Gradient Method for Continuous Influence Maximization with Budget-Saving Considerations

Wei Chen,[*,1] **Weizhong Zhang,**[2] **Haoyu Zhao**[3]
[1]Microsoft Research, Beijing, China
[2,3]IIIS, Tsinghua University, Beijing, China
weic@microsoft.com, {zwz15, zhaohy16}@mails.tsinghua.edu.cn

Abstract

Continuous influence maximization (CIM) generalizes the original influence maximization by incorporating general marketing strategies: a marketing strategy mix is a vector $\boldsymbol{x} = (x_1, \ldots, x_d)$ such that for each node v in a social network, v could be activated as a seed of diffusion with probability $h_v(\boldsymbol{x})$, where h_v is a strategy activation function satisfying DR-submodularity. CIM is the task of selecting a strategy mix $\boldsymbol{x}$ with constraint $\sum_i x_i \leq k$ where k is a budget constraint, such that the total number of activated nodes after the diffusion process, called influence spread and denoted as $g(\boldsymbol{x})$, is maximized. In this paper, we extend CIM to consider budget saving, that is, each strategy mix $\boldsymbol{x}$ has a cost $c(\boldsymbol{x})$ where c is a convex cost function, and we want to maximize the balanced sum $g(\boldsymbol{x}) + \lambda(k - c(\boldsymbol{x}))$ where λ is a balance parameter, subject to the constraint of $c(\boldsymbol{x}) \leq k$. We denote this problem as CIM-BS. The objective function of CIM-BS is neither monotone, nor DR-submodular or concave, and thus neither the greedy algorithm nor the standard result on gradient method could be directly applied. Our key innovation is the combination of the gradient method with reverse influence sampling to design algorithms that solve CIM-BS: For the general case, we give an algorithm that achieves $\left(\frac{1}{2} - \varepsilon\right)$-approximation, and for the case of independent strategy activations, we present an algorithm that achieves $\left(1 - \frac{1}{e} - \varepsilon\right)$ approximation.

1 Introduction

Influence maximization is the task of selecting a small number of seed nodes in a social network such that the influence spread from the seeds when following an influence diffusion model is maximized. It models the viral marketing scenario and has been extensively studied (cf. (Kempe, Kleinberg, and Tardos 2015; Chen, Lakshmanan, and Castillo 2013; Li et al. 2018)). Continuous influence maximization (CIM) generalizes the original influence maximization by incorporating general marketing strategies: a marketing strategy mix is a vector $\boldsymbol{x} = (x_1, \ldots, x_d)$ such that for each node v in a social network, v could be activated as a seed of diffusion with probability $h_v(\boldsymbol{x})$, where h_v is a strategy activation function satisfying monotonicity and DR-submodularity. CIM is the task of selecting a strategy mix $\boldsymbol{x}$ with constraint $\sum_i x_i \leq k$ where k is a budget constraint, such that the total number of activated nodes after the diffusion process, called influence spread and denoted as $g(\boldsymbol{x})$, is maximized. CIM is proposed in (Kempe, Kleinberg, and Tardos 2015), and recently followed up by a few studies (Yang et al. 2016; Chen, Wu, and Yu 2019).

In this paper, we extend CIM to consider budget saving: each strategy mix $\boldsymbol{x}$ has a cost $c(\boldsymbol{x})$ where c is a convex cost function, and we want to maximize the balanced sum $g(\boldsymbol{x}) + \lambda(k - c(\boldsymbol{x}))$ where λ is a balance parameter, subject to the constraint of $c(\boldsymbol{x}) \leq k$. We denote this problem as CIM-BS. The objective reflects the realistic consideration of balancing between increasing influence spread and saving marketing budget. In general we have $g(\boldsymbol{x})$ monotone (increasing) and DR-submodular (diminishing return property formally defined in Section 2), but $\lambda(k - c(\boldsymbol{x}))$ is concave and likely to be monotonically decreasing, and thus the objective function $g(\boldsymbol{x}) + \lambda(k - c(\boldsymbol{x}))$ is neither monotone, nor DR-submodular or concave, and thus neither the greedy algorithm nor the standard result on gradient method could be directly applied for a theoretical guarantee.

In this paper, we apply the gradient method (Nesterov 2013; Parikh, Boyd, and others 2014) to solve CIM-BS with theoretical approximation guarantees. This is the only case we know of that the gradient method is applied to influence maximization with a theoretical guarantee while the greedy method cannot. The gradient method may be applied to the original objective function $g(\boldsymbol{x}) + \lambda(k - c(\boldsymbol{x}))$, but $g(\boldsymbol{x})$ is a complicated combinatorial function and its exact gradient is infeasible to compute. We could use stochastic gradient instead, but it results in large variance and very slow convergence. Instead, we integrate the gradient method with the reverse influence sampling (RIS) approach (Borgs et al. 2014; Tang, Xiao, and Shi 2014; Tang, Shi, and Xiao 2015), which is the main technical innovation in our paper. RIS is proposed for improving the efficiency of the influence maximization task, but when integrating with the gradient method, it brings two additional benefits: (a) it allows the efficient computation of the exact gradient of an estimator function of $g(\boldsymbol{x})$, which avoids slow convergence caused by the large variance in the stochastic gradient, and (b) for a class of independent strat-

[*]All authors contributed equally.

egy activation functions h_v where each strategy dimension independently attempts to activate node v, the new objective function is in the form of coverage functions (Karimi et al. 2017b), which allows a tight concave upper bound function and leads to a better approximation ratio.

For the general case, we apply the proximal gradient method originally designed for concave functions to work with RIS and achieve an approximation of $\left(\frac{1}{2} - \varepsilon\right)$ (Theorem 3). This requires an adaptation of the proximal gradient method for the functions of the form $f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})$ where f_1 is non-negative, monotone and DR-submodular and f_2 is non-negative and concave, and the result of this adaption (Theorem 2) may be of independent interest. For the independent strategy activation case, we apply the projected subgradient method on a tight concave upper bound of the objective function and achieve a $\left(1 - \frac{1}{e} - \varepsilon\right)$ approximation (Theorem 4). We test our algorithms on a real-world dataset and validate its effectiveness comparing with other algorithms.

In summary, our contributions include: (a) we propose the study of CIM-BS problem to balance influence spread with budget saving; and (b) we integrate the gradient method with reverse influence sampling and provide two algorithms with theoretical approximation guarantees, on the objective function that is neither monotone, nor DR-submodular or concave. Our study is one of the first studies that introduce the gradient method to influence maximization, and hopefully it will enrich the scope of the influence maximization research.

Due to the space constraint, detailed proofs and full experiment results are moved to the full technical report (Chen, Zhang, and Zhao 2019).

Related works. Influence maximization was first proposed by Kempe et al. (Kempe, Kleinberg, and Tardos 2015) as a discrete optimization problem, and has been extensively studied since (cf. (Chen, Lakshmanan, and Castillo 2013; Li et al. 2018)). CIM is also proposed in (Kempe, Kleinberg, and Tardos 2015). Yang et al. (Yang et al. 2016) propose heuristic algorithms to solve CIM more efficiently, while Wu et al. (Chen, Wu, and Yu 2019) consider discrete version of CIM and apply RIS to solve it efficiently. Profit maximization in (Lu and Lakshmanan 2012; Tang, Tang, and Yuan 2016) introduces linear cost with no budget constraint to influence maximization. Our CIM-BS problem is new and more general than both CIM and profit maximization studied before. The RIS approach is originally proposed in (Borgs et al. 2014), and is further improved in (Tang, Xiao, and Shi 2014; Tang, Shi, and Xiao 2015; Nguyen, Thai, and Dinh 2016).

