Faster Approximation Algorithm for the k-Regret Minimizing Set and Related Problems*

Nirman Kumar† Stavros Sintos‡

Abstract

Efficient multi-criteria decision making often requires looking at a small set of representative objects from a large collection. A recently proposed method for finding representative objects is the k-regret minimizing set (k-RMS problem). Intuitively, given a large set of objects (points) in d dimensions, the goal is to choose a small representative subset, such that for every user preference, there is always an object in the subset whose preference score is not much worse than the score of the k-th most preferred object in the original set. We propose two new efficient approximation algorithms for the k-regret minimizing set problem with provable theoretical guarantees. Our algorithms improve on the space and time complexities of previous approximation algorithms for the k-RMS problem. In addition, we run extensive experiments on real and synthetic data sets showing that simple modifications of our theoretical algorithms run significantly faster than the previous implementations of the k-RMS problem. Finally, we present an efficient approximation algorithm with theoretical guarantees for an extension of the k-RMS problem, which is called the Top-k regret minimizing set problem.

1 Introduction

Imagine a user querying amazon.com for 'digital camera'. Such a search query can easily bring up thousands of results, displaying which takes hundreds of pages. The website would like to present the best choices in the first few pages, such that almost all users find what they are looking for. However, given the many parameters on which a digital camera depends, it is a complex problem to determine which are more important than others. This is an example of the problem of multi-criteria decision making. Assuming that the parameters of the objects (digital cameras) are numerical, there are two common approaches taken to present users with representative choices. The first one is to fix a ranking function for the objects, and present the top-k objects in order of decreasing scores for some number k. The second one is to present the user with a *pareto-optimal set* or the *skyline*, i.e., a subset of the objects such that no other object in the set is strictly better in all parameters than any in the subset. Both of these have some shortcomings. The top-k method is tied with a specific ranking function, and it may be difficult to choose the 'correct' ranking function. The skyline method has the disadvantage that an extremely large subset of the objects can be on the skyline, particularly so in high dimensions, i.e., the more the number of parameters the less effective it becomes. To alleviate these shortcomings multiple approaches have been proposed. Agarwal et al. [1] introduced the notion of *coreset* which is a small representative set of the data that capture the geometric properties of the original set. In particular, they find a set of objects that approximate the *width* of the set for any user preference. Nanongkai et al. [17] suggested the 1-regret minimizing set approach: users can have different preference functions, but, for a given tolerance of $x\%$, it is desired to compute a subset of objects such that for every user, at least one of the subset is within $x\%$ of her top choice. This was generalized later to the k regret minimizing set [11], where the preference scores of the users top-k choice, is compared with the best in the subset. Following the definition of k-regret minimizing set in [11], we propose faster approximation algorithms for finding small representative sets.

Problem formulation. Let P be a set of n points (objects) in d dimensions, where d is a constant. Each point $p = (p_1, \ldots, p_d)$ in P has non-negative coordinates, i.e., $p_i \geq 0$ for every $i \leq d$. Let $\mathbb{X} = \{x \in \mathbb{R}^d \mid x_i \geq 0 \ \ \forall i\}$ be the positive (first) orthant of $\mathbb{R}^d$. Notice that $\mathsf{P} \subset \mathbb{X}$. A user preference (vector) is represented as a point $u = (u_1, \ldots, u_d) \in \mathbb{X}$. One of the most common scoring function in the top-k and k-regret minimizing set queries is a linear function over the parameters of an object. Given a preference $u \in \mathbb{R}^d$, the *score* of an object p is defined as $\omega(u, p) = \langle u, p \rangle = \sum_{i=1}^{d} u_i p_i$.

For a vector $u \in \mathbb{X}$ and an integer $k \geq 1$, $\varphi_k(u, \mathsf{P})$ denotes the point p in P with the k-th largest score and $\omega_k(u, \mathsf{P})$ denotes its score. Let $\Phi_k(u, \mathsf{P}) = \{\varphi_j(u, \mathsf{P}) \mid 1 \leq j \leq k\}$ be the set of the k top points with respect to preference u.

*Work by Sintos is supported by NSF under grants CCF-15-13816, CCF-15-46392, and IIS-14-08846, by ARO grant W911NF-15-1-0408, and by Grant 2012/229 from the U.S.-Israel Binational Science Foundation.

†Department of Computer Science, University of Memphis

‡Department of Computer Science, Duke University

For a subset $\mathsf{Q} \subseteq \mathsf{P}$ and a preference u, define the regret of Q for preference u, denoted by $\ell_k(u, \mathsf{Q}, \mathsf{P})$, as $\ell_k(u, \mathsf{Q}, \mathsf{P}) = \frac{\max\{0, \omega_k(u,\mathsf{P}) - \omega_1(u,\mathsf{Q})\}}{\omega_k(u,\mathsf{P})}$. Intuitively, this is the relative error of $\omega_k(u, \mathsf{P})$ and $\omega_1(u, \mathsf{Q})$. The maximum *regret ratio* of Q is defined as $\ell_k(\mathsf{Q}, \mathsf{P}) = \max_{u \in \mathbb{X}} \ell_k(u, \mathsf{Q}, \mathsf{P})$. A set Q is a (k, ϵ)-regret set of P if and only if $\ell_k(\mathsf{Q}, \mathsf{P}) \leq \epsilon$.

