item can't be seeded with a higher probability than the probability it realizes. Thus, there exists $\mathbf{t} \in [0, 1]^{|N(S)|}$ such that for every $i \in \mathcal{N}(S)$ $\mathbf{q}_i = \mathbf{t}_i \mathbf{p}_i$. Since in each realization only $k - |S|$ element are chosen, we know that $\mathbf{t}^T \mathbf{p} < k - |S|$. Thus, \mathbf{t} can be understood as a fractional solution for the second stage set of the non-adaptive policy. We can round \mathbf{t} using the *pipage rounding* [1] technique in order to get a vector with only one fractional solution and no loss in value. Note that we can assume without loss of generality that the fractional entry has the smallest marginal density out of all non-zero entries and that there are no fractional entries if $\mathbf{t}^T \mathbf{p} < k - |S|$.

First assume that $|S| < k/2$. To get T we take the items for which the rounded vector entry is 1. From submodularity and the fact that $\mathbf{t}^T \mathbf{p} \geq k/2$ we get that the solution is a valid non-adaptive policy and has at most $2/k$ loss of the fractional solution. If $|S| \geq k/2$ we can instead remove an item of S (and all the entries that are connected only to it) with the least marginal value and include the fractional entry in T (if it is not connected to that item). From submodularity, we can divide the value of the solution between the different items of S such that at least one of them has marginal value less than $2/k$ so we get a valid solution with at most $2/k$ loss. So we constructed a valid non-adaptive policy whose value is at least $(1 - 1/e - 2/k)\text{OPT}_A$ and thus get our result. \square

We next show that a non-adaptive policy can be converted to an adaptive policy with small loss by using the contention resolution scheme [7].

Lemma 2.2. *For every $\epsilon \in (0, 1/5)$ and for any non-adaptive policy (S, T) such that $k - |S| > \epsilon^{-4}$, there exist an adaptive policy with value $\geq (1 - 2\epsilon) F(T)$.*

Proof. Given a solution (S, T) consider an adaptive policy which seeds the same set S in the first stage; in the second stage, for every realization R of neighbors of S the policy selects each node $j \in R \cap T$ with probability $(1 - \epsilon)$ into a set \hat{T} and seeds the nodes in \hat{T} if $|\hat{T}| > \xi = k - |S|$ and otherwise does not seed any nodes. We next compute the probability of any element j to be seeded given that it is in \hat{T} .

$$\begin{aligned}
\Pr [j \text{ is seeded} \mid j \in \hat{T}] &= \Pr [|\hat{T}| \leq \xi \mid j \in \hat{T}] \\
&= \Pr [|\hat{T} \setminus \{j\}| \leq \xi - 1] \\
&= 1 - \Pr [|\hat{T} \setminus \{j\}| > \xi - 1] \\
&\geq 1 - \Pr \left[|\hat{T} \setminus \{j\}| > \left(\frac{\xi - 1}{(1 - \epsilon)(\xi - \mathbf{p}_j)} \right) \mathbb{E} [|\hat{T} \setminus \{j\}|] \right] \\
&\geq 1 - \exp \left(- \frac{(1 - \epsilon)(\xi - \mathbf{p}_j)}{3} \left(\frac{\xi - 1}{(1 - \epsilon)(\xi - \mathbf{p}_j)} - 1 \right)^2 \right) \\
&\geq 1 - \exp \left(- \frac{\epsilon^2 \xi - 2}{3} \right) \\
&\geq 1 - \exp(-\epsilon^{-1}) \\
&\geq 1 - \epsilon
\end{aligned}$$

We derive the first inequality from $(1 - \epsilon)(\xi - \mathbf{p}_j) \geq \mathbb{E}[|\hat{T} \setminus \{j\}|]$. Notice that since $\epsilon < 0.5$, we have that $\frac{\xi - 1}{(1 - \epsilon)(\xi - \mathbf{p}_j)} \in (1, 2)$, and thus the second inequality follows from Chernoff bound (See appendix

for the exact bound used). The third inequality follows from simple arithmetic derivations. The fourth is from substituting ξ with ϵ^{-4} and using the fact that $\epsilon < 1/5$.

We therefore have that the probability j is seeded given that it is realized (in R) is at least $(1 - \epsilon)^2 \geq (1 - 2\epsilon)$. We also have that the seeded set is always of size at most ξ . In addition for any element j and any two realizations R_1 and R_2 such that $j \in R_1 \subseteq R_2$ we have that the probability j is seeded when the realization is R_1 is higher than if the realization is R_2 . These three conditions define a monotone $(1 - 2\epsilon)$ -balanced *contention resolution scheme* [7] and thus using the results from [7] we get that the expected value of this process (and thus of the adaptive policy) is at least $(1 - 2\epsilon)F(T)$. \square

Combining these results we can prove the following theorem:

Theorem 2.3. *For every $\epsilon > 0$, given an algorithm that finds a **non-adaptive** policy with value at least γOPT_{NA} , there is a $(1 - 1/e)\gamma - \epsilon$ approximation algorithm for the optimal **adaptive** policy.*

Proof. Run the non-adaptive algorithm to get a non-adaptive policy (S, T) , with an approximation of $\gamma > 0$ for OPT_{NA} . First, assume that both $k > 4/\epsilon$ and $k - |S| > (4/\epsilon)^4$. We can then use the same adaptive policy as in Lemma 2.2 with parameter $\epsilon/4$ and let $\text{Adapt}(S, T)$ be the value of that policy.

$$\begin{aligned} \text{Adapt}(S, T) &\geq (1 - \epsilon/2) F(T) \\ &\geq (1 - \epsilon/2) \gamma \text{OPT}_{NA} \\ &\geq ((1 - \epsilon/2)(1 - 1/e - \epsilon/2)) \gamma \text{OPT}_A \\ &\geq ((1 - 1/e)\gamma - \epsilon) \text{OPT}_A \end{aligned}$$

where the first inequality is due to Lemma 2.2 and the third is due to Lemma 2.1.

If $k - |S| < c = (4/\epsilon)^4$ we can iteratively remove from S the $\lceil c \rceil$ elements that contribute the least to the value of the solution, as well as the elements of T that are connected only to the removed elements. Let S', T' be the result of this procedure. Notice that $k - |S'| > c$. As we removed the elements with the least value we know that $F(T') \geq (1 - \frac{\lceil c \rceil}{|S|})F(T)$, and thus by the same argument as in the previous case we get that $\text{Adapt}(S', T') \geq (1 - \frac{\lceil c \rceil}{|S|})\text{Adapt}(S, T)$. It is easy to check that if $k > O(\frac{\lceil c \rceil}{\epsilon})$ we still get the desired approximation ratio.

