Mode finding for SLC Distributions via Regularized Submodular Maximization

Ehsan Kazemi [1 2] **Shervin Minaee** [3] **Moran Feldman** [4] **Amin Karbasi** [1]

Abstract

In this paper, we propose scalable methods for maximizing a regularized submodular function $f = g - \ell$ expressed as the difference between a monotone submodular function g and a modular function ℓ. Indeed, submodularity is inherently related to the notions of diversity, coverage, and representativeness. In particular, finding the mode (i.e., the most likely configuration) of many popular probabilistic models of diversity, such as determinantal point processes, submodular probabilistic models, and strongly log-concave distributions, involves maximization of (regularized) submodular functions. Since a regularized function f can potentially take on negative values, the classic theory of submodular maximization, which heavily relies on the non-negativity assumption of submodular functions, may not be applicable. To circumvent this challenge, we develop the first streaming and distributed algorithm for maximizing regularized submodular functions. Furthermore, we show that how can we find the mode of a strongly log-concave (SLC) distribution by regularized submodular maximization.

1. Introduction

Finding a diverse set of items, also known as data summarization, is one of the central tasks in machine learning. It usually involves either maximizing a utility function that promotes coverage and representativeness (Wei et al., 2015) (we call this an optimization perspective) or sampling from discrete probabilistic models that promote negative correlations and show repulsive behaviors (Gotovos et al., 2015) (we call this a sampling perspective). Celebrated examples of probabilistic models that encourage negative dependency include determinantal point processes (Kulesza & Taskar, 2012), strongly Rayleigh measures (Borcea et al., 2009) strongly log-concave distributions (Gurvits, 2009), and probabilistic submodular models (Djolonga & Krause, 2014; Iyer & Bilmes, 2015). In fact, the two above views are tightly related in the sense that oftentimes the mode (i.e., most likely configuration) of a diversity promoting distribution is a simple variant of a (regularized) submodular function. For instance, determinantal point processes are log-submodular. Or, as we show later, a strongly log-concave distribution is indeed a regularized log-submodular plus a log quadratic term. The aim of this paper is to show how such an optimization task can be done at scale.

From the optimization perspective, to effectively select a diverse subset of items, we need to define a measure that captures the amount of representativeness that lies within a selected subset. Oftentimes, such a measure naturally satisfies the intuitive diminishing returns condition which can be formally captured by *submodularity*. Given a ground set $\mathcal{N}$ of size n, consider a set function $g\colon 2^{\mathcal{N}} \to \mathbb{R}$ assigning a utility $g(A)$ to every subset $A \subseteq \mathcal{N}$. We say that g is submodular if for any pair of subsets $A \subseteq B \subseteq \mathcal{N}$ and an element $u \notin B$, we have $g(A \cup \{u\}) - g(A) \geq g(B \cup \{u\}) - g(B)$, which intuitively means that the increase in "representativeness" following the addition of an element u is smaller when u is added to a larger set. Additionally, a set function is said to be monotone if $g(A) \leq g(B)$ for all $A \subseteq B \subseteq \mathcal{N}$; that is, adding more data can only increase the representativeness of any subset. Many of the previous work in data summarization and diverse subset selection that take an optimization perspective, simply aim to maximize a monotone submodular function (Mirzasoleiman et al., 2013). Monotonicity has the advantage of promoting coverage, but it also enhances the danger of over-fitting to the data as adding more elements can never decrease the utility. To address this issue, as it is often done in machine learning, we need to add a simple penalty or a regularizer term. Formally, we cast this optimization problem as an instance of **regularized submodular** maximization, in which we are asked to find a set S of size at most k that maximizes

$$OPT = \underset{S \subseteq \mathcal{N}, |S| \leq k}{\arg\max} \left[g(S) - \ell(S) \right] , \tag{1}$$

where g is a non-negative monotone submodular function and ℓ is a non-negative modular function.[1] The role of the

[1]Yale Institute for Network Science, Yale University [2]Now at Google, Zürich, Switzerland [3]Expedia Group, Seattle, WA [4]Department of Computer Science, University of Haifa, Israel. Correspondence to: Ehsan Kazemi <ehsan.kazemi@yale.edu>.

ICML 2020 workshop on Negative Dependence and Submodularity for ML, Vienna, Austria, PMLR 119, 2020.

[1]A set function $\ell\colon 2^{\mathcal{N}} \to \mathbb{R}$ is modular if there is a value ℓ_u for every $u \in \ell$ such that $\ell(S) = \sum_{u \in S} \ell_u$ for every set $S \subseteq \mathcal{N}$.

modular function ℓ is to discount the benefit of adding elements. We highlight that $g - \ell$ is still submodular. However, it may no longer be non-negative, an assumption that is essential for deriving competitive algorithms with constant-factor approximation guarantees (for more information, see the survey by (Buchbinder & Feldman, 2018)).

In many practical scenarios, random access to the entire data is not possible and only a small fraction of the data can be loaded to the main memory, there is only time to read the data once, and the data arrives at a very fast pace. Furthermore, the amount of collected data is often too large to solve the optimization problem on a single machine. Prior to this work, no streaming or distributed algorithm to solve Problem (1) was established. However, based on ideas from (Sviridenko et al., 2017; Feldman, 2019), Harshaw et al. (2019) proposed DISTORTED-GREEDY, an efficient offline algorithm to (approximately) solve this problem. This algorithm iteratively and greedily finds elements that maximize a distorted function. On the other hand, DISTORTED-GREEDY, as a centralized algorithm, requires a memory that grows linearly with the size of the data, and it needs to make multiple passes ($\Theta(n)$ in the worst case) over the data; therefore it fails to satisfy the above mentioned requirements of modern applications. In this paper, we propose **scalable methods** (in both streaming and distributed settings) for maximizing a **regularized** submodular function. In the following, we explain our main theoretical results.

Our Results. In Section 2, we introduce DISTORTED-STREAMING, the first one-pass streaming algorithm for maximizing a regularized submodular function subject to a k-cardinality constraint. Theorem 1 guarantees the performance of THRESHOLD-STREAMING.

Theorem 1. *For every $\varepsilon, r > 0$,* THRESHOLD-STREAMING *produces a set $S \subseteq \mathcal{N}$ of size at most k for Problem (1), obeying $g(S) - \ell(S) \geq \max_{T \subseteq \mathcal{N}, |T| \leq k} [(h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)]$, where $h(r) = \frac{2r+1-\sqrt{4r^2+1}}{2}$.*

We should point out that previous studies of Problem (1), for various theoretical and practical reasons, have only focused on the case in which $r = 1$ and T is the set OPT of size at most k maximizing $g(T) - \ell(T)$ (Sviridenko et al., 2017; Feldman, 2019; Harshaw et al., 2019). In this case, we get the following corollary from the result of Theorem 1.

Corollary 2. *For every $\varepsilon > 0$,* THRESHOLD-STREAMING *produces a set $S \subseteq \mathcal{N}$ of size at most k for Problem (1) obeying $g(S) - \ell(S) \geq (\phi^{-2} - \varepsilon) \cdot g(OPT) - \ell(OPT)$, where ϕ is the golden ratio (and thus, $\phi^{-2} \approx 0.382$).*

In Appendix C, we develop DISTORTED-DISTRIBUTED-GREEDY, the first distributed algorithm for regularized submodular maximization in a MapReduce model of computation. This algorithm allows us to distribute data across several machines and use their combined computational resources. For the classic case of an unregulated monotone submodular function, our distributed algorithm improves over the space and communication complexity of the existing work (Barbosa et al., 2016) by a factor of $\Theta(1/\varepsilon)$. The approximation guarantee of DISTORTED-DISTRIBUTED-GREEDY is given in Theorem 3.

Theorem 3. DISTORTED-DISTRIBUTED-GREEDY *(Algorithm 4) returns a set $D \subseteq \mathcal{N}$ of size at most k such that*

$$\frac{\mathbb{E}[g(D) - \ell(D)]}{1 - \varepsilon} \geq (1 - e^{-1}) \cdot g(OPT) - \ell(OPT).$$

Finally, as our algorithms can efficiently find diverse elements from massive datasets, in Section 3, we explore the power of the regularized submodular maximization on finding the mode of SLC distributions.

2. Streaming Algorithm

In this section, we present our proposed streaming algorithm for Problem (1). To explain our algorithm, let us first define T to be a subset of $\mathcal{N}$ of size at most k such that

$$T \in \operatorname*{arg\,max}_{S \subseteq \mathcal{N}, |S| \leq k} \left[(h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)\right] ,$$

where r is some positive real value to be discussed later, and $h(r) = \frac{2r+1-\sqrt{4r^2+1}}{2}$. A basic version of our proposed algorithm, named THRESHOLD-STREAMING, is given as Algorithm 1. We note that this algorithm guesses, in the first step, a value $\tau > 0$ which obeys $k\tau \leq h(r) \cdot g(T) - r \cdot \ell(T) \leq (1 + \varepsilon)k\tau$. In Algorithm 1, to avoid unnecessary technicalities, we simply assume that the algorithm can guess such a value for τ based on some oracle. In Appendix B, we explain how a technique from (Badanidiyuru et al., 2014) can be used for that purpose at the cost of increasing the space complexity of the algorithm by a factor of $O(\varepsilon^{-1}(\log k + \log r^{-1}))$. Algorithm 1 starts with an empty set S. While the data stream is not empty yet and the size of set S is still smaller than k, for every incoming element u, the value of $g(u \mid S) - \alpha(r) \cdot \ell(\{u\})$ is calculated, where we define $\alpha(r) = \frac{2r+1+\sqrt{4r^2+1}}{2}$. If this value is at least τ, then u is added to S by the algorithm. The theoretical guarantee of Algorithm 1 is provided in Theorem 1 and the proof of this theorem is given in Appendix A.1.