Recently, a number of studies have applied gradient methods to DR-submodular maximization: Bian et al. (Bian et al. 2017) apply the Frank-Wolfe algorithm to achieve $1 - 1/e$ approximation for down-closed sets; Hassani et al. (Hassani, Soltanolkotabi, and Karbasi 2017) apply stochastic projected gradient descent to achieve $1/2$ approximation; Karimi et al. (Karimi et al. 2017a) achieve $1 - 1/e$ approximation for coverage functions, which we adopt for the independent strategy activation case; Mokhtari et al. (Mokhtari, Hassani, and Karbasi 2018) apply more complicated conditional gradient method to achieve $1 - 1/e$ approximation. Our study is not a simple adoption of such methods to CIM-BS, because our objective function is not DR-submodular, and gradient computation cannot be treated as an oracle —- we have to provide exact gradient computation and an end-to-end integration with the RIS approach.

2 Preliminary and Model

In this paper, we focus on the triggering model for influence maximization problem. We use a directed graph $G = (V, E)$ to represent a social network, where V is the set of nodes representing individuals, and E is the set of directed edges with edge (u, v) representing that u could directly influence v. Let $n = |V|$ and $m = |E|$. In the diffusion process, each node is either active or inactive, and a node will stay active if it is activated. In the triggering model, every node $v \in V$ has a distribution D_v on the subsets of v's in-neighbors $N^-(v) = \{u | (u, v) \in E\}$. Before the diffusion starts, each node $v \in V$ samples a triggering set $T_v \subseteq N^-(v)$ from the distribution D_v, denoted $T_v \sim D_v$. At time $t = 0$, the nodes in a pre-determined *seed set* S are activated. For any time $t = 1, 2, \ldots$, the node v is activated if at least one of nodes in its triggering set T_v is activated at time $t - 1$. The whole propagation stops when no new node is activated in a step. An important quantity is the *influence spread* of the seed set S, denoted as $\sigma(S)$, which is defined as the expected number of the final activated nodes with seed set S. The classical *influence maximization* problem is to maximize $\sigma(S)$ such that $|S| \leq k$ for some given budget k.

A generalization of the classical influence maximization problem is the *continuous influence maximization (CIM)* problem with general marketing strategies (Kempe, Kleinberg, and Tardos 2015). A mix of marketing strategies is represented by a d-dimensional vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}_+^d$, where $\mathbb{R}_+$ is the set of non-negative real numbers. In the general case, we consider strategy mix $\boldsymbol{x}$ in a general convex set $\mathcal{D} \subseteq \mathbb{R}_+^d$, but most commonly we consider $\mathcal{D} = \mathbb{R}_+^d$ or $\mathcal{D}$ has an upper bound in each dimension, e.g. $\mathcal{D} = [0, 1]^d$. Given the strategy $\boldsymbol{x}$, each node $v \in V$ is independently activated as a seed with probability $h_v(\boldsymbol{x})$, where $h_v : \mathbb{R}_+^d \to [0, 1]$ is referred to as a *strategy activation function*. Once a set of seeds S is activated by a marketing strategy mix $\boldsymbol{x}$, the influence propagates from seeds in S following the triggering model. Then we define the influence spread of strategy mix $\boldsymbol{x}$, $g(\boldsymbol{x})$, as the expected number of nodes activated by $\boldsymbol{x}$, and formally,

$$\begin{aligned} g(\boldsymbol{x}) &= \mathbb{E}_S[\sigma(S)] \\ &= \sum_{S \subseteq V} \sigma(S) \cdot \prod_{u \in S} h_u(\boldsymbol{x}) \cdot \prod_{v \notin S} (1 - h_v(\boldsymbol{x})). \end{aligned} \tag{1}$$

The above formula means that we enumerate through all possible seed sets S, and due to independent seed activation by $\boldsymbol{x}$ the probability of S being the seed set is $\prod_{u \in S} h_u(\boldsymbol{x}) \cdot \prod_{v \notin S} (1 - h_v(\boldsymbol{x}))$ and its influence spread is $\sigma(S)$.

In many situations, each strategy dimension in $\boldsymbol{x}$ activates each node independently. That is, for each node v and each strategy $j \in [d]$, there is a function $q_{v,j}$ such that strategy j with amount x_j activates node v with probability $q_{v,j}(x_j)$. Then we have $h_v(\boldsymbol{x}) = 1 - \prod_{j \in [d]} (1 - q_{v,j}(x_j))$. We call this case *independent strategy activation*. Independent strat-

egy activation models many scenarios such as personalized marketing and event marketing (Chen, Wu, and Yu 2019).

In this paper, we focus on an extension of the continuous influence maximization problem — *continuous influence maximization with budget saving (CIM-BS)*. We have a total budge k, and for every strategy mix $\boldsymbol{x}$, there is a cost $c(\boldsymbol{x})$. We do not want the cost to exceed the budget, and we want to maximize the budget balanced influence spread: a combination of the expected influence spread and the remaining budget. More formally, we have the following definition.

Definition 1 (Continuous Influence Maximization with Budget Saving)**.** *The* continuous influence maximization with budget saving (CIM-BS) *is the problem of given (a) a social network $G = (V, E)$ and the triggering model $\{D_v\}_{v\in V}$ on G, (b) strategy activation functions $\{h_v\}_{v\in V}$, (c) cost function c, total budget k, and a balance parameter $\lambda \geq 0$, finding a strategy mix $\boldsymbol{x}^* \in \mathbb{R}_+^d$ to maximize its balanced sum of influence spread and budget savings, i.e., find $\boldsymbol{x}^*$ such that $\boldsymbol{x}^* \in \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{D}, c(\boldsymbol{x})\leq k} (g(\boldsymbol{x}) + \lambda(k - c(\boldsymbol{x})))$.*

Note that when $\lambda = 0$ and $c(\boldsymbol{x}) = \sum_{i\in[d]} x_i$, the CIM-BS problem falls back to the CIM problem defined in (Kempe, Kleinberg, and Tardos 2015), and also appears in (Yang et al. 2016; Chen, Wu, and Yu 2019). When $c(\boldsymbol{x})$ is a linear function and the budget constraint $c(\boldsymbol{x}) \leq k$ is dropped, the problem resembles the profit maximization studied in (Lu and Lakshmanan 2012; Tang, Tang, and Yuan 2016). However, the general version of the problem as defined here with $\lambda > 0$, a general cost function $c(\boldsymbol{x})$ and constraint $c(\boldsymbol{x}) \leq k$ together is new. Henceforth, let $s(\boldsymbol{x}) = \lambda(k - c(\boldsymbol{x}))$. Intuitively, $s(\boldsymbol{x})$ is the budget-saving part of the objective. The overall objective of $g(\boldsymbol{x})+s(\boldsymbol{x})$ is trying to find the balance between maximizing influence and saving budget. We call $g(\boldsymbol{x})+s(\boldsymbol{x})$ the *budget-balanced influence spread*. For convenience, we denote $\mathcal{P} = \{\boldsymbol{x} \mid \boldsymbol{x} \in \mathcal{D}, c(\boldsymbol{x}) \leq k\}$.