If k does not satisfy one of the conditions above (so smaller than some constant) we can find an optimal adaptive policy by a brute force search over sets of size at most k in the first stage (we can approximate their value to any desired accuracy by sampling realization and finding the optimal second stage set). \square

2.1 Optimization via Non-Adaptive Policies

Given the blackbox reduction in Theorem 2.3, one can consider the problem of designing algorithms for non-adaptive policies. We now describe the simple greedy algorithm `NONADAPTIVEGREEDY` which is similar to the one sketched in the Introduction: at each step, as long as it doesn't exceed the total budget, the algorithm adds the densest ϵ -block. For non-adaptive policies, an ϵ -block is a node $x \in X$ and a subset of its neighbors whose *expected* cardinality is at most $1/\epsilon$. A formal description of the algorithms follows in Algorithm 1. It assumes a black-box access to an algorithm that finds an approximate optimal ϵ -block called `FINDOPTIMALNONADAPTIVEBLOCK`.

The next lemma shows that for any $\alpha > 0$, a procedure which guarantees an α -approximation for the optimal ϵ -block translates to a $(1 - 1/e^\alpha - \epsilon)$ -approximation guarantee for the optimal non-adaptive policy.

Algorithm 1 NONADAPTIVEGREEDY

Input: $f : 2^{\mathcal{N}(X)} \rightarrow \mathbb{R}_+$, budget k .

- 1: $S \leftarrow \emptyset, \mathbf{q} \leftarrow \vec{0}$.
 - 2: **while** $|S| + \sum_{j \in T} p_j \leq k - \frac{3}{\epsilon}$ **do**
 - 3: $(x, B) \leftarrow \text{FINDOPTIMALNONADAPTIVEBLOCK}(S, T)$
 - 4: $(S, T) = (S \cup x, T \cup B)$
 - 5: **end while**
 - 6: **return** (S, T)
-

Lemma 2.4. $\forall \epsilon > 0$, assume that in every iteration FINDOPTIMALNONADAPTIVEBLOCK returns a block which is an α -approximation to the optimal ϵ -block. Then, when $k = \Omega(1/\epsilon^2)$ Algorithm NONADAPTIVE returns a solution (S, T) such that $F(T) \geq (1 - 1/e^\alpha - O(\epsilon)) \text{OPT}_{NA}$.

Proof. Let (S_j, T_j) be the solution at the beginning of iteration j . For a block (x, B) let $F_{T_j}(B) = \sum_{i \in [m]} p(R_i)(f((T_j \cup B) \cap R_i) - f(T_j \cap R_i))$. Let (x_j, B_j) be the solution returned by FINDOPTIMALNONADAPTIVEBLOCK at iteration j and let (x_O, B_O) be the optimal ϵ -block in this iteration. First we observe that similarly to the non-stochastic case the optimal ϵ -block is a $(1 - 2\epsilon)$ -approximation of the densest block. Consider an iteration j . For any block (x, B) we have that if $C(B) < 1/\epsilon$ than the optimal ϵ -block has at least the same marginal density. Otherwise, let B_ϵ be the set of elements of B of highest marginal value of cost at most $1/\epsilon$.

$$\frac{F_{T_j}(B_O)}{1 + C(B_O)} \geq \frac{F_{T_j}(B_\epsilon)}{1 + C(B_\epsilon)} \geq \frac{\frac{1/\epsilon - 1}{C(B)} F_{T_j}(B)}{1 + 1/\epsilon} \geq \frac{1 - \epsilon}{1 + \epsilon} \frac{F_{T_j}(B)}{C(B)} \geq (1 - 2\epsilon) \frac{F_{T_j}(B)}{1 + C(B)}$$

where the first inequality is because B_O is the optimal ϵ -block and B_ϵ is a candidate, the second is because B_ϵ is at least of size $1/\epsilon - 1$ and from the submodularity of the function, and the other steps are just simple algebra.

We can think of the optimal non-adaptive solution as a set $O \subseteq X$ and arbitrarily partition the nodes in $\mathcal{N}(O)$ such that for each node $o_\ell \in O$ we associate a set of children $O_\ell \subseteq \mathcal{N}(o_\ell)$. The cost associates with each node and its children is simply $1 + C(O_\ell)$. Thus, we have that:

$$\frac{F_{T_j}(B_j)}{1 + C(B_j)} \geq \alpha \frac{F_{T_j}(B_O)}{1 + C(B_O)} \geq \alpha (1 - 2\epsilon) \max_{\ell} \frac{F_{T_j}(O_\ell)}{1 + C(O_\ell)} \geq \alpha (1 - 2\epsilon) \frac{\sum_{\ell} F_{T_j}(O_\ell)}{\sum_{\ell} 1 + C(O_\ell)} \geq \alpha (1 - 2\epsilon) \frac{F_{T_j}(O)}{k}$$

where the second inequality is from the last equation, and the last is from submodularity.

We proceed by induction to show that at any iteration we have that

$$F(T_{j+1}) \geq \left(1 - \prod_{\ell=1}^j \left(1 - \alpha (1 - 2\epsilon) \frac{1 + C(B_\ell)}{k} \right) \right) \text{OPT}_{NA}$$

The base case is trivial. Now assume it is true for $F(T_j)$. Then we have that

$$\begin{aligned}
F(T_{j+1}) &\geq F_{T_j} + \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_j)}{k} (\text{OPT}_{NA} - F(T_j)) \\
&= \left(1 - \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_j)}{k}\right) F(T_j) + \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_j)}{k} \text{OPT}_{NA} \\
&\geq \left(1 - \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_j)}{k}\right) \left(1 - \prod_{\ell=1}^{j-1} \left(1 - \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_\ell)}{k}\right)\right) \text{OPT}_{NA} \\
&\quad + \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_j)}{k} \text{OPT}_{NA} \\
&= \left(1 - \prod_{\ell=1}^j \left(1 - \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_\ell)}{k}\right)\right) \text{OPT}_{NA}
\end{aligned}$$

where the first inequality is from the previous derivation and the second inequality is by the induction hypothesis.

After the last iteration the cost of the solution is greater than $k - \frac{3}{\epsilon} > (1 - 3\epsilon)k$. Therefore,

$$\begin{aligned}
F(T) &\geq \left(1 - \prod_{\ell=1}^t \left(1 - \alpha(1-2\epsilon) \frac{1 + \mathcal{C}(B_\ell)}{k}\right)\right) \text{OPT}_{NA} \\
&\geq \left(1 - \left(1 - \frac{\alpha(1-2\epsilon)(1-3\epsilon)}{t}\right)^t\right) \text{OPT}_{NA} \\
&\geq \left(1 - \frac{1}{e^{\alpha(1-5\epsilon)}}\right) \text{OPT}_{NA} \\
&\geq \left(1 - \frac{1}{e^\alpha} - O(\epsilon)\right) \text{OPT}_{NA}
\end{aligned}$$

Where the second inequality is because setting all of the $\mathcal{C}(B_j)$ to be equal minimizes the function and we know that their sum is at least $(1 - 3\epsilon)k$. \square

2.2 Finding optimal non-adaptive ϵ -block

Lemma 2.4 shows that finding good approximation for non-adaptive policies reduces to computing approximations for the optimal ϵ -block. In this section we show that for some special cases we can find an optimal ϵ -block and thus a $(1 - 1/e)$ approximation for the optimal non-adaptive policy in polynomial time. Unfortunately, for general submodular functions we show it is unlikely that it can be approximated arbitrary well.

