Stochastic Submodular Cover with Limited Adaptivity*

Arpit Agarwal† Sepehr Assadi‡ Sanjeev Khanna‡

Abstract

In the submodular cover problem, we are given a non-negative monotone submodular function \( f \) over a ground set \( E \) of items, and the goal is to choose a smallest subset \( S \subseteq E \) such that \( f(S) = Q \) where \( Q = f(E) \). In the stochastic version of the problem, we are given \( m \) stochastic items which are different random variables that independently realize to some item in \( E \), and the goal is to find a smallest set of stochastic items whose realization \( R \) satisfies \( f(R) = Q \). The problem captures as a special case the stochastic set cover problem and more generally, stochastic covering integer programs.

A fully adaptive algorithm for stochastic submodular cover chooses an item to realize and based on its realization, decides which item to realize next. A non-adaptive algorithm on the other hand needs to choose a permutation of items beforehand and realize them one by one in the order specified by this permutation until the function value reaches \( Q \). The cost of the algorithm in both case is the number (or costs) of items realized by the algorithm. It is not difficult to show that even for the coverage function there exist instances where the expected cost of a fully adaptive algorithm and a non-adaptive algorithm are separated by \( \Omega(Q) \). This strong separation, often referred to as the adaptivity gap, is in sharp contrast to the separations observed in the framework of stochastic packing problems where the performance gap for many natural problem is close to the poly-time approximability of the non-stochastic version of the problem. Motivated by this striking gap between the power of adaptive and non-adaptive algorithms, we consider the following question in this work: does one need full power of adaptivity to obtain a near-optimal solution to stochastic submodular cover? In particular, how does the performance guarantees change when an algorithm interpolates between these two extremes using a few rounds of adaptivity.

Towards this end, we define an \( r \)-round adaptive algorithm to be an algorithm that chooses a permutation of all available items in each round \( k \in [r] \), and a threshold \( \tau_k \), and realizes items in the order specified by the permutation until the function value is at least \( \tau_k \). The permutation for each round \( k \) is chosen adaptively based on the realization in the previous rounds, but the ordering inside each round remains fixed regardless of the realizations seen inside the round. Our main result is that for any integer \( r \), there exists a poly-time \( r \)-round adaptive algorithm for stochastic submodular cover whose expected cost is \( O(Q^{1/r}) \) times the expected cost of a fully adaptive algorithm. Prior to our work, such a result was not known even for the case of \( r = 1 \) and when \( f \) is the coverage function. On the other hand, we show that for any \( r \), there exist instances of the stochastic submodular cover problem where no \( r \)-round adaptive algorithm can achieve better than \( \Omega(Q^{1/r}) \) approximation to the expected cost of a fully adaptive algorithm. Our lower bound result holds even for coverage function and for algorithms with unbounded computational power. Thus our work shows that logarithmic rounds of adaptivity are necessary and sufficient to obtain near-optimal solutions to the stochastic submodular cover problem, and even few rounds of adaptivity are sufficient to sharply reduce the adaptivity gap.

1 Introduction

Submodular functions naturally arise in many applications domains including algorithmic game theory, machine learning, and social choice theory, and have been extensively studied in combinatorial optimization. Many computational problems can be modeled as the submodular cover problem where we are given a non-negative monotone submodular function \( f \) over a ground set \( E \), and the goal is to choose a smallest subset \( S \subseteq E \) such that \( f(S) = Q \) where \( Q = f(E) \). A well-studied special case is the set cover problem where the function \( f \) is the coverage function and the items correspond to subsets of an underlying universe. Even this special case is known...
to be NP-hard to approximate to a factor better than $\Omega(\log Q)$ [22, 25, 34, 35], and on the other hand, the classic paper of Wolsey [43] shows that the problem admits a poly-time $O(\log Q)$-approximation for any integer-valued monotone submodular function.

In this work we consider the stochastic version of the problem that naturally arises when there is uncertainty about items. For instance, in stochastic influence spread in networks, the set of nodes that can be influenced by any particular node in the network is a random variable whose value depends on the realized state of the influencing node (e.g., being successfully activated). In sensor placement problems, each sensor can fail partially or entirely with certain probability and the coverage of a sensor depends on whether the sensor failed or not. In data acquisition for machine learning (ML) tasks, each data point is apriori a random variable that can take different values, and one may wish to build a dataset representing a diverse set of values. For example, if one wants to build a ML model for identifying a new disease from gene patterns, one would start by building a database of gene patterns associated to that disease. In this case, each person’s gene pattern is a random variable that can realize to different values depending on his/her race, sex etc. For some further examples, we refer the reader to [33] (an application in databases) and [3] (an application in document retrieval).

In the stochastic submodular cover problem, we are given $m$ stochastic items which are different random variables that independently realize to an element of $E$, and the goal is to find a lowest cost set of stochastic items whose realization $R$ satisfies $f(R) = Q$. In network influence spread problems each item corresponds to a node in the network, and its realization corresponds to the set of nodes it can influence. In sensor placement problems an item corresponds to a sensor and its realization corresponds to the area that it covers upon being deployed. In the case of data acquisition, an item corresponds to a data point and its realization corresponds to the value it takes upon being queried. The problem captures as a special case the stochastic set cover problem and more generally, stochastic covering integer programs.

In stochastic optimization, a powerful computational resource is adaptivity. An adaptive algorithm for stochastic submodular cover chooses an item to realize and based on its realization, decides which item to realize next. A non-adaptive algorithm on the other hand needs to choose a permutation of items and realize them in the order specified by the permutation until the function value reaches $Q$. The cost of the algorithm in both cases is the number (or costs) of items realized by the algorithm. It is well-understood that in general, adaptive algorithms perform better than non-adaptive algorithms in terms of cost of coverage. However, in practical applications a non-adaptive algorithm is better from the point of view of practitioners as it eliminates the need of sequential decision making and instead requires them to make just one decision. This motivates the study of separation between the performance of adaptive and non-adaptive algorithms, known as the adaptivity gap. For many stochastic packing problems, the adaptivity gap is only a constant. For instance, the adaptivity gap for budgeted stochastic max coverage where you are given a constraint on the number of items that can be chosen and the goal is to maximize coverage, the adaptivity gap is bounded by $1-1/e$ [5]. In a sharp contrast, covering version of the problem admits an adaptivity gap of $\Omega(Q)$ [26].

Motivated by this striking separation between the power of adaptive and non-adaptive algorithms, we consider the following question in this work: does one need full power of adaptivity to obtain a near-optimal solution to stochastic submodular cover? In particular, how does the performance guarantees change when an algorithm interpolates between these two extremes using a few rounds of adaptivity.

Towards this end, we define an $r$-round adaptive algorithm to be an algorithm that chooses a permutation of all available items in each round $k \in [r]$, and a threshold $\tau_k$, and realizes items in the order specified by the permutation until the function value is at least $\tau_k$. A non-adaptive algorithm would then correspond to the case $r = 1$ (with $\tau_1 = Q$), and an adaptive algorithm would correspond to the case $r = m$ (with $\tau_k = 0$ for all $k \in [r]$). The permutation for each round $k$ is chosen adaptively based on the realization in the previous rounds, but the ordering inside each round remains fixed regardless of the realizations seen inside the round.

Our main result is that for any integer $r$, there exists a poly-time $r$-round adaptive algorithm for stochastic submodular cover whose expected cost is $\tilde{O}(Q^{1/r})$ times the expected cost of a fully adaptive algorithm, where the $\tilde{O}$ notation is hiding a logarithmic dependence on the number of items and the maximum cost of any item. Prior to our work, such a result was not known even for the case of $r = 1$ and when $f$ is the coverage function. Indeed achieving such a result was cast as an open problem by Goemans and Vondrak [20] who achieved an $O(n^2)$ bound (corresponding to $O(Q^2)$) on the adaptivity gap of
stochastic set cover. Furthermore, we show that for any \( r \), there exist instances of the stochastic submodular cover problem where no \( r \)-round adaptive algorithm can achieve better than \( \Omega(Q^{1/r}) \) approximation to the expected cost of a fully adaptive algorithm. Our lower bound result holds even for coverage functions and for algorithms with unbounded computational power. Thus our work shows that logarithmic rounds of adaptivity are necessary and sufficient to obtain near-optimal solutions to the stochastic submodular cover problem, and even few rounds of adaptivity are sufficient to sharply reduce the adaptivity gap.

### 1.1 Problem Statement

Let \( X := \{X_1, \ldots, X_m\} \) be a collection of \( m \) independent random variables each supported on the same ground set \( E \) and \( f \) be an integer-valued\(^1\) non-negative monotone submodular function \( f : 2^E \rightarrow \mathbb{N}_+ \). We will refer to random variables \( X_i \)'s as items and any set \( S \subseteq X \) as a set of items. For any \( i \in [m] \), we use \( x_i \in E \) to refer to a realization of item (random variable) \( X_i \) and define \( X := \{x_1, \ldots, x_m\} \) as the realization of \( X \). We slightly abuse notation\(^2\) and extend \( f \) to the ground set of items \( X \) such that for any set \( S \subseteq X \), \( f(S) := f(\cup_{x_i \in S} x_i) \): this definition means that for any realization \( S \) of \( f \), \( f(S) = f(\cup_{x_i \in S} x_i) \). Finally, there is an integer-valued cost \( c_i \in [C] \) associated with item \( X_i \in X \).

Let \( Q := f(E) \). For any set of items \( S \subseteq X \), we say that a realization \( S \) of \( f \) is feasible if \( f(S) = Q \). We will assume that any realization \( X \) of \( X \) is always feasible, i.e. \( f(X) = Q \).\(^3\) We will say that a realization \( X \) of \( X \) is covered by a realization \( S \subseteq X \) of \( S \) if \( f(S) \) is feasible. The goal in the stochastic submodular cover problem is to find a set of items \( S \subseteq X \) with the minimum cost which gets realized to a feasible set. In order to do so, if we include any item \( X_i \) to \( S \), we pay a cost \( c_i \), and once included, \( X_i \) would be realized to some \( x_i \in E \) and is fixed from now on. Once a decision made regarding inclusion of an item in \( S \), this item cannot be removed from \( S \).

For any set of items \( S \subseteq X \), we define \( \text{cost}(S) \) to be the total cost of all items in \( S \), i.e. \( \text{cost}(S) = \sum_{i \in [m]} c_i \cdot 1 [X_i \in S] \), where \( 1[\cdot] \) is an indicator function. For any algorithm \( \mathcal{A} \), we refer to the total cost of solution \( S \) returned by \( \mathcal{A} \) on an instantiation \( X \) of \( S \) as the cost of \( \mathcal{A} \) on \( X \) denoted by \( \text{cost}(\mathcal{A}(X)) \). We are interested in minimizing the expected cost of the algorithm \( \mathcal{A} \), i.e., \( \mathbb{E}_{X \sim \mathcal{X}} [\text{cost}(\mathcal{A}(X))] \).