We say that a vector function $f : \mathcal{D} \to \mathbb{R}$ is DR-submodular if for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$ with $\boldsymbol{x} \leq \boldsymbol{y}$ (coordinate-wise), for any unit vector $\boldsymbol{e}_i$ with the i-th dimension 1 and all other dimensions 0, and for any $\delta > 0$, we have $f(\boldsymbol{x}+\delta\cdot\boldsymbol{e}_i)-b(\boldsymbol{x}) \geq f(\boldsymbol{y}+\delta\cdot\boldsymbol{e}_i)-f(\boldsymbol{y})$. DR-submodularity characterizes the diminishing marginal return on function f as vector $\boldsymbol{x}$ increases, hence the name. We also say that f is monotone if for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$ with $\boldsymbol{x} \leq \boldsymbol{y}$, $f(\boldsymbol{x}) \leq f(\boldsymbol{y})$; f is convex if for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, any $\lambda \in [0, 1]$, $f(\lambda\boldsymbol{x} + (1 - \lambda)\boldsymbol{y}) \leq \lambda f(\boldsymbol{x}) + (1 - \lambda)f(\boldsymbol{y})$; f is L-Lipschitz if for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, $|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq L \cdot ||\boldsymbol{x} - \boldsymbol{y}||_2$, where $||\cdot||_2$ is the vector 2-norm; f is β-smooth if it has gradients everywhere and for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, $||\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})||_2 \leq \beta||\boldsymbol{x} - \boldsymbol{y}||_2$. Note that when the gradients exist, the L-Lipschitz condition is equivalent to $||\nabla f(\boldsymbol{x})||_2 \leq L$ for all $\boldsymbol{x} \in \mathcal{D}$.

In this paper, we assume that the strategy activation function h_v is monotone and DR-submodular, which implies that the influence spread function g is also monotone and DR-submodular, same as assumed in (Kempe, Kleinberg, and Tardos 2015; Chen, Wu, and Yu 2019). It is reasonable in that, with more marketing effort, the probability of seed activation would increase (monotonicity) but the marginal effect may be decreasing. For the case of independent strategy activation ($h_v(\boldsymbol{x}) = 1 - \prod_{j\in[d]}(1 - q_{v,j}(x_j))$), we assume $q_{v,j}$ is non-decreasing and concave, which implies that h_v is monotone and DR-submodular (Chen, Wu, and Yu 2019). In term of the cost function c, we assume that c is convex and L_c-Lipschitz. The most common function is the simple summation (or 1-norm of $\boldsymbol{x}$): $c(\boldsymbol{x}) = \sum_{i\in[d]} x_i$, but more general convex functions are also common in the economics literature (e.g. (Mankiw 2014)), for example $c(x) = ||\boldsymbol{x}||_2$.

An important remark is now in order. When h_v's are monotone and DR-submodular and c is convex, g is monotone and DR-submodular and s is concave, and as a result $g + s$ may be neither monotone nor DR-submodular. This means the greedy hill-climbing algorithm of (Kempe, Kleinberg, and Tardos 2015; Chen, Wu, and Yu 2019) no longer has theoretical approximation guarantee for the CIM-BS problem. This motivates us to apply the gradient method to solve CIM-BS.

3 Gradient Method with Reverse Influence Sampling

Gradient method has been applied to many continuous optimization problems. For our CIM-BS problem, a natural option is to apply the gradient method directly on the objective function $g + s$. However, the influence spread function g is a complicated combinatorial function, such that its gradient ∇g is too complex to compute in practice. We could apply stochastic gradient on g (see Appendix D in (Chen, Zhang, and Zhao 2019)) but it has very large variance due to the significant amount of randomness from both strategies activating seeds and influence propagation from seeds, which leads to very slow convergence of the method. Instead, in this section, we propose a novel integration of the gradient method with the reverse influence sampling (RIS) approach (Borgs et al. 2014) for CIM-BS. The key insight is that RIS allows the efficient computation of the exact gradient of an alternative objective function $\hat{g}_{\mathcal{R}} + s$ while maintaining an approximation guarantee of $1/2 - \varepsilon$. Moreover, when independent strategy activation is satisfied by the model, the alternative objective enables a tight concave upper bound, which leads to a $1 - 1/e - \varepsilon$ approximation.

In Section 3.1, we first review existing results on RIS with the continuous domain. Then in Sections 3.2 and 3.3, we present the gradient method, its integration with RIS, and its theoretical analysis, which are our main technical contribution.

3.1 Properties of the Reverse Reachable Sets

The central concept in the RIS approach is the *reverse reachable set*, as defined below.

Definition 2 (Reverse Reachable Set)**.** *Under the triggering model, a* reverse reachable (RR) set *with a root node v, denoted R_v, is the random set of nodes that v reaches in one reverse propagation: sample all triggering sets $T_u, u \in V$, such that edges $\{(w, u)|u \in V, w \in T_u\}$ together with nodes V form a live-edge graph, and R_v is the set of nodes that can reach v (or v can reach reversely) in this live-edge graph. An RR set R without specifying a root is one with root v selected uniformly at random from V .*

An RR set R_v includes nodes that would activate v in one sample propagation. Then, the key insight is that for a

Algorithm 1 Grad-RIS: Gradient-RIS Meta-Algorithm for CIM-BS.

Input: Directed graph G, triggering model $\{D_v\}_{v\in V}$, domain $\mathcal{P}$, strategy activation functions $\{h_v\}_{v\in V}$, cost function c, budget k, balance parameter λ, Lipschitz constants L_1, L_2 for the functions $g+s$ and $\hat{g}_{\mathcal{R}}+s$, approximation parameter ε, confidence parameter ℓ, gradient algorithm $\mathcal{A}$ with the approximation guarantee α

Output: A strategy mix $\boldsymbol{x}$

1: $\mathcal{R}, LB \leftarrow \mathsf{Sampling}$(all parameters received)
2: $\boldsymbol{x} \leftarrow \mathcal{A}(\mathcal{R}, \hat{g}_{\mathcal{R}}+s, \varepsilon LB)$ /* $\hat{g}_{\mathcal{R}}$ defined in Eq. (2), $s(\boldsymbol{x}) = \lambda(k - c(\boldsymbol{x}))$ */
3: **return** $\boldsymbol{x}$

collection of RR sets, if some node u appears in many of these RR sets, it means u is likely to activate many nodes, and thus has high influence. Technically, RR sets connect with the influence spread of a seed set S with the following equation (Borgs et al. 2014; Tang, Shi, and Xiao 2015): $\sigma(S) = \mathbb{E}_R[\Pr\{S \cap R \neq \emptyset\}]$. For CIM-BS, we have the following connection as given in (Chen, Wu, and Yu 2019):

Lemma 1 ((Chen, Wu, and Yu 2019))**.** *For any strategy $\boldsymbol{x} \in \mathcal{P}$, we have $g(\boldsymbol{x}) = n \cdot \mathbb{E}_R\left[1 - \prod_{u\in R}(1 - h_u(\boldsymbol{x}))\right]$.*

Intuitively, the above lemma means that a node $u \in R$ would activate R's root if u itself is activated, which happens with probability $h_u(x)$, and thus strategy mix successfully activate R's root with probability $1 - \prod_{u\in R}(1 - h_u(\boldsymbol{x}))$. We can generate θ independent RR-sets $\mathcal{R} = \{R_1, \ldots, R_\theta\}$, and take the average among them as defined below:

$$\hat{g}_{\mathcal{R}}(\boldsymbol{x}) = \frac{n}{\theta} \sum_{R\in\mathcal{R}} \left(1 - \prod_{v\in R}(1 - h_v(\boldsymbol{x}))\right). \qquad (2)$$

We can see that $\hat{g}_{\mathcal{R}}(\boldsymbol{x})$ is an unbiased estimator of $g(\boldsymbol{x})$. If θ is large enough, then $\hat{g}_{\mathcal{R}}(\boldsymbol{x})$ should be close to $g(\boldsymbol{x})$ at every $\boldsymbol{x} \in \mathcal{P}$. We also have the following lemma from (Chen, Wu, and Yu 2019).