2.2.1 Approximating optimal ϵ -block for large probabilities

In the first special case we consider, all the probabilities on nodes are larger than some constant. Here, the optimization problem is easy. Given some constant $\epsilon > 0$ the algorithm simply enumerates over all $x \in X$ and over all possible subsets of items $T \in \mathcal{N}(x)$ s.t. $C(S) \leq 1/\epsilon$.

Corollary 2.5. *Let $\delta = \min_{i \in [n]} p_i$. Then for any constant $\epsilon > 0$, we can approximate the optimal non-adaptive policy to within $(1 - 1/e - \epsilon)$ in time $\text{poly}(n^{1/\delta})$.*

In the rest of this section we turn to the more challenging task of seeding nodes with small probabilities.

2.2.2 Approximating the optimal ϵ -block for MRS objective

In case the probabilities are small enumerating over all possible solutions is computationally infeasible. The problem of finding ϵ -blocks reduces to the following fundamental problem.

Definition 2.6. SUBMODULAR-OPTIMIZATION-WITH-SMALL-PROBABILITIES- δ (SOSP- δ) *problem:* We are given a monotone submodular function f and probabilities of each element realizing \mathbf{p} . The probabilities satisfy $\max_i \mathbf{p}_i \leq \delta$. Our goal is to find a set T of expected size k that maximizes the expected value of f .

We'll look at the fractional version of this problem in which the items of T are chosen independently with probabilities \mathbf{q} (this can be thought of as a multi-linear relaxation [6] of SOSP). Since we are only interested in small values of δ it is easy to round fractional solutions using the *pipage rounding* [1] technique with very small loss. Formally, we want to solve

$$\begin{aligned} \max_{\mathbf{q}} \quad & \mathbb{E}[f(T)] = \sum_T \left(\prod_{i \in T} \mathbf{q}_i \prod_{i \notin T} (1 - \mathbf{q}_i) \right) f(T) \\ \text{s.t.} \quad & \sum_i \mathbf{q}_i \leq k \\ & \mathbf{q}_i \in [0, \mathbf{p}_i] \quad \forall i \in [n] \end{aligned}$$

At a first glance, it may seem like no algorithm should be able to get an approximation better than $1 - 1/e$ for this problem: when $\delta = 1$ the problem identifies with submodular maximization under a cardinality constraint, and due to Feige we know that no algorithm can do better than $1 - 1/e$ unless P=NP even for coverage functions [14] (even for the fractional version). It seems like shrinking the constraint polytope by a factor of δ should not make a difference in the optimization. Surprisingly, as we next show, for submodular functions in a class known as *matroid rank sum* (which includes coverage functions), the above optimization problem can be solved nearly optimally. At a high level, we show that for such functions the problem can be well approximated through a convex program, which then enables us to produce a solution whose approximation becomes optimal as δ vanishes.²

Theorem 2.7. *Suppose that f can be represented as a matroid rank sum (MRS) function. Then, there exists a $(1 - \delta/2)$ -approximation algorithm for SOSP- δ using convex programming.*

Proof. Suppose that we relax the program, so that the probability of seeding node i is $1 - e^{-\mathbf{q}_i}$ (but the cost remain the same). We can now optimize the following program:

$$\begin{aligned} \max_{\mathbf{q}} \quad & \sum_T \left(\prod_{i \in T} (1 - e^{-\mathbf{q}_i}) \prod_{i \notin T} (e^{-\mathbf{q}_i}) \right) f(T) \\ & \sum_i \mathbf{q}_i \leq 1/\epsilon \\ & \mathbf{q}_i \in [0, \mathbf{p}_i] \quad \forall i \in [n] \end{aligned}$$

Dughmi et al. [12] consider essentially the same program in the context of Poisson rounding and show that this program is concave when f is a MRS function. (They use it to achieve a $(1 - 1/e)$ -approximation for general probabilities and do not consider the special case of small δ .) Thus, we can optimize this program (to within arbitrarily good approximation) in polynomial time.

²Note that this is not due to the low cost of elements which allows for example for a greedy algorithm to be nearly optimal for the knapsack problem.

Observe that $1 - e^{-\mathbf{q}_i} \leq \mathbf{q}_i$. Therefore we only decreased the probability of seeding each node, so by monotonicity of f , our expected value will be at least as good as the solution of the new concave program.

We lose at most a factor of $(1 - e^{-\delta})/\delta$ for any submodular function f (and in particular for matroid rank sum). Think of the process where the elements are added one by one, and consider the marginal contribution of each one. Let $\mu = \mu(\mathbf{q})$ denote the original distribution on sets (i.e. $\Pr_\mu[T] = \prod_{i \in T} \mathbf{q}_i \cdot \prod_{i \notin T} (1 - \mathbf{q}_i)$), and let $\nu = \nu(\mathbf{q})$ denote the transformed distribution (i.e. $\Pr_\nu[T] = \prod_{i \in T} (1 - e^{-\mathbf{q}_i}) \cdot \prod_{i \notin T} e^{-\mathbf{q}_i}$); let F_μ and F_ν denote the value of the objective function under each distribution.

$$\begin{aligned}
F_\nu(\mathbf{q}) &= \sum_i (1 - e^{-\mathbf{q}_i}) \mathbb{E}_{T \sim \nu} [f(\{i\} \cup (T \cap [i-1])) - f(T \cap [i-1])] \\
&\geq \sum_{i: a \in A_i} \left(\frac{1 - e^{-\delta}}{\delta} \cdot \mathbf{q}_i \right) \mathbb{E}_{T \sim \nu} [f(\{i\} \cup (T \cap [i-1])) - f(T \cap [i-1])] \\
&\geq \sum_{i: a \in A_i} \left(\frac{1 - e^{-\delta}}{\delta} \cdot \mathbf{q}_i \right) \mathbb{E}_{T \sim \mu} [f(\{i\} \cup (T \cap [i-1])) - f(T \cap [i-1])] \\
&= \frac{1 - e^{-\delta}}{\delta} \cdot F_\mu(\mathbf{q}).
\end{aligned}$$

The first step follows by considering the expected increment for adding i , with respect to ν . The second step follows by lower bounding $e^{-\mathbf{q}_i}$ and monotonicity. The third step follows by submodularity. Finally, the last step follows by again considering the expected increment for adding i , this time with respect to μ . \square

Finally, we obtain a tight approximation for the non-adaptive solutions to adaptive seeding instances with MRS objective and arbitrary probabilities by carefully combining the two special cases.