---

**Example 1.1.** *(Stochastic Set Cover)* A canonical example of the stochastic submodular cover problem is the stochastic set cover problem. Let \( U \) be a universe of \( n \) “elements” (not to be mistaken with “items”) and \( X = \{X_1, \ldots, X_m\} \) be a collection of \( m \) random variables where each random variable \( X_i \) is supported on subsets of \( U \), i.e., realizes to some subset \( T_i \subseteq U \). We refer to each random variable \( X_i \) as a stochastic set. In the stochastic set cover problem, the goal is to pick a smallest (or minimum weight) collection \( S \) of items (or equivalently sets) in \( X \) such that the realized sets in this collection cover the universe \( U \).

We consider the following types of algorithms (sometimes referred to as policies in the literature) for the stochastic submodular cover problem:

- **Non-adaptive**: A non-adaptive algorithm simply picks a fixed ordering of items in \( X \) and insert the items one by one to \( S \) until the realization \( S \) of \( S \) become feasible.
- **Adaptive**: An adaptive algorithm on the other hand picks the next item to be included in \( S \) adaptively based on the realization of previously chosen items. In other words, the choice of each item to be included in \( S \) is now a function of the realization of items already in \( S \).
- **\( r \)-round adaptive**: We define \( r \)-round adaptive algorithms as an “interpolation” between the above two extremes. For any integer \( r \geq 1 \), an \( r \)-round adaptive algorithm chooses the items to be included in \( S \) in \( r \) rounds of adaptivity: In each round \( i \in [r] \), the algorithm chooses a threshold \( \tau_i \in \mathbb{N}_+ \) and an ordering over items, and then inserts the items one by one according to this ordering to \( S \) until for the realized set \( S \), \( f(S) \geq \tau_i \). Once this round is finished, the algorithm decides on an ordering over the remaining items adaptively, based on the realization of all items chosen so far.

In above definitions, a non-adaptive algorithm corresponds to case of \( r = 1 \) round adaptive algorithm (with \( \tau_1 = Q \)) and a (fully) adaptive algorithm corresponds to the case of \( r = m \) (here \( \tau_i \) is irrelevant and can be thought as being zero).

**Adaptivity gap.** We use \( \text{OPT} \) to refer to the optimal adaptive algorithm for the stochastic sub-
modular cover problem, i.e., an adaptive algorithm with minimum expected cost. We use the expected cost of \( \text{OPT} \) as the main benchmark against which we compare the cost of other algorithms. In particular, we define *adaptivity gap* as the ratio between the expected cost of the best non-adaptive algorithm for the submodular cover problem and the expected cost of \( \text{OPT} \). Similarly, for any integer \( r \), we define the \( r \)-round adaptivity gap for \( r \)-rounds adaptive algorithms in analogy with above definition.

**Remark 1.1.** The notion of “best” non-adaptive or \( r \)-round adaptive algorithm defined above allow unbounded computational power to the algorithm. Hence, the only limiting factor of the algorithm is the information-theoretic barrier caused by the uncertainty about the underlying realization (due to the limit of adaptivity in decision making).

### 1.2 Our Contributions

In this paper, we establish tight bounds (up to logarithmic factor) on the \( r \)-round adaptivity gap of the stochastic submodular cover problem for any integer \( r \geq 1 \). Our main result is an \( r \)-round adaptive algorithm (for any integer \( r \geq 1 \)) for the stochastic submodular cover problem.

**Result 1. (Main Result)** For any integer \( r \geq 1 \) and any monotone submodular function \( f \), there exists an \( r \)-round adaptive algorithm for the stochastic submodular cover problem for function \( f \) and set of items \( X \) with cost of each item bounded by \( C \) that incurs expected cost \( O(Q^{1/r} \cdot \log Q \cdot \log(mC)) \) times the expected cost of the optimal adaptive algorithm.

A direct corollary of Result 1 is that the \( r \)-round adaptivity gap of the submodular cover problem is \( O(Q^{1/r}) \). This in particular implies using only \( O \left( \frac{\log Q}{\log \log Q} \right) \) rounds of adaptivity, one can reduce the cost of the algorithm to within poly-logarithmic factor of the optimal fully adaptive algorithm. In other words, one can “harness” the (essentially) full power of adaptivity, in only logarithmic number of rounds.

Various stochastic covering problems can be cast as submodular cover problem, including the stochastic set cover problem and the stochastic covering integer programs studied previously in the literature [21, 26, 27]. As such, Result 1 directly extends to these problems as well. In particular, as a (very) special case of Result 1, we obtain that the adaptivity gap of the stochastic set cover problem is \( O(n) \) (here \( n \) is the size of the universe), improving upon the \( O(n^2) \) bound of Goemans and Vondrak [26] and settling an open question in their work regarding the adaptivity gap of this problem (an \( \Omega(n) \) lower bound was already shown in [26]).

We further prove that the \( r \)-round adaptivity gaps in Result 1 are almost tight for any \( r \geq 1 \).

**Result 2.** For any integer \( r \geq 1 \), there exists a monotone submodular function \( f : 2^X \rightarrow \mathbb{N}_+ \), in particular a coverage function, with \( Q := f(E) \) such that the expected cost of any \( r \)-round adaptive algorithm for the submodular cover problem for function \( f \), i.e., the stochastic set cover problem, is \( \Omega \left( \frac{1}{r} \cdot Q^{1/r} \right) \) times the expected cost of the optimal adaptive algorithm.

Result 2 implies that the \( r \)-round adaptivity gap of the submodular cover problem is \( \Omega \left( \frac{1}{r} \cdot Q^{1/r} \right) \), i.e., within poly-logarithmic factor of the upper bound in Result 1. An immediate corollary of this result is that \( \Omega \left( \frac{\log Q}{\log \log Q} \right) \) rounds of adaptivity are necessary for reducing the cost of the algorithms to within logarithmic factors of the optimal adaptive algorithm. We further point out that interestingly, the optimal adaptive algorithm in instances created in Result 2 only requires \( r + 1 \) rounds; as such, Result 2 in fact is proving a lower bound on the gap between the cost of \( r \)-round and \((r + 1)\)-round adaptive algorithms.

We remark that our algorithm in Result 1 is polynomial time (for polynomially-bounded item costs), while the lower bound in Result 2 holds again algorithms with unbounded computational power (see Remark 1.1).

### 1.3 Related Work

The problem of submodular cover was perhaps first studied by [43], who showed that a greedy algorithm achieves an approximation ratio of \( \log(Q) \). Subsequent to this there has been a lot of work on this problem in various settings [9, 10, 21, 26, 27, 28, 31, 32]. To our knowledge, the question of adaptivity in stochastic covering problems was first studied in [26] for the special case of stochastic set cover and covering integer programs. It was shown that the adaptivity gap of this problem is \( \Omega(n) \), where \( n \) is the size of the universe to be covered. A non-adaptive algorithm for this problem with an adaptivity gap of \( O(n^2) \) was also presented.

Subsequently there has been a lot of work on stochastic set cover and the more general stochastic submodular cover problem in the fully adaptive setting. A special case of stochastic set cover was studied by [33] in the adaptive setting, and an adaptive greedy algorithm was studied. In [27] the notion

---

*The paper originally claimed an approximation ratio of \( \log(n) \) for this algorithm, however, the claim was later retracted by the authors due to an error in the analysis [37].*
of “adaptive submodularity” was defined for adaptive optimization, which demands that given any partial realization of items, the marginal function with respect to this realization remains monotone submodular. This paper also presented an adaptive greedy algorithm for the problem of stochastic submodular cover, and stochastic submodular maximization subject to cardinality constraints.\footnote{It was originally claimed that this algorithm achieves an approximation ratio of } log(Q) where Q is the desired coverage, however, the claim was later retracted due to an error in the analysis \cite{27}. The authors have claimed an approximation ratio of log^2(Q) since then.

studies under various types of constraints, including knapsack constraints. An approximation ratio of \( \tau \) for this problem under knapsack constraint was given, where \( \tau \) is the smallest probability of any element in the ground set being realized by any item. The question of adaptivity has also been studied for other stochastic problems such as stochastic packing, knapsack, matching etc. (see, e.g., \cite{7,8,13,19,20,44} and references therein).

There has also been a lot of work under the framework of 2-stage or multi-stage stochastic programming \cite{16,39–41}. In this framework, one has to make sequential decisions in a stochastic environment, and there is a parameter \( \lambda \) such that the cost of making the same decision increases by a factor \( \lambda \) after each stage. The stochastic program in each stage is defined in terms of the expected cost in the later stages. The central question in these problems is—when can we find good solutions to this complex stochastic program, either by directly solving it or by finding approximations to it? This largely depends on the complexity of the stochastic program at hand. For example, if the distribution of the environment is explicitly given, then one might be able to solve the stochastic program exactly by using integer programming, and this question becomes largely computational in nature. This is fundamentally different than the information theoretic question we consider in this paper.

Aside from the stochastic setting, algorithms with limited adaptivity have been studied across a wide spectrum of areas in computer science including in sorting and selection (e.g., \cite{14,17,42}), multi-armed bandits (e.g., \cite{1,38}), algorithms design (e.g., \cite{11,12,23,24}), among others; we refer the interested reader to the wide spectrum of areas in computer science including the references therein.

Remark: Our study of \( r \)-round adaptive algorithm for submodular cover is reminiscent of a recent work of Chakrabarti and Wirth \cite{15} on multi-pass streaming algorithms for the set cover problem. They showed that allowing additional passes over the input in the streaming setting (similar-in-spirit to more rounds of adaptivity) can significantly improve the performance of the algorithms and established tight pass-approximation tradeoffs that are similar (but not identical) to \( r \)-round adaptivity gap bounds in Results 1 and Results 2. In terms of techniques, our upper bound result—our main contribution—is almost entirely disjoint from the techniques in \cite{15} (and works for the more general problem of submodular cover, whereas the results in \cite{15} are specific to set cover). While our lower bound uses similar instances as \cite{15} but is based on an entirely different analysis.
1.4 Organization In Section 2 we present some preliminaries for our problem. In Section 3 we present a technical overview of our main results. In Section 4 we present a non-adaptive selection algorithm that will be used to prove our upper bound result in Section 5. We present the lower bound result in Section 6.

2 Preliminaries

Notation. Throughout this paper we will use symbols $S, T,$ and $R$ to denote subsets of the ground set $E$, and use symbols $A$ and $B$ to denote subsets of $[m]$, i.e., indices of items. We will also use symbols $S, T$ and $R$ to denote subsets of $X$ which realize to subsets $S, T$ and $R$ of the ground set $E$.

Submodular Functions: Let $E$ be a finite ground set and $\mathbb{N}_+$ be the set of non-negative integers. For any set function $f : 2^E \to \mathbb{N}_+$, and any set $S \subseteq E$, we define the 
\textit{marginal contribution} to $f$ as the set function $f_S : 2^E \to \mathbb{N}_+$ such that for all $T \subseteq E$, $f_S(T) = f(S \cup T) - f(S)$.

When clear from the context, we abuse the notation and for $e \in E$, we use $f(e)$ and $f_S(e)$ instead of $f(\{e\})$ and $f_S(\{e\})$.

A set function $f : 2^E \to \mathbb{N}_+$ is submodular iff for all $S \subseteq T \subseteq E$ and $e \in E$: $f_S(e) \geq f_T(e)$. Function $f$ is additionally monotone iff $f(S) \leq f(T)$. Throughout the paper, we solely focus on monotone submodular functions unless stated explicitly otherwise.