It is well known that $\ell_k(\mathsf{Q}, \mathsf{P})$ is a monotone decreasing (if $A \subseteq B \subseteq \mathsf{P}$, then $\ell_k(A, \mathsf{P}) \geq \ell_k(B, \mathsf{P})$) and scale invariant ($\ell_k(tu, \mathsf{Q}, \mathsf{P}) = \ell_k(u, \mathsf{Q}, \mathsf{P})$ for $t \geq 0$) function. Because of the scale invariance, without loss of generality, we can consider the users' preferences as unit vectors and define the set $\mathbb{U} = \{u \in \mathbb{X} \mid \|u\| = 1\}$.

The goal is to compute a small subset $\mathsf{Q} \subseteq \mathsf{P}$ with small regret ratio, which we refer to as the k-RMS problem, or k-regret minimizing set problem. Given a set P of n points in $\mathbb{R}^d$, two versions of the k-RMS problem have been studied: (i)min-*size*: Given a parameter $\epsilon > 0$, compute a smallest size subset with regret ratio at most ϵ: $\text{argmin}_{\mathsf{Q} \subseteq \mathsf{P}: \ell_k(\mathsf{Q}) \leq \epsilon} |\mathsf{Q}|$. (ii) min-*error*: Given an integer r, compute a subset of P of size r that minimizes the regret ratio: $\text{argmin}_{\mathsf{Q} \subseteq \mathsf{P}: |\mathsf{Q}| \leq r} \ell_k(\mathsf{Q}, \mathsf{P})$.

We focus on the min-size version of the problem. Throughout, we assume that s_ϵ is the smallest size of a (k, ϵ)-regret of point set P. All our presented algorithms for the min-size version of the k-RMS problem can be extended efficiently to the min-error version by running a binary search on the different regret ratios (at most n different regret ratios).

Top-k RMS. A generalization of the k-RMS problem, proposed in [4, 23]. The goal is to minimize the dissatisfaction of the k-th top tuple in Q versus the k-th top tuple in P, i.e. find a subset $\mathsf{Q} \subseteq \mathsf{P}$ such that $\omega_k(u, \mathsf{Q}) \geq (1 - \epsilon)\omega_k(u, \mathsf{P})$, for any $u \in \mathbb{X}$. In [23] the authors showed that there is always such a set of size $O(\frac{k}{\epsilon^{(d-1)/2}})$ that can be found in $O(n \log n + \frac{k}{\epsilon^{3d/2}})$ time.

Here, we define and solve a more general problem. The Top-k RMS problem is defined as follows: Given a set of n points $\mathsf{P} \subseteq \mathbb{X}$, an integer $k \geq 1$, and a parameter $\epsilon \in (0, 1)$, the goal is to find a set $\mathsf{Q} \subseteq \mathsf{P}$ of minimum size such that $\omega_i(u, \mathsf{Q}) \geq (1 - \epsilon)\omega_i(u, \mathsf{P})$ for all $u \in \mathbb{X}$ and for every $1 \leq i \leq k$. Namely, the goal is to find a subset $\mathsf{Q} \subseteq \mathsf{P}$ to approximate top-k queries [23]. A set $\mathsf{Q} \subseteq \mathsf{P}$ with this property is called a (k, ϵ)-top regret set of P. Given the parameters ϵ, k, we define $s_{\epsilon,k}$ as the size of the minimum (k, ϵ)-top regret set of P. The algorithm proposed by [23] can also be used for the Top-k RMS problem.

Related work. The 1-RMS problem was first defined by Nanongkai et. al. [17]. Chester et al. [11] extended the notion of 1-regret minimizing set to the k-regret minimizing set. They showed that the problem is NP hard when the dimension d is large and they gave an exact algorithm for $d = 2$. By experiments on real data sets, they also showed that the size of a k-regret set can be much smaller than the size of the minimum 1-regret set.

Recently, three methods have been proposed for the k-RMS problem. One is based on a greedy selection of the next point to add in the subset that reduces the maximum regret ratio [11, 17] maximally. Even though theoretical guarantees for such greedy algorithms have not been shown, they compute small regret sets. The second one is based on the notion of coresets [1], a widely used notion in computational geometric approximation. Papers [3, 7] show that a coreset for the *directional width* is also a valid $(1, \epsilon)$-regret set. While these algorithms are fast, they produce large $(1, \epsilon)$-regret sets of size $O(\frac{1}{\epsilon^{(d-1)/2}})$ and they do not scale efficiently with k. A third method [3, 4], models the k-RMS problem as a set cover or a hitting set problem and uses known approximation algorithms with provable guarantees. In particular, given a parameter $0 < \gamma < 1$, in [3] the authors get a $(k, (1-\gamma)\epsilon+\gamma)$-regret set of size $O(s_\epsilon \log s_\epsilon)$ in $O(\frac{n}{\gamma^{d-1}} \log n \log \frac{1}{\epsilon})$ expected time, while in [4] they get a $(1, (1-\gamma)\epsilon+\gamma)$-regret set of size $O(s_\epsilon \log \frac{1}{\gamma})$ in $O(\frac{n}{\gamma^{d-1}})$ time. Recall that s_ϵ is the minimum size (k, ϵ)-regret set of P.