Lemma 2 ((Chen, Wu, and Yu 2019))**.** *If h_v is monotone and DR-submodular for all $v \in V$, then g and $\hat{g}_{\mathcal{R}}$ are also monotone and DR-submodular.*

3.2 Algorithmic Framework Integrating Gradient Method with RIS

We first introduce the general algorithmic framework that integrates any gradient algorithm with the RIS approach. We assume a generic gradient algorithm $\mathcal{A}$ that takes a set of RR sets $\mathcal{R} = \{R_1, \ldots, R_\theta\}$, an objective function $\hat{g}_{\mathcal{R}} + s$, and an additive error ε as input, and return a solution $\hat{\boldsymbol{x}}$ that guarantees $(\hat{g}_{\mathcal{R}}+s)(\hat{\boldsymbol{x}}) \geq \alpha \cdot \max_{\boldsymbol{y}\in\mathcal{P}}(\hat{g}_{\mathcal{R}}+s)(\boldsymbol{y}) - \varepsilon$, in time $T = \text{poly}(\frac{1}{\varepsilon})$, where α is some constant approximation ratio. We call such an $\mathcal{A}$ an (α, ϵ)-*approximate gradient algorithm*.

Algorithm 1 gives the meta-algorithm. It first calls the Sampling procedure to sample enough RR sets $\mathcal{R}$, together with an estimated lower bound LB of the optimal solution. Then it calls the gradient algorithm $\mathcal{A}$ with $\mathcal{R}$, using $\hat{g}_{\mathcal{R}} + s$ as the objective function and εLB as the additive error.

Algorithm 2 Sampling Procedure.

Input: Same as in Algorithm 1

Output: The RR-sets $\mathcal{R}$ and an estimated lower bound LB

1: $LB \leftarrow 1, \mathcal{R}_0 \leftarrow \phi, \theta_0 = 0, \varepsilon' \leftarrow \sqrt{2}\varepsilon/3$
2: **for** $i = 1, 2, \ldots, \lfloor \log_2(n + \lambda k) - 1 \rfloor$ **do**
3: $\quad x_i \leftarrow (n + \lambda k)/2^i$
4: $\quad \theta_i \leftarrow \lceil n\left(2 + \frac{2}{3}\varepsilon'\right) \cdot$
5: $\qquad \frac{\ln \mathcal{N}(\mathcal{P}, \frac{\varepsilon/3}{L_2} x_i) + \ell \ln n + \ln 2 + \ln \log_2 (n+\lambda k)}{\varepsilon'^2 x_i} \rceil$
6: $\quad$ Generate $\theta_i - \theta_{i-1}$ independent RR-sets $\mathcal{R}'$
7: $\quad \mathcal{R}_i \leftarrow \mathcal{R}' \cup \mathcal{R}_{i-1}$
8: $\quad \boldsymbol{y}_i \leftarrow \mathcal{A}(\mathcal{R}_i, \hat{g}_{\mathcal{R}_i} + s, \varepsilon x_i/3)$
9: $\quad$ **if** $(\hat{g}_{\mathcal{R}_i} + s)(\boldsymbol{y}_i) \geq (1 + \varepsilon' + \varepsilon/3) \cdot x_i$ **then**
10: $\qquad LB \leftarrow \frac{(\hat{g}_{\mathcal{R}_i}+s)(\boldsymbol{y}_i)}{1+\varepsilon'+\varepsilon/3}$; $\theta \leftarrow \theta_i$
11: $\qquad$ **break**
12: $\quad$ **end if**
13: **end for**
14: $\theta^{(1)} = \frac{8n \cdot \ln\left(4n^\ell\right)}{LB \cdot (\alpha - \varepsilon/3)^2 \varepsilon^2/9}$
15: $\theta^{(2)} = \frac{2\alpha' \cdot n \cdot \ln\left(4n^\ell \mathcal{N}(\mathcal{P}, \frac{\varepsilon/3}{L_1+L_2} LB)\right)}{(\varepsilon/3 - \frac{1}{4}(\alpha-\varepsilon/3)^2\varepsilon/3)^2 LB}$
16: $\tilde{\theta} \leftarrow \max\{\theta^{(1)}, \theta^{(2)}\}$.
17: Generate $\tilde{\theta}$ independent RR-sets $R_1, \ldots, R_{\tilde{\theta}}$, $\mathcal{R} \leftarrow \{R_1, \ldots, R_{\tilde{\theta}}\}$
18: **return** $(\mathcal{R}, LB)$

The Sampling procedure is to sample enough RR sets for the theoretical guarantee. We adapt the sampling procedure of the IMM algorithm (Tang, Shi, and Xiao 2015), as shown in Algorithm 2. We use the IMM sampling procedure mainly because of its clarity in analysis and theoretical guarantee, while other sampling procedures (e.g. (Nguyen, Thai, and Dinh 2016)) could be adapted too. The main structure of the sampling procedure is the same as in IMM, where we repeatedly halving the guess x_i of the lower bound LB of the optimal budget-balanced influence spread to find a good lower bound estimate, and then use LB to estimate the final number of RR sets needed and regenerate these RR sets (the regeneration is the workaround 1 proposed in (Chen 2019) to fix a bug in the original IMM). There are two important differences worth to mention. First, in line 8, we call the gradient algorithm $\mathcal{A}$ to find an approximate solution $\boldsymbol{y}_i$, which replaces the original greedy algorithm in IMM. Second, and more importantly, the original IMM algorithm works on a finite solution space — at most $\binom{n}{k}$ feasible seed sets of size k, and $\binom{n}{k}$ is used to bound the number of RR sets needed. However, in CIM-BS, we are working on an infinite solution space, and thus we cannot directly have such a bound. To tackle this problem, we utilize the concept of ε-net and covering number to turn the infinite solution space into a finite space:

Definition 3 (ε-Net and Covering Number)**.** *A finite set N is called an ε-net for $\mathcal{P}$ if for every $\boldsymbol{x} \in \mathcal{P}$, there exists $\pi(\boldsymbol{x}) \in N$ such that $||\boldsymbol{x} - \pi(\boldsymbol{x})||_2 \leq \varepsilon$. The smallest cardinality of an ε-net for $\mathcal{P}$ is called the covering number: $\mathcal{N}(\mathcal{P}, \varepsilon) = \inf\{|N| : N \text{ is an } \varepsilon\text{-net of } \mathcal{P}\}$.*

As a concrete example, suppose we have 1-norm or 2-norm cost function $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1$or $||\boldsymbol{x}||_2$. With budget k, we know that $\mathcal{P}$ is bounded by the ball $\mathbb{B}_1(k)$ or $\mathbb{B}_2(k)$ with radius k. Then, As shown in (van Handel 2014), the covering number satisfies $\mathcal{N}(\mathcal{P}, \varepsilon) \leq \mathcal{N}(\mathbb{B}_1(k), \varepsilon) \leq \mathcal{N}(\mathbb{B}_2(k), \varepsilon) \leq (3k/\varepsilon)^d$.