We use the following two well-known facts about submodular functions throughout the paper.

\textbf{Fact 2.1.} Let $f(\cdot)$ be a monotone submodular function, then:
$$\forall S, T \subseteq E \quad f(S) \leq f(T) + \sum_{e \in S \setminus T} f_T(e).$$

\textbf{Fact 2.2.} Let $f(\cdot)$ be a monotone submodular function, then for any $S \subseteq E$, $f_S(\cdot)$ is also monotone submodular.

3 Technical Overview

We give here an overview of the techniques used in our upper and lower bound results.

\textbf{Upper bound on $r$-round adaptivity gap.} In this discussion we focus mainly on our non-adaptive ($r = 1$) algorithm, which already deviates significantly from the previous work of Goemans and Vondrak [26]. A non-adaptive algorithm simply picks a permutation of items and realize them one by one in a set $S$ until $f(S) = Q$. Hence, the “only” task in designing a non-adaptive algorithm is to find a “good” ordering of items, that is, an ordering such that its prefix that covers $Q$ has a low expected cost.

Consider the following algorithmic task: In the setting of stochastic submodular cover problem, suppose we are given a (ordered) set $S$ of stochastic items. Can we pick a low-cost (ordered) set $T$ of stochastic items non-adaptively (without looking at a realization of $S$ or $T$) so that the coverage of $S \cup T$ is sufficiently larger than $S$, i.e., $E[f_S(T)]$ is large? Assuming we can do this, we can use this primitive to find sets with large coverage non-adaptively and iteratively, by starting from the empty-set and using this primitive to increase the coverage further repeatedly.

Recall that in the non-stochastic setting, the greedy algorithm is precisely solving this problem, i.e., finds a set $T$ such that $f_S(T) \geq f_S(Q - f_S(S))$, where with a slight abuse of notation, $\text{OPT}$ here denotes the optimal non-stochastic cover of $f(E)$. This suggests that one can always find a “low” cost set $T$ with a large marginal contribution to $S$. For the stochastic problem, however, it is not at all clear whether there always exists a “low” cost (compared to adaptive $\text{OPT}$) $T$ whose expected marginal contribution to $S$ is large. This is because there are many different realizations possible for $S$, and each realization $S$, in principle may require a dedicated set of items $T(S)$ to achieve a large value $E[f_S(T(S)) | S]$. As such, while adaptive $\text{OPT}$ can first discover the realization $S$ of $S$ and based on that choose $T(S)$ to increase the expected coverage, a non-adaptive algorithm needs to instead pick $\cup_{S \in S} T(S)$, which can have a much larger cost (but the same marginal contribution). This suggests that cost of non-adaptive algorithm can potentially grow with the size of all possible realizations of $S$. We point out that this task remains challenging even if all remaining inputs other than $S$ are non-stochastic, i.e., always realize to a particular item.

Nevertheless, it turns out that no matter the size of the set of all realizations of $S$, one can always find a set of stochastic items $T$ such that $E[f_S(T)] = \Omega(1) \cdot E[Q - f(S)]$ while $\text{cost}(T) = O(Q \cdot E[\text{cost}(\text{OPT})])$, i.e., achieve a marginal contribution proportional to $E[Q - f(S)]$ while paying cost which is $O(Q)$ times larger than $\text{OPT}$ (here $\text{OPT}$ corresponds to an optimal adaptive algorithm corresponding the residual problem of covering $Q - f(S)$). Compared to the non-stochastic setting, this cost is $O(Q)$ times larger than the analogous cost in the non-stochastic setting (see Example 4.1). This part is one of
the main technical ingredients of our paper (see Theorem 4.1). We briefly describe the main ideas behind this proof.

The idea behind our algorithm is to sample several realizations $S_1, \ldots, S_\Psi$ from $S$ and pick a low cost dedicated set $T_i$ for each $S_i$ such that $\mathbb{E}[f_{S_i}(T_i)]$ is large (here, the randomness is only on realizations of $T_i$). This step is quite similar to solving the non-adaptive submodular maximization problem with knapsack constraint for which we design a new algorithm based on an adaptation of Wolsey's LP [43] (see Theorem 4.2 and discussion before that for more details and comparison with existing results). This allows us to bound the cost of each set $T_i$ by $O(\mathbb{E}[\text{cost(OPT)}])$. The final (ordered) set returned by this algorithm is then $T := T_1 \cup \ldots \cup T_\Psi$. The ordering within items of $T$ does not matter.

The main step of this argument is however to bound the value of $\Psi$, i.e., the number of samples, by $O(Q)$. This step is done by bounding the total contribution of sets $T_1, \ldots, T_\Psi$ on their own, i.e., $\mathbb{E}[f(T_1 \cup \ldots \cup T_\Psi)]$ independent of the set $S$. The intuition is that if we choose, say $T_1$, with respect to some realization $S$ of $S$, but $T_1$ does not have a marginal contribution to most realizations $S'$ of $S$, then this means that by picking another set $T_2$, the set $T_1 \cup T_2$ needs to have a coverage larger than both $T_1$ and $T_2$. As a result, if we repeat this process sufficiently many times, we should eventually be able to increase $\mathbb{E}[f_S(T)]$, simply because otherwise $f(T) > Q$, a contradiction.

We now use this primitive to design our non-adaptive algorithm as follows: we keep adding set of items to the ordering using the primitive above in iterative phases. In each phase $p$, we run the above primitive multiple times to find a set $S_p$ with $\mathbb{E}[Q - f(S_p)] | \mathcal{E}_{p-1}] = o(1)$, where $\mathcal{E}_{p-1}$ is the event that the realization of items picked in previous phases of the algorithm did not cover $Q$ entirely. We further bound the cost of the set $S_p$ with the expected cost of OPT conditioned on the event $\mathcal{E}_{p-1}$, i.e., $\mathbb{E}[\text{cost(OPT)}] | \mathcal{E}_{p-1}]$. Notice that this quantity can potentially be much larger than the expected cost of OPT. However, since the probability that in the permutation returned by the non-adaptive algorithm, we ever need to realize the sets in $S_p$ is bounded by $\mathbb{P}(\mathcal{E}_{p-1})$, we can pay for the cost of these sets in expectation. By repeating these phases, we can reduce the probability of not covering $Q$ exponentially fast and finalize the proof.

We then extend this algorithm to an $r$-round adaptive algorithm for any $r \geq 1$. For simplicity, let us only mention the extension to 2 rounds (extending to $r$ is then straightforward). We spend the first round to find a (ordered) set $S$ with $f(S) \geq Q - \sqrt{Q}$ with high probability for any realizations $S$ of $S$. We extend our main primitive above to ensure that if $\mathbb{E}[Q - f(S)] \geq \sqrt{Q}$, then we can find a set $T$ with $\mathbb{E}[f_S(T)] = \Omega(1) \cdot \mathbb{E}[Q - f(S)]$ and $\text{cost}(T) = \tilde{O}(\sqrt{Q} \cdot \mathbb{E}[\text{cost(OPT)}])$ (as opposed to $O(Q)$ in the original statement). This is achieved by the fact that when the deficit $Q - f(S)$ is sufficiently large then the rate of coverage per cost is higher, as opposed to when the deficit $Q - f(S)$ is very small. Precisely, we exploit the fact that the gap of $Q - f(S)$ is sufficiently large to reach the contradiction in the original argument with only $O(\sqrt{Q})$ sets $T_1, T_2, \ldots$. We then run the previous algorithm using this primitive by setting the threshold $\tau_1 = Q - \sqrt{Q}$. In the next round, we simply run our previous algorithm on the function $f_S(\cdot)$ where $S$ is the realization in the first round. As $f_S(\cdot)$ has maximum value at most $O(\sqrt{Q})$, by the previous argument we only need to pay $\tilde{O}(\sqrt{Q})$ times expected cost of OPT, hence our total cost is $\tilde{O}(\sqrt{Q} \cdot \mathbb{E}[\text{cost(OPT)}])$. Extending this approach to $r$-round algorithms is now straightforward using similar ideas as the thresholding greedy algorithm for set cover (see, e.g. [18]).

**Lower bound on adaptivity gap.** We prove our lower bound for the stochastic set cover problem, a special case of stochastic submodular cover problem (see Example 1.1). Let us first sketch our lower bound for two round algorithms. Let $S := \{U_1, \ldots, U_k\}$ be a collection of $k = \text{poly}(n)$ sets to be determined later (recall that $n$ is the size of the universe $U$ we aim to cover). Consider the following instance of stochastic set cover: there exists a single stochastic set $T$ which realizes to one set $S_p$ chosen uniformly at random from sets $U_1, \ldots, U_k$, i.e., complements of the sets in $S$. We further have $k$ additional stochastic sets where $T_i$ realizes to $U_i \setminus \{e\}$ for $e$ chosen uniformly at random from $U_i$. Finally, for any element $e \in U$, we have a set $T_e$ with only one realization which is the singleton set $\{e\}$ (i.e., $T_e$ always covers $e$).

Consider first the following adaptive strategy: pick $T$ in the first round and see its realization, say, $U_i$. Pick $T_i$ in the second round and see its realization, say $U_i \setminus \{e\}$. Pick $T_e$ in the third round. This collection of sets is $(U \setminus U_i) \cup (U_i \setminus \{e\}) \cup \{(e)\} = U$, hence it is a feasible cover. As such, in only 3 rounds of adaptivity, we were able to find a solution with cost only 3.

A two-round algorithm is however one round short of following the above strategy. One approach to remedy this would be to try to make a “shortcut” by picking more than one sets in each round of this
process, e.g., pick the set \( T_i \) also in the first round. However, it is easy to see that as long as we do not pick \( \Omega(k) \) sets in the first round, or \( \Omega(|U_i|) \) sets in the second round, we have a small chance of making such a shortcut. We are not done yet as it is possible that the algorithm covers the universe using entirely different sets (i.e., do not follow this strategy). To ensure that cannot help either, we need the sets in \( U_1, \ldots, U_k \) to have “minimal” intersection; this in turns limits the size of each set \( U_i \) and hence the eventual lower bound we obtain using this argument.

We design a family of instances that allows us to extend the above argument to \( r \)-round adaptive algorithms. We construct these instances using the *edifice* set-system of Chakrabarti and Wirth [15] that poses a “near laminar” property, i.e., any two sets are either subset-superset of one another or have “minimal” intersection. We remark that this set-system was originally introduced by [15] for designing multi-pass streaming lower bounds for the set cover problem. While the instances we create in this work are similar to the instances of [15], the proof of our lower bound is entirely different (lower bound of [15] is proven using a reduction in communication complexity, while our proof is a direct argument in the stochastic setting).