Theorem 2.8. *For any $\epsilon > 0$ there is a polynomial-time algorithm that returns a $(1 - 1/e - \epsilon)$ -approximation of the optimal non-adaptive policy for any matroid rank sum (MRS) function.*

Proof. Run Algorithm NONADAPTIVE with subroutine FINDOPTIMALNONADAPTIVEBLOCK that finds ϵ' -blocks implemented as follows: Enumerate over all feasible subsets of nodes with probabilities at least δ . For each subset, let k' be the remaining budget for this block. Solve the concave program for budget in $\{\epsilon'', 2\epsilon'', \dots, k'\}$. By Lemma 2.7, when our enumeration reaches the optimal subset of large-probabilities elements, and we use the approximately correct additional budget (we spend at most an additional ϵ budget), the solution of the concave program is a $((1 - \delta/2) - \epsilon'')$ -approximation to the densest subset.

The concave program might return a solution that does not correspond to a set (have some \mathbf{q}_i that does not equal to 0 or \mathbf{p}_i). However, using the *pipage rounding* [1] technique the algorithm can round it (make \mathbf{q}_i equals either \mathbf{p}_i or 0) to have at most one undetermined item without any loss of value. If such an item remains, the algorithm compares the density of the solution that includes that item to the solution that does not include it and chooses the one with maximum marginal density. It is easy to verify that one of those solution has a higher density than the density of the fractional solution. This procedure might cause the block to cost δ more so in total $1 + 1/\epsilon' + \delta$.

We chose ϵ', ϵ'' and $\delta \leq \epsilon'$ (thus the total cost of a block is at most $3/\epsilon'$) such that the total loss due to Lemma 2.4 is ϵ . For large values of k the theorem follows by Lemma 2.4 and by noticing that in the analysis of that lemma we only analyze iterations where there is at least $3/\epsilon'$ budget left so this procedure returns a valid block in each such iteration. If k is not large enough (smaller than some constant that depends on ϵ), we can enumerate over all first stage set of size at most k

and complete the solution by solving a monotone submodular maximization on the second stage to get a $(1 - 1/e)$ approximation. \square

Combining this with the results of the previous section we get that for MRS functions we have a $(1 - 1/e)^2$ approximation for the adaptive seeding problem as desired.

2.2.3 Hardness of approximating optimal ϵ -block for general submodular functions

Unfortunately, for general submodular functions the Submodular-optimization-with-small-probabilities problem cannot be approximated arbitrary well even with a constant budget (and as a special case, computing optimal densest ϵ -blocks is hard). Our algorithm heavily relies on the reduction to a concave program, yet as pointed out by [12, 13], this program is not concave in general. While this means our current approach fails it does not mean the problem is computationally hard.

Standard techniques for showing hardness of submodular maximization seem to fail for this problem: Feige’s construction [14] would also show hardness for the max-cover version, but we know that this problem is easy in this setting. The symmetry gap [42] should give an information-theoretic lower bound (in the oracle model), but with a constant budget it is easy to design an exponential-time, poly-information algorithm that achieves an arbitrarily good approximation.

We therefor look for a construction where the optimal set behaves *locally* very differently from a random set. In other words, when we look at a random $(1/\epsilon)$ -subset of the optimal set, it should be different from what we expect from a random $(1/\epsilon)$ -subset somewhere else in the graph. Intuitively, this is very similar to the PLANTED-CLIQUE problem, where every subset of the clique is of course also a clique, but the rest of the graph may be arbitrarily (constant) sparse [2].

Theorem 2.9. *If the Submodular-optimization-with-small-probabilities problem with a constant budget k can be approximated within any constant factor better than $(1 - e^{-k/2}) / (1 - (\frac{k}{2} + 1)e^{-k})$, then there is a polynomial time algorithm for the PLANTED-CLIQUE problem that succeeds with high probability. In particular, for $k = 1.7$,*

$$(1 - e^{-k/2}) / \left(1 - \left(\frac{k}{2} + 1\right) e^{-k}\right) \approx 0.865$$

Proof. We reduce from the Densest l -subgraph problem. In particular, Alon et al. [2] proved that for any constant $\epsilon > 0$, given a graph $G = (V, E)$ it is PLANTED-CLIQUE hard³ to distinguish between:

- Completeness: G contains a clique of size l ; or
- Soundness: every l -vertex subgraph of G is ϵ -sparse (i.e. it contains at most an ϵ -fraction of the $\binom{l}{2}$ edges an l -clique would contain.)

Given G , we construct a monotone submodular function that gives 1 for every subset that contains an edge, and otherwise approaches 1 exponentially with the size of the subset:

$$f(T) = \begin{cases} 1 & \exists (u, v) \in E \text{ s.t. } \{u, v\} \subseteq T \\ 1 - 2^{-|T|} & \text{otherwise} \end{cases}$$

We set the maximal probabilities such that a budget of k is sufficient to bid exactly for the entire l -clique: $p_i = k/l$. Monotonicity is trivial.

³See [2] for precise statement.

- **Submodularity:** Adding u to T may increase f by at most $2^{-|T|}$. However, adding u to any $T' \subsetneq T$ would increase f by at least $2^{-(|T'|-1)} \geq 2^{-|T|}$.

Now, observe that regardless of the choice of \mathbf{q} , the random variable $|T|$ (the size of the realized set) behaves approximately like a Poisson distribution with parameter k . More precisely, the total variation distance between $|T|$ and $\mathbf{Pois}(k)$ is bounded $\sum \mathbf{q}_i^2 \leq k^2/l < \epsilon$ (e.g. [25]).

- **Completeness:** Let \mathbf{q}^{opt} be 1 on the l -clique and 0 otherwise; then

$$\begin{aligned} \mathbb{E}[f(T^{\text{opt}})] &= 1 - \frac{1}{2} \Pr[|T| = 1] - \Pr[|T| = 0] \\ &\geq 1 - \frac{1}{2} \Pr[\mathbf{Pois}(k) = 1] - \Pr[\mathbf{Pois}(k) = 0] - \epsilon \\ &= 1 - \left(\frac{k}{2} + 1\right) e^{-k} - \epsilon \end{aligned}$$

- **Soundness:** Suppose that every l -subgraph of G is ϵ -sparse. Observe that by submodularity of f , we can assume w.l.o.g. that \mathbf{q}^{alg} defines a set, i.e. it is \mathbf{p}_i for l vertices that contain at most $\epsilon \binom{l}{2}$ edges (and 0 everywhere else). Then,

$$\mathbb{E}[f(T^{\text{alg}})] \leq 1 - \sum_{i=0}^{\infty} \Pr[|T| = i] 2^{-i} + \Pr[T \text{ contains an edge}]$$

For each of the $\epsilon \binom{l}{2}$ potential edges, the probability that both vertices belong to T is $\binom{|T|}{2} / \binom{l}{2}$. Taking a union bound over all of them, we have

$$\Pr[T \text{ contains an edge}] \leq \epsilon \binom{|T|}{2} < \epsilon |T|^2$$