Besides the ε-net, we also need to have the upper bounds L_1 and L_2 on the Lipschitz constants of functions $g + s$ and $\hat{g}_{\mathcal{R}} + s$. We defer the discussion on L_1 and L_2 to the next subsection. Covering number and L_1, L_2 together are used to bound the number of RR sets needed, as used in lines 4 and 15 of Algorithm 2. We denote Algorithms 1 and 2 together as Grad-RIS, and we show that Grad-RIS achieves the following approximation guarantee:

Theorem 1. *For any $\varepsilon, \ell, \alpha > 0$, for any $(\alpha, \varepsilon/3)$-approximate gradient algorithm $\mathcal{A}$ for $\hat{g}_{\mathcal{R}} + s$, with probability at least $1 - \frac{1}{n^\ell}$,* Grad-RIS *outputs a solution $\boldsymbol{x}$ that is an $(\alpha - \varepsilon)$-approximation of the optimal solution OPT_{g+s} of CIM-BS, i.e. $(g + s)(\boldsymbol{x}) \geq (\alpha - \varepsilon)OPT_{g+s}$.*

The proof of the above theorem follows the proof structure of IMM (Tang, Shi, and Xiao 2015), where the number of the RR sets needed is carefully adapted to accommodate the covering number of the ε-net, and the Lipschitz constants of the objective functions.

3.3 Gradient Algorithms

In this subsection, we will show two gradient algorithms that approximately maximize the function $\hat{g}_{\mathcal{R}}(\boldsymbol{x}) + s(\boldsymbol{x})$. They are the instantiations of the generic algorithm $\mathcal{A}$ in Section 3.2: the first one works on the general model and uses proximal gradient to achieve $(\frac{1}{2}, \varepsilon)$-approximation, while the second one works on the special case of independent strategy activation and uses gradient on a concave upper bound to achieve $(1 - \frac{1}{e}, \varepsilon)$-approximation.

General Case: ProxGrad-RIS**.** We first consider the general case where the strategy activation functions h_v's are monotone, DR-submodular, L_h-Lipschitz and β_h-smooth, and the cost function c is convex and L_c-Lipschitz. In this case, we have that g and $\hat{g}_{\mathcal{R}}$ are monotone and DR-submodular (Lemma 2), and budget-saving function s is concave. To solve this problem, we adapt the (stochastic) proximal gradient algorithm (Parikh, Boyd, and others 2014; Nitanda 2014) to provide a $\frac{1}{2}$-approximate solution to the following problem: given a convex set $\mathcal{P}$, a β-smooth, non-negative, monotone, and DR-submodular function $f_1(\boldsymbol{x})$ on $\mathcal{P}$ and a non-negative and concave function $f_2(\boldsymbol{x})$ on $\mathcal{P}$, find a solution in $\mathcal{P}$ maximizing $f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})$. The original proximal gradient is for the case when both f_1 and f_2 are concave, and we adapt it to the case when f_1 is monotone and DR-submodular to provide an approximate solution. The reason we use proximal gradient is that our budget-saving function s may not be smooth (e.g. when the cost function is the 2-norm function). We present the general solution first, since it may be of independent interest. The following is the iteration procedure for the stochastic proximal gradient algorithm.

$$\begin{cases} \boldsymbol{x}^{(t+1)} = \text{prox}_{-\eta_t f_2}(\boldsymbol{x}^{(t)} + \eta_t \boldsymbol{v}^{(t)}), \\ \quad \text{where } \mathbb{E}[\boldsymbol{v}^{(t)}] = \nabla f_1(\boldsymbol{x}^{(t)}), \\ \text{prox}_{\phi}(\boldsymbol{x}) := \text{argmin}_{\boldsymbol{y} \in \mathcal{P}}(\phi(\boldsymbol{y}) + \frac{1}{2}||\boldsymbol{x} - \boldsymbol{y}||_2^2), \\ \quad \text{for any convex function } \phi, \end{cases} \tag{3}$$

where $\eta_t \leq \frac{1}{\beta}$ is the step size and $\boldsymbol{v}^{(t)}$ is the stochastic gradient at $\boldsymbol{x}^{(t)}$. We use Δ to denote an upper bound of the diameter of $\mathcal{P}$, i.e. $\Delta \geq \max_{\boldsymbol{x}, \boldsymbol{y} \in \mathcal{P}} ||\boldsymbol{x} - \boldsymbol{y}||_2$. The following is the main result for the above stochastic proximal gradient algorithm, with its proof adapted from the original proof.

Theorem 2. *Suppose that $\mathcal{P}$ is a convex set, function $f_1(\boldsymbol{x})$ is β-smooth, non-negative, monotone, and DR-submodular on $\mathcal{P}$, $f_2(\boldsymbol{x})$ is non-negative and concave on $\mathcal{P}$. Let $\boldsymbol{x}^*$ be the point that maximizes $f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})$. Suppose that for some $\sigma > 0$, the stochastic gradient $\boldsymbol{v}^{(t)}$ satisfies $\mathbb{E}||\boldsymbol{v}^{(t)} - \nabla f_1(\boldsymbol{x}^{(t)})||_2^2 \leq \sigma^2$ for all t, then for all $T > 0$, if we set $\eta_t = \eta = 1/(\beta + \frac{\sigma}{\Delta}\sqrt{2T})$, and iterate as shown in (3) starting from $\boldsymbol{x}^{(0)} \in \mathcal{P}$, we have*

$$\mathbb{E}\left[\max_{t=0,1,2,\ldots,T}(f_1 + f_2)(\boldsymbol{x}^{(t)})\right] \geq \frac{1}{2}(f_1 + f_2)(\boldsymbol{x}^*) - \frac{\beta\Delta^2}{4T} - \frac{\sigma\Delta}{\sqrt{2T}}.$$

Note that if we use exact gradient instead of the stochastic gradient, we simply set $\sigma = 0$ in the above theorem. To apply the proximal gradient algorithm and Theorem 2 to maximize $\hat{g}_{\mathcal{R}} + s$, we compute the exact gradient of $\hat{g}_{\mathcal{R}}$ and also derive the Lipschitz and smoothness constants, as shown below. For RR set sequence $\mathcal{R} = \{R_1, \ldots, R_\theta\}$, let $\nu^{(1)}(\mathcal{R}) = \sum_{R \in \mathcal{R}} |R|/\theta$ be the average RR set size in $\mathcal{R}$, $\nu^{(2)}(\mathcal{R}) = \sum_{R \in \mathcal{R}} |R|^2/\theta$ be the average squared size in $\mathcal{R}$, and $\nu^{(3)}(\mathcal{R}) = \sum_{R \in \mathcal{R}} |R|^3/\theta$ be the average cubed size.

Lemma 3. *If functions $h_v(\boldsymbol{x})$'s are L_h-Lipschitz, then function $\hat{g}_{\mathcal{R}}(\boldsymbol{x})$ is $(\nu^{(1)}(\mathcal{R})nL_h)$-Lipschitz, and function $g(\boldsymbol{x})$ is $(n^2 L_h)$-Lipschitz. If functions $h_v(\boldsymbol{x})$'s are β_h-smooth, then function $\hat{g}_{\mathcal{R}}(\boldsymbol{x})$ is $(\nu^{(1)}(\mathcal{R})n\beta_h + \nu^{(2)}(\mathcal{R})nL_h^2)$-smooth. The gradient of function $\hat{g}_{\mathcal{R}}(\boldsymbol{x})$ is*

$$\nabla \hat{g}_{\mathcal{R}}(\boldsymbol{x}) = \frac{n}{\theta} \sum_{R \in \mathcal{R}, v' \in R} \nabla h_{v'}(\boldsymbol{x}) \prod_{v \in R, v \neq v'} (1 - h_v(\boldsymbol{x})). \tag{4}$$

With Lemma 3 and Theorem 2, we can conclude the gradient algorithm $\mathcal{A}$ working with Grad-RIS with the following settings: (a) we use the proximal gradient iteration given in Eq. (3), with stochastic gradient $\boldsymbol{v}^{(t)}$ replaced with the exact gradient $\nabla \hat{g}_{\mathcal{R}}(\boldsymbol{x}^{(t)})$ as given in Eq. (4); (b) we set step size $\eta_t = 1/(\nu^{(1)}(\mathcal{R})n\beta_h + \nu^{(2)}(\mathcal{R})nL_h^2)$ when calling the algorithm with $\mathcal{R}$; (c) we set number of steps $T = 3(\nu^{(1)}(\mathcal{R})n\beta_h + \nu^{(2)}(\mathcal{R})nL_h^2) \cdot \Delta^2/4\varepsilon$; (d) we use $L_1 = L_2 = n^2 L_h + \lambda L_c$ (since $n \geq \nu^{(1)}(\mathcal{R})$) as parameters in Grad-RIS. We refer to the full algorithm with the above setting as ProxGrad-RIS. The following theorem summarizes the approximation guarantee of ProxGrad-RIS.