4 The Non-Adaptive Selection Algorithm

We introduce a key primitive of our approach in this section for solving the following task: Suppose we have already chosen a subset \( S \subseteq X \) of items but we are not aware of the realization of these items; our goal is to non-adaptively add another set \( T \) to \( S \) to increase its expected coverage. Formally, given any monotone submodular function \( g : 2^E \rightarrow \mathbb{N}_+ \), let \( Q_g := g(E) \) be the required coverage on \( g \). Also, for any realization \( S \) of \( g \), we use \( \Delta(S) := Q_g - g(S) \) to refer to the *deficit* in covering \( Q_g \), and denote by \( \Delta := \mathbb{E} [\Delta(S)] \) the expected deficit of set \( S \). Our goal is now to add (still non-adaptively) a “low-cost” (compared to adaptive OPT) set \( T \) to \( S \) to decrease the expected deficit. It is easy to see that such a primitive would be helpful for finding sets with “large” coverage non-adaptively and iteratively, by starting from the empty-set and use this primitive to reduce the deficit further by picking another set and then repeat the process starting from this set.

Let us start by giving an example which shows some of the difficulty of this task.

**Example 4.1.** Consider an instance of stochastic set cover: there exists a single set, say \( X_1 \) which realizes to \( U \setminus \{e^*\} \) for an element \( e^* \) chosen uniformly at random from \( U \) and \( n \) singleton sets \( X_2, \ldots, X_{n+1} \), each covering a unique element in \( U \). If we have already chosen \( X_1 \), and want to chose more sets in order to decrease the expected deficit, then it is easy to see that even though the cost of OPT is only 2, no collection of \( o(n) \) sets can decrease the expected deficit by one. This should be contrasted with the non-stochastic setting in which there always exists a single set that reduces a deficit of \( \Delta \) by \( \Delta/\text{cost(OPT)} \).

We are now state our main result in this section.

**Theorem 4.1.** Let \( X \) be a collection of items, and let \( g \) be any monotone submodular function such that \( g(X) = Q_g \) for every realization \( X \) of \( X \). Let \( S \subseteq X \) be any subset of items and define \( \Delta := \mathbb{E} [Q_g - g(S)] \). Given any parameter \( \alpha \geq Q_g/\Delta \), there is a randomized non-adaptive algorithm that outputs a set \( T \subseteq X \setminus S \) such that cost of \( T \) is \( O(\alpha) \cdot \mathbb{E} [\text{cost(OPT)}] \) in expectation over the randomness of the algorithm and \( \mathbb{E} [Q_g - g(S \cup T)] \leq \frac{5}{6} \Delta \) over the randomness of the algorithm and realizations of \( S \) and \( T \). Here OPT is an optimal fully-adaptive algorithm for the stochastic submodular cover problem with the function \( g^6 \).

The goal in Theorem 4.1, is to select a set of items that can decrease the deficit of a *typical* realization \( S \) of \( S \) (i.e., the expected deficit). In order to do so, we first design a non-adaptive algorithm that finds a low-cost set that can decrease the deficit of a *particular* realization \( S \) of \( S \). This step is closely related to solving a stochastic submodular maximization problem subject to a knapsack constraint. Indeed, when costs of all the items are the same, i.e., when we want to minimize the number of items in the solution, one can use the algorithm of [5] (with some small modification) for stochastic submodular maximization subject to cardinality constraint for this purpose. Also, when the random variables \( X_i \)’s have binary realizations, i.e. take only two possible values, then one can use the algorithm of [29] for this purpose. However, we are not aware of a solution for the knapsack constraint of the problem in its general form with the bounds required in our algorithms, and hence we present an algorithm for this task as well.

The main step of our argument is however on how to...
use this algorithm to prove Theorem 4.1, i.e., move from per-realization guarantee, to the expectation guarantee.

4.1 A Non-Adaptive Algorithm for Increasing Expected Coverage We start by presenting a non-adaptive algorithm that picks a low-cost (compared to the expected cost of OPT) set of items deterministically, while achieving a constant factor of coverage of OPT. For any set $A \subseteq [m]$, i.e., the set of indices of stochastic items, and any realization $X$ of $X$, we define $X_A := \{x_i \mid i \in A\}$, i.e., the realization of all items corresponding to indices in $A$.

**Theorem 4.2.** There exists a non-adaptive algorithm that takes as input a set of items $X$, a monotone submodular function $f$, and a parameter $\bar{Q}$ such that $f(X) = \bar{Q}$ for any realization $X$ of $X$, and outputs a set $A \subseteq [m]$ such that (i) $\text{cost}(X_A) \leq 3 \cdot \mathbb{E}[\text{cost}(\text{OPT})]$ and (ii) $\mathbb{E}_{X \sim X}[f(X_A)] \geq \bar{Q}/3$. Here, OPT is the optimum adaptive algorithm for submodular cover on $X$ with function $f$ and parameter $Q = \bar{Q}$.

As argued before, Theorem 4.2 can be interpreted as an algorithm for submodular maximization subject to knapsack constraint.

To prove Theorem 4.2, we design a simple greedy algorithm (similar to the greedy algorithm for submodular maximization) and analyze it using a linear programming (LP) relaxation in the spirit of Wolsey’s LP [43] defined in the following section.

**Extension of Wolsey’s LP for Stochastic Submodular Cover** Let us define the function $F : 2^{[m]} \to \mathbb{N}_+$ as follows: for any $A \subseteq [m],\quad (4.1) \quad F(A) := \mathbb{E}_{X_A \sim X}[f(X_A)].$

As we assume in the lemma statement that $\bar{Q} := \mathbb{E}_{X \sim X}[f(X)]$, we have $F([m]) = \bar{Q}$ as well. For any $B \subseteq [m]$, we further define the marginal contribution function $F_B : 2^{[m]} \to \mathbb{N}_+$ where $F_B(A) := F(A \cup B) - F(B)$ for all $A \subseteq [m] \setminus B$. The following proposition is straightforward.

**Proposition 4.1.** Function $F$ is a monotone submodular function.

**Proof.** $F$ is a convex combination of submodular functions, one for each realization of $X$. \hfill \Box

We will use a linear programming (LP) relaxation in the spirit of Wolsey’s LP [43] for the submodular cover problem (when applied to the function $F$). Consider the following linear programming relaxation:

\[
P = \min_{\mathclap{y \in [0,1]^m}} \sum_{i=1}^m c_i \cdot y_i \quad (4.2)
\]

s.t. $\sum_{i \in [m] \setminus A} F_A(i) \cdot y_i \geq \bar{Q} - 2F(A), \forall A \subseteq [m]$

The difference between LP (4.2) and Wolsey’s LP is in RHS of the constraint which is $\bar{Q} - F(A)$ in case Wolsey’s LP. In the non-stochastic setting, one can prove that Wolsey’s LP lower bounds the value of optimum submodular cover for function $F$. To extend this result to the stochastic case (for the function $f$) however, it suffices to modify the constraint as in LP (4.2), as we prove in the following lemma.

**Lemma 4.1.** The cost of an optimal adaptive algorithm OPT for submodular cover on function $f$ is lower bounded by the optimal cost $P$ of LP (4.2), i.e., $P \leq \mathbb{E}[\text{cost}(\text{OPT})]$.

**Proof.** For a realization $X$ of $X$ and any $i \in [m]$, define an indicator random variable $w_i(X)$ that takes value 1 iff OPT chooses $x_i$ on the realization $X$, i.e., $w_i(X) = 1[x_i \in \text{OPT}(X)]$.

Let $w_i$ be the probability that OPT chooses $x_i$, i.e.,

$$w_i = \mathbb{P}_{X \sim X} (w_i(X) = 1) = \mathbb{E}_{X}(w_i(X)).$$

We have that,

$$\mathbb{E}[\text{cost}(\text{OPT})] = \mathbb{E}_{X} \left[ \sum_{i=1}^m 1[x_i \in \text{OPT}(X)] \cdot c_i \right] = \sum_{i=1}^m w_i \cdot c_i.$$

In the following, we prove that $w := (w_1, \ldots, w_m)$ is a feasible solution to LP (4.2), which by above equation would immediately imply that $P \leq \mathbb{E}[\text{cost}(\text{OPT})]$.

Clearly $w \in [0,1]^m$, so it suffices to prove that the constraint holds for any set $A \subseteq [m]$. The main step in doing so is the following claim.

**Claim 4.2.** For any set $A \subseteq [m]$, and any two realizations $X$ and $X'$ of $X$:

$$f(X_A) + f(X'_A) + \sum_{i \in [m] \setminus A} f_{X_A}(x_i) \cdot w_i(X) \geq \bar{Q}.$$

**Proof.** Recall that we assume $f(X) = \bar{Q}$ always, and hence $f(\text{OPT}(X)) = \bar{Q}$ as well. Moreover, for any
Claim 4.2, which finalizes the proof.

We have,
\[ f(X_A) + f(X'_A) + \sum_{i \in [m] \setminus A} f_{X_A}(x_i) \cdot w_i(X) = f(X_A) + f(X'_A) + \sum_{x_i \in C} f_{X'_A}(x_i) \]
(by submodularity)
\[ \geq f(X_A) + f(X'_A \cup C) \]
(by monotonicity)
\[ \geq f(X_B) + f(X_C) \]
(by submodularity and since \(X_B \cup X_C = \text{OPT}(X)\))
\[ = f(X_B \cup X_C) = \tilde{Q}, \]
which finalizes the proof.  \( \square \) Claim 4.2

Fix any set \( A \subseteq [m] \). We first take an expectation over all realizations of \( X \) in LHS of Claim 4.2:
\[ \tilde{Q} \leq \mathbb{E}_X \left[ f(X_A) + f(X'_A) + \sum_{i \in [m] \setminus A} f_{X_A}(x_i) \cdot w_i(X) \right] \]
\[ = \mathbb{E}_X \left[ f(X_A) + f(X'_A) + \sum_{i \in [m] \setminus A} \mathbb{E}_X \left[ f_{X_A}(x_i) \right] \cdot w_i(X) \right] \]
\[ = \mathbb{E}_X \left[ f(X_A) + f(X'_A) \right] + \sum_{i \in [m] \setminus A} \mathbb{E}_X \left[ f_{X_A}(x_i) \right] \cdot \mathbb{E}_X \left[ w_i(X) \right], \]
as random variables \( f_{X_A}(X_i) \) and \( w_i(X) \) are independent since the choice of \( X_i \) by \( \text{OPT} \) is independent of what \( X_i \) realizes to. We further point out that \( \mathbb{E}_X \left[ f(X_A) \right] \) in the RHS of last equation above is equal to \( F(A) \) by definition in Eq (4.1) and \( \mathbb{E}_X \left[ w_i(X) \right] = w_i \).

We further take an expectation over all realizations of \( X' \) in the RHS above:
\[ \tilde{Q} \leq \mathbb{E}_{X'} \left[ F(A) + f(X'_A) + \sum_{i \in [m] \setminus A} \mathbb{E}_X \left[ f_{X_A}(x_i) \right] \cdot w_i \right] \]
\[ = F(A) + f(X'_A) + \sum_{i \in [m] \setminus A} \mathbb{E}_X \left[ f_{X_A}(x_i) \right] \cdot w_i \]
\[ = 2 \cdot F(A) + \sum_{i \in [m] \setminus A} F_A(i) \cdot w_i, \]
as \( F_A(i) = \mathbb{E}_{X'} \mathbb{E}_X \left[ f(X'_A \cup X_i) - f(X'_A) \right] \). Rewriting the above equation, we obtain that the constraint associated with set \( A \) is satisfied by \( w \). This concludes the proof that \( w \) is a feasible solution.  \( \square \) Lemma 4.1

The Non-Adaptive-Greedy Algorithm

We now design an algorithm, namely NON-ADAPT-GREEDY, based on “the greedy algorithm” (for submodular optimization) applied to the function \( F \) in the last section and then use LP (4.2) to analyze it. We emphasize that the use of the LP is only in the analysis and not in the algorithm.