Next, we study the effect of the parameter r on the performance of Algorithm 1 under different settings. First note that the bound given by Corollary 2 reduces to a trivial lower bound of 0 when $\phi^{-2} \cdot g(OPT) \leq \ell(OPT)$. A similar phenomenon happens for the bound of (Harshaw et al., 2019, Theorem 3) when $(1 - e^{-1}) \cdot g(OPT) \leq \ell(OPT)$. Namely, in this regimen their bound (i.e., $(1 - e^{-1}) \cdot g(OPT) - \ell(OPT)$) becomes trivial. We now explain how a carefully chosen value for r can be used to prevent this issue.

For a set S, let β_S denote the ratio between the utility of S and its linear cost, i.e., $\beta_S = \frac{g(S)-\ell(S)}{\ell(S)}$. Using this ter-

Algorithm 1: THRESHOLD-STREAMING

Guess a value τ such that $k\tau \leq h(r) \cdot g(T) - r \cdot \ell(T) \leq (1+\varepsilon)k\tau$.
Let $\alpha(r) \leftarrow \frac{2r+1+\sqrt{4r^2+1}}{2}$.
Let $S \leftarrow \varnothing$.
while $|S| < k$ *and there are more elements* **do**
  Let u be the next elements in the stream.
  if $g(u \mid S) - \alpha(r) \cdot \ell(\{u\}) \geq \tau$ **then**
    Add u to the set S.
return *the better solution among* S *and* $\varnothing$.

Algorithm 2: DISTORTED-STREAMING

$\Lambda \leftarrow \{\varepsilon(1+\delta)^i \mid 0 \leq i \leq \lfloor \log_{1+\delta}(1/(2\varepsilon)) \rfloor\}$.
for *every* $\zeta \in \Lambda$ ***in parallel*** **do**
  Calculate $\beta \leftarrow \frac{4\zeta}{(1-2\zeta)^2}$.
  Calculate $r \leftarrow \frac{\beta}{2\sqrt{1+2\beta}}$. `// This is the formula from Corollary 4.`
  Run THRESHOLD-STREAMING (Algorithm 1) with r.
return *the best among the solutions found by all the copies of* THRESHOLD-STREAMING *executed*.

minology, we get that the guarantees of Corollary (2) and (Harshaw et al., 2019, Theorem 3) become trivial when $\beta_{OPT} \leq \phi^2 - 1 = \phi$ and $\beta_{OPT} \leq 1/(e-1)$, respectively. In Corollary 4, we show that by knowing the value of β_{OPT} we can find a value for r in Algorithm 1 which makes Theorem 1 yield the strongest guarantee for (1), and moreover, this guarantee is non-trivial as long as $\beta_{OPT} > 0$ (if $\beta_{OPT} \leq 0$, then the empty set is a trivial optimal solution).

Corollary 4. *Assume the value of β_{OPT} is given, where OPT is the optimal solution of Problem (1). Setting $r = r_{OPT} = \frac{\beta_{OPT}}{2\sqrt{1+2\beta_{OPT}}}$ makes Algorithm 1 return a solution S with the guarantee*

$$g(S) - \ell(S) \geq \left(\frac{1 + \beta_{OPT} - \sqrt{1+2\beta_{OPT}}}{2\beta_{OPT}} - \varepsilon' \right) \cdot \left(g(OPT) - \ell(OPT)\right) \ ,$$

where $\varepsilon' = \varepsilon \cdot (1 + 1/\beta_{OPT})$.

To apply the result of Corollary 4 to obtain the strongest guarantee for Problem (1), we need access to the set OPT (and consequently β_{OPT} and r_{OPT}); but, unfortunately, none of these is known a priori. We propose an efficient approach to find an accurate enough estimate of r_{OPT}. Let $\zeta_{OPT} = \frac{1+\beta_{OPT}-\sqrt{1+2\beta_{OPT}}}{2\beta_{OPT}}$ be the approximation ratio that can be obtained for the unknown value of β_{OPT} via Corollary 4 (except for the ε' error term). This definition implies that we always have $0 \leq \zeta_{OPT} < 1/2$. Thus, we can find an accurate guess for ζ_{OPT} by dividing the interval $[\varepsilon, 1/2)$ to small intervals (values of ζ_{OPT} below $\varepsilon < \varepsilon'$ are not of interest because Corollary 4 gives a trivial guarantee for them). Moreover, given a guess for ζ_{OPT}, we can calculate the corresponding values of β_{OPT} and r_{OPT}. The full version of the proposed algorithm (DISTORTED-STREAMING) is given as Algorithm 2.

One can see that the value of r passed to every copy of THRESHOLD-STREAMING by DISTORTED-STREAMING is at least 2ε, which implies that the number of elements kept by each such copy is at most $O(\varepsilon^{-1}(\log k + \log \varepsilon^{-1}))$. Furthermore, the number of elements kept by DISTORTED-STREAMING is larger than that by a factor of at most $1 + \log_{1+\delta}(1/(2\varepsilon)) = O(\delta^{-1} \cdot \log(\varepsilon)^{-1})$. The following observation studies the approximation guarantee of DISTORTED-STREAMING.

Lemma 5. *Despite not assuming access to β_{OPT},* DISTORTED-STREAMING *outputs a set S obeying*

$$g(S) - \ell(S) \geq ((1-\delta') \cdot \zeta_{OPT} - \varepsilon') \cdot \left(g(OPT) - \ell(OPT)\right) \ ,$$

where $\delta' = \delta/2$ and $\varepsilon' = \frac{\varepsilon}{2\zeta_{OPT}}$.

3. Mode Finding of SLC Distributions

In Section 1, we discussed the power of sampling from discrete probabilistic models (specifically SLC distributions), which encourages negative correlation. In this regard, recently, Robinson et al. (2019) established a notion of γ-additively weak submodularity for SLC functions. By using this newly defined property, we guarantee the performance of our proposed algorithms for mode finding of a class of distributions that are derived from SLC functions. Following is the definition of γ-additively weak submodular functions.

Definition 6 (Definition 1, (Robinson et al., 2019))**.** *A set function $\rho \colon 2^{\mathcal{N}} \to \mathbb{R}$ is γ-additively weak submodular if for any $S \subseteq \mathcal{N}$ and $u, v \in \mathcal{N} \setminus S$ with $u \neq v$, we have $\rho(S) + \rho(S \cup \{u, v\}) \leq \gamma + \rho(S \cup \{u\}) + \rho(S \cup \{v\})$.*

In Appendix D.1, we show that a γ-additively weak submodular function can be written, with a little modification, as a submodular function Λ, which in its own turn can be rewritten as the difference between a non-negative monotone submodular function and a modular function. With this new formulation, we improve the theoretical guarantees of Robinson et al. (2019) in the offline setting, and provide our streaming and distributed solutions for the mode finding problem under a cardinality constraint k. Specifically, by using either our proposed streaming or distributed algorithms (depending on whether the setting is a streaming or a distributed setting), we can get a scalable solution with a guarantee with respect to ρ, and in particular, a guarantee for the task of finding the mode of an SLC distribution. We should point out that it is also possible to use the distorted

greedy algorithm (Harshaw et al., 2019, Algorithm 1) to optimize Λ in the offline setting.

Corollary 7. *Assume $\rho\colon 2^{\mathcal{N}} \to \mathbb{R}$ is a γ-additively weak submodular function. Then, when given Λ as the objective function,* DISTORTED-DISTRIBUTED-GREEDY *(Algorithm 4) returns a solution R such that*

$$\mathbb{E}[\rho(R)] \geq (1-\varepsilon)\cdot\left[(1-e^{-1})\cdot\rho(\texttt{OPT}) - e^{-1}\cdot\ell(\texttt{OPT})\right] - \frac{\gamma\cdot[(1-e^{-1})\cdot l\cdot(l-1) - \mathbb{E}[|R|\cdot(|R|-1)]]}{2},$$

where $\texttt{OPT} \in \arg\max_{|S|\leq k}\rho(S)$ and $l = |\texttt{OPT}| \leq k$.

The following corollary shows the guarantee obtained by THRESHOLD-STREAMING as a function of the input parameter r. When the best choice for r is unknown, DISTORTED-STREAMING roughly obtains this guarantee for the best value of r, as discussed in Section 2.

Corollary 8. *Assume $\rho\colon 2^{\mathcal{N}} \to \mathbb{R}$ is a γ-additively weak submodular function. Then, when given Λ as the objective function,* THRESHOLD-STREAMING *(Algorithm 1) returns a solution R such that $\rho(R)$ is at least*

$$(h(r)-\varepsilon)\cdot\rho(\texttt{OPT}) - (\alpha(r)-r-1+\varepsilon)\cdot\ell(\texttt{OPT}) - \frac{\gamma\cdot[(h(r)-\varepsilon)\cdot l\cdot(l-1) - |R|\cdot(|R|-1)]}{2},$$

where $\texttt{OPT} \in \arg\max_{|S|\leq k}\rho(S)$ and $l = |\texttt{OPT}| \leq k$.