Different variations of the k-RMS problem have been studied as well. Nanongkai et al. [16] defined the interactive 1-regret minimization problem, where the user can interact with the system saying which tuples are more preferred. Peng and Wong [18] define the notion of happy points, which is a subset of the skyline points, to solve the 1-RMS problem. Catallo et al. [8] use an algorithm to first narrow down the database and then solve the 1-RMS problem. Soma and Yoshida [21] tackle the 1-RMS problem in cases where the regret function is submodular. Finally, Faulkner et al.[12] studied the 1-RMS problem for non-linear utility functions.

Our results.

(i) We show an algorithm that finds a set Q which is a $(k, (1-\gamma)\epsilon+\gamma)$-regret set in $O(n + \frac{k^2}{\gamma^{d-1}} + \frac{k+\log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}})$ expected time. The running time also holds with high probability. The size of Q is $O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $O(s_\epsilon)$ for $d \leq 3$. Notice that for $k = O(\sqrt{n})$ our new algorithm is faster than the best previous algorithm. In practice, k is usually small enough to be considered a constant. Our algorithm needs $O(n + \frac{1}{\gamma^{3(d-1)/2}})$ space, which is smaller than the $O(\frac{n}{\gamma^{d-1}})$ space needed by both approximation algorithms in [3] and [4]. Using this result, we can also get an algorithm for the min-*error* version. Given r, we can compute a set Q of size $O(r \log r)$ with

$\ell_k(\mathsf{Q},\mathsf{P}) \leq (1-\gamma)\ell_r + \gamma$ in $O(n + \frac{k^2}{\gamma^{d-1}} + \frac{k+\log^2\frac{1}{\gamma}}{\gamma^{3(d-1)/2}} \log\frac{k}{\gamma})$ expected time, where ℓ_r is the minimum k-regret ratio of a subset of P of size at most r.

(ii) We can improve the running time of (i), if $k \geq c\ln\frac{n}{\delta}$, for a constant c, and a parameter $\delta \in (0,1)$. We give a probabilistic algorithm that works for $k \geq c\ln\frac{n}{\delta}$ and finds a set Q which is a $(4k,(1-\gamma)\epsilon+\gamma)$-regret set with probability $1-\delta$ in $O(n + \frac{\log^2\left(\frac{n}{\delta}\right)}{\gamma^{d-1}} + \frac{\log\frac{n}{\delta}+\log^2(\frac{1}{\gamma})}{\gamma^{3(d-1)/2}})$ expected time. This algorithm is faster than the previous known algorithms when $k \geq c\ln\frac{n}{\delta}$. The size of Q is $O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $O(s_\epsilon)$ for $d \leq 3$.

(iii) We propose a multi-hitting set based approximation algorithm for the Top-k RMS problem. By modeling the problem as a multi-hitting set problem we show the first algorithm to approximate the optimal (k,ϵ)-top regret set. In particular, we get a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of size $O(s_{\epsilon,k}\log\frac{k}{\gamma})$ in $O(\frac{kn\log n}{\gamma^{d-1}})$ time, under a mild assumption on the point set P.

(iv) We run experiments on real and synthetic data sets comparing our new proposed algorithms with existing known techniques. For the k-RMS problem we found that our new algorithms are faster, and produce small and competitive regret sets compared to the best known algorithms.

For better readability we have moved some the technical proofs to the Appendix, although they are presented in full.

2 Algorithms for RMS problem

In this section we present two efficient algorithms for the k-RMS problem. The first algorithm presented in Subsection 2.1 works for any k, but it is faster and uses less space than the previous known algorithms when $k = o(\sqrt{n})$. In Subsection 2.2, we propose a different algorithm for the k-RMS problem that works for $k \geq c\log\frac{n}{\delta}$, for a sufficiently large constant c and $\delta \in (0,1)$, and runs faster than all known algorithms.

2.1 Algorithm 1. In [3], the authors present an approximation algorithm for the k-RMS problem, based on the hitting set problem. For a parameter $\gamma > 0$, their algorithm outputs a $(k,(1-\gamma)\epsilon+\gamma)$-regret set in expected time $O(\frac{n}{\gamma^{d-1}}\log n\log\frac{1}{\epsilon})$. Notice that the algorithm for the 1-RMS problem presented in [4] has almost the same running time and the same approximation factor.

The algorithms mentioned above spend too much time because all their operations run on the entire input set P. Our main idea is to work with two sketches $\mathsf{P}_1,\mathsf{P}_k$ of the database P instead of the entire set. To construct our sketches of P, we use the notion of (k,ϵ)-kernel introduced in [2]. P_k is used to find quickly the top-k item in each direction, while P_1 is used to construct the sets, for the hitting set instance, faster. We define a set N, which is called $\frac{\gamma}{2d}$-*net*, of $O(\frac{1}{\gamma^{d-1}})$ vectors around the origin, such that for any $v \in \mathbb{X}$ there is $u \in \mathsf{N}$ with angular distance at most $\frac{\gamma}{2d}$. For each $u \in \mathsf{N}$, we find the top-k point in P_k and construct a set that contains all points in P_1 with score at least $(1-\epsilon)\omega_k(u,\mathsf{P})$. Then, we solve the hitting set instance where the elements are the points in P_1 and the sets are the constructed sets for all $u \in \mathsf{N}$.