NON-ADAPT-GREEDY(\( X, f, \tilde{Q} \)). Given a monotone submodular function \( f \), the set of stochastic items \( X \), and a parameter \( \tilde{Q} = f(X) \) for all \( X \), outputs a set \( A \) of (indices of) stochastic items.

1. Initialize: Set \( A \leftarrow \emptyset \) and \( F \) be the function associated to \( f \) in Eq (4.1).
2. While \( F(A) < \tilde{Q}/3 \) do:
   (a) Let \( j^* \leftarrow \arg \max_{j \in [m]} F_A(j)/c_j \).
   (b) Update \( A \leftarrow A \cup \{j^*\} \).
3. Output: \( A \).

It is clear that the set \( A \) output by NON-ADAPT-GREEDY achieves \( \mathbb{E}_{X_A} \left[ f(X_A) \right] = F(A) \geq \tilde{Q}/3 \) (as \( F([m]) = \tilde{Q} \), the termination condition would always be satisfied eventually). We will now bound the cost paid by the greedy algorithm in terms of the optimal value \( P \) of LP (4.2).

**Lemma 4.3.** \( \text{cost}(X_A) \leq 3P \).

To prove Lemma 4.3 we need some definition. Let the sequence of items picked by the greedy algorithm be \( j_1, j_2, j_3, \ldots \), where \( j_i \) is the index of the item picked in iteration \( i \). Moreover, for any \( i \), define \( A_{<i} := \{ j_1, \ldots, j_{i-1} \} \), i.e., the set of items chosen before iteration \( i \). We first prove the following bound on the ratio of coverage rate to costs in each iteration.

**Lemma 4.4.** In each iteration \( i \) of the non-adaptive greedy algorithm we have,
\[ \frac{F_{A_{<i}}(j_i)}{c_{j_i}} \geq \frac{\tilde{Q} - 2F(A_{<i})}{P}, \]
where \( P \) is the optimal value of LP (4.2).

**Proof.** Fix any iteration \( i \). Recall that in each iteration, we pick the item \( j_i \in \arg \max_{j \in [m]} F_{A_{<i}}(j)/c_j \).
Suppose towards a contradiction that in some iteration $i$:

\begin{equation}
\forall j \in [m] \quad \frac{F_{A_{<i}}(j)}{c_j} < \frac{\bar{Q} - 2F(A_{<i})}{P}.
\end{equation}

Let $y^*$ be an optimal solution to LP (4.2), then by the constraint of the LP for set $A_{<i}$ we have

\[
\bar{Q} - 2F(A_{<i}) \leq \sum_{j \in [m] \setminus A_{<i}} F_{A_{<i}}(j) \cdot y^*_j
\leq \sum_{j \in [m] \setminus A_{<i}} y^*_j \cdot c_j \cdot \frac{\bar{Q} - 2F(A_{<i})}{P}
\leq \frac{\bar{Q} - 2F(A_{<i})}{P} \sum_{j \in [m]} y^*_j c_j = \bar{Q} - 2F(A_{<i}),
\]

where the last equality is because by definition $\sum_{j \in [m]} y^*_j c_j = P$. By above equation, $\bar{Q} - 2F(A_{<i}) < \bar{Q} - 2F(A_{<i})$, a contradiction. \hfill \Box 

**Lemma 4.4**

**Proof.** [Proof of Lemma 4.3] Fix any iteration $i$ in the algorithm where $F(A_{<i}) \leq \bar{Q}/3$. By Lemma 4.4,

\begin{equation}
F_{A_{<i}}(j_i) \geq c_{j_i} \cdot \frac{\bar{Q} - 2F(A_{<i})}{3P}.
\end{equation}

Let $k$ be the first index where $F_{A_{<k}} < \bar{Q}/3$ but $F_{A_{<k+1}} \geq \bar{Q}/3$ (i.e., the iteration the algorithm terminates). Note that \( \text{cost}(A_{<k}) = \sum_{i=1}^{k} c_{j_i} \). We start by bounding the first $k - 1$ terms in \( \text{cost}(A_{<k}) \):

\[
\frac{\bar{Q}}{3} > F(A_{<k}) = \sum_{i=1}^{k-1} F_{A_{<i}}(j_i) \geq \sum_{i=1}^{k-1} c_{j_i} \cdot \frac{\bar{Q}}{3P} = \sum_{i=1}^{k-1} c_{j_i} < P.
\]

Now consider the last term in \( \text{cost}(A) \), i.e., \( c_{j_k} \).

Again, by Lemma 4.4, we have,

\[
c_{j_k} \leq \frac{F_{A_{<k}}(j_k) \cdot P}{\bar{Q} - 2F(A_{<k})} \leq \left( \frac{\bar{Q} - F(A_{<k})}{\bar{Q} - 2F(A_{<k})} \right) \cdot P \leq 2P,
\]

using the fact that $F(A_{<k}) < \bar{Q}/3$. As such, $\text{cost}(A_{<k}) \leq 3P$ finalizing the proof. \hfill \Box 

**4.2 Proof of Theorem 4.1** We use the algorithm in Theorem 4.2 to present the following algorithm for reducing the expected deficit of any given set $S$ in Theorem 4.1.

\text{SELECT}(X, g, S, \alpha). Given a collection of indices $X$, a monotone submodular function $g$ with $g(X) = Q_g$ for every $X \sim X$, collection of items $S$ with expected deficit $\Delta = E[Q_g - g(S)]$, picks a set $T$ of items to decrease the expected deficit.

1. Let $\Psi := 6\alpha$.

2. For $i = 1, \cdots, \Psi$ do:

   (a) Sample a realization $S_i \sim S$.

   (b) $T_i \leftarrow \text{NON-ADAPT-GREEDY}(X \setminus S_i, g, \Delta(S_i))$ (recall that $\Delta(S_i) = Q_g - g(S_i)$).

3. Return all items in the sets $T := T_1 \cup T_2 \cdots \cup T_\Psi$.

The \text{SELECT} algorithm repeatedly calls the NON-ADAPT-GREEDY algorithm for samples drawn from realizations of the set $S$. By Fact 2.2, for any realization $S_i$ of $S$, $g(S_i)$ is also a monotone submodular function. Moreover, by the assumption that $g(X) = Q_g$ always, we have that $g(S_i) = Q_g - f(S_i)$ always as well. Hence, the parameters given to function NON-ADAPT-GREEDY in \text{SELECT} are valid.

We first bound the expected cost of \text{SELECT}.

\begin{claim}
\[ E[\text{cost}(T)] = O(\alpha) \cdot E[\text{cost}(\text{OPT})]. \]
\end{claim}

**Proof.** Cost of $T$ is the cost of the sets $T_1, \cdots, T_\Psi$ chosen by NON-ADAPT-GREEDY on $g_S$ for each of the $\Psi$ realizations of $S$. By Theorem 4.2, we can bound the cost of each $T_i$ using $\text{OPT}$ conditioned on realization $S_i$ for $S$ (as we consider $g_S$). As such,

\[
E[\text{cost}(T)] = \sum_{i=1}^{\Psi} E_{S_i \sim S} [\text{cost}(T_i)] \\
\leq \sum_{i=1}^{\Psi} E_{S_i \sim S} \left[ 3 \cdot E_{X \sim X} [\text{cost}(\text{OPT}(X)) | S = S_i] \right] \\
= 3\Psi \cdot E_{X \sim X} [\text{cost}(\text{OPT}(X))].
\]

where the inequality (a) follows from Theorem 4.2 because even though the $\text{OPT}$ used in Theorem 4.2...
is an optimal algorithm on the problem instance
\((\hat{Q},X,S)\), but the cost of \(\mathbb{E}_X[\text{cost}(\text{OPT}(X)) \mid S = S_i]\) can only be larger than the cost of OPT on the instance \((\hat{Q},X \setminus S)\). The bound now follow from the value of \(\Psi = 6\alpha\). \(\square\) Claim 4.5

We now prove that the expected deficit of \(f(S \cup T)\)
is dropped by at least a \(\Delta/6\) factor. The following lemma is at the heart of the proof.

**Lemma 4.6.** \(\mathbb{E}[\Delta(S \cup T)] \leq 5\Delta / 6\).

**Proof.** We start by introducing the notation needed. We are computing \((2)\) the randomness in the indexed \(\text{Select}\) in iterations 1 through \(i\), and \(S_{\leq i}\) to denote the tuple of realizations \((S_{1} \cdots , S_{i})\) (we define \(T_{<i}\) and \(S_{<i}\) analogously, where \(T_{<i} = S_{<i} = \emptyset\)). We also denote by \(T_{\leq i}(S_{\leq i})\) the sets selected in iterations 1 to \(i\) given \(S_{\leq i}\).

Consider any \(i \in [\Psi]\). For a realization \(S_{i} \sim S\), we are computing \(\text{NON-ADAPT-GREEDY}\) on \(g_{S_{i}}\), with parameter \(Q = \Delta(S_{i})\). As such, by Theorem 4.2, for the set \(T_{i}(S_{i})\) returned, we have \(\mathbb{E}_X[g_{S_{i}}(T_{i}(S_{i}))] \geq Q / 3 = \Delta(S_{i})/3\). Consequently,

\[
\mathbb{E}_{S_{\leq i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{S_{i}}(T_{i}(S_{i}))] \geq \mathbb{E}_{S_{\leq i} \sim S} \frac{\Delta(S_{i})}{3} = \frac{\Delta}{3}.
\]

We now use this equation to argue that adding each set \(T_{i}\) can decrease the expected deficit. Before that, let us briefly touch upon the difficulty in proving this statement and the intuition behind the proof. In \(\text{SELECT}\), we first pick a realization \(S_{i}\) of \(S\) and then add “enough” sets to \(T_{i}\) to (almost) cover the deficit introduced by \(S_{i}\). This corresponds to Eq (4.5). However, our goal is to decrease the expected deficit of \(S\) (not a deficit of a single realization). As such, the quantity of interest is in fact the following instead:

\[
\mathbb{E}_{X}[g_{S}(T_{i})] = \mathbb{E}_{S_{\leq i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{S_{i}}(T_{i}(S_{i}))],
\]
i.e., the marginal contribution of \(T_{i}(S_{i})\) (chosen by picking a set \(S_{i}\)) to a “typical” set \(S_{i}' \sim S_{i}\) (not exactly the set \(S_{i}\)). The set \(T_{i}\) we picked in this step is not necessarily covering the deficit introduced by \(S_{i}'\) as well (in the context of the stochastic set cover problem, think of \(S_{i}\) and \(S_{i}'\) as covering a completely different set of elements and \(T_{i}\) being a deterministic set covering \(U \setminus S_{i}\)). As such, it is not at all clear that picking the set \(T_{i}\) should make “any progress” towards reducing the expected deficit.