Finally, since $|T|$ is distributed ϵ -like $\mathbf{Pois}(k)$,

$$\Pr[T \text{ contains an edge}] < \epsilon \cdot (k^2 + k) + \epsilon$$

Therefore,

$$\begin{aligned} \mathbb{E}[f(T^{\text{alg}})] &\leq 1 - \sum_{i=0}^{\infty} \Pr[\mathbf{Pois}(k) = i] 2^{-i} + O(\epsilon) \\ &= 1 - \sum_{i=0}^{\infty} \frac{k^i}{i!} e^{-k} 2^{-i} + O(\epsilon) \\ &= 1 - \left(\sum_{i=0}^{\infty} \frac{\left(\frac{k}{2}\right)^i}{i!} e^{-k/2} \right) \cdot e^{-k/2} + O(\epsilon) \\ &= 1 - e^{-k/2} + O(\epsilon) \end{aligned}$$

Thus, for budget k , it is PLANTED-CLIQUE hard to find any approximation which is better than $(1 - e^{-k/2}) / (1 - (\frac{k}{2} + 1) e^{-k})$. For example, set $k = 1.7$ to get the approximation factor of $(1 - e^{-1.7/2}) / (1 - (\frac{1.7}{2} + 1) e^{-1.7}) < 0.865$. (Recall that since we have fractional costs, the budget may be fractional as well.) \square

This lower bound implies that the non-adaptive framework we use here cannot obtain the $(1 - 1/e)^2$ approximation ratio⁴. This obstacle motivates our use of ϵ -locally adaptive policies discussed in the following section.

3 Approximation via ϵ -locally-adaptive policies

In this section we prove our main result:

Theorem 3.1. *For every constant $\epsilon > 0$ there is an algorithm that runs in polynomial time and returns an adaptive policy which is a $((1 - 1/e)^2 - \epsilon)$ -approximation to the optimal adaptive policy with general monotone submodular functions.*

Proof outline. Our entire proof relies on the novel definition of a restricted class of adaptive policies which we call ϵ -locally-adaptive. Informally, we say that a policy is ϵ -locally-adaptive, if it can be divided into ϵ -blocks. In this context, an ϵ -block is a subset of X of constant size (for technical reasons these are not singletons as in previous cases), and for each realization an adaptively chosen set of constant size of its neighbors. We prove that a greedy algorithm that in each iteration finds the optimal ϵ -block gives a $(1 - 1/e - \epsilon)$ -approximation to the optimal ϵ -locally-adaptive policy. This adaptive variant of ϵ -blocks allows us to find the optimal subset for each realization (much in the same way as in the warmup presented in the introduction) and thus find the optimal block. Thus, while the non-adaptive block structure allows for greedy optimizations, the power of adaptivity within a block circumvents the hardness result of the previous section. We then prove that the optimal ϵ -locally-adaptive policy is a $(1 - 1/e)$ -approximation to the optimal adaptive policy by using the fact that locally-adaptive policies dominate non-adaptive policies. In particular, we show we can convert a non-adaptive policy to an ϵ -locally-adaptive policy, with arbitrarily small loss in value, and thus our bound follows. A natural question is then whether locally-adaptive policies are as good as adaptive policies. We answer that question negatively by presenting an example that exhibits a gap (≈ 0.853) between the optimal locally-adaptive policy and the optimal adaptive policy. We conclude this expository subsection by formally defining ϵ -locally-adaptive policies.

Definition 3.2. *An (adaptive) ϵ -block is a set $S \subseteq X$ of size at most $1/\epsilon^2$ and for each realization R_i a set $T_i \subseteq \mathcal{N}(S) \cap R_i$ of size at most $2/\epsilon$. The cost of a block B is $\mathcal{C}(B) = |S| + \max_i(|T_i|)$. An ϵ -locally-adaptive policy is a set \mathcal{B} of (not necessarily disjoint) ϵ -blocks.⁵*

Let $T_{i,B}$ be the set seeded by block B in realization R_i and let $\mathcal{T}_i(\mathcal{B}) = \bigcup_{B \in \mathcal{B}} T_{i,B}$. We abuse notation and generalize the value and cost functions to be applied on these policies. That is, we let the value of such a policy be $F(\mathcal{B}) = \sum_{i=1}^m p(R_i) f(\mathcal{T}_i(\mathcal{B}))$ and its cost $\mathcal{C}(\mathcal{B}) = \sum_{B \in \mathcal{B}} \mathcal{C}(B)$. The optimal ϵ -locally-adaptive policy with budget k is then:

$$\text{OPT}_{LA}^\epsilon = \max_{\mathcal{B}} \{F(\mathcal{B}) : \mathcal{C}(\mathcal{B}) \leq k, \forall B \in \mathcal{B} : |S_B| \leq 1/\epsilon^2, \forall i : T_{i,B} \subseteq \mathcal{N}(S_B) \cap R_i, |T_{i,B}| \leq 2/\epsilon\}$$

3.1 The Algorithm

We now describe the `LOCALLYADAPTIVEGREEDY` algorithm. We run a greedy algorithm that in each iteration adds a new ϵ -block to the current solution. The algorithm always adds a block with an optimal marginal density, i.e. a block which maximizes the ratio between the expected marginal contribution and cost. A formal description of the algorithm is included below. The

⁴Note that this does not exclude worse approximations guaranteed via non-adaptive policies. In [35] non-adaptive policies are used to obtain a (poor) constant approximation guarantee under knapsack constraints. In this paper, we are interested in obtaining the $(1 - 1/e)^2$ approximation bounds.

⁵Note that the blocks are not necessarily independent - T_i can depend on the entire R_i and not only on $\mathcal{N}(S) \cap R_i$.

FINDOPTIMALADAPTIVEBLOCK subroutine simply enumerates over all subsets of size less than $1/\epsilon^2$ of X and all budgets of size at most $2/\epsilon$ and returns the pair with the highest marginal density.

Algorithm 2 LOCALLYADAPTIVEGREEDY

Input: budget k, ϵ

- 1: $\mathcal{B} \leftarrow \emptyset$
 - 2: **while** $\mathcal{C}(\mathcal{B}) < k - \frac{3}{\epsilon^2}$ **do**
 - 3: $\mathcal{B} \leftarrow \mathcal{B} \cup \text{FINDOPTIMALADAPTIVEBLOCK}(\mathcal{B}, \epsilon)$
 - 4: **end while**
 - 5: **return** \mathcal{B}
-

Polynomial-size representation. As there are possibly exponential many realizations we can't hope to output a full explicit description of a locally-adaptive policy. Our algorithm instead outputs for each block its first stage set S_i and a budget k_i for it to optimize in the second stage as well as an order over the blocks. At every realization the policy seeds the second stage nodes by going over the blocks by order and optimizing the choices of each block given only the choices made by the previous blocks. Note that this implicitly determines the content of each block. In our algorithm we implicitly assume the order on the blocks of \mathcal{B} is the order in which the algorithms adds them to \mathcal{B} . Note that we can approximate $F(\mathcal{B})$ and $F_{\mathcal{B}}(B)$ (the marginal value of block B for policy \mathcal{B}) for such a policy to any desired accuracy by sampling realizations and running this process on each of them (we thus assume in the analysis that we have an oracle for their value).