Notice that if we replace the sketches $\mathsf{P}_1,\mathsf{P}_k$ with P the algorithm described above is the same as the algorithm presented in [3]. The main technical difficulty is to show that using the sketches of P to construct the hitting set instance still allows a $(k,(1-\gamma)\epsilon+\gamma)$-regret set that does not contain many points, which requires new ideas. The set we output has size $O(s_\epsilon\log s_\epsilon)$ for $d \geq 4$ and size $O(s_\epsilon)$ for $d \leq 3$ matching the performance of the algorithm in [3].

Before describing the new algorithm we recall the notion of (k,ϵ)-kernel. Given a set P with n points, a subset $\mathsf{Q} \subseteq \mathsf{P}$ is a (k,ϵ)-kernel if $\omega_a(u,\mathsf{Q}) + \omega_b(-u,\mathsf{Q}) \geq (1-\epsilon)[\omega_a(u,\mathsf{P}) + \omega_b(-u,\mathsf{P})]$, for all $u \in \mathbb{R}^d$ and $1 \leq a,b \leq k$. Since $\mathsf{P} \subset \mathbb{X}$, by adding a dummy point at the origin that is always included in Q, and setting $b = 1$, it is straightforward to see that Q is a valid (k,ϵ)-regret set, as $\omega_1(-u,\mathsf{Q}) = \omega_1(-u,\mathsf{P}) = 0$. As such, $\omega_i(u,\mathsf{Q}) \geq (1-\epsilon)\omega_i(u,\mathsf{P})$, for all $u \in \mathbb{X}$ and $1 \leq i \leq k$. In [2], the authors showed that a (k,ϵ)-kernel Q of size $O(\frac{k}{\epsilon^{(d-1)/2}})$ can be computed in $O(n + \frac{k^2}{\epsilon^{d-1}})$ time. From the discussion above, the same result also holds for finding a (k,ϵ)-regret set.

We now present our algorithm. Let $B \subseteq \mathsf{P}$ be the set of size at most d, that contains for each coordinate j, the point with the highest j coordinate (break ties arbitrarily), and let Basis(P) be the method to find the basis B. Scale(P) is a procedure that scales the set P according to a well known transformation presented in [3, 17]. It is a non-uniform scaling of P that takes $O(n)$ time such that all points lie in the unit hypercube, i.e., $0 \leq p_j \leq 1$, $\forall p \in \mathsf{P}$ and $0 \leq j \leq d$. For the transformation, we divide the j-th coordinate of all points by M_j, the highest j coordinate among all points, for all $j = 1,2,\ldots,d$. Note that after Scale(P) is applied, for each coordinate j there is a point $p_i \in B$ with $p_{ij} = 1$. Finally, let Greedy_HS be the greedy algorithm in [6] for the geometric hitting set problem.

Algorithm 1: KernelHS

1: $B :=$ Basis(P)
2: $\mathsf{P} :=$ Scale(P)
3: $\mathsf{P}_1 := (1,\gamma/3)$-kernel of P
4: $\mathsf{P}_k := (k,\gamma/3)$-kernel of P
5: $\mathsf{N} := \frac{\gamma}{6d}$-net of $\mathbb{U}$

6: $R_u := \{p \in \mathsf{P}_1 \mid \omega(u,p) \geq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P}_k)\}$
7: $\mathcal{R}_{\mathsf{N}} := \{R_u \mid u \in \mathsf{N}\}$
8: $\mathsf{Q}' :=$ GREEDY_HS$(\mathsf{P}_1, \mathcal{R}_{\mathsf{N}})$
9: Return $\mathsf{Q} := \mathsf{Q}' \cup B$

Running time and Space. BASIS and SCALE procedures take $O(n)$ time. The computation of P_1 takes $O(n + \frac{1}{\gamma^{d-1}})$ [1, 9] and the computation of P_k takes $O(n + \frac{k^2}{\gamma^{d-1}})$ [2]. N can be constructed in $O(|\mathsf{N}|) = O(\frac{1}{\gamma^{d-1}})$ time. For each $u \in \mathsf{N}$, $\omega_k(u, \mathsf{P}_k)$ can be found in time linear to $|\mathsf{P}_k|$, i.e., $O(\frac{k}{\gamma^{(d-1)/2}})$, and the set R_u can be found in additional $O(\frac{1}{\gamma^{(d-1)/2}})$ time. Hence $\mathcal{R}_{\mathsf{N}}$ can be constructed in $O(|\mathsf{N}| \times \frac{k}{\gamma^{(d-1)/2}}) = O(\frac{k}{\gamma^{3(d-1)/2}})$ time.
Notice that $|\mathsf{P}_1| = O(\frac{1}{\gamma^{(d-1)/2}})$ and $|\mathcal{R}_{\mathsf{N}}| = O(\frac{1}{\gamma^{d-1}})$, so the greedy algorithm in [6] takes $O\left(\frac{\log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}}\right)$ expected time (the bound on the running time also holds with high probability). Overall, the algorithm runs in $O\left(n + \frac{k^2}{\gamma^{d-1}} + \frac{k+\log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}}\right)$ time. The family of sets $|\mathcal{R}_{\mathsf{N}}|$ has $O(\frac{1}{\gamma^{3(d-1)/2}})$ elements, so the algorithm needs $O(n + \frac{1}{\gamma^{3(d-1)/2}})$ space.