Note that in (Robinson et al., 2019, Theorem 12) and Corollary 7, if the value of the linear function is considerably larger than the values of functions η or ρ, then the parts that depend on the optimal solution OPT in the right-hand side of these results could be negative, which makes the bounds trivial. The main explanation for this phenomenon is that the distorted greedy algorithm does not take into account the relative importance of g and ℓ to the value of the optimal solution. On the other hand, the **distinguishing feature** of our streaming algorithm is that, by guessing the value of $\beta_{\texttt{OPT}}$, it can find the best possible scheme for assigning weights to the importance of the submodular and modular terms. Therefore, DISTORTED-STREAMING, even in the scenarios where the linear cost is large, can find solutions with a non-trivial provable guarantee. In the experiments presented at Section 3.1, we showcase two facts that: (i) DISTORTED-STREAMING could be used for mode finding of strongly log-concave distributions with a provable guarantee, and (ii) choosing an accurate estimation of β_{S^*} plays an important role in this optimization procedure.

3.1. Experimental Evaluations

In this section, we compare the performance of DISTORTED-STREAMING with the performance of distorted greedy, vanilla greedy and sieve streaming on the problem of mode finding for an SLR distribution. We consider a distribution $\nu(S) \propto \sqrt{\det(L_S)}\cdot\mathbf{1}\{|S|\leq d\}$, where L is an $n\times n$ PSD matrix. Here, L_S corresponds to the submatrix of L, where the rows and columns are indexed by elements of S (Robinson et al., 2019). In the optimization procedure, our goal is to maximize $\rho(S) \triangleq \log(\nu(S))$. To generate the random matrix L, we first sample a diagonal matrix D and a random PSD matrix Q, and then assign $L = QDQ^{-1}$. Each diagonal element of D is from a log-normal distribution with a probability mass function $p(x) = \frac{1}{\sigma x\sqrt{2\pi}}\exp(-\frac{(\ln(x)-\mu)^2}{2\sigma^2})$, where μ and σ are the mean and standard deviation of the normally distributed logarithm of the variable, respectively. This log-normal distribution allows us to have a PSD matrix where the eigenvalues have a heavy-tailed distribution. In these experiments, we set $n = 1000, d = 100, \mu = 1.0$ and $\sigma = 1.0$.

In Fig. 1, we observe that DISTORTED-STREAMING outperforms sieve streaming. This is mainly a result of the fact that DISTORTED-STREAMING estimates the value of $\beta_{\texttt{OPT}}$ and uses the best possible value for r. Furthermore, we see that vanilla greedy performs better than distorted greedy, and for cardinality constraints larger than $k = 20$, the performance of distorted greedy degrades. This observation could be explained by the fact that the linear cost for each element u is comparable to the value of $g(u)$ (or marginal gain of u to any set S). Therefore, distorted greedy does not pick any element in the first few iterations when k is large enough, i.e., when $(1-\frac{1}{k})^{k-(i+1)}$ is small. It is worth mentioning that while the performance of the greedy algorithm is the best for this specific application, only DISTORTED-STREAMING and distorted greedy have a theoretical guarantee.

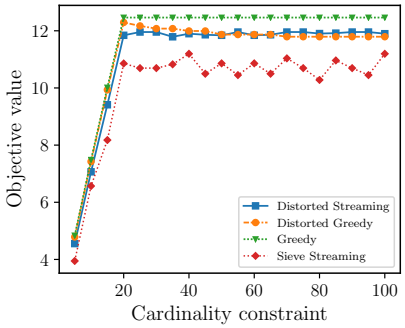

Figure 1. We want to find the mode of a distribution $\nu(S) \propto \sqrt{\det(L_S)}\cdot\mathbf{1}\{|S|\leq d\}$ for a PSD matrix L. For the objective value, we report $\log(\nu(S))$.

4. Conclusion

In this paper, we proposed scalable methods for maximizing a *regularized* submodular function expressed as the difference between a non-negative monotone submodular function g and a modular function ℓ. We developed the first streaming and distributed algorithms for maximizing a regularized submodular function subject to a k-cardinality constraint. We studied the applications of regularized submodular maximization in mode finding of SLC distributions.

Acknowledgements

The research of Moran Feldman was partially supported by ISF grant 1357/16. Amin Karbasi is partially supported by NSF (IIS- 1845032), ONR (N00014-19-1-2406), and AFOSR (FA9550-18-1-0160).

References

Badanidiyuru, A., Mirzasoleiman, B., Karbasi, A., and Krause, A. Streaming Submodular Maximization:Massive Data Summarization on the Fly. In *International Conference on Knowledge Discovery and Data Mining, KDD*, pp. 671–680, 2014.

Barbosa, R. d. P., Ene, A., Nguyen, H., and Ward, J. The power of randomization: Distributed submodular maximization on massive datasets. In *International Conference on Machine Learning*, pp. 1236–1244, 2015.

Barbosa, R. d. P., Ene, A., Nguyen, H. L., and Ward, J. A New Framework for Distributed Submodular Maximization. In *Symposium on Foundations of Computer Science, (FOCS)*, pp. 645–654, 2016.

Borcea, J., Brändén, P., and Liggett, T. Negative dependence and the geometry of polynomials. *Journal of the American Mathematical Society*, 22(2):521–567, 2009.

Buchbinder, N. and Feldman, M. Submodular Functions Maximization Problems. In *Handbook of Approximation Algorithms and Metaheuristics*, pp. 771–806. Chapman and Hall/CRC, 2018.

Djolonga, J. and Krause, A. From map to marginals: Variational inference in bayesian submodular models. In *Advances in Neural Information Processing Systems*, pp. 244–252, 2014.

Feldman, M. Guess free maximization of submodular and linear sums. In *Algorithms and Data Structures (WADS)*, pp. 380–394, 2019.

Gotovos, A., Hassani, H., and Krause, A. Sampling from probabilistic submodular models. In *Advances in Neural Information Processing Systems*, pp. 1945–1953, 2015.

Gurvits, L. A polynomial-time algorithm to approximate the mixed volume within a simply exponential factor. *Discrete & Computational Geometry*, 41(4):533–555, 2009.

Harshaw, C., Feldman, M., Ward, J., and Karbasi, A. Submodular Maximization beyond Non-negativity: Guarantees, Fast Algorithms, and Applications. In *International Conference on Machine Learning*, pp. 2634–2643, 2019.

Iyer, R. and Bilmes, J. Submodular point processes with applications to machine learning. In *Artificial Intelligence and Statistics*, pp. 388–397, 2015.

Kulesza, A. and Taskar, B. Determinantal Point Processes for Machine Learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

Lovász, L. Submodular functions and convexity. In *Mathematical Programming The State of the Art*, pp. 235–257. Springer, 1983.

Mirrokni, V. and Zadimoghaddam, M. Randomized composable core-sets for distributed submodular maximization. In *Ssymposium on Theory of Computing (STOC)*, pp. 153–162. ACM, 2015.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pp. 2049–2057, 2013.

Robinson, J., Sra, S., and Jegelka, S. Flexible Modeling of Diversity with Strongly Log-Concave Distributions. In *Advances in Neural Information Processing Systems*, pp. 15199–15209, 2019.

Sviridenko, M., Vondrák, J., and Ward, J. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Math. Oper. Res.*, 42(4):1197–1218, 2017.

Wei, K., Iyer, R., and Bilmes, J. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*, pp. 1954–1963, 2015.

A. Proofs

A.1. Proof of Theorem 1

In this section, we provide the proof of Theorem 1. Let T be the subset of $\mathcal{N}$ of size at most k maximizing $(h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)$ among all such subsets. If $h(r) \cdot g(T) - r \cdot \ell(T) \leq 0$, then the empty set is a solution set obeying all the requirements of Theorem 1. Thus, in the rest of this section, we assume $h(r) \cdot g(T) - r \cdot \ell(T) > 0$, which implies that the value τ guessed by Algorithm 1 is positive.

The following two lemmata prove together that Algorithm 1 has the approximation guarantee of Theorem 1. The first of these lemmata handles the case in which the size of the solution S of Algorithm 1 reaches the maximum possible size k. In the proofs of both lemmata u_i denotes the i-th element added to S by Algorithm 1.

Lemma 9. *If $|S| = k$, then $g(S) - \ell(S) \geq (h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)$.*

Proof. Observe that

$$\begin{aligned} g(S) - \alpha(r) \cdot \ell(S) &= \sum_{i=1}^{k} [g(u_i \mid \{u_1, u_2, \ldots, u_{i-1}\}) - \alpha(r) \cdot \ell(\{u\})] \geq k\tau \\ &\geq \frac{h(r) \cdot g(T) - r \cdot \ell(T)}{1 + \varepsilon} \geq (1 - \varepsilon) \cdot h(r) \cdot g(T) - r \cdot \ell(T) \ , \end{aligned}$$

where the first inequality holds since Algorithm 1 chose to add u_i to S, and the set S at that time was equal to $\{u_1, u_2, \ldots, u_{i-1}\}$.

