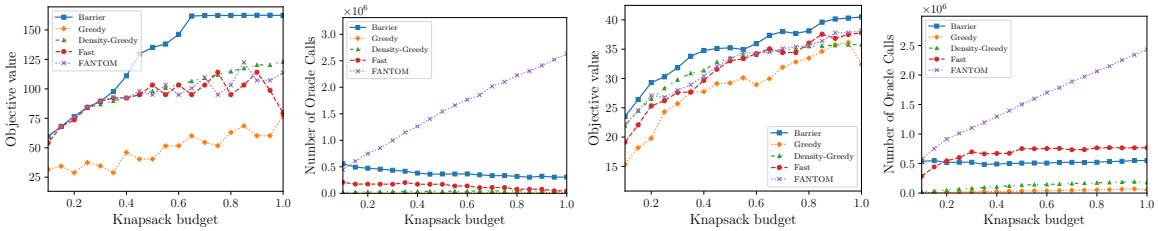
We thank all the reviewers for their constructive comments. We are also encouraged that the reviewers "like[d] the main ideas of [our] paper", mentioned that we study an "important problem", stated that is a "very strong theoretical result" with "comprehensive experiment for practical performance", and added that "the paper is well written". In this response, we address the specific questions asked by each reviewer one by one.

Reviewer #1 Q. Why Barrier-Greedy is not compared to FANTOM? We had performed the first set of experiments where FANTOM was included. We decided to exclude FANTOM for two main reasons: (i) The first iteration of FANTOM is similar to FAST, where FANTOM uses an iterated greedy algorithm with density threshold to guarantee the performance of the algorithm for non-monotone submodular functions. As a result, the performance of the two algorithms would be quite close for monotone submodular functions (See Figure 1a and 1c). (ii) Due to the several iterations of the iterated greedy in FANTOM, its computational complexity highly increases (See Figures 1b and 1d) without any gained benefit for the utility. Given these, we thought that for a more fair comparison it is better to not report the performance of FANTOM. Definitely, in the revised version, we can report the results of FANTOM for all our experiments.



(a) Figure 1a of the paper (b) Figure 1c of the paper (c) Figure 2a of the paper (d) Figure 2c of the paper

Figure 1: A few sample experiments from the paper where FANTOM is included.

Reviewer #1 Q. The experiments are designed so that r is small and the running time looks linear. We agree that generally r could be in the order of n. In that case, the term r^3 dominates the term nr and the linearity argument is not valid anymore. We will clarify this in the revised version.

Reviewer #1 Q. Compare Barrier-Heuristic with Barrier-Greedy in the experimental section. We will add the experimental results for Barrier-Greedy in applications with more than one knapsack constraint. In our initial experiments, we do not observe a dramatic reduction and Barrier-Greedy is at least as good as the other baseline algorithms.

Reviewer #1 Q. Parameterizing the running time with the size of a feasible solution is not very common. We will state the time complexities in terms of n and will add a corollary for cases with $r \ll n$.

Reviewer #2 Q. This algorithm is not useful in only the matroid case. Moreover, there is no guarantee in the only knapsack case. We should point out that the power of our algorithm is in solving the monotone submodular maximization problem subject to the intersection of these two constraints. Indeed, the guarantee of our algorithm for only k-matroids is $(k+1)$ (which is slightly worse than greedy). For the case of only ℓ knapsacks, the approximation factor is $\ell + 1$. We will add this discussion to the revised version of the paper.

Reviewer #2 Q. Importance of the constraints studied in this paper. We believe the problem of maximizing a submodular function subject to the interaction of k-matroids and ℓ-knapsack constraints has both theoretical and practical importance. Indeed, in many optimization tasks that are subject to some structural constraints, we have a limited budget. The combination of matroids and knapsack constraints is a natural way to model these problems. Along with our work, Mirzasoleiman et al. [30] used the same set of constraints to model recommendation systems, image summarization, and revenue maximization tasks.

Reviewer #2 Q. Add a table contrasting their results with earlier results and comparing the running times. We will add a table to compare our results with previous works.

Reviewer #2 Reviewer #3 Q. Scalability of Barrier-Greedy and Barrier-Greedy++. Although our algorithms are not completely suitable for very large scale problems (at least theoretically), we provide the state of the art result that lies in between fast algorithms with suboptimal approximation ratios and slower algorithms. We should mention that Barrier-Greedy and Barrier-Heuristic algorithms are quite fast in practice (they are comparable with FAST) and this makes them applicable in practical scenarios. We also agree that Barrier-Greedy++ is not scalable enough to be used for large scale applications. The main purpose of Barrier-Greedy++ is to provide a tighter approximation guarantee at the expense of a higher computational cost. Indeed, there is a trade-off between the quality of the solution we desire to obtain (higher for Barrier-Greedy++) and the time we are willing to spend (faster for Barrier-Greedy).

Reviewer #4 Q. Access to "proof sketches" to see how things are proved, instead of putting the whole proof in the appendix. We will add a more detailed proof sketch to the main text.