Correctness. For each $u \in \mathsf{N}$, from the definition of R_u, there is a point $q \in \mathsf{Q}'$ such that $\omega(u,q) \geq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P}_k) \geq (1-\gamma/3)^2(1-\epsilon)\omega_k(u,\mathsf{P})$, or equivalently, $\omega_1(u,\mathsf{Q}') \geq (1-\gamma/3)^2(1-\epsilon)\omega_k(u,\mathsf{P})$. Indeed, this is sufficient, as the analysis in [3] shows that $\omega_1(u,\mathsf{Q}) \geq (1-\gamma)(1-\epsilon)\omega_k(u,\mathsf{P})$ for each $u \in \mathbb{X}$, and hence Q is a $(k,(1-\gamma)\epsilon+\gamma)$-regret set of P.
It remains to show that $|\mathsf{Q}| = O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $|\mathsf{Q}| = O(s_\epsilon)$ for $d \leq 3$. We assume that $d \geq 4$ and later we can easily extend to $d \leq 3$. We introduce some useful notation. Let μ be the optimum solution of the hitting set problem on the set system $\{\mathsf{P}_1, \mathcal{R}_{\mathsf{N}}\}$. Equivalently, we define μ_{N} as the minimum (k,ϵ)-regret set of P when we only consider directions in N, as opposed to all possible directions in $\mathbb{X}$. From [6], we know that $|\mathsf{Q}'| = O(|\mu| \log |\mu|)$. It also holds that $|\mu_{\mathsf{N}}| \leq s_\epsilon$, so if we can show that $|\mu| = O(\mu_{\mathsf{N}})$ then the result follows. Indeed, the following lemma, whose proof can be found in Appendix A, shows precisely that.

LEMMA 2.1. $|\mu| \leq (d+1)|\mu_{\mathsf{N}}|$.

Set B contains $O(d)$ points and hence $|\mathsf{Q}| = O(s_\epsilon \log s_\epsilon)$. If $d \leq 3$ then the better bound $|\mathsf{Q}'| = O(|\mu_{\mathsf{N}}|)$ is true, and hence $|\mathsf{Q}| = O(s_\epsilon)$. We conclude to the following theorem.

THEOREM 2.1. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $k \geq 1$ an integer, and $\epsilon > 0$, $\gamma > 0$ two parameters. Let s_ϵ be the minimum size of a (k,ϵ)-regret set of P. A subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in $O(n + \frac{k^2}{\gamma^{d-1}} + \frac{k+\log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}})$ expected time such that Q is a $(k,(1-\gamma)\epsilon+\gamma)$-regret set of P. The size of Q is $O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $O(s_\epsilon)$ for $d \leq 3$. The algorithm needs $O(n + \frac{1}{\gamma^{3(d-1)/2}})$ space.*

Discussion. In practice k is often a constant. In this case, Algorithm 1 is the fastest over all known approximation algorithms for the k-RMS problem. For constant k, the running time is $O(n + \frac{1}{\gamma^{3(d-1)/2}} \log^2 \frac{1}{\gamma})$, while the best known approximation for the k-RMS problem in [3] (and [4] for the 1-RMS problem) runs in $O(\frac{n}{\gamma^{d-1}} \log n \log \frac{1}{\epsilon})$ time. In theory, k can be much larger. If $k = \sqrt{n}$ then the running time becomes $O(\frac{n}{\gamma^{d-1}})$, which is not better than the algorithm presented in [3]. In the next section, we propose an algorithm with a better running time if $k \geq c \log n$ for a constant c. Finally, notice that if $\gamma = \epsilon$, then Q is a $(k, 2\epsilon)$-regret set of P.

min-error version. Let $\ell_r(\mathsf{P})$ be the smallest k-regret ratio of a subset of P with size at most r, and let $\ell_k(\mathsf{Q},\mathsf{P})$ be the k-regret ratio of a set $\mathsf{Q} \subseteq \mathsf{P}$. From [3], we know that we can find a set Q such that $\ell_k(\mathsf{Q},\mathsf{P}) \leq (1-\gamma)\ell_r(\mathsf{P}_k)+\gamma$, by running a binary search on all possible regret ratios. By setting $\gamma \leftarrow \gamma/3$ and following the same argument as in [3, Lemma 8] it is not hard to show that $\ell_r(\mathsf{P}_k) \leq (1-\gamma)\ell_r(\mathsf{P})+\gamma$. We conclude to the following result.