We now make two observations. First, we observe that

$$\alpha(r) = \frac{2r + 1 + \sqrt{4r^2 + 1}}{2} \geq \frac{1 + \sqrt{1}}{2} = 1 \ ,$$

and second, we observe that $h(r) \leq 1/2$ because

$$h(r) \leq 1/2 \iff \frac{2r + 1 - \sqrt{4r^2 + 1}}{2} \leq 1/2 \iff 2r \leq \sqrt{4r^2 + 1} \iff 4r^2 \leq 4r^2 + 1 \ .$$

Using, these observations and the above inequality, we now get

$$g(S) - \ell(S) \geq g(S) - \alpha(r) \cdot \ell(S) \geq (1 - \varepsilon) \cdot h(r) \cdot g(T) - r \cdot \ell(T) \geq (h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T) \ . \qquad \square$$

The following lemma proves the approximation ratio of Algorithm 1 for the case in which the solution set S does not reach its maximum allowed size k before the stream ends.

Lemma 10. *If $|S| < k$, then $g(S) - \ell(S) \geq (h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)$.*

Proof. Consider an arbitrary element $u \in OPT \setminus S$. Since $|S| < k$, the fact that u was not added to S implies

$$g(u \mid S') - \alpha(r) \cdot \ell(\{u\}) < \tau \ ,$$

where S' is the set S at the time in which u arrived. By the submodularity of g, we also get

$$g(u \mid S) - \alpha(r) \cdot \ell(\{u\}) < \tau \ .$$

Adding the last inequality over all elements $u \in T \setminus S$ implies

$$\begin{aligned} g(T) - g(S) - \alpha(r) \cdot \ell(T) \leq g(T \mid S) - \alpha(r) \cdot \ell(T) &\leq \sum_{u \in T \setminus S} [g(u \mid S) - \alpha(r) \cdot \ell(\{u\})] \\ &< k\tau \leq h(r) \cdot g(T) - r \cdot \ell(T) \ , \end{aligned}$$

where the first inequality follows from the monotonicity of g, and the second inequality holds due to the submodularity of g and the non-negativity of ℓ. Rearranging this inequality yields

$$(1 - h(r)) \cdot g(T) + (r - \alpha(r)) \cdot \ell(T) < g(S) \ . \tag{2}$$

Recall that $\tau > 0$. Thus, using the same argument used in the proof of Lemma 9, we get

$$g(S) - \alpha(r) \cdot \ell(S) = \sum_{i=1}^{|S|} [g(u_i \mid \{u_1, u_2, \ldots, u_{i-1}\}) - \alpha(r) \cdot \ell(\{u\})] \geq |S|\tau \geq 0 \ .$$

Adding a $1/\alpha(r)$ fraction of this equation to a $1 - 1/\alpha(r)$ fraction of Equation (2) yields

$$g(S) - \ell(S) > (1 - 1/\alpha(r))(1 - h(r)) \cdot g(T) + (1 - 1/\alpha(r))(r - \alpha(r)) \cdot \ell(T) \ .$$

The following two calculations now complete the proof of the lemma (since $\varepsilon \cdot g(T)$ is non-negative).

$$\begin{aligned}
(1 - 1/\alpha(r))(1 - h(r)) &= \left(1 - \frac{2}{2r + 1 + \sqrt{4r^2 + 1}}\right)\left(1 - \frac{2r + 1 - \sqrt{4r^2 + 1}}{2}\right) \\
&= \frac{2r - 1 + \sqrt{4r^2 + 1}}{2r + 1 + \sqrt{4r^2 + 1}} \cdot \frac{1 - 2r + \sqrt{4r^2 + 1}}{2} = \frac{4r^2 + 1 - (2r - 1)^2}{2(2r + 1 + \sqrt{4r^2 + 1})} \\
&= \frac{(2r + 1)^2 - (4r^2 + 1)}{2(2r + 1 + \sqrt{4r^2 + 1})} = \frac{2r + 1 + \sqrt{4r^2 + 1}}{2r + 1 + \sqrt{4r^2 + 1}} \cdot \frac{2r + 1 - \sqrt{4r^2 + 1}}{2} = h(r) \ ,
\end{aligned}$$

and

$$\begin{aligned}
(1 - 1/\alpha(r))(r - \alpha(r)) &= \left(1 - \frac{2}{2r + 1 + \sqrt{4r^2 + 1}}\right)\left(r - \frac{2r + 1 + \sqrt{4r^2 + 1}}{2}\right) \\
&= -\frac{2r - 1 + \sqrt{4r^2 + 1}}{2r + 1 + \sqrt{4r^2 + 1}} \cdot \frac{1 + \sqrt{4r^2 + 1}}{2} = -\frac{2r - 1 + (4r^2 + 1) + 2r \cdot \sqrt{4r^2 + 1}}{2[2r + 1 + \sqrt{4r^2 + 1}]} \\
&= -\frac{2r \cdot [1 + 2r + \sqrt{4r^2 + 1}]}{2[2r + 1 + \sqrt{4r^2 + 1}]} = -r \ .
\end{aligned}$$

□

A.2. Proof of Corollary 4

In this section, we first restate Corollary 4 and then provide its proof.

Corollary 4. *Assume the value of β_{OPT} is given, where OPT is the optimal solution of Problem (1). Setting $r = r_{OPT} = \frac{\beta_{OPT}}{2\sqrt{1+2\beta_{OPT}}}$ makes Algorithm 1 return a solution S with the guarantee*

$$g(S) - \ell(S) \geq \left(\frac{1 + \beta_{OPT} - \sqrt{1 + 2\beta_{OPT}}}{2\beta_{OPT}} - \varepsilon'\right) \cdot (g(OPT) - \ell(OPT)) \ ,$$

where $\varepsilon' = \varepsilon \cdot (1 + 1/\beta_{OPT})$.

Proof. First, let us define $T_r^* = \arg\max_{T \in \mathcal{N}, |T| \leq k} [(h(r) - \varepsilon) \cdot g(T) - r \cdot \ell(T)]$. From Theorem 1 and the definition of T_r^*, we have

$$g(S) - \ell(S) \geq (h(r) - \varepsilon) \cdot g(T_r^*) - r \cdot \ell(T^*) \geq (h(r) - \varepsilon) \cdot g(OPT) - r \cdot \ell(OPT) \ .$$

Furthermore, from the definition of β_{OPT}, we have

$$\begin{aligned}
(h(r) - \varepsilon) \cdot g(OPT) - r \cdot \ell(OPT) &= \frac{(h(r) - \varepsilon) \cdot g(OPT) - r \cdot \ell(OPT)}{g(OPT) - \ell(OPT)} \cdot (g(OPT) - \ell(OPT)) \\
&= \frac{(h(r) - \varepsilon) \cdot (1 + \beta_{OPT}) - r}{\beta_{OPT}} \cdot (g(OPT) - \ell(OPT)) \ .
\end{aligned}$$

It can be verified that r_{OPT} is the value that maximizes the above expression, and plugging this value into the expression proves the corollary. □

A.3. Proof of Lemma 5

In this section, we first restate Lemma 5 and then provide its proof.

Lemma 5. *Despite not assuming access to β_{OPT},* DISTORTED-STREAMING *(Algorithm 2) outputs a set S obeying*

$$g(S) - \ell(S) \geq ((1-\delta') \cdot \zeta_{OPT} - \varepsilon') \cdot (g(OPT) - \ell(OPT)) \ ,$$

where $\delta' = \delta/2$ and $\varepsilon' = \frac{\varepsilon}{2\zeta_{OPT}}$.

Proof. For values of $\zeta_{OPT} < \varepsilon$, the right-hand side of the lower bounded provided by the lemma is negative, and thus, it gives a trivial lower bound (note that $\varepsilon' \geq \varepsilon$ and $\delta' > 0$ since ζ_{OPT} is always smaller than $1/2$). For this reason, in the rest of proof, we assume $\zeta_{OPT} \geq \varepsilon$. First, note that there must be a value $\zeta \in \Lambda$ such that $\zeta \leq \zeta_{OPT} < (1+\delta) \cdot \zeta$, and let us denote $\omega = \frac{\zeta_{OPT}}{\zeta}$. It is clear that $1 \leq \omega < 1 + \delta$. Moreover, using the definition of ω we get that the value of β corresponding to ζ is $\beta = \frac{4\omega\zeta_{OPT}}{(\omega - 2\zeta_{OPT})^2}$, and the value of r corresponding to this ζ is

$$\begin{aligned} r &= \frac{\beta}{2\sqrt{1+2\beta}} = \frac{4\omega\zeta_{OPT}/(\omega - 2\zeta_{OPT})^2}{2\sqrt{1 + 8\omega\zeta_{OPT}/(\omega - 2\zeta_{OPT})^2}} \\ &= \frac{4\omega\zeta_{OPT}}{2(\omega - 2\zeta_{OPT}) \cdot \sqrt{(\omega - 2\zeta_{OPT})^2 + 8\omega\zeta_{OPT}}} = \frac{4\omega\zeta_{OPT}}{2(\omega - 2\zeta_{OPT}) \cdot \sqrt{(\omega + 2\zeta_{OPT})^2}} = \frac{2\omega\zeta_{OPT}}{\omega^2 - 4\zeta_{OPT}^2} \ . \end{aligned}$$