The way we get around this difficulty is to additionally consider the marginal contribution of the sets \(T_{1}, \ldots , T_{\Psi}\) to each other. If \(T_{i}\) cannot decrease the expected deficit of most realizations \(S\) chosen from \(S\), then this means that by picking another set \(T_{2}(S)\) (for a realization \(S\) of \(S\)), the set \(T_{1} \cup T_{2}\) needs to have a coverage larger than both \(T_{1}\) and \(T_{2}\) individually (in the context of the set cover problem, since \(T_{1}\) is “useless” in covering deficit created by \(S\), and \(T_{2}\) can cover this deficit, this means that \(T_{1}\) and \(T_{2}\) should not have many elements in common typically). We formalize this intuition in the following claim (compare Eq (4.7) in this claim with Eq (4.6)).

**Claim 4.7.** Suppose at the start of iteration \(i\) the following holds

\[
\mathbb{E}_{S_{i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{S_{i}}(T_{<i}(S_{<i}))] < \frac{\Delta}{6}.
\]

Then,

\[
\mathbb{E}_{S_{<i} \sim S} \mathbb{E}_{S_{\leq i} \sim S}[g_{T_{<i}(S_{<i})}(T_{i}(S_{i}))] > \frac{\Delta}{6}.
\]

**Proof.** By subtracting Eq (4.7) from Eq (4.5), and using linearity of expectation we get that:

\[
\frac{\Delta}{6} < \mathbb{E}_{S_{i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{S_{i}}(T_{i}(S_{i})) - g_{S_{i}}(T_{<i}(S_{<i}))]
\]

(by monotonicity)

\[
\leq \mathbb{E}_{S_{i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{T_{\leq i}(S_{\leq i})} \cup S_{i}] - g_{T_{<i}(S_{<i})} \cup S_{i}]
\]

(by submodularity as \(T_{<i}(S_{<i}) \subseteq T_{\leq i}(S_{\leq i})\))

\[
= \mathbb{E}_{S_{i} \sim S} \mathbb{E}_{S_{<i} \sim S}[g_{T_{\leq i}(S_{\leq i})} - g_{T_{<i}(S_{<i})}]
\]

(4.8)

finalizing the proof. \(\square\) Claim 4.7

Suppose towards a contradiction that \(\mathbb{E}[\Delta(S \cup T)] > 5\Delta / 6\). This implies that,

\[
5\Delta / 6 < \mathbb{E}_{S \sim S}[Q_{g} - g(S \cup T)] = \mathbb{E}_{S \sim S}[Q_{g} - g(S) - g_{S}(T)]
\]

\[
\Rightarrow \mathbb{E}_{S \sim S}[g_{S}(T)] < \Delta / 6.
\]
By monotonicity of $f$ and since $T = T_1 \cup \ldots \cup T_\Psi$, this implies that for all $i \in [\Psi]$, 
\[
\Delta/6 > \mathbb{E}_{S \leq S_2} \mathbb{E}_X [g_S(T_{\leq i})] = \mathbb{E}_{S_i \leq S} \mathbb{E}_{S_{<i}} \mathbb{E}_X [g_S(T_{<i}(S_{<i}))].
\]
Hence, we can apply Claim 4.7 to obtain that for any $i \in [\Psi]$, 
\[
\mathbb{E}_{S_{<i}\sim S_2} \mathbb{E}_X [g_{T_{<i}(S_{<i})}(T_i(S_i))] > \frac{\Delta}{6}.
\]
As such, by linearity of expectation and above equation, 
\[
\mathbb{E}_{S \sim S_2} \mathbb{E}_X [g(T(S_{\leq i}))] = \sum_{i=1}^{\Psi} \mathbb{E}_{S_i \sim S_2} \mathbb{E}_X [g_{T_{<i}(S_{<i})}(T_i(S_i))] \\
> \Psi \cdot \frac{\Delta}{6} = 6\alpha \cdot \frac{\Delta}{6} \\
\geq Q_g = \mathbb{E}_X [g(X)].
\]
where the last inequality follows due to the condition that $\alpha \geq Q_g/\Delta$. The above is a contradiction as $T \subseteq X$ and $g$ is monotone. Hence, $\mathbb{E}[\Delta(S \cup T)] \leq 5\Delta/6$, finalizing the proof. \hfill Lemma 4.6

Theorem 4.1 now follows immediately from Claim 4.5 and Lemma 4.6.

5 Algorithms for the Stochastic Submodular Cover Problem

In this section, we present our main algorithmic result which formalizes Result 1.

Theorem 5.1. Let $E$ be a ground-set of items, $f : 2^E \rightarrow \mathbb{N}_+$ be a monotone submodular function with $Q := f(E)$, and $X := \{X_1, \ldots, X_m\}$ be a collection of $m$ stochastic items with support in $E$. Let $c_i \in [C]$ be the integer-valued cost of item $X_i$. For any integer $r \geq 1$, there exists an $r$-round adaptive algorithm for the stochastic submodular cover problem for function $f$ and items $X$ with expected cost $O\left(r \cdot Q^{1/r} \cdot \log Q \cdot \log(mC)\right)$ times the expected cost of the optimal adaptive algorithm.

Theorem 5.1 immediately implies that the $r$-round adaptivity gap of the stochastic submodular cover problem is $O(Q^{1/r})$. The rest of this section is devoted to the proof of Theorem 5.1.

Overview. The underlying strategy behind our algorithm is as follows: in each round of the algorithm, reduce the deficit of the currently realized set $T$ chosen in the previous rounds (i.e., the quantity $Q - f(T)$) by a factor of roughly $Q^{1/r}$. This suggests that after $r$ rounds the deficit should reach zero, hence we obtain a submodular cover. In order to do so, the algorithm needs to specify an ordering of items without knowing the realizations of these items in advance (i.e., non-adaptively). This step is itself done by running the algorithm in Theorem 4.1 over multiple iterative phases to reduce the deficit without knowing realization of any chosen items in this round. We now present our algorithm in details, starting with its main component for reducing the deficit in each round.

5.1 The Reduce Subroutine Let $T_k$ be the items selected by the $r$-round adaptive algorithm in rounds up to (and including) $k$, and $T_k$ be their realization. In round $k$, the algorithm creates an ordering of all the available items and sets a threshold $\tau_k := Q - Q^{(r-k)/r}$ for coverage in this round: after deciding on an ordering of the items non-adaptively, the algorithm picks items according to this ordering one by one until the total coverage of the function reaches $\tau_k$. In this section, we design an algorithm, namely Reduce, which returns an ordered set $S \subseteq X \setminus T_{k-1}$ in round $k$ such that items in $S$ are enough to reach the coverage threshold for this round with high probability. If there are items that are not included in $S$ by Reduce, we will simply add them at the end of $S$ in any arbitrary order.

The input to the function Reduce in round $k$ is the set of items $X \setminus T_{k-1}$, and the function marginal $f_{T_{k-1}}$; by Fact 2.2, $f_{T_{k-1}}$ is also a monotone submodular function. The execution of Reduce is partitioned over $\Gamma := O(\log(mC))$ phases, where in each phase, the algorithm picks a new set of items to be added to the (ordered) set returned by it. The final set of items returned by Reduce are ordered in increasing order of the phases (with arbitrary ordering in each phase).

For any phase $p \in [\Gamma]$, we define $S_p$ as the ordered set of items selected in phase 1 up to (and including) $p$. Let $Q_k := Q - f(T_{k-1})$; this is the deficit of the set $T_{k-1}$ with respect to function $f$. For any set $S$ of items, we define the following event $\mathcal{E}_k(S)$:

\[
\mathcal{E}_k(S) := 1\{Q_k - f_{T_{k-1}}(S) \geq Q_k/Q^{1/r}\}.
\]

Intuitively speaking, $\mathcal{E}_k(S)$ happens if the set of items $S$ cannot cover most of $Q_k$ yet.

In each phase, Reduce makes $\Lambda := O(\log Q)$ calls to SELECT subroutine (Theorem 4.1). Each call in phase $p$ is to increase the coverage of the set $S_{p-1}$ to eventually achieve a larger coverage in $S_p$. Instead of passing $S_{p-1}$ directly to SELECT, we instead pass the
Claim 5.2. Reduce move on to the next round. We present the pseudo-algorithm below.

\begin{algorithm}
\caption{Reduce($X, f_{T_{k-1}}$): Given a set $X$ of items and a monotone submodular function $f_{T_{k-1}}$, outputs an ordered set of items $S$ to be used in round $k$ of the $r$-round adaptive algorithm.}

1. Initialize: Set $\Lambda \leftarrow 12 \log(Q)$, and $\Gamma \leftarrow 2 \log(mC)$.

2. Set $S_0 \leftarrow \emptyset$.

3. For phases $p = 1, \cdots, \Gamma$ do:
   \begin{enumerate}
   \item Set $R_0 \leftarrow \emptyset$ and let $S_{p-1} := S_{p-1} \mid \mathcal{E}_k(S_{p-1})$.
   \item For iterations $i = 1, \cdots, \Lambda$ do:
     \begin{enumerate}
     \item Select $R_i \leftarrow R_{i-1} \cup \text{Select}(X \setminus \{R_{i-1} \cup S_{p-1}^\prime, f_{T_{k-1}}, R_{i-1} \cup S_{p-1}^\prime, 2Q^{1/r})$.
     \item $S_p \leftarrow S_{p-1} \cup R_i$.
     \end{enumerate}
   \end{enumerate}

4. Return the set $S_{\Gamma}$, ordered according to the order in which items were added to $S_{\Gamma}$.
\end{algorithm}

Before analyzing Reduce we need the following straightforward extension of Theorem 4.1.

Claim 5.1. (Extension of Theorem 4.1) Let $f_T$ be any monotone submodular function, for some $T \subseteq E$, such that $Q' := Q - f(T)$. Let $S \subseteq X$ be any subset of items, and $\mathcal{E}$ be an event which is a function of $S$ and $S := S \mid \mathcal{E}$. Let $\Delta := \mathbb{E}[Q' - f_T(S)]$, then Select, given parameter $\alpha \geq Q'/\Delta$, and $6\alpha$ samples from $S$, outputs a set $R \subseteq X \setminus S$ such that cost of $R$ is $O(\alpha) \cdot \mathbb{E}[\text{cost(OPT)}]$, in expectation over the randomness of the algorithm and $\mathbb{E}[Q' - f_T(S \cup R)] \leq 5\Delta/6$ over the randomness of the algorithm and realizations of $S$ and $R$.

Claim 5.1 can be proven as follows: in Select we only need samples from the distribution $\mathcal{S}$, hence by sampling from the distribution of $\mathcal{S}$ instead we obtain the same result conditioned on event $\mathcal{E}$.

We start by bounding the cost of the sets returned by Reduce in each phase. Note that not all these sets are going to be chosen by the $r$-round algorithm in round $k$ (as we may cover $\tau_k$ before reaching these sets and move on to next round) and hence this cost is not a lower bound on cost of the $r$-round algorithm.