3.2 Analysis

Lemma 3.3. *For any $\epsilon > 0$, let \mathcal{B} be the solution returned by LOCALLYADAPTIVEGREEDY with input k, ϵ . Then, $F(\mathcal{B}) \geq \left(1 - 1/e - O\left((\epsilon\sqrt{k})^{-2}\right)\right) \text{OPT}_{LA}^\epsilon$.*

Proof. We first show that the marginal density of the ϵ -block chosen in each iteration of the algorithm is at least the marginal density of the optimal locally-adaptive policy. Note that we always have enough budget left to add a full sized block. Let \mathcal{B}_j be the solution at the beginning of iteration j and let B_j be the block added at iteration j with $k_j = \max_i(|T_{i,B_j}|)$. Let \mathcal{B}_O denote an optimal solution with value OPT_{LA}^ϵ . For every iteration of the algorithm we have that:

$$\begin{aligned}
\frac{F_{\mathcal{B}_j}(\mathcal{B}_O)}{k} &= \frac{\mathbb{E}_{R_i} \left[f_{\mathcal{T}_i(\mathcal{B}_j)}(\mathcal{T}_i(\mathcal{B}_O)) \right]}{k} \\
&\leq \frac{\sum_{B \in \mathcal{B}_O} \mathbb{E}_{R_i} \left[f_{\mathcal{T}_i(\mathcal{B}_j)}(T_{i,B}) \right]}{\sum_{B \in \mathcal{B}_O} \mathcal{C}(B)} \\
&\leq \max_{B \in \mathcal{B}_O} \frac{\mathbb{E}_R \left[f_{\mathcal{T}_i(\mathcal{B}_j)}(T_{i,B}) \right]}{\mathcal{C}(B)} \\
&\leq \frac{\mathbb{E}_{R_i} \left[f_{\mathcal{T}_i(\mathcal{B}_j)}(T_{i,B_j}) \right]}{|S_j| + k_j} \\
&= \frac{F_{\mathcal{B}_j}(B_j)}{|S_j| + k_j}
\end{aligned}$$

The first inequality is from the submodularity of f , and the third holds because the algorithm enumerates over all ϵ -blocks as candidates in each iteration - including over the blocks of \mathcal{B}_O .

We proceed by induction to show that at any iteration we have that

$$F(\mathcal{B}_{j+1}) \geq \left(1 - \prod_{l=1}^j \left(1 - \frac{|S_l| + k_l}{k}\right)\right) \text{OPT}_{LA}^\epsilon$$

The base case is trivial. Assume it is true for $F(\mathcal{B}_j)$. Then we have that

$$\begin{aligned} F(\mathcal{B}_{j+1}) &\geq F(\mathcal{B}_j) + \frac{|S_j| + k_j}{k} (\text{OPT}_{LA}^\epsilon - F(\mathcal{B}_j)) \\ &= \left(1 - \frac{|S_j| + k_j}{k}\right) F(\mathcal{B}_j) + \frac{|S_j| + k_j}{k} \text{OPT}_{LA}^\epsilon \\ &\geq \left(1 - \frac{|S_j| + k_j}{k}\right) \left(1 - \prod_{l=1}^{j-1} \left(1 - \frac{|S_l| + k_l}{k}\right)\right) \text{OPT}_{LA}^\epsilon + \frac{|S_j| + k_j}{k} \text{OPT}_{LA}^\epsilon \\ &= \left(1 - \prod_{l=1}^j \left(1 - \frac{|S_l| + k_l}{k}\right)\right) \text{OPT}_{LA}^\epsilon \end{aligned}$$

where the first inequality is from the previous derivation and the second inequality is by the induction hypothesis.

When the algorithm ends the cost of the solution is greater than $k' > \left(1 - \frac{3}{k\epsilon^2}\right)k$. Therefore,

$$\begin{aligned} F(\mathcal{B}) &\geq \left(1 - \prod_{j=1}^t \left(1 - \frac{|S_j| + k_j}{k}\right)\right) \text{OPT}_{LA}^\epsilon \\ &\geq \left(1 - \left(1 - \frac{k'}{kt}\right)^t\right) \text{OPT}_{LA}^\epsilon \\ &\geq \left(1 - \frac{1}{e^{k'/k}}\right) \text{OPT}_{LA}^\epsilon \\ &\geq \left(1 - \frac{1}{e} - O\left(\frac{1}{k\epsilon^2}\right)\right) \text{OPT}_{LA}^\epsilon \end{aligned}$$

where the second inequality is because the solution's cost is at least k' and this expression is minimized when the cost is evenly spread over the iterations. \square

Adaptivity gap of ϵ -locally-adaptive policies. We now show that ϵ -locally-adaptive policies can arbitrarily approximate non-adaptive policies, which implies our bound. The high level idea is to show that there are good non-adaptive policies that have a *block* structure and thus can be converted to locally-adaptive policies. We first define the notion of ϵ -local for non-adaptive policies:

Definition 3.4. A budgeted ϵ -block is a triplet (S, k, T) such that $S \subseteq X$ is of size at most $1/\epsilon^2$, $\frac{1}{\epsilon} \leq k \leq \frac{2}{\epsilon}$ and $T \subseteq \mathcal{N}(S)$ satisfies $\sum_{j \in T} p_j \leq k$. An ϵ -local non-adaptive policy is a set \mathcal{L} of budgeted ϵ -blocks. The cost of \mathcal{L} is $\mathcal{C}(\mathcal{L}) = \sum_{B \in \mathcal{L}} |S_B| + k_B$.

We now prove that a non-adaptive policy can be converted into an ϵ -local non-adaptive policy. Let $\mathcal{T}(\mathcal{L})$ be the union of all of \mathcal{L} 's blocks second stage sets.

Lemma 3.5. For any non-adaptive policy (S, T) in which $|S| + \mathcal{C}(T) > 3/\epsilon^3$, there exists an ϵ -local non-adaptive policy \mathcal{L} of the same cost such that $F(\mathcal{T}(\mathcal{L})) \geq (1 - 3\epsilon)F(T)$.