To calculate the value of $h(r)$ corresponding to this value of r, we note that:

$$\sqrt{4r^2+1} = \sqrt{4\left(\frac{2\omega\zeta_{OPT}}{\omega^2 - 4\zeta_{OPT}^2}\right)^2 + 1} = \frac{\omega^2 + 4\zeta_{OPT}^2}{\omega^2 - 4\zeta_{OPT}^2} \ .$$

If we plug this equality into the definition of $h(r)$, we get

$$h(r) = \frac{2r + 1 - \sqrt{4r^2+1}}{2} = \frac{1}{2} \cdot \left[1 + \frac{4\omega\zeta_{OPT}}{\omega^2 - 4\zeta_{OPT}^2} - \frac{\omega^2 + 4\zeta_{OPT}^2}{\omega^2 - 4\zeta_{OPT}^2}\right] = \frac{2\omega\zeta_{OPT} - 4\zeta_{OPT}^2}{\omega^2 - 4\zeta_{OPT}^2} = \frac{2\zeta_{OPT}}{\omega + 2\zeta_{OPT}} \ .$$

We are now ready to plug the calculated value of r into Theorem 1, which yields that the output set S' of the instance of THRESHOLD-STREAMING initialized with this value of r obeys

$$\begin{aligned} g(S') - \ell(S') &\geq (h(r) - \varepsilon) \cdot g(OPT) - r \cdot \ell(OPT) \qquad (3) \\ &= \frac{(h(r) - \varepsilon) \cdot g(OPT) - r \cdot \ell(OPT)}{g(OPT) - \ell(OPT)} \cdot (g(OPT) - \ell(OPT)) \\ &= \frac{(h(r) - \varepsilon) \cdot (1 + \beta_{OPT}) - r}{\beta_{OPT}} \cdot (g(OPT) - \ell(OPT)) \ , \qquad (4) \end{aligned}$$

where the last equality follows from the definition of β_{OPT}. Let us now lower bound the coefficient of $g(OPT) - \ell(OPT)$ in the rightmost hand side of the last equality. Recalling that $\beta_{OPT} = \frac{4\zeta_{OPT}}{(1-2\zeta_{OPT})^2}$, we get $\frac{1+\beta_{OPT}}{\beta_{OPT}} = \frac{1+4\zeta_{OPT}^2}{4\zeta_{OPT}}$. Thus, the above mentioned coefficient can be written as

$$\begin{aligned} \frac{(h(r) - \varepsilon) \cdot (1 + \beta_{OPT}) - r}{\beta_{OPT}} &= \frac{1 + \beta_{OPT}}{\beta_{OPT}} \cdot h(r) - \frac{r}{\beta_{OPT}} - \frac{(1 + \beta_{OPT}) \cdot \varepsilon}{\beta_{OPT}} \\ &= \frac{1 + 4\zeta_{OPT}^2}{4\zeta_{OPT}} \cdot \frac{2\zeta_{OPT}}{\omega + 2\zeta_{OPT}} - \frac{\omega \cdot (1 - 2\zeta_{OPT})^2}{2 \cdot (\omega^2 - 4\zeta_{OPT}^2)} - \frac{(1 + 4\zeta_{OPT}^2) \cdot \varepsilon}{4\zeta_{OPT}} \\ &= \frac{2\omega\zeta_{OPT} - \zeta_{OPT} - 4\zeta_{OPT}^3}{\omega^2 - 4\zeta_{OPT}^2} - \frac{(1 + 4\zeta_{OPT}^2) \cdot \varepsilon}{4\zeta_{OPT}} \\ &= \left(1 - \frac{(\omega - 1)^2}{\omega^2 - 4\zeta_{OPT}^2}\right) \cdot \zeta_{OPT} - \frac{(1 + 4\zeta_{OPT}^2) \cdot \varepsilon}{4\zeta_{OPT}} \ . \end{aligned}$$

We can observe that the coefficient of ζ_{OPT} on the rightmost side is a decreasing function of ω for $\omega \geq 4\zeta_{OPT}^2$. Together with the facts that $\zeta_{OPT} < 1/2$ and $\omega \geq 1$, this implies

$$\begin{aligned}\frac{(h(r)-\varepsilon)\cdot(1+\beta_{OPT})-r}{\beta_{OPT}} &\geq \left(1-\frac{\delta^2}{(1+\delta)^2-4\zeta_{OPT}^2}\right)\cdot\zeta_{OPT}-\frac{\varepsilon}{2\zeta_{OPT}}\\ &\geq \left(1-\frac{\delta^2}{2\delta}\right)\cdot\zeta_{OPT}-\frac{\varepsilon}{2\zeta_{OPT}}=\left(1-\frac{\delta}{2}\right)\cdot\zeta_{OPT}-\frac{\varepsilon}{2\zeta_{OPT}}\ .\end{aligned}$$

Plugging this inequality into Eq. (3), we get that the set S' produced by THRESHOLD-STREAMING for the above value of r has at least the value guaranteed by the lemma for the output set S of DISTORTED-STREAMING. The lemma now follows since the set S is chosen as the best set among multiple options including S'. □

B. Guessing τ in Algorithm 1

In this section we explain how one can guess the value τ in Algorithm 1, which is a value obeying $k\tau \leq h(r)\cdot g(T)-r\cdot\ell(T) \leq (1+\varepsilon)k\tau$, at the cost of increasing the space complexity of the algorithm by a factor of $O(\varepsilon^{-1}(\log k+\log r^{-1}))$. We assume that $h(r)\cdot g(T)-r\cdot\ell(T)$—and thus, also τ—is positive.

Observe that

$$\begin{aligned}\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})] \leq h(r)\cdot g(T)-r\cdot\ell(T) &\leq \sum_{u\in T}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]\\ &\leq k\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]\ ,\end{aligned}$$

where the first inequality holds since $\{u\}$ is a candidate to be T for every $u\in\mathcal{N}$, and the second inequality follows from the submodularity of g. Thus, if we knew the value of $\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]$ from the very beginning, we could simply run in parallel an independent copy of Algorithm 1 for every value of τ that has the form $(1+\varepsilon)^i$ for some integer i and falls within the range

$$\left[k^{-1}\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})],(1+\varepsilon)\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]\right]\ .$$

Clearly, at least one of the values we would have tried obeys $k\tau \leq h(r)\cdot g(T)-r\cdot\ell(T)\leq(1+\varepsilon)k\tau$, and the number of values we would have needed to try is upper bounded by

$$1+\log_{1+\varepsilon}\left(\frac{(1+\varepsilon)\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})}{k^{-1}\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]}\right)=2+\log_{1+\varepsilon}k=O(\varepsilon^{-1}\log k)\ .$$

Unfortunately, the value of $\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]$ is not known to us in advance. To compensate for this, we make the following two observations. The first observation is that $k^{-1}\cdot\max_{u\in\mathcal{N}'}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]$, where $\mathcal{N}'$ is the set of elements viewed so far, is a lower bound on the value of $k^{-1}\cdot\max_{u\in\mathcal{N}}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]$. Following is the second observation, which shows that copies of Algorithm 1 with τ values that are much larger than this lower bound cannot accept any element of $\mathcal{N}'$, and thus, need not be maintained explicitly.

Observation 11. *If $\tau > (\alpha(r)/r)\cdot\max_{u\in\mathcal{N}'}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]$, then Algorithm 1 accepts no element of $\mathcal{N}'$.*

Proof. Algorithm 1 accepts an element $u\in\mathcal{N}'$ if $g(u\mid S)-\alpha(r)\cdot\ell(\{u\})\geq\tau$. However, the condition of the observation implies

$$\begin{aligned}g(u\mid S)-\alpha(r)\cdot\ell(\{u\}) \leq g(\{u\})-\alpha(r)\cdot\ell(\{u\}) &= \frac{\alpha(r)}{r}\cdot[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]\\ &\leq \frac{\alpha(r)}{r}\cdot\max_{u\in\mathcal{N}'}[h(r)\cdot g(\{u\})-r\cdot\ell(\{u\})]<\tau\ ,\end{aligned}$$

where the first inequality follows from the submodularity of g, and the equality follows from the following calculation.

$$\alpha(r)\cdot h(r)=\frac{2r+1+\sqrt{4r^2+1}}{2}\cdot\frac{2r+1-\sqrt{4r^2+1}}{2}=\frac{(2r+1)^2-(4r^2+1)}{4}=\frac{4r}{4}=r\ .$$

□

The above observations imply that it suffices to explicitly maintain a copy of Algorithm 1 for values of τ that are equal to $(1+\varepsilon)^i$ for some integer i and fall within the range

$$\left[k^{-1} \cdot \max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})], \frac{\alpha(r)}{r} \cdot \max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})]\right] \ . \tag{5}$$