Claim 5.2. For any $p \in [\Gamma]$, $\mathbb{E}[\text{cost}(S_p \setminus S_{p-1})] = O(Q^{1/r} \cdot \log Q) \cdot \mathbb{E}[\text{cost(OPT)}] \mathcal{E}_k(S_{p-1})]$. 

Proof. We call Select with the parameter $2Q^{1/r}$ for $O(\log Q)$ iterations. By Claim 5.1, cost of each iteration of phase $p$ is at most $O(Q^{1/r})$ times the expected cost of OPT conditioned on $\mathcal{E}_k(S_{p-1})$. Hence, total cost of phase $p$ is $\mathbb{E}[\text{cost}(S_p \setminus S_{p-1})] = O(Q^{1/r} \cdot \log Q) \cdot \mathbb{E}[\text{cost(OPT)}] \mathcal{E}_k(S_{p-1})]$. 

We now prove the main property of the Reduce subroutine, i.e., that the sets returned by it can cover the required threshold $\tau_k$ with high probability.

Lemma 5.3. Suppose $S_{\Gamma} := \text{Reduce}(X, f_{T_{k-1}})$. Then,

\[ \Pr(\mathcal{E}_k(S_{\Gamma})) \leq 1/(mC)^2, \]

with respect to the randomness of the algorithm and the realizations of $S_{\Gamma}$.

Proof. We prove that the probability of the event $\mathcal{E}_k(S_p)$ decreases after each phase $p$ by a constant factor. Fix a phase $p \in [\Gamma]$. For a realization $S$ we define deficit $\Delta(S) = Q_k - f_{T_{k-1}}(S)$. Recall that $R_i$ is the set of items picked up to (and including) iteration $i$ in phase $p$ on calls to Select with parameter $\alpha = 2Q^{1/r}$. By Claim 5.1 we know that each iteration reduces the expected deficit by a constant factor. More formally, fix an $R_{i-1}$ selected up to iteration $i - 1$. If $\mathbb{E}[\Delta(R_{i-1} \cup S_{p-1})|\mathcal{E}_{p-1}] \geq Q_k/2Q^{1/r}$, then the condition of Claim 5.1 that $\alpha \geq Q'/\Delta$ is satisfied with $\Delta = \mathbb{E}[\Delta(R_{i-1} \cup S_{p-1})|\mathcal{E}_{p-1}]$, $\alpha = 2Q^{1/r}$, and $Q' = Q_k$. We then have

\[ \mathbb{E}[\Delta(R_{i-1} \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \leq \frac{5}{6} \mathbb{E}[\Delta(R_{i-1} \cup S_{p-1})|\mathcal{E}_k(S_{p-1})], \]

Copyright © 2019 by SIAM
Unauthorized reproduction of this article is prohibited
Lemma 5.3

The expected deficit can be written as

\[ \mathbb{E}[\Delta(R_A \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \geq \frac{Q_k}{2Q^{1/r}}. \]

Due to the fact that \( f_{T_{k-1}} \) is a monotone function, we have

\[ \mathbb{E}[\Delta(R_i \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \geq \mathbb{E}[\Delta(R_A \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \]

for all \( R_i \). Then using Eq. (5.10) and the above equation, we can observe that the condition of Claim 5.1 is satisfied for every \( R_i \). This implies that after \( \Lambda \) iterations the expected deficit can be written as

\[ \mathbb{E}[\Delta(R_A \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \leq \frac{5}{6} \cdot \mathbb{E}[\Delta(S_{p-1})|\mathcal{E}_k(S_{p-1})] \]

(Recall that \( \Lambda = 12\log Q \))

\[ \leq \frac{5}{6} \cdot 12\log Q \cdot Q_k \]

\[ < \frac{Q_k}{2Q^{1/r}}. \]

Eq. (5.10) and Eq. (5.11) lead to a contradiction. Hence, we will have that

\[ \mathbb{E}[\Delta(S_p)|\mathcal{E}_k(S_{p-1})] = \mathbb{E}[\Delta(R_A \cup S_{p-1})|\mathcal{E}_k(S_{p-1})] \]

\[ \leq \frac{Q_k}{2Q^{1/r}}. \]

5.2 The r-Round Adaptive Algorithm

We are now ready to present our \( r \)-round algorithm which is based on successive applications of the REDUCE subroutine.

\[ \text{r-Round-Adaptive}(X, f, Q): \text{Given a set of items } X, \text{ a monotone submodular function } f, \text{ and the desired coverage value } Q, \text{ outputs a set } T \text{ such that its realization } T \text{ is feasible.} \]

1. Initialize: Set \( T_0 \leftarrow \emptyset, T_0 \leftarrow \emptyset \)
2. For \( k = 1, 2, \ldots, r \) do:
   a. Set threshold \( \tau_k \leftarrow Q - Q^{(r-k)/r} \)
   b. \( T \leftarrow \text{REDUCE}(X \setminus T_{k-1}, f_{T_{k-1}}) \)
   c. Add the remaining items \( X \setminus (T \cup T_{k-1}) \) at the end of \( T \) in any arbitrary order.
   d. Observe the realizations \( T' \) of the set of items \( T' \subseteq T \) selected by running through the ordered set \( T \) until a total coverage of \( \tau_k \) is reached, i.e. \( f(T_{k-1} \cup T') \geq \tau_k \)
   e. \( T_k \leftarrow T' \cup T_{k-1} \text{ and } T_k \leftarrow T' \cup T_{k-1} \)
3. Return \( T_r \) with realization \( T_r \) as the final answer.

We are now ready to prove Theorem 5.1 by analyzing the above algorithm. The overall plan is to bound the cost of each round of the \( r \)-round algorithm. In each round the algorithm selects an ordering returned by a call to REDUCE and adds the remaining items at the end of this ordering. As argued earlier, not all the sets in the ordering are going to be chosen by the \( r \)-round algorithm in round \( k \). We will use Claim 5.2 and Lemma 5.3 to bound the expected cost of the items selected from the ordering in round \( k \) in terms of the expected cost of OPT. In order to do so, we first lower bound the cost of OPT.

Claim 5.4. For any (possibly randomly chosen) collection \( S \subseteq X \), and any event \( \mathcal{E} \) which is a function of \( S \), the expected cost of OPT can be lower bounded as

\[ \mathbb{E}[\text{cost(OPT)}] \geq \mathbb{E}[\text{cost(OPT)}|\mathcal{E}] \cdot \mathbb{E}[\mathcal{E}]. \]

Proof. The expected cost of OPT can be written as

\[ \mathbb{E}[\text{cost(OPT)}] = \mathbb{E}[\text{cost(OPT)}|\mathcal{E}] + \mathbb{E}[\text{cost(OPT)}|\neg \mathcal{E}] \]

\[ \geq \mathbb{E}[\mathcal{E}] \cdot \mathbb{E}[\text{cost(OPT)}|\mathcal{E}] \cdot \mathbb{E}[\text{cost(OPT)}|\mathcal{E}]. \]

Note that the above also holds even if the collection \( S \) is itself randomly chosen. Lemma 5.4
We now prove the lemma bounding the expected cost of each round of \( r\text{-Round-Adaptive} \). We will define the notation \( \text{cost}(\text{Round}_k) \) to be the total cost of all the items added to the feasible set in round \( k \). More formally,

\[
\text{cost}(\text{Round}_k) := \text{cost}(T_k \setminus T_{k-1}).
\]

Now, we will provide a bound on \( \mathbb{E}[\text{cost}(\text{Round}_k)] \).

**Lemma 5.5.** For any round \( k \leq r \), given \( T_{k-1} \), the expected cost paid by the \( r\text{-Round-Adaptive} \) algorithm in round \( k \) can be bounded as

\[
\mathbb{E}[\text{cost}(\text{Round}_k)|T_{k-1}] \\
\leq O(Q^{1/r} \log(Q) \log(mC)) \cdot \mathbb{E}[\text{cost}(\text{OPT})|T_{k-1}].
\]

**Proof.** Recall that in round \( k \) we call REDUCE with parameter \( f_{T_{k-1}} = f_{T_{k-1}} \) such that \( Q_k = Q - f(T_{k-1}) \). Also, recall that in phase \( p \), REDUCE adds items \( S_p \setminus S_{p-1} \) to the ordering \( S_f \) returned by it. Using Claim 5.2 we have that

\[
\mathbb{E}[\text{cost}(S_p \setminus S_{p-1})|T_{k-1}] \\
= O(Q^{1/r} \cdot \log(Q)) \cdot \mathbb{E}[\text{cost}(\text{OPT})|T_{k-1}, \mathcal{E}_k(S_{p-1})].
\]

Also, recall that while running through the ordered set of round \( k \) we select items from \( S_{p-1} \setminus S_{p-1} \) only if the realization is such that the items in \( S_{p-1} \) are not able to reach the required coverage threshold \( \tau_k \). More formally, we only pay for the cost of items in \( S_{p-1} \setminus S_{p-1} \) when the event \( \mathcal{E}_k(S_{p-1}) \) occurs. Hence, we will pay the cost of phase \( p \) items with probability \( \Pr(\mathcal{E}_k(S_{p-1})) \). Also, in the case that all the items \( S_f \) returned by REDUCE are not able to reach the required coverage threshold, we trivially bound the cost by \( mC \). Since, \( Q_k \leq Q(r-k+1)/r \), this event happens with probability at most \( \Pr(\mathcal{E}_k(S_f)) \) which is upper bounded by \( 1/(mC)^2 \) using Lemma 5.3. Combining all this, we have that, given \( T_{k-1} \),

\[
\mathbb{E}[\text{cost}(\text{Round}_k)|T_{k-1}] \\
\leq \sum_{p=1}^{r} \Pr(\mathcal{E}_k(S_{p-1})) \cdot \mathbb{E}[\text{cost}(S_p \setminus S_{p-1})|T_{k-1}] \\
+ \Pr(\mathcal{E}_k(S_f)) \cdot mC \\
\leq \sum_{p=1}^{r} \Pr(\mathcal{E}_k(S_{p-1})) \cdot O\left(Q^{1/r} \log(Q)\right) \\
\cdot \mathbb{E}[\text{cost}(\text{OPT})|T_{k-1}, \mathcal{E}_k(S_{p-1})] \\
+ \Pr(\mathcal{E}_k(S_f)) \cdot mC \\
\leq O\left(Q^{1/r} \log(Q) \log(mC)\right) \\
\cdot \mathbb{E}[\text{cost}(\text{OPT})|T_{k-1}] + \Pr(\mathcal{E}_k(S_f)) \cdot mC
\]

where the last equality is due to the fact that once we fix the randomness due to coins up to round \( k-1 \), then the realizations \( T_{k-1} \) form a partition over the space of all realizations \( X \). Since the choice of the randomness was arbitrary, we have that \( \mathbb{E}[\text{cost}(\text{Round}_k)] \leq O\left(Q^{1/r} \log(Q) \log(mC)\right) \cdot \mathbb{E}[\text{cost}(\text{OPT})|T_{k-1}] \).