Proof. Let $k = |S| + \mathcal{C}(T)$. Fix some order $(x_1 \dots x_{|S|})$ on the elements of S . We say that a second-stage node $y \in \mathcal{N}(S) \cap T$ belongs to x_j if x_j is the smallest-indexed node in S with an edge to y . For a set $Q \subseteq S$, let $\mathcal{R}(Q)$ be the nodes that belong to nodes in Q . Iteratively construct the blocks of \mathcal{L} by adding nodes from S into sets S_i by the order $(x_1 \dots x_{|S|})$, until either

- $2/\epsilon \geq \mathcal{C}(\mathcal{R}(S_i)) > 1/\epsilon$ - set $T_i = \mathcal{R}(S_i)$ and $k_i = \mathcal{C}(\mathcal{R}(S_i))$.
- $\mathcal{C}(\mathcal{R}(S_i)) > 2/\epsilon$ - let Φ be the maximal set of nodes of $\mathcal{R}(x)$, where x is the last node added to S_i , that can be added to $\mathcal{R}(S_i \setminus x)$ such that the cost remains under $2/\epsilon$ (there is at least one such node since a node's cost is at most 1); set $T_i = \mathcal{R}(S_i \setminus x) \cup \Phi$ and $k_i = \mathcal{C}(T_i)$; start the next set S_{i+1} again with x but ignore the nodes in Φ .
- $|S_i| = 1/\epsilon^2$ or we are done - set $T_i = \mathcal{R}(S_i)$ and $k_i = 1/\epsilon$.

The first condition incurs no extra cost. The second condition applies at most $\lceil \epsilon \mathcal{C}(T) \rceil$ times and incurs an extra cost of 1 for the duplicated node, and in total at most $\epsilon \mathcal{C}(T) + 1$. The third condition applies at most $\lceil \epsilon^2 |S| \rceil$ times and it incurs an extra cost of at most $1/\epsilon$, and in total at most $\epsilon |S| + 1/\epsilon$. The total additional cost incurred is therefore at most $\epsilon k + 1 + 1/\epsilon \leq 2\epsilon k$.

Iteratively remove from \mathcal{L} the blocks whose marginal density is the lowest until $\mathcal{C}(\mathcal{L}) \leq k$. Their total cost at most $2\epsilon k + (1/\epsilon^2 + 2/\epsilon) \leq 3\epsilon k$. Thus, by submodularity, $F(\mathcal{T}(\mathcal{L})) \geq (1 - 3\epsilon)F(T)$. \square

Our last step is to show that given an ϵ -local non-adaptive policy, we can construct an ϵ -locally-adaptive policy with almost the same value. The following lemma follows from Lemma 2.2 by analyzing each block of the policy independently. The ϵ^4 is needed to match the condition on the budget left for the second stage in that Lemma.

Lemma 3.6. *For any ϵ such that $\epsilon \in (0, 1/5)$, and for any ϵ^4 -local non-adaptive policy \mathcal{L} there exists an ϵ^4 -locally-adaptive policy \mathcal{B} with value $\geq (1 - 2\epsilon) F(\mathcal{T}(\mathcal{L}))$.*

Putting it all together, we have that for every constant $\epsilon > 0$ there exist $\epsilon_1, \epsilon_2, \epsilon_3$ such that:

$$F(\mathcal{B}) \geq (1 - 1/e - \epsilon_1) \text{OPT}_{LA}^{\epsilon_2} \geq (1 - 1/e - \epsilon_3) \text{OPT}_{NA} \geq ((1 - 1/e)^2 - \epsilon) \text{OPT}_A$$

where the first inequality follows from Lemma 3.3, the second from Lemma 3.6 and the third from the results of the previous section. This completes our proof of Theorem 3.1. \square

3.3 Separation between ϵ -locally-adaptive and adaptive policies

We complement this section with an example that exhibits a gap between the values of the optimal ϵ -local-adaptive policy and the optimal adaptive policy. It remains open whether the optimal gap is $(1 - 1/e)$ as our upper bound suggests. Any better upper bound on the gap will immediately imply a better approximation bound for Algorithm `LOCALLYADAPTIVEGREEDY`.

Lemma 3.7. *There are instances where every ϵ -locally-adaptive policy achieves at most ≈ 0.853 of the value of the optimal adaptive policy.*

Proof. We construct an instance where it is advantageous to move large amounts of the budget after seeing the realizations. Since this is only possible in a fully adaptive (i.e. not ϵ -locally-adaptive) policy, we obtain a separation between the two classes of solutions.

Let m be a large parameter, and consider an instance where the first stage has $|X| = m$ nodes. Each node $x \in X$ is connected to one *special* node $y_x \in Y$, and it also has m^2 *regular* neighbors

$z_x^i \in Z_x \subset Z$. The special nodes in Y realize with probability $1/m$, while the regular nodes in Z realize with probability 1. Let the budget be $k = m^2 + m + 1$. Finally, consider the function:

$$f(T) = 1 - \prod_{x \in X} \left(1 - \frac{1}{2} |T \cap y_x| - \frac{1}{2m^2} |T \cap Z_x| \right).$$

To see that f is indeed submodular, consider the following observation by Dughmi and Vondrak [13]: For any monotone submodular functions g_1 and g_2 with values in $[0, 1]$, $g(S) := 1 - (1 - g_1(S))(1 - g_2(S))$ is also monotone submodular. Applying this lemma recursively, we see that for any number of monotone submodular functions $g_1 \dots g_t$ with values in $[0, 1]$, $g(S) := 1 - \prod_i (1 - g_i(S))$ is also monotone submodular. Finally, f is submodular since it can be written in this form for coverage functions $f_x(T) = \frac{1}{2} |T \cap y_x| - \frac{1}{2m^2} |T \cap Z_x|$.

Optimal adaptive: The optimal adaptive solution seeds all the first-stage nodes. With probability $1 - 1/e$, one of the special nodes y_x realizes, in which case the optimal policy seeds y_x , as well as all the nodes $z_x \in Z_x$. In this case, $f(T) = 1$. With probability $1/e$, none of the special nodes realize. In this case, the adaptive policy picks x arbitrarily, and seeds all the nodes $z_x \in Z_x$. In this case $f(T) = 1/2$. In expectation, $\mathbb{E}[f(T)] = 1 - 1/(2e)$.

ϵ -locally adaptive: Given a realization, we say that a block S_i is *special* if it contains a node with a realized special neighbor, and *regular* otherwise. Observe that in any locally adaptive policy, the probability that a block S_i is special is less than $1/(m\epsilon^2)$ by union bound. Therefore, the expected total budget k_i allocated to special blocks S_i is at most $O(m/\epsilon^2)$, i.e. negligible in comparison to $|Z_x| = m^2$. Therefore any locally adaptive policy must spend all but a negligible amount of its budget on regular subsets. (It may still seed all the realized special nodes, but with only a negligible fraction of their neighbors). Let ξ denote the number of realized special nodes. Conditioning on ξ , we have,

$$\mathbb{E}[f(T) \mid \xi] \leq 1 - 2^{-\xi} \cdot \prod_{x \in X} \left(1 - \frac{1}{2m^2} |T \cap Z_x| \right) + o(1).$$

The optimal way to spend the budget on regular blocks is to pick some x arbitrarily, and seed (almost) all the nodes $z_x \in Z_x$, leaving sufficient budget to seed any realized special nodes. In particular, for any feasible T ,

$$\prod_{x \in X} \left(1 - \frac{1}{2m^2} |T \cap Z_x| \right) \geq 1 - \sum_{x \in X} \frac{1}{2m^2} |T \cap Z_x| \geq 1/2.$$