THEOREM 2.2. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $k \geq 1$ an integer, and $r > 0$, $\gamma \in (0,1)$ two parameters. Let $\ell_r(\mathsf{P})$ be the minimum k-regret ratio of a subset of P of size at most r. A subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in $O(n + \frac{k^2}{\gamma^{d-1}} + \frac{k+\log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}} \log \frac{k}{\gamma})$ expected time such that $\ell_k(\mathsf{Q},\mathsf{P}) \leq (1-\gamma)\ell_r(\mathsf{P})+\gamma$. The size of Q is $O(r \log r)$ for $d \geq 4$ and $O(r)$ for $d \leq 3$. The algorithm needs $O(n + \frac{1}{\gamma^{3(d-1)/2}})$ space.*

2.2 Algorithm 2. Algorithm 1 is not very efficient if k is large because of the construction of the $(k,\gamma/3)$-kernel, which takes $O(n + \frac{k^2}{\gamma^{d-1}})$ time. Notice that the (k,γ)-kernel is a stronger tool than what we really need. Indeed, we want to approximate the score of the top-k point in P and not the scores of the top-i points for each $i \leq k$. We sample a subset of points $S \subseteq \mathsf{P}$ such that the score of the top-k point in P can be approximated by the score of the top-k' point in S, for $k' \ll k$, i.e., k' is much smaller than k. Hence, we are able to construct the $(k',\gamma/3)$-kernel of S faster than the $(k,\gamma/3)$-kernel of P.
The algorithm description follows. For a number $0 \leq p \leq 1$, a p-sample set of a point set P is a (random) set $S \subseteq \mathsf{P}$

such that each point of P is added in S with probability p. Let $\gamma, \delta \in (0,1)$ be parameters and $k \leq n$ an integer where $k \geq c \log \frac{n}{\delta}$ (notice that $k \geq c \log n$, assuming $\delta \geq 1/\operatorname{poly}(n)$) for a sufficiently large constant c that we specify in Appendix B.

Algorithm 2: Large$k$KernelHS

$B :=$ Basis(P)
$\mathsf{P} :=$ Scale(P)
$\mathsf{P}_1 :=$ $(1, \gamma/3)$-kernel of P
$p = c\frac{\log \frac{n}{\delta}}{k}$
$S :=$ p-sample set of P
$S' :=$ $(\lceil 2kp \rceil, \gamma/3)$-kernel of S
$\mathsf{N} :=$ $\frac{\gamma}{6d}$-net of $\mathbb{U}$
$R_u := \{p \in \mathsf{P}_1 \,|\, \omega(u,p) \geq (1-\gamma/3)(1-\epsilon)\omega_{\lceil 2kp \rceil}(u, S')\}$
$\mathcal{R}_{\mathsf{N}} := \{R_u \mid u \in \mathsf{N}\}$
$\mathsf{Q}' :=$ Greedy_HS($\mathsf{P}_1, \mathcal{R}_{\mathsf{N}}$)
Return $\mathsf{Q} := \mathsf{Q}' \cup B$

Running time. As we had in Algorithm 1, the Basis and Scale procedures take $O(n)$ time. The set P_1 can be constructed in $O(n + \frac{1}{\gamma^{d-1}})$ time, while the set S' is constructed in $O(n + \frac{(kp)^2}{\gamma^{d-1}})$ time. From the definition of p, notice that $kp = O(\log \frac{n}{\delta})$, so the running time to construct the $(\lceil 2kp \rceil, \gamma/3)$-kernel is $O(n + \frac{\log^2(\frac{n}{\delta})}{\gamma^{d-1}})$. N is constructed in $O(\frac{1}{\gamma^{d-1}})$ time. Set S' has $O(\frac{kp}{\gamma^{(d-1)/2}}) = O(\frac{\log \frac{n}{\delta}}{\gamma^{(d-1)/2}})$ points so we need $O(\frac{\log \frac{n}{\delta}}{\gamma^{(d-1)/2}})$ time to find $\omega_{\lceil 2kp \rceil}(u, S')$ for each $u \in \mathsf{N}$. Set P_1 has $O(\frac{1}{\gamma^{(d-1)/2}})$ points and hence the total time to construct R_u is $O(\frac{\log \frac{n}{\delta}}{\gamma^{(d-1)/2}})$. There are $O(\frac{1}{\gamma^{d-1}})$ directions in N and hence we need $O(\frac{\log \frac{n}{\delta}}{\gamma^{3(d-1)/2}})$ time to construct $\mathcal{R}_{\mathsf{N}}$. Using the greedy algorithm we compute Q' in $O(\frac{1}{\gamma^{3(d-1)/2}} \log^2 \frac{1}{\gamma})$ time, so the overall algorithm runs in $O(n + \frac{\log^2 \frac{n}{\delta}}{\gamma^{d-1}} + \frac{\log \frac{n}{\delta} + \log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}})$ time.

Correctness. In Appendix B we prove the following lemma.

Lemma 2.2. *The set Q computed by Algorithm 2 is a $(4k, (1-\gamma)\epsilon + \gamma)$-regret set of P with probability $1-\delta$. The size of Q is $O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $O(s_\epsilon)$ for $d \leq 3$.*

The final result is captured in the following theorem.