In particular, we know that when the value of $\max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})]$ increases (due to the arrival of additional elements), we can start a new copy of Algorithm 1 for the values of τ that have the form $(1+\varepsilon)^i$ for some integer i and now enter the range. By Observation 11, these instances will behave in exactly the same way as if they had been created at the very beginning of the stream. A formal description of the algorithm we obtain using this method is given as Algorithm 3. We note that the space complexity of this algorithm is larger than the space complexity of Algorithm 1 only by an $O(\varepsilon^{-1}(\log k + \log r^{-1}))$ factor because the number of values of the form $(1+\varepsilon)^i$ that can fall within the range (5) is at most

$$\begin{aligned}
&1 + \log_{1+\varepsilon}\left(\frac{\frac{\alpha(r)}{r} \cdot \max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})]}{k^{-1} \cdot \max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})]}\right) = 1 + \log_{1+\varepsilon}\left(\frac{k \cdot \alpha(r)}{r}\right) \\
&\quad = 1 + \log_{1+\varepsilon}\left(\frac{k \cdot (2r + 1 + \sqrt{4r^2+1})}{2r}\right) \leq 1 + \log_{1+\varepsilon}(k + k/r) \\
&\quad \leq 1 + \log_{1+\varepsilon} k + \log_{1+\varepsilon}(k/r) = O(\varepsilon^{-1}(\log k + \log r^{-1})) \ .
\end{aligned}$$

Algorithm 3: DISTORTED-STREAMING: Guessing τ

Let $M \leftarrow -\infty$ and $I \leftarrow \varnothing$. `// M represents` $\max_{u \in \mathcal{N}'}[h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})]$ `and I is the list of copies of Algorithm 1 currently maintained.`
while *there are more elements in the stream* **do**
  Let u be the next element of the stream.
  Update $M \leftarrow \max\{M, h(r) \cdot g(\{u\}) - r \cdot \ell(\{u\})\}$.
  Let $J = \{i \in \mathbb{Z} \mid k^{-1}M \leq (1+\varepsilon)^i \leq r^{-1}M \cdot \alpha(r)\}$.
  Delete every copy of Algorithm 1 in I corresponding to a value $\tau = (1+\varepsilon)^i$ for an integer i that now falls outside the set J.
  Add to I a new copy of Algorithm 1 with $\tau = (1+\varepsilon)^i$ for every integer $i \in J$, unless such a copy already exists there.
  Pass the element u to all the copies of Algorithm 1 in I.
return *the set S maximizing $g(S) - \ell(S)$ among all the output sets of all the copies of Algorithm 1 in I.*

C. Distributed Algorithm

The exponential growth of data makes it difficult to process or even store the entire data on a single machine. For this reason, there is an urgent need to develop distributed or parallel computing methods to process massive datasets. Distributed algorithms in a Map-Reduce model have shown promising results in several problems related to submodular maximization (Mirzasoleiman et al., 2013; Mirrokni & Zadimoghaddam, 2015; Barbosa et al., 2015; 2016). In this section we present a distributed solution for Problem (1), named DISTORTED-DISTRIBUTED-GREEDY, which appears as Algorithm 4. Our algorithm uses DISTORTED-GREEDY proposed by Harshaw et al. (2019) as a subroutine.

Out distributed solution is based on the framework suggested by Barbosa et al. (2016) for converting greedy-like sequential algorithms into a distributed algorithm. However, its analysis is based on a generalization of ideas from (Barbosa et al., 2015) rather than being a direct adaptation of the analysis given by (Barbosa et al., 2016). This allows us to get an algorithm which uses asymptotically the same number of computational rounds as the algorithm of (Barbosa et al., 2016), but does not require to keep multiple copies of the data as is necessary for the last algorithm.[2] We would also like to point out that

[2]Our technique can be used to get the same improvement for the setting of (Barbosa et al., 2016). However, as this is not the main subject of this paper, we omit the details.

Barbosa et al. (2016) have proposed a variant of their algorithm that avoids data replication, but it does so at the cost of increasing the number of rounds from $\Theta(1/\varepsilon)$ to $\Theta(1/\varepsilon^2)$.

DISTORTED-DISTRIBUTED-GREEDY is a distributed algorithm within a Map-Reduce framework using $\lceil 1/\varepsilon \rceil$ rounds of computation, where $\varepsilon \in (0, 1/2]$ is a parameter controlling the quality of the output produced by the algorithm—for simplicity, we assume that $1/\varepsilon$ is an integer from this point on. In the first round of computation, DISTORTED-DISTRIBUTED-GREEDY randomly distributes the elements among m machines by independently sending each element $u \in \mathcal{N}$ to a uniformly random machine. Each machine i then runs DISTORTED-GREEDY on its data and forwards the resulting solution $S_{1,i}$ to all other machines (in general, we denote by $S_{r,i}$ the solution calculated by machine i in round r). The next rounds repeat this operation, except that the data of each machine now includes both: (i) elements sent to this machine during the random partition and (ii) the elements that belong to any solutions calculated (by any machine) during the previous rounds. At the end of the last round, machine number 1 outputs the final solution, which is the best solution among the solution computed by this machine in the last round and the solutions computed by all the machines in the previous rounds. Theorem 3 analyzes the approximation guarantee of DISTORTED-DISTRIBUTED-GREEDY, and its proof is given in Appendix C.1.

Algorithm 4: DISTORTED-DISTRIBUTED-GREEDY

for $r = 1$ **to** $\lceil \varepsilon^{-1} \rceil$ **do**
  for *each* $u \in \mathcal{N}$ **do**
    Assign element u to a machine chosen uniformly at random (and independently) among m machines.
    Let $\mathcal{N}_{r,i}$ denote the elements assigned to machine i in this round.
  for $i = 1$ **to** m **do**
    Run DISTORTED-GREEDY on the set $\mathcal{N}_{r,i} \cup (\cup_{r'=1}^{r-1} \cup_{i'=1}^{m} S_{r',i'})$ to get the solution $S_{r,i}$ of size at most k .
  if $r < \varepsilon^{-1}$ **then** Forward the solutions $S_{r,i}$, for every integer $1 \leq i \leq m$, to all the machines.
  else return *a set* D *maximizing* $g(D) - \ell(D)$ *among all sets in* $\{S_{r,1}\} \cup \{S_{r',i'} \mid 1 \leq r' < r \text{ and } 1 \leq i' \leq m\}$.

C.1. Proof of Theorem 3

Theorem 3 guarantees the performance of DISTORTED-DISTRIBUTED-GREEDY (Algorithm 4). In this section we first restate and then prove Theorem 3.

Theorem 3. DISTORTED-DISTRIBUTED-GREEDY *(Algorithm 4) returns a set* $D \subseteq \mathcal{N}$ *of size at most* k *such that*

$$\mathbb{E}[g(D) - \ell(D)] \geq (1 - \varepsilon) \left[(1 - e^{-1}) \cdot g(OPT) - \ell(OPT) \right].$$

We define the submodular function $f(S) \triangleq g(S) - \ell(S)$. It is easy to see that f is a submodular function (although it is not guaranteed to be either monotone or non-negative). The Lovász extension of f is the function $\hat{f} \colon [0,1]^{\mathcal{N}} \to \mathbb{R}$ given by

$$\hat{f}(\mathbf{x}) = \mathop{\mathbb{E}}_{\theta \in \mathcal{U}(0,1)} \left[f\left(\{ i : x_i \geq \theta \} \right) \right] \ ,$$

where $\mathcal{U}(0,1)$ is the uniform distribution within the range $[0,1]$ (Lovász, 1983). Note that the Lovász extension of a modular set function is the natural linear extension of the function. It was also proved in (Lovász, 1983) that the Lovász extension of a submodular function is convex. Finally, we need the following well-known properties of Lovász extensions, which follow easily from its definition.

Observation 12. *For every set* $S \subseteq \mathcal{N}$, $\hat{f}(\mathbf{1}_S) = f(S)$. *Additionally,* $\hat{f}(c \cdot \mathbf{p}) \geq c \cdot \hat{f}(\mathbf{p})$ *for every* $c \in [0,1]$ *and* $\mathbf{p} \in [0,1]^{\mathcal{N}}$ *whenever* $f(\varnothing)$ *is non-negative.*

Let us denote by DISTORTED-GREEDY(A) the set produced by DISTORTED-GREEDY when it is given the elements of a set $A \subseteq \mathcal{N}$ as input. Using this notation, we can now state the following lemma. We omit the simple proof of this lemma, but note that it is similar to the proof of (Barbosa et al., 2015, Lemma 2).

Lemma 13. *Let* $A \subseteq \mathcal{N}$ *and* $B \subseteq \mathcal{N}$ *be two disjoint subsets of* $\mathcal{N}$. *Suppose that, for each element* $u \in B$, *we have* DISTORTED-GREEDY$(A \cup \{u\})$ = DISTORTED-GREEDY(A). *Then,* DISTORTED-GREEDY$(A \cup B)$ = DISTORTED-GREEDY(A).