Finally, notice that the distribution of ξ is approximately **Pois**(1). The expected value of the locally adaptive policy is therefore:

$$\begin{aligned} \mathbb{E}[f(T)] &\approx 1 - \frac{1}{2} \cdot \sum_{i=0}^{\infty} \Pr[\mathbf{Pois}(1) = i] 2^{-i} \\ &= 1 - \frac{1}{2} \cdot \sum_{i=0}^{\infty} \frac{1}{i!} e^{-1} 2^{-i} \\ &= 1 - \frac{1}{2} \left(\sum_{i=0}^{\infty} \frac{\left(\frac{1}{2}\right)^i}{i!} e^{-1/2} \right) \cdot e^{-1/2} \\ &= 1 - \frac{1}{2} e^{-1/2} \end{aligned}$$

Thus the ratio between the values of the policies is $\frac{1 - 1/(2e)}{1 - \frac{1}{2} e^{-1/2}} \approx 0.853$. \square

References

- [1] Alexander A. Ageev and Maxim Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. Comb. Optim.*, 8(3):307–328, 2004.
- [2] Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest k-subgraph from average case hardness, 2011.
- [3] Arash Asadpour, Hamid Nazerzadeh, and Amin Saberi. Stochastic submodular maximization. In *Internet and Network Economics*, pages 477–489. Springer, 2008.
- [4] Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Everyone’s an influencer: quantifying influence on twitter. In *WSDM*, 2011.
- [5] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. Maximizing social influence in nearly optimal time. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, SODA*, volume 14. SIAM, 2014.
- [6] Grigori Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *Integer programming and combinatorial optimization*, pages 182–196. Springer, 2007.
- [7] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 783–792. ACM, 2011.
- [8] Ning Chen. On the approximability of influence in social networks. In *SODA*, pages 1029–1037, 2008.
- [9] Wei Chen, Fu Li, Tian Lin, and Aviad Rubinfeld. Combining Traditional Marketing and Viral Marketing with Amphibious Influence Maximization. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC ’15, Portland, OR, USA, June 15-19, 2015*, pages 779–796, 2015.
- [10] Brian C Dean, Michel X Goemans, and J Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 208–217. IEEE, 2004.
- [11] Pedro Domingos and Matthew Richardson. Mining the network value of customers. In *KDD*, pages 57–66, 2001.
- [12] Shaddin Dughmi, Tim Roughgarden, and Qiqi Yan. From convex optimization to randomized mechanisms: toward optimal combinatorial auctions. In *Proceedings of the 43rd annual ACM symposium on Theory of computing, STOC ’11*, pages 149–158, New York, NY, USA, 2011. ACM.
- [13] Shaddin Dughmi and Jan Vondrák. Limitations of randomized mechanisms for combinatorial auctions. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 502–511. IEEE, 2011.
- [14] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.
- [15] Scott L Feld. Why your friends have more friends than you do. *American Journal of Sociology*, pages 1464–1477, 1991.

- [16] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42(1):427–486, 2011.
- [17] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *KDD*, pages 1019–1028, 2010.
- [18] Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, and R Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1522–1538. SIAM, 2012.
- [19] Anupam Gupta, Martin Pál, R Ravi, and Amitabh Sinha. Boosted sampling: approximation algorithms for stochastic optimization. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 417–426. ACM, 2004.
- [20] Thibaut Horel and Yaron Singer. Scalable methods for adaptively seeding a social network. *WWW*, 2015.
- [21] Nicole Immorlica, David Karger, Maria Minkoff, and Vahab S Mirrokni. On the costs and benefits of procrastination: approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 691–700. Society for Industrial and Applied Mathematics, 2004.
- [22] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD*, pages 137–146, 2003.
- [23] Jon Kleinberg, Yuval Rabani, and Éva Tardos. Allocating bandwidth for bursty connections. *SIAM Journal on Computing*, 30(1):191–217, 2000.
- [24] Silvio Lattanzi and Yaron Singer. The power of random neighbors in social networks. *WSDM*, 2015.
- [25] Lucien Le Cam. An approximation theorem for the poisson binomial distribution. *Pacific Journal of Mathematics*, 10(4):1181–1197, 1960.
- [26] Jure Leskovec, Lada A. Adamic, and Bernardo A. Huberman. The dynamics of viral marketing. In *ACM Conference on Electronic Commerce*, pages 228–237, 2006.
- [27] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne M. VanBriesen, and Natalie S. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.
- [28] Michael Mathioudakis, Francesco Bonchi, Carlos Castillo, Aristides Gionis, and Antti Ukkonen. Sparsification of influence networks. In *KDD*, 2011.
- [29] Vahab S. Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *ACM Conference on Electronic Commerce*, pages 70–77, 2008.
- [30] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.
- [31] Elchanan Mossel and Sébastien Roch. On the submodularity of influence in social networks. In *STOC*, pages 128–134, 2007.

- [32] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions ii. *Math. Programming Study* 8, pages 73–87, 1978.
- [33] R Ravi and Amitabh Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. In *Integer programming and combinatorial optimization*, pages 101–115. Springer, 2004.
- [34] Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD*, pages 61–70, 2002.
- [35] Aviad Rubinfeld, Lior Seeman, and Yaron Singer. Approximability of adaptive seeding under knapsack constraints. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, pages 797–814, 2015.
- [36] Lior Seeman and Yaron Singer. Adaptive seeding in social networks. In *Proceedings of the The 54th Annual Symposium on Foundations of Computer Science*, 2013.
- [37] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*, volume 9. Society for Industrial Mathematics, 2009.
- [38] David B Shmoys and Chaitanya Swamy. Stochastic optimization is (almost) as easy as deterministic optimization. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 228–237. IEEE, 2004.
- [39] David B Shmoys and Chaitanya Swamy. An approximation scheme for stochastic linear programming and its application to stochastic integer programs. *Journal of the ACM (JACM)*, 53(6):978–1012, 2006.
- [40] Aravind Srinivasan. Approximation algorithms for stochastic and risk-averse optimization. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1305–1313. Society for Industrial and Applied Mathematics, 2007.
- [41] Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 67–74. ACM, 2008.
- [42] Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013.
- [43] De-Nian Yang, Hui-Ju Hung, Wang-Chien Lee, and Wei Chen. Maximizing acceptance probability for active friending in online social networks. In *KDD'13*, 2013.

A Chernoff bound

We use the following bound in the paper (See for example [30, Theorem 4.4]).

Theorem A.1 (Chernoff bound). *Suppose X_1, \dots, X_n are independent random variables taking values in $[0, 1]$ and let X denote their sum and μ be the expected value of X . Then, for $0 < \delta < 1$ we have that*

$$\Pr[X > (1 + \delta)\mu] \leq \exp\left(-\frac{\delta^2\mu}{3}\right)$$