Theorem 2.3. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $\delta, \epsilon, \gamma \in (0,1)$ three real parameters, and an integer $k \geq c \log \frac{n}{\delta}$ for a constant c. Let s_ϵ be the minimum size of a (k,ϵ)-regret set of P. A subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in $O(n + \frac{\log^2 \frac{n}{\delta}}{\gamma^{d-1}} + \frac{\log \frac{n}{\delta} + \log^2 \frac{1}{\gamma}}{\gamma^{3(d-1)/2}})$ expected time such that Q is a $(4k, (1-\gamma)\epsilon + \gamma)$-regret set of P with probability $1-\delta$. The size of Q is $O(s_\epsilon \log s_\epsilon)$ for $d \geq 4$ and $O(s_\epsilon)$ for $d \leq 3$. The algorithm needs $O(n + \frac{1}{\gamma^{3(d-1)/2}})$ space.*

3 Top-k RMS problem

In this section we propose an efficient approximation algorithm for the Top-k RMS problem. Using the results in [2, 23] we can easily get a (k,ϵ)-top regret set $\mathsf{Q} \subseteq \mathsf{P}$ of size $O(\frac{k}{\epsilon^{(d-1)/2}})$ in $O(n + \frac{k^2}{\epsilon^{d-1}})$ time. While it shows that every point set admits a (k,ϵ)-top regret set of small size, a specific instance of P may have a much smaller (k,ϵ)-top regret set. In this section, we present the first efficient scheme to approximate the optimal (k,ϵ)-top regret set, by formulating the Top-k RMS problem problem as a multi-hitting set problem.

An instance of the multi-hitting set problem, $\Sigma = (X, \mathcal{R}, A)$, consists of a set X of objects, a family $\mathcal{R}$ of subsets of X, and an array A of $|\mathcal{R}|$ elements that gives the 'hitting' requirement of each subset; $A[R] \in [1, |R|]$ is an integer for $R \in \mathcal{R}$ that specifies the number of times R must be hit. A subset $H \subseteq X$ is a multi-hitting set for Σ if for each $R \in \mathcal{R}$, there are at least $A[R]$ elements $X_1, X_2, \ldots, X_{A[R]}$ in $H \cap R$. The multi-hitting set problem is to compute a multi-hitting set of the minimum size. The multi-hitting set problem is an NP-Complete problem, and a well-known greedy $O(\log n)$-approximation algorithm is known [20].

Given an instance of the Top-k RMS problem, we give the construction of the multi-hitting set instance. Let $\mathbb{A} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_d\}$ be the set of unit vectors along the axes. Let $C = \min_{u \in \mathbb{A}} \omega_k(u, \mathsf{P})/\omega_1(u, \mathsf{P})$. Given a parameter $\gamma > 0$, let N be a $\frac{C\gamma}{2d}$-net of $\mathbb{X}$ with size $O(\frac{1}{C^{d-1}\gamma^{d-1}})$, where we assume without loss of generality that $\|u\| = 1$ for $u \in \mathsf{N}$. Let Scale(P) be the procedure defined in Section 2 (see also [3]). For each $u \in \mathsf{N}$ we define k sets, $R_u^i = \{p \in \mathsf{P} \mid \omega(u,p) \geq (1-\epsilon)\omega_i(u, \mathsf{P})\}$ with coverage requirement $A[R_u^i] = i$ for $i = 1, 2, \ldots, k$. Let $\mathcal{R}_{\mathsf{N}} = \{R_u^i \mid u \in \mathsf{N}, i \in [1,k]\}$. Notice that $|\mathsf{P}| = n$ and $|\mathcal{R}_{\mathsf{N}}| = O(\frac{k}{C^{d-1}\gamma^{d-1}})$.

We define the k-basis B_k of P by $B_k = \bigcup_{u \in \mathbb{A}} \Phi_k(u, \mathsf{P})$. Notice that $|B_k| \leq d \cdot k$. Let BasisK(P) be the method to compute the k-basis B_k. Let MultiGreedy_HS be the greedy algorithm in [20] for the multi-hitting set problem.

Algorithm 3: TopkRMS

$B_k :=$ BasisK(P)
$\mathsf{P} :=$ Scale(P)
$C = \min_{u \in \mathbb{A}} \frac{\omega_k(u,\mathsf{P})}{\omega_1(u,\mathsf{P})}$
$\mathsf{N} :=$ $\frac{C\gamma}{2d}$-net of $\mathbb{U}$
$R_u^i := \{p \in \mathsf{P} \,|\, \omega(u,p) \geq (1-\epsilon)\omega_i(u,\mathsf{P})\} \forall u \in \mathsf{N},\ i \in [1,k]$
$A[R_u^i] = i,\ \forall u \in \mathsf{N},\ i \in [1,k]$

7: $\mathcal{R}_{\mathsf{N}} := \{R_u^i \mid u \in \mathsf{N}, i \in [1,k]\}$
8: $\mathsf{Q}' :=$ MULTIGREEDY_HS$(\mathsf{P}, \mathcal{R}_{\mathsf{N}}, A)$
9: Return $\mathsf{Q} := \mathsf{Q}' \cup B_k$

Running time. Up to step 4, the algorithm takes $O(n + \frac{1}{C^{d-1}\gamma^{d-1}})$ time. Constructing R_u^i takes $O(n)$ time for each $i = 1, 2, \ldots, k$, so the construction of $\mathcal{R}_{\mathsf{N}}$ takes $O(\frac{kn}{C^{d-1}\gamma^{d-1}})$. The greedy algorithm for the multi-hitting set problem in [20] can be implemented in $O(\frac{kn \log n}{C^{d-1}\gamma^{d-1}})$ time so the total running time of Algorithm 3 is $O(\frac{kn \log n}{C^{d-1}\gamma^{d-1}})$.

Correctness. We first show the following lemma.