We now need some additional notation. Let S^* denote an optimal solution for Problem (1), and let $\mathcal{N}(1/m)$ represent the distribution over random subsets of $\mathcal{N}$ where each element is sampled independently with probability $1/m$. To see why this distribution is important, recall that $\mathcal{N}_{r,i}$ is the set of elements assigned to machine i in round i by the random partition, and that every element is assigned uniformly at random to one out of m machines, which implies that the distribution of $\mathcal{N}_{r,i}$ is identical to $\mathcal{N}(1/m)$ for every two integers $1 \leq i \leq m$ and $1 \leq r \leq \varepsilon^{-1}$. We now define for every integer $0 \leq r \leq \varepsilon^{-1}$ the set $C_r = \cup_{r'=1}^{r} \cup_{i=1}^{m} S_{r',i}$ and the vector $\mathbf{p}^r \in [0,1]^{\mathcal{N}}$ whose u-coordinate, for every $u \in \mathcal{N}$, is given by

$$p_u^r = \begin{cases} \Pr_{A \sim \mathcal{N}(1/m)}[u \notin C_{r-1} \text{ and } u \in \textsc{Distorted-Greedy}(A \cup C_{r-1} \cup \{u\})] & \text{if } u \in S^* \\ 0 & \text{otherwise} \end{cases} .$$

The next lemma proves an important property of the above vectors.

Lemma 14. *For every element $u \in S^*$ and $0 \leq r \leq 1/\varepsilon$, $\Pr[u \in C_r] = \sum_{r'=1}^{r} p_u^{r'}$.*

Proof. Since u is assigned in round r' to a single machine uniformly at random,

$$\begin{aligned} \Pr[u \in C_{r'} \setminus C_{r'-1}] &= \Pr[u \in \cup_{i=1}^{m} S_{r',i} \setminus C_{r'-1}] = \frac{1}{m} \sum_{i=1}^{m} \Pr[u \in S_{r',i} \setminus C_{r'-1} \mid u \in \mathcal{N}_{r',i}] \\ &= \frac{1}{m} \sum_{i=1}^{m} \Pr[u \notin C_{r'-1} \text{ and } u \in \textsc{Distorted-Greedy}(\mathcal{N}_{r',i} \cup (\cup_{r''=1}^{r'-1} \cup_{i'=1}^{m} S_{r'',i'})) \mid u \in \mathcal{N}_{r',i}] \\ &= \frac{1}{m} \sum_{i=1}^{m} \Pr_{A \sim \mathcal{N}(1/m)}[u \notin C_{r'-1} \text{ and } u \in \textsc{Distorted-Greedy}(A \cup C_{r'-1} \cup \{u\})] = p_u^{r'} , \end{aligned}$$

where the first equality holds since $C_{r'}$ can be obtained from $C_{r'-1}$ by adding to the last set all the elements of $\cup_{i=1}^{m} S_{r',i}$ that do not already belong to $C_{r'-1}$, and the last equality holds since the distribution of $\mathcal{N}_{r',i}$ conditioned on u belonging to this set is equal to the distribution of $A \cup \{u\}$ when A is distributed like $\mathcal{N}(1/m)$.

Since $C_1 \subseteq C_2 \subseteq \ldots \subseteq C_r$, the events $\Pr[u \in C_{r'} \setminus C_{r'-1}]$ must be disjoint for different values of r', which implies

$$\sum_{r'=1}^{r} p_u^{r'} = \sum_{r'=1}^{r} \Pr[u \in C_{r'} \setminus C_{r'-1}] = \Pr[u \in C_r] - \Pr[u \in C_0] = \Pr[u \in C_r] ,$$

where the last equality holds since $C_0 = \varnothing$ by definition. □

Using the last lemma, we can now prove lower bounds on the expected values of the sets $S_{r,i}$.

Lemma 15. *Let $\hat{g}$ and $\hat{\ell}$ be the Lovász extensions of the functions g and ℓ, respectively. Then, for every two integers $1 \leq r \leq \varepsilon^{-1}$ and $1 \leq i \leq m$,*

$$\mathbb{E}[f(S_{r,i})] \geq (1 - e^{-1}) \cdot \hat{g}(\mathbf{1}_{S^*} - \mathbf{p}^r) - \hat{\ell}(\mathbf{1}_{S^*} - \mathbf{p}^r) ,$$

and

$$\mathbb{E}[f(S_{r,i})] \geq (1 - e^{-1}) \cdot \hat{g}(\textstyle\sum_{r'=1}^{r-1} \mathbf{p}^{r'}) - \hat{\ell}(\sum_{r'=1}^{r-1} \mathbf{p}^{r'}) .$$

Proof. Let $R = \{u \in S^* \mid u \notin \textsc{Distorted-Greedy}(\mathcal{N}_{r,i} \cup C_{r-1} \cup \{u\})\}$, and let $O_{r,i}$ be some random subset of S^* to be specified later which includes only elements of $\mathcal{N}_{r,i} \cup C_{r-1} \cup R$. By Lemma 13,

$$\begin{aligned} S_{r,i} &= \textsc{Distorted-Greedy}(\mathcal{N}_{r,i} \cup (\cup_{r'=1}^{r-1} \cup_{i'=1}^{m} S_{r',i'})) \\ &= \textsc{Distorted-Greedy}(\mathcal{N}_{r,i} \cup C_{r-1}) = \textsc{Distorted-Greedy}(\mathcal{N}_{r,i} \cup C_{r-1} \cup R) . \end{aligned}$$

Due to this equality and the fact that $|O_{r,i}| \leq |S^*| \leq k$, the guarantee of Distorted-Greedy (Harshaw et al., 2019, Theorem 3) implies:

$$f(S_{r,i}) = g(S_{r,i}) - \ell(S_{r,i}) \geq (1 - e^{-1}) \cdot g(O_{r,i}) - \ell(O_{r,i}) .$$

Therefore,

$$\begin{aligned}\mathbb{E}[f(S_{r,i})] &\geq \mathbb{E}[(1-e^{-1})\cdot g(O_{r,i}) - \ell(O_{r,i})] = (1-e^{-1})\cdot \mathbb{E}[g(O_{r,i})] - \mathbb{E}[\ell(O_{r,i})] \\ &\geq (1-e^{-1})\cdot \hat{g}(\mathbb{E}[\mathbf{1}_{O_{r,i}}]) - \hat{\ell}(\mathbb{E}[\mathbf{1}_{O_{r,i}}]) \ ,\end{aligned} \tag{6}$$

where the second inequality holds since $\hat{g}$ is convex and $\hat{\ell}$ is linear (see the discussion before Observation 12).

To prove the first part of the lemma, we now choose

$$O_{r,i} = (C_{r-1}\cap S^*)\cup R = (C_{r-1}\cap S^*)\cup \{u\in S^* : u\notin \textsc{Distorted-Greedy}(\mathcal{N}_{r,i}\cup C_{r-1}\cup\{u\})\} \ .$$

One can verify that this choice obeys our assumptions about $O_{r,i}$; and moreover, since the distribution of $\mathcal{N}_{r,i}$ is the same as that of $\mathcal{N}(1/m)$, we get:

$$\Pr[u\in O_{r,i}] = 1-\Pr[u\notin O_{r,i}] = 1-p_u^r \quad \forall\, u\in S^* \qquad \text{and} \qquad \mathbb{E}\left[\mathbf{1}_{O_{r,i}}\right] = \mathbf{1}_{S^*} - \mathbf{p}^r \ .$$

The first part of the lemma now follows by combining the last equality with Inequality (6).

To prove the second part of this lemma, we choose $O_{r,i} = C_{r-1}\cap S^*$. One can verify that this choice again obeys our assumptions about $O_{r,i}$; and moreover, by Lemma 14, $\mathbb{E}[\mathbf{1}_{O_{r,i}}] = \sum_{r'=1}^{r-1}\mathbf{p}^{r'}$. The second part of the lemma now follows by combining this equality with Inequality (6). □

We are now ready to prove Theorem 3.

Proof of Theorem 3. Let D be the output set of Algorithm 4. The definition of D and Lemma 15 together guarantee that for every $1\leq r\leq \varepsilon^{-1}-1$ we have

$$\mathbb{E}[f(D)] \geq \mathbb{E}[f(S_{r,1})] \geq (1-e^{-1})\cdot\hat{g}(\mathbf{1}_{S^*}-\mathbf{p}^r) - \hat{\ell}(\mathbf{1}_{S^*}-\mathbf{p}^r) \ ,$$

and additionally,

$$\mathbb{E}[f(D)] \geq \mathbb{E}[f(S_{1/\varepsilon,1})] \geq (1-e^{-1})\cdot\hat{g}(\textstyle\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r) - \hat{\ell}(\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r) \ .$$

Therefore,

$$\begin{aligned}\mathbb{E}[f(D)] &\geq \varepsilon\cdot\sum_{r=1}^{1/\varepsilon-1}[(1-e^{-1})\cdot\hat{g}(\mathbf{1}_{S^*}-\mathbf{p}^r) - \hat{\ell}(\mathbf{1}_{S^*}-\mathbf{p}^r)] + \varepsilon[(1-e^{-1})\cdot\hat{g}(\textstyle\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r) - \hat{\ell}(\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r)] \\ &\geq (1-e^{-1})\cdot\hat{g}\left(\varepsilon\cdot\sum_{r=1}^{1/\varepsilon-1}(\mathbf{1}_{S^*}-\mathbf{p}^r) + \varepsilon\cdot\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r\right) - \hat{\ell}\left(\varepsilon\cdot\sum_{r=1}^{1/\varepsilon-1}(\mathbf{1}_{S^*}-\mathbf{p}^r) + \varepsilon\cdot\sum_{r=1}^{1/\varepsilon-1}\mathbf{p}^r\right) \\ &= (1-e^{-1})\cdot\hat{g}\left((1-\varepsilon)\cdot\mathbf{1}_{S^*}\right) - \hat{\ell}\left((1-\varepsilon)\cdot\mathbf{1}_{S^*}\right) \geq (1-\varepsilon)\cdot\left[(1-e^{-1})\cdot g(S^*) - \ell(S^*)\right] \ ,\end{aligned}$$

where the second inequality holds since $\hat{\ell}$ is linear and $\hat{g}$ is convex, and the last inequality follows again from the linearity of $\hat{\ell}$ and Observation 12 because $f(\varnothing) = g(\varnothing) - \ell(\varnothing) = g(\varnothing) \geq 0$. □

D. Supplementary Material for Section 3

In Section 3, we discussed that the problem of finding the mode of a class of SLC distributions could be reduced to the maximization of a γ-additively weak submodular function. In Appendix D.1, we prove the theoretical guarantees of our proposed algorithms for maximizing a γ-additively weak submodular function. In Section 3.1, we empirically evaluate the performance of DISTORTED-STREAMING on the problem of mode finding for an SLR distribution.