Theorem 3. *For any $\varepsilon, \ell > 0$, with probability at least $1 - \frac{1}{n^\ell}$,* ProxGrad-RIS *outputs a solution $\boldsymbol{x}$ that is a $(\frac{1}{2} - \varepsilon)$-approximation of the optimal solution OPT_{g+s} of CIM-BS, i.e. $(g+s)(\boldsymbol{x}) \geq (\frac{1}{2} - \varepsilon)OPT_{g+s}$.*

We remark that the actual computation of the proximal step $\text{prox}_{-\eta_t f_2}(\cdot)$ in Eq.(3) depends on domain $\mathcal{D}$ and cost function c. When $\mathcal{D} = \mathbb{R}_+^d$ and c is 1-norm or 2-norm function, we can derive efficient algorithm for the proximal step, as summarized below.

Lemma 4. *When $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1$ and $\mathcal{D} = \mathbb{R}_+^d$, the proximal step can be done in time $O(d \log d)$. When $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2$ and $\mathcal{D} = \mathbb{R}_+^d$, the proximal step can be done in $O(d)$.*

Independent Strategy Activation Case: UpperGrad-RIS**.** Next, we introduce an $\left(1 - \frac{1}{e}\right)$-approximation for maximizing $\hat{g}_\mathcal{R} + s$ under the case of independent strategy activation. Recall that in the independent strategy activation case, each function $h_v(\boldsymbol{x}) = 1 - \prod_{j \in [d]}(1 - q_{v,j}(x_j))$, where $q_{v,j}(x_j)$ is monotone and concave in x_j. In this case, we can write $\hat{g}(\boldsymbol{x})$ into the following form.

$$\hat{g}_\mathcal{R}(\boldsymbol{x}) = \frac{n}{\theta} \sum_{R \in \mathcal{R}} \left(1 - \prod_{v \in R}(1 - h_v(\boldsymbol{x}))\right)$$

$$= \frac{n}{\theta} \sum_{R \in \mathcal{R}} \left(1 - \prod_{v \in R} \left(\prod_{j \in [d]}(1 - q_{v,j}(\boldsymbol{x}))\right)\right).$$

The above form makes $\hat{g}_\mathcal{R}(\boldsymbol{x})$ belong to coverage functions, which has the following concave upper and lower bounds ((Karimi et al. 2017a)):

Proposition 1 ((Karimi et al. 2017a))**.** *For any $\boldsymbol{x} \in [0,1]^l$, let $\alpha(\boldsymbol{x}) = 1 - \prod_{i=1}^l (1 - x_i)$ and $\beta(\boldsymbol{x}) = \min\{1, \sum_{i=1}^l x_i\}$, we have $(1 - 1/e)\beta(\boldsymbol{x}) \leq \alpha(\boldsymbol{x}) \leq \beta(\boldsymbol{x})$.*

By Proposition 1, we can optimize $\bar{g}_\mathcal{R} + s$ where $\bar{g}_\mathcal{R}(\boldsymbol{x})$ is the upper bound of $\hat{g}_\mathcal{R}(\boldsymbol{x})$ defined as:

$$\bar{g}_\mathcal{R}(\boldsymbol{x}) := \frac{n}{\theta} \sum_{R \in \mathcal{R}} \min\{1, \sum_{j \in [d], v \in R} q_{v,j}(\boldsymbol{x})\}. \tag{5}$$

Since the function $g_\mathcal{R}(\boldsymbol{x})$ is non-smooth, we use the projected subgradient method to maximize the function $\bar{g}_\mathcal{R} + s$ (Nesterov 2013), as summarized by the following lemma.

Lemma 5. *In the case of independent strategy activation, suppose that $\bar{g}_\mathcal{R} + s$ is $L_{\bar{g}+s}$-Lipschitz. If we use projected subgradient descent to optimize the function $(\bar{g}_\mathcal{R} + s)(\boldsymbol{x})$ with step size $\eta_t = \frac{\Delta}{L_{\bar{g}+s}\sqrt{t}}$ and let $\boldsymbol{y}$ denote the output where $(\bar{g}_\mathcal{R}+s)(\boldsymbol{y}) \geq \max_{\boldsymbol{x} \in \mathcal{P}}(\bar{g}_\mathcal{R}+s)(\boldsymbol{x}) - \varepsilon$. Then $(\hat{g}_\mathcal{R}+s)(\boldsymbol{y}) \geq \left(1 - \frac{1}{e}\right)\max_{\boldsymbol{x} \in \mathcal{P}}(\hat{g}_\mathcal{R} + s)(\boldsymbol{x}) - \varepsilon$, and $\boldsymbol{y}$ can be solved in $\frac{(\Delta L_{\bar{g}+s})^2}{\varepsilon^2}$ iterations.*

The following lemma presents the Lipschitz constants and subgradients needed in Lemma 5.

Lemma 6. *Suppose that functions $q_{v,j}(x_j)$'s are L_q-Lipschitz, then function $g(\boldsymbol{x})$ is $n^2\sqrt{d}L_q$-Lipschitz, and functions $\hat{g}_\mathcal{R}(\boldsymbol{x})$ and $\bar{g}_\mathcal{R}(\boldsymbol{x})$ are $\nu^{(1)}(\mathcal{R})n\sqrt{d}L_q$-Lipschitz. The subgradient of the function $\bar{g}_\mathcal{R}(\boldsymbol{x})$ is*

$$\frac{n}{\theta} \sum_{R \in \mathcal{R}} \begin{cases} 0, \text{ if } \sum_{j \in [d], v \in R} q_{v,j}(x_j) \geq 1, \\ \sum_{v \in R, j \in [d]} \nabla q_{v,j}(x_j), \text{ if } \sum_{j \in [d], v \in R} q_{v,j}(x_j) < 1. \end{cases} \tag{6}$$