Then, using Eq. (5.13) and the above, the total cost can be bounded as

\[
\text{cost}(r\text{-Round-Adaptive}) \\
= O\left(rQ^{1/r} \log(Q) \log(mC)\right) \cdot \mathbb{E}[\text{cost}(\text{OPT})].
\]

**Theorem 5.1**
Remark 5.6. We can implement the $r$-round algorithm in polynomial time as long as the costs are polynomially bounded, i.e., achieve a pseudo-polynomial time algorithm. Indeed, the only “time consuming” step of the algorithm is to sample from the conditional distribution $S|E$ for some event $E$. This is however only needed as long as $\Pr(E) \geq 1/(mc)^{O(1)}$. Hence, one can use rejection sampling with the total running time bounded by $\text{poly}(QmC)$ to implement this step. The probability that we do not get the required number of samples from the event $E$ with $\Pr(E) \geq 1/mC$ after $\text{poly}(QmC)$ trials is negligible, and we can pay for the cost in case this bad event happens.

6 A Lower Bound for $r$-Round Adaptive Algorithms

In this section, we prove a lower bound on the approximation ratio of any $r$-round adaptive algorithm for the submodular cover problem and formalize Result 2. We prove this lower bound for the stochastic set cover problem (see Example 1.1) which is a special case of the stochastic submodular cover problem.

Theorem 6.1. For any integer $r \geq 1$, any $r$-round adaptive algorithm for the stochastic set cover problem on instances with $m$ stochastic sets from a universe of size $n$ elements such that $m = n^{\Omega(r)}$ has expected cost $\Omega\left(\frac{1}{r^2} \cdot n^{1/r}\right)$ times the cost of the optimal adaptive algorithm.

Theorem 6.1 formalizes Result 2 as by definition, $Q = n$ in the stochastic set cover problem.

Overview. Consider first an instance of the stochastic set cover problem with the following property. There exists a single stochastic set, say $T$, which realizes to $U \setminus \{e^*\}$ for $e^*$ chosen uniformly at random from $U$ (support of $T$ has $n$ sets). The remaining sets in this instance are $n$ singleton sets that each deterministically realize to some unique element $e \in U$. Solving such an instance adaptively with just two sets, and indeed even in two rounds of adaptivity, is trivial: choose the set $T$ and observe its realization in the first round; next choose the singleton set that covers $e^*$. However, consider any non-adaptive algorithm for this problem: even though it is obvious that the set $T$ needs to be the first set in the ordering returned by the algorithm, there is no “good” choice for the ordering of the remaining sets as the algorithm is oblivious to the identity of $e^*$ at this point. It is then fairly easy to see that no matter what ordering the non-adaptive algorithm chooses, in expectation $\Omega(n)$ sets need to be picked before it could cover $e^*$ and hence the universe $U$. An adaptivity gap of $\Omega(n)$ now follows easily from this argument.

Our main contribution in this section is to design a family of instances in this spirit that allows us to extend the above argument to $r$-round adaptive algorithms. Roughly speaking, these instances are constructed in a way that at the beginning of each round, the algorithm has access to a set that covers a “large” portion of the remaining universe “randomly”, but since the realization of this set is not known to the algorithm, unless it picks many more sets, it would not be able to also cover the “remainder of universe” (left out by the realization of the aforementioned set). Morally speaking, this corresponds to replacing the set $\{e^*\}$ with larger subsets of $U$ in the above argument and then recurse on each subset individually.

The rest of this section is devoted to the proof of Theorem 6.1. We start by introducing an algebraic construction of a set-system, named an edifice, due to Chakrabarti and Wirth [15] and use it to introduce a family of “hard” instances for the stochastic set cover problem. We then prove that any algorithm with limited rounds of adaptivity on these instances necessarily incurring a large cost compared to the optimal adaptive algorithm and prove Theorem 6.1.

Edifice Set-System An edifice over a universe $U$ of $n$ items is a collection of sets in which for any two sets, either one of them is a subset of the other, or the two sets have a small intersection. Formally:

Definition 1. (Edifice Set-System [15]) For integers $k \leq s \leq b \leq d$, a $(s, b, k, d)$-edifice $T$ over a universe $U$ is a complete $d$-ary $k$-level rooted tree together with a collection of associated sets, satisfying the following properties:

(I) Each node $v$ in $T$ is associated with a set $U_v \subseteq U$ such that the set associated to the root of $T$ is $U$, and $U_u \subseteq U_v$ if $u$ is a child of $v$ in $T$.

(II) If $v$ is a leaf of $T$, then $|U_v| = b$.

(III) For each leaf $u$ and each node $v$ not an ancestor of $u$ in $T$, $|U_u \cap U_v| \leq s$.

In this definition, we say that root is at level 1 of the tree and the leaf-vertices are at level $k$.

Edifices are typically interesting when the parameter $s$ is small and parameter $b$ is large compared to the size of the universe, i.e., when we have large sets which are almost disjoint from each other in a recursive manner suggested by the tree-structure of an
edifice. For our purpose, we are interested in edifices with parameters $r = k ≈ s$ ($r$ is the number of rounds we want to prove the lower bound for), $b ≈ n^{1/k}$, and $d = n^{O(1)}$ ($n$ is the number of elements in the universe). The existence of such edifices follows from the results in [15] (see Theorem 3.5; see also RND-set systems in [6] for a similar construction), which we summarize in the following proposition.

**Proposition 6.1.** ([15]) For infinitely many integers $N$ and any integer $k ≥ 1$, there exists a $(4k, N, k, N^2)$-edifice over a universe $U$ of size $N^k$.

### Hard Instances for Stochastic Set Cover

Fix an integer $k ≥ 1$ and a sufficiently large integer $N ≥ k$ and let $U$ be a universe of size $N^k$ elements. Define $T$ as any arbitrary $(4k, N, k, N^2)$-edifice over $U$ which is guaranteed to exist by Proposition 6.1. We define the following family of “hard” instances for stochastic set cover.

#### Family $X^{(k)}$:

A collection of stochastic sets over universe $U$ using edifice $T$.

- For any vertex $u ∈ T$ and any element $e ∈ U$; there exists a dedicated stochastic set $X_u$ and $X_e$ in $X^{(k)}$, respectively, defined as follows.

- For any non-leaf vertex $u ∈ T$ with child-vertices $v_1, \ldots, v_d$, the stochastic set $X_u$ realizes to one of the sets $T_{u,v_1}, \ldots, T_{u,v_d}$ uniformly at random where $T_{u,v_i} := U_u \setminus U_{v_i}$.

- For any leaf vertex $u ∈ T$ with $U_u = \{e_1, \ldots, e_N\}$ (recall that $|U_u| = N$ be Definition 1), the stochastic set $X_u$ realizes to one of the sets $T_{u,e_1}, \ldots, T_{u,e_N}$ uniformly at random where $T_{u,e_i} := U_u \setminus \{e_i\}$.

- For any element $e ∈ U$, $X_e$ deterministically realizes to the singleton set $\{e\}$.

For any realization of $X^{(k)}$, we define the *canonical path* of the realization as the root-to-leaf path $P = v_1, v_2, \ldots, v_k$ over the vertices of the edifice $T$ as follows:

1. $v_1$ is the root of the tree $T$.
2. For any $1 ≤ i ≤ k$, $v_i$ is the child-vertex of $v_{i-1}$ corresponding to $T_{v_{i-1}, v_i} = X_{v_{i-1}}$.

We have the following simple claim on the cost of the optimal adaptive algorithm on the family $X^{(k)}$ for any integer $k ≥ 1$.

#### Claim 6.1. For any integer $k ≥ 1$, the expected cost of OPT on $X^{(k)}$ is at most $k + 1$.

**Proof.** We prove that the following algorithm has expected cost $k + 1$; clearly optimal adaptive algorithm can only have a lower expected cost.

Consider the adaptive algorithm that constructs the canonical path of the underlying realization one vertex at a time: it first chooses $v_1$ which is the root of $T$ and add $X_{v_1}$ to $S$. Next, based on the realization of $X_{v_1}$, it can determine the second vertex $v_2$ in the canonical path and adds $X_{v_2}$ to $S$. It continues like this until it has added all sets $X_{v_1}, \ldots, X_{v_k}$ to $S$ where $P := v_1, \ldots, v_k$ is the canonical path of the realization. Finally, a realization of $X_{v_k}$ for a leaf $v_k$ corresponds to a set $T_{v_k,e}$ that covers all of $U_{v_k}$ (the set associated with the leaf-vertex $v_k$ in the edifice) except for a single element $e$. The algorithm then picks the set $X_e$ which deterministically realizes to $\{e\}$.

Clearly, the number of stochastic sets picked by this algorithm is $k + 1$. We argue that these sets cover the universe $U$ entirely. This is because, $X_{v_1}$ covers $U \setminus U_{v_2}$, $X_{v_2}$ covers $U_{v_2} \setminus U_{v_3}$, and so on until $X_{v_k}$ covers $U_{v_k} \setminus \{e\}$. As such, $X_{v_1} \cup \ldots \cup X_{v_k}$ covers $U \setminus \{e\}$ and picking $X_e$ would cover the whole universe as $X_e$ always realizes to $\{e\}$.

The following lemma is where we use the properties of edifice $T$ and is crucial to our analysis.

#### Lemma 6.2.

Let $U_{v_k}$ be the set associated to the k-th vertex $v_k$ in the canonical path of $X^{(k)}$ in edifice $T$ and $C$ be any collection of sets in $X^{(k)} \setminus X_{v_k}$. Then $|\bigcup_{T ∈ C} T \cap U_{v_k}| ≤ 4|C| \cdot k$.

**Proof.** Fix any set $T ∈ C$. We prove that $|T \cap U_{v_k}| ≤ 4k$ which would immediately imply the lemma.

If $T$ is a realization of some set $X_e$ for some element $e ∈ U$, then $|T| = 1$ and hence the claim immediately holds. Hence, suppose that $T$ is a realization of $X_{v_k}$ for some vertex $v ∈ T$.

If $v$ is an ancestor of $v_k$, then $T = U_{v} \setminus U_{v'}$ where $v'$ is either another ancestor of $v_k$ or it is equal to $v_k$ itself by definition of the canonical path. In either case, by property (I) of edifices in Definition 1, $U_{v_k} ⊆ U_{v'}$ and hence $T \cap U_{v_k} = \emptyset$.

If $v$ is not an ancestor of $v_k$, then $T ⊆ U_v$ as $X_v ⊆ U_v$ and by property (III) of edifices in Definition 1, $|U_v \cap V_{v_k}| ≤ 4k$ (here parameter $s = 4k$) and hence $|T \cap V_{v_k}| ≤ 4k$, finalizing the proof.

The proof of Theorem 6.1 then uses Lemma 6.2 to show that any $(r = k)$-round adaptive algorithm
for stochastic set cover on $X^{(k)}$ should incur a cost of roughly $\sqrt[1/k]{n}$. It is worth remarking that the adaptive algorithm in Claim 6.1 that achieves the cost of $k+1$ requires only $k+1$ rounds of adaptivity; as such, our results are in fact proving a separation between the cost of any $k$-round and $k+1$-round adaptive algorithms. The remainder of proof of Theorem 6.1 is rather technical and is hence postponed to the full version of the paper [2].

Acknowledgements We thank the anonymous reviewers for helpful comments.

References