D.1. Proof of Corollaries 7 and 8

In this section, we show how we can maximize a γ-additively weak submodular function ρ in streaming and distributed settings. Towards this goal, we need to find a submodular function Λ that is close to ρ, which is done in Lemma 16. Then, we explain in Lemma 17 how to present Λ as the difference between a monotone submodular function and a linear function,

which allows us to optimize Λ using the results of Theorem 1 and Theorem 3. Finally, we show that even though we use these algorithms to optimize Λ, the solution they provide has a good guarantee with respect to the original γ-additively weak submodular function ρ.

Lemma 16. *For a γ-additively weak submodular function ρ, the function $\Lambda(S) \triangleq \rho(S) - \frac{\gamma}{2} \cdot |S| \cdot (|S| - 1)$ is submodular.*

Proof. For every set S and two distinct elements $u, v \notin S$, the γ-additively weak submodularity of ρ implies

$$\rho(S) + \rho(S \cup \{u, v\}) \leq \gamma + \rho(S \cup \{u\}) + \rho(S \cup \{v\}) \ .$$

Rearranging this inequality now gives

$$\begin{aligned}\rho(S) - \frac{\gamma \cdot |S| \cdot (|S| - 1))}{2} + \rho(S \cup \{u, v\}) - \frac{\gamma \cdot (|S| + 2) \cdot (|S| + 1)}{2} \\ \leq \rho(S \cup \{u\}) + \rho(S \cup \{v\}) - \frac{2 \cdot \gamma \cdot (|S| + 1) \cdot |S|}{2} \ ,\end{aligned}$$

which, by the definition of Λ, is equivalent to

$$\Lambda(S) + \Lambda(S \cup \{u, v\}) \leq \Lambda(S \cup \{u\}) + \Lambda(S \cup \{v\}) \ . \qquad \square$$

Now, let us define the modular function $\ell(S) = \sum_{u \in S} \ell_u$, where $\ell_u \triangleq \max\{\Lambda(\mathcal{N} \setminus u) - \Lambda(\mathcal{N}), 0\} = \max\{\rho(\mathcal{N} \setminus u) - \rho(\mathcal{N}) + \gamma \cdot (|\mathcal{N}| - 1), 0\}$.

Lemma 17. *The function $g(S) \triangleq \Lambda(S) + \ell(S)$ is monotone and submodular. Furthermore, if $\rho(\varnothing) \geq 0$, then $g(S)$ is also non-negative because $\ell(\varnothing) = 0$.*

Proof. To see that $g(S)$ is submodular, recall that $\Lambda(S)$ is submodular and that the summation of a submodular function with a modular function is still submodular. To prove the monotonicity of $g(S)$, we show that for all sets $S \subseteq \mathcal{N}$ and elements $u \in \mathcal{N} \setminus S$: $g(u \mid S) \geq 0$.

$$\begin{aligned}g(u \mid S) = \Lambda(u \mid S) + \ell(u \mid S) = \Lambda(u \mid S) + \ell_u = \Lambda(u \mid S) + \max\{\Lambda(\mathcal{N} \setminus u) - \Lambda(\mathcal{N}), 0\} \\ \geq \Lambda(u \mid S) + \Lambda(\mathcal{N} \setminus u) - \Lambda(\mathcal{N}) = \Lambda(u \mid S) - \Lambda(u \mid \mathcal{N} \setminus u) \geq 0 \ ,\end{aligned}$$

where the last inequality follows from the submodularity of Λ. $\square$

We now show that by optimizing Λ under a cardinality constraint k, by using either our proposed streaming or distributed algorithms (depending on whether the setting is a streaming or a distributed setting), we can get a scalable solution with a guarantee with respect to ρ, and in particular, a guarantee for the task of finding the mode of an SLC distribution.

Corollary 7. *Assume $\rho \colon 2^{\mathcal{N}} \to \mathbb{R}$ is a γ-additively weak submodular function. Then, when given Λ as the objective function,* DISTORTED-DISTRIBUTED-GREEDY *(Algorithm 4) returns a solution R such that*

$$\mathbb{E}[\rho(R)] \geq (1 - \varepsilon) \cdot \left[(1 - e^{-1}) \cdot \rho(\texttt{OPT}) - e^{-1} \cdot \ell(\texttt{OPT})\right] - \frac{\gamma \cdot [(1 - e^{-1}) \cdot l \cdot (l - 1) - \mathbb{E}[|R| \cdot (|R| - 1)]]}{2} \ ,$$

where $\texttt{OPT} \in \arg\max_{|S| \leq k} \rho(S)$ and $l = |\texttt{OPT}| \leq k$.

Proof. Using the guarantee of Theorem 3 for the performance of DISTORTED-DISTRIBUTED-GREEDY for maximizing the function $\Lambda(S) = g(S) - \ell(S)$ in the distributed setting under a cardinality constraint k, we get

$$\mathbb{E}[g(R) - \ell(R)] \geq (1 - \varepsilon) \cdot \left[(1 - e^{-1}) \cdot g(\texttt{OPT}) - \ell(\texttt{OPT})\right] \ ,$$

which implies, by the definition of g,

$$\frac{\mathbb{E}[\Lambda(R)]}{1 - \varepsilon} \geq (1 - e^{-1}) \cdot (\Lambda(\texttt{OPT}) + \ell(\texttt{OPT})) - \ell(\texttt{OPT}) = (1 - e^{-1}) \cdot \Lambda(\texttt{OPT}) - e^{-1} \cdot \ell(\texttt{OPT}) \ .$$

Using the definition of Λ now, we finally get

$$\mathbb{E}[\rho(R)] \geq (1-\varepsilon)\cdot\left[(1-e^{-1})\cdot\rho(\texttt{OPT}) - e^{-1}\cdot\ell(\texttt{OPT})\right] \\ - \frac{\gamma\cdot[(1-\varepsilon)\cdot(1-e^{-1})\cdot|\texttt{OPT}|\cdot(|\texttt{OPT}|-1) - \mathbb{E}[|R|\cdot(|R|-1)]]}{2} \ ,$$

which implies the corollary since $(1-e^{-1})\cdot|\texttt{OPT}|\cdot(|\texttt{OPT}|-1)$ is non-negative. □

Corollary 8. *Assume $\rho\colon 2^{\mathcal{N}}\to\mathbb{R}$ is a γ-additively weak submodular function. Then, when given Λ as the objective function,* THRESHOLD-STREAMING *(Algorithm 1) returns a solution R such that*

$$\rho(R) \geq (h(r)-\varepsilon)\cdot\rho(\texttt{OPT}) - \left(\frac{\sqrt{4r^2+1}-1}{2}+\varepsilon\right)\cdot\ell(\texttt{OPT}) - \frac{\gamma\cdot[(h(r)-\varepsilon)\cdot l\cdot(l-1) - |R|\cdot(|R|-1)]}{2} \ ,$$

where $\texttt{OPT}\in\arg\max_{|S|\leq k}\rho(S)$ and $l=|\texttt{OPT}|\leq k$.

Proof. By Theorem 1,

$$g(R) - \ell(R) \geq (h(r)-\varepsilon)\cdot g(\texttt{OPT}) - r\cdot\ell(\texttt{OPT}) \ ,$$

which implies, by the definition of g,

$$\Lambda(R) \geq (h(r)-\varepsilon)\cdot(\Lambda(\texttt{OPT})+\ell(\texttt{OPT})) - r\cdot\ell(\texttt{OPT}) = (h(r)-\varepsilon)\cdot\Lambda(\texttt{OPT}) - (r-h(r)+\varepsilon)\cdot\ell(\texttt{OPT}) \ .$$

Using the definition of Λ now, we finally get

$$\rho(R) \geq (h(r)-\varepsilon)\cdot\rho(\texttt{OPT}) - (r-h(r)+\varepsilon)\cdot\ell(\texttt{OPT}) \\ - \frac{\gamma\cdot[(h(r)-\varepsilon)\cdot|\texttt{OPT}|\cdot(|\texttt{OPT}|-1) - |R|\cdot(|R|-1)]}{2} \ .$$

The corollary now follows by observing that $r-h(r)=\frac{\sqrt{4r^2+1}-1}{2}$. □