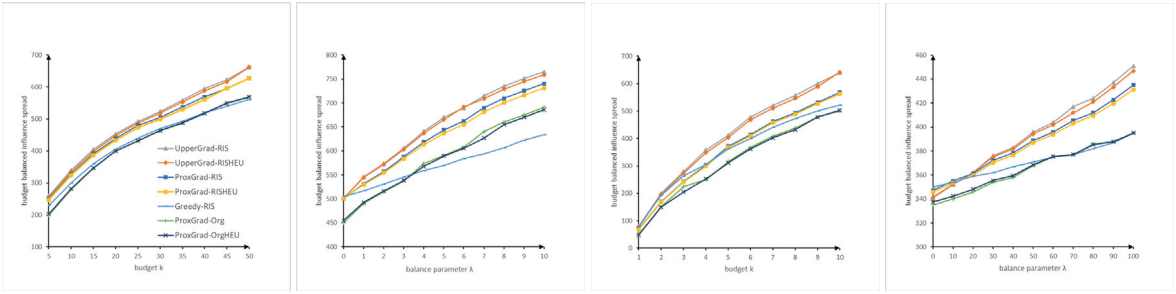
Combining Lemma 6 with Lemma 5, we can conclude our subgradient algorithm based on the upper bound function $\bar{g}_\mathcal{R} + s$: (a) we use the projected subgradient algorithm with the subgradient of $\bar{g}_\mathcal{R}$ given in Eq.(6); (b) we set step size $\eta_t = \Delta/(\nu^{(1)}(\mathcal{R})n\sqrt{d}L_q\sqrt{t})$; (c) we use $T = 9(\Delta\nu^{(1)}(\mathcal{R})n\sqrt{d}L_q + \lambda L_c)^2/\varepsilon^2$ iterations to get ε accuracy; and (d) we set $L_1 = L_2 = n^2\sqrt{d}L_q + \lambda L_c$ in Grad-RIS. We refer to the full algorithm with the above setting as UpperGrad-RIS. The following theorem summarizes the approximation guarantee of UpperGrad-RIS.

Theorem 4. *For any $\varepsilon, \ell > 0$, with probability at least $1 - \frac{1}{n^\ell}$,* UpperGrad-RIS *outputs a solution $\boldsymbol{x}$ that is an $(1 - 1/e - \varepsilon)$-approximation of the optimal solution OPT_{g+s} of CIM-BS, i.e. $(g+s)(\boldsymbol{x}) \geq (1 - 1/e - \varepsilon)OPT_{g+s}$.*

Total Time Complexity. For the time complexity of ProxGrad-RIS and UpperGrad-RIS, we make the following reasonable assumptions: (1) the time for sampling a trigger set $T_v \sim D_v$ is proportional to the in-degree of v; (2) the optimal influence spread $\max_{\boldsymbol{x} \in \mathcal{P}} g(\boldsymbol{x})$ among strategy mixes is at least the optimal single node influence spread $\max_{v \in V} \sigma(\{v\})$; and (3) $\lambda k \leq n$, otherwise the budget saving is more important than influencing the entire network, and CIM-BS problem no longer makes much sense. The following theorem summarizes the time complexity result when $\mathcal{D} = \mathbb{R}_+^d$ and $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1$ or $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2$. The more general result is given in (Chen, Zhang, and Zhao 2019). Notation $\tilde{O}(\cdot)$ ignores poly-logarithmic factors.

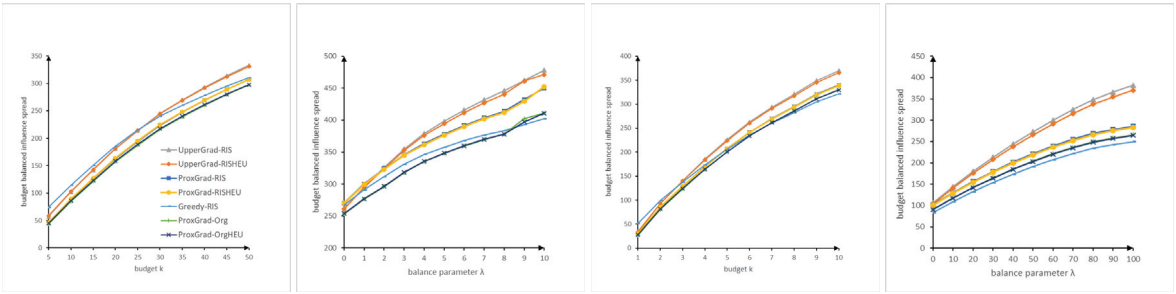
Theorem 5. *Suppose that $\mathcal{D} = \mathbb{R}_+^d$ and $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1$ or $||\boldsymbol{x}||_2$, $h_v(\boldsymbol{x})$'s are L_h-Lipschitz and β_h-smooth. If $\nabla h_v(\boldsymbol{x})$ can be computed in time T_h, the expected running time of* ProxGrad-RIS *is bounded by $\tilde{O}\left(\frac{\beta_h n^2 + L_h^2 n^3}{\varepsilon} \cdot \frac{T_h(m+n)\cdot(d+\ell)}{\varepsilon^2}\right)$. Under independent strategy activation, if $q_{v,j}(x_j)$'s are L_q-Lipschitz and the gradient and function value of $q_{v,l}(x_j)$ can be computed in time T_q, the expected running time of* UpperGrad-RIS *is bounded by $\tilde{O}\left(\frac{n^4 d L_q^2}{\varepsilon^2} \cdot \frac{T_q(m+n)\cdot(d+\ell)}{\varepsilon^2}\right)$.*

From the time complexity result, we can see that the two gradient algorithms still have high-order dependency on the graph size. This is mainly because we need the conservative bounds on the number of gradient algorithm iterations for the theoretical guarantee (terms $\frac{\beta_h n^2 + L_h^2 n^3}{\varepsilon}$ and $\frac{n^4 d L_q^2}{\varepsilon^2}$). In our actual algorithms, we already use $\nu^{(1)}(\mathcal{R})$ and $\nu^{(2)}(\mathcal{R})$ instead of n and n^2 in the upper bound of the gradient decent steps for $\hat{g}_\mathcal{R} + s$, so our actual performance would be reduced by corresponding factors. For details, please see (Chen, Zhang, and Zhao 2019) for the results and discussions on using the moments of RR set size in the time complexity bounds.



(a) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1, \lambda = 5$ (b) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1, k = 50$ (c) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2, \lambda = 50$ (d) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2, k = 5$

Figure 1: Budget-balanced influence spread results for the personalized marketing scenario. Legends in (a) apply to all figures.



(a) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1, \lambda = 5$ (b) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_1, k = 50$ (c) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2, \lambda = 50$ (d) $c(\boldsymbol{x}) = ||\boldsymbol{x}||_2, k = 5$

Figure 2: Budget-balanced influence spread results for the segment marketing scenario. Legends in (a) apply to all figures.

4 Experiments

Experiment setup. We test on the DM network, which is a network of data mining researchers extracted from the Arnet-Miner archive (arnetminer.org), with 679 nodes and 3, 374 edges, and edge weights are learned from a topic affinity model and obtained from the authors (Tang et al. 2009).

Besides our ProxGrad-RIS and UpperGrad-RIS algorithms, we test two more algorithms: (a) ProxGrad-Org: stochastic proximal gradient algorithm on the original objective function (see Appendix D in (Chen, Zhang, and Zhao 2019)); (b) Greedy-RIS: simply replace the gradient algorithm $\mathcal{A}$ in Grad-RIS with the greedy algorithm for the objective $\hat{g}_{\mathcal{R}}(\boldsymbol{x}) + s(\boldsymbol{x})$ on generated RR sets $\mathcal{R}$, and the greedy algorithm stops either when the budget is exhausted or the marginal gain is negative. For the three gradient-based algorithms ProxGrad-RIS, UpperGrad-RIS, and ProxGrad-Org, we further test their heuristic versions that may lead to faster running time: instead of using a conservative number of iteration steps for theoretical guarantees, we heuristically terminate the gradient iteration if the difference in the objective function values for two consecutive iterations is within a small value of 0.3 (we justify the choice of this parameter in our full report). We put suffix HEU for the three versions of the heuristic gradient termination algorithms.

For parameter settings, we set $\varepsilon = 0.1$ and $\ell = 1$ for all algorithms. For Greedy-RIS, we set the greedy step size to be 0.1 on each dimension. For 1-norm cost function ($c(\boldsymbol{x}) = ||\boldsymbol{x}||_1$), we test (a) vary k from 5 to 50 while keeping $\lambda = 5$, and (b) vary λ from 0 to 10 while keeping $k = 50$. For 2-norm cost function ($c(\boldsymbol{x}) = ||\boldsymbol{x}||_2$), we test (c) varying vary k from 1 to 10 while keeping $\lambda = 50$, and (d) vary λ from 0 to 100 while keeping $k = 5$. The reason we use a smaller budget k for 2-norm cost function is because $||\boldsymbol{x}||_2 \leq ||\boldsymbol{x}||_1/\sqrt{d}$, and thus we need a significantly small budget for 2-norm in order to have a similar feasible region. Parameter λ is adjusted accordingly so that $\lambda \cdot k$ is at the same scale as the influence spread, otherwise either influence spread or budget saving is dominant, and the problem is degenerated. We test two cases: the personalized marketing case and the segment marketing case (Yang et al. 2016; Chen, Wu, and Yu 2019).

Experimental results. Figure 1 and 2 show the budget-balanced influence spread results of the personalized marketing and the segment marketing scenarios respectively. Each data point on an curve is the average of five solutions of five runs of the same algorithm, and the influence spread of each solution is an average of 1000 simulation runs. In all cases, UpperGrad-RIS/UpperGrad-RISHEU have the best performance, followed by ProxGrad-RIS/ProxGrad-RISHEU, which coincides with our theoretical analysis that UpperGrad-RIS has a better theoretical guarantee. Both algorithms outperform two base lines in most cases, especially when λ is getting large, indicating that our algorithms handle the balance between budget saving and influence spread better.

Table 1 shows the running time of the personalized marketing scenario. Each running time number is the average of five runs. The result shows that ProxGrad-RIS and UpperGrad-RIS are 30+ times faster than ProxGrad-Org. This is mainly due to the high variance in the stochastic gradient for the original objective function, as we discussed before.

Table 1: Running time results for the personalized marketing scenario (in seconds).

	$c(\boldsymbol{x}) = \|\|\boldsymbol{x}\|\|_1$, $k = 50$ and $\lambda = 5$	$c(\boldsymbol{x}) = \|\|\boldsymbol{x}\|\|_2$, $k = 5$ and $\lambda = 50$
ProxGrad-RIS	33.2	27.3
ProxGrad-RISHEU	6.5	5.5
UpperGrad-RIS	81.8	72.3
UpperGrad-RISHEU	13.2	13.9
Greedy-RIS	10.2	8.7
ProxGrad-Org	1043.9	1021.6
ProxGrad-OrgHEU	243.4	187.8

Moreover, ProxGrad-RIS and UpperGrad-RIS is slower than Greedy-RIS. This is mainly because our conservative bounds on the number of gradient iterations make ProxGrad-RIS and UpperGrad-RIS slow, while Greedy-RIS only use the heuristic greedy approach with step size 0.1 without any theoretical guarantee. But when we apply heuristic termination, UpperGrad-RISHEU is close to Greedy-RIS while ProxGrad-RISHEU becomes faster than Greedy-RIS. Therefore, our gradient-based algorithms could achieve faster running time with heuristic termination while still providing better influence spread quality than the greedy heuristic. Running time result on the segment marketing scenario is similar.

Due to the space constraint, more detailed experiment setup and results including a new dataset are reported in the full technical report (Chen, Zhang, and Zhao 2019).

5 Conclusion and Further Work

In this paper, we tackle the new problem of continuous influence maximization with budget saving (CIM-BS), whose objective function is neither monotone, nor DR-submodular or concave. We use the gradient method to solve CIM-BS, and provide innovative integration with the reverse influence sampling method to achieve theoretical approximation guarantees. One important direction of future study is to make the gradient method more scalable, which requires more detailed study of convergence behavior and properties of the gradient method in the influence maximization domain. Another direction is to investigate if the gradient method can be applied to other influence maximization settings such as competitive influence maximization. Gradient method is a rich and powerful approach that has been already applied to many application domains, and thus we hope our work could inspire more studies incorporating the gradient method into the influence maximization research.

References

Bian, A. A.; Mirzasoleiman, B.; Buhmann, J. M.; and Krause, A. 2017. Guaranteed non-convex optimization: Submodular maximization over continuous domains. In *AISTATS*.

Borgs, C.; Brautbar, M.; Chayes, J.; and Lucier, B. 2014. Maximizing social influence in nearly optimal time. In *SODA*.

Chen, W.; Lakshmanan, L. V.; and Castillo, C. 2013. *Information and Influence Propagation in Social Networks*. Morgan & Claypool Publishers.

Chen, W.; Wu, R.; and Yu, Z. 2019. Scalable lattice influence maximization. Technical Report arXiv:1802.04555v2, arXiv.

Chen, W.; Zhang, W.; and Zhao, H. 2019. Gradient method for continuous influence maximization with budget-saving considerations. available at shorturl.at/cwBNS, to appear in arXiv.

Chen, W. 2019. An issue in the martingale analysis of the influence maximization algorithm imm. In *CSoNet*.

Hassani, S. H.; Soltanolkotabi, M.; and Karbasi, A. 2017. Gradient methods for submodular maximization. In *NIPS*.

Karimi, M.; Lucic, M.; Hassani, H.; and Krause, A. 2017a. Stochastic submodular maximization: The case of coverage functions. In *NIPS*.

Karimi, M. R.; Lucic, M.; Hassani, S. H.; and Krause, A. 2017b. Stochastic submodular maximization: The case of coverage functions. In *NIPS*, 6856–6866.

Kempe, D.; Kleinberg, J. M.; and Tardos, É. 2015. Maximizing the spread of influence through a social network. *Theory of Computing* 11(4):105–147. First appeared in KDD'03.

Li, Y.; Fan, J.; Wang, Y.; and Tan, K. 2018. Influence maximization on social graphs: A survey. *IEEE Trans. Knowl. Data Eng.* 30(10):1852–1872.

Lu, W., and Lakshmanan, L. V. S. 2012. Profit maximization over social networks. In *ICDM*, 479–488.

Mankiw, N. G. 2014. *Principles of economics*. Cengage Learning.

Mokhtari, A.; Hassani, H.; and Karbasi, A. 2018. Conditional gradient method for stochastic submodular maximization: Closing the gap. In *AISTATS*.

Nesterov, Y. 2013. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Nguyen, H. T.; Thai, M. T.; and Dinh, T. N. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *SIGMOD*, 695–710.

Nitanda, A. 2014. Stochastic proximal gradient descent with acceleration techniques. In *NIPS*.

Parikh, N.; Boyd, S.; et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1(3):127–239.

Tang, J.; Sun, J.; Wang, C.; and Yang, Z. 2009. Social influence analysis in large-scale networks. In *KDD*.

Tang, Y.; Shi, Y.; and Xiao, X. 2015. Influence maximization in near-linear time: a martingale approach. In *SIGMOD*.

Tang, J.; Tang, X.; and Yuan, J. 2016. Profit maximization for viral marketing in online social networks. In *ICNP*.

Tang, Y.; Xiao, X.; and Shi, Y. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*.

van Handel, R. 2014. Probability in high dimension. Technical report, Princeton University.

Yang, Y.; Mao, X.; Pei, J.; and He, X. 2016. Continuous influence maximization: What discounts should we offer to social network users? In *SIGMOD*, 727–741.