LEMMA 3.1. *A solution of the multi-hitting set instance $\Sigma_{\mathsf{N}} = (\mathsf{P}, \mathcal{R}_{\mathsf{N}}, A)$ is a (k,ϵ)-top regret set of P in N, and vice versa.*

Proof. Our proof is by contradiction. Let Q be a multi-hitting set of Σ_{N}. Assume that there is a direction $u \in \mathsf{N}$ and an integer $i \in [1,k]$ such that $\omega_i(u,\mathsf{Q}) < (1-\epsilon)\omega_i(u,\mathsf{P})$. Consider the set $R_u^i \in \mathcal{R}_{\mathsf{N}}$. For each point $p \in R_u^i$, we have that $\omega(u,p) \geq (1-\epsilon)\omega_i(u,\mathsf{P})$ (it follows by definition). Since $A[R_u^i] = i$, there are at least i distinct points $q_1, \ldots, q_i \in \mathsf{Q} \cap R_u^i$, and without loss of generality assume that $\omega(u,q_1) \geq \ldots \geq \omega(u,q_i)$. Hence we have, $\omega_i(u,\mathsf{Q}) \geq \omega(u,q_i) \geq (1-\epsilon)\omega_i(u,\mathsf{P})$, which is a contradiction.
Similarly, we can show the other direction. Let Q be a (k,ϵ)-top regret set of P in N. Assume that Q is not a multi-hitting set of Σ_{N}, so assume that there is a set $R_u^i \in \mathcal{R}_{\mathsf{N}}$ such that there are not at least i points in Q that are contained in R_u^i. Notice that all points $p \in \mathsf{P}$ with $\omega(u,p) \geq (1-\epsilon)\omega_i(u,\mathsf{P})$ belong to set R_u^i. Since Q is a (k,ϵ)-top regret set, for each $j \leq k$, $\omega_j(u,\mathsf{Q}) \geq (1-\epsilon)\omega_j(u,\mathsf{P})$. Thus, for $j = i$, there are at least i points with score greater or equal to $(1-\epsilon)\omega_i(u,\mathsf{P})$. All of these points also belong in R_u^i.

From Lemma 3.1, we have that Q' is a (k,ϵ)-top regret set of P in N. Next, we show that $\mathsf{Q}' \cup B_k$ is a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of P in any direction $\mathbb{X}$. The proof of the next lemma can be found in Appendix C.

LEMMA 3.2. *The set Q computed by Algorithm 3 is a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of P.*

From [20] we have that Q' is a $O(\log \frac{k}{C\gamma})$ approximation of the optimum solution, i.e., $|\mathsf{Q}'| = O(s_{\epsilon,k} \log(\frac{k}{C\gamma}))$, where $s_{\epsilon,k}$ is the smallest possible (k,ϵ)-top regret set of P in N. B_k contains at most $d \cdot k = O(s_{\epsilon,k})$ because $k \leq s_{\epsilon,k}$ in any case. Hence, $|\mathsf{Q}| = O(s_{\epsilon,k} \log(\frac{k}{C\gamma}))$.
Our result is encapsulated in the following theorem.

THEOREM 3.1. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $k \geq 1$ an integer, and $\epsilon, \gamma \in (0,1)$ two parameters. Let $C = \min_{u \in \mathbb{A}} \omega_k(u,\mathsf{P})/\omega_1(u,\mathsf{P})$ and let $s_{\epsilon,k}$ be the minimum size of a (k,ϵ)-top regret set of P. A subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in $O(\frac{kn \log n}{C^{d-1}\gamma^{d-1}})$ time such that Q is a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of P. The size of Q is $O(s_{\epsilon,k} \log \frac{k}{C\gamma})$.*

In many applications C can be considered as a constant number, and hence we have the following.

COROLLARY 3.1. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $k \geq 1$ an integer, and $\epsilon, \gamma \in (0,1)$ two parameters. Let $s_{\epsilon,k}$ be the minimum size of a (k,ϵ)-top regret set of P. If the ratio $\min_{u \in \mathbb{A}} \omega_k(u,\mathsf{P})/\omega_1(u,\mathsf{P})$ is bounded by a constant, then a subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in $O(\frac{kn \log n}{\gamma^{d-1}})$ time such that Q is a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of P. The size of Q is $O(s_{\epsilon,k} \log \frac{k}{\gamma})$.*

Exploiting the geometry. Using the algorithm of Chekuri et. al. [10], for the set multi-cover problem in geometric settings, i.e., when the VC dimension is a constant, we get better results for the approximation ratio. However, their algorithm uses linear programming so the running time is not as efficient. The corollary below summarizes the main gain in approximation ratio.

COROLLARY 3.2. *Let $\mathsf{P} \subset \mathbb{X}$ be a set of n points in $\mathbb{R}^d$, $k \geq 1$ an integer, and $\epsilon, \gamma \in (0,1)$ two parameters. Let $s_{\epsilon,k}$ be the minimum size of a (k,ϵ)-top regret set of P. A subset $\mathsf{Q} \subseteq \mathsf{P}$ can be computed in polynomial time such that Q is a $(k,(1-\gamma)\epsilon+\gamma)$-top regret set of P. The size of Q is $O(s_{\epsilon,k} \log s_{\epsilon,k})$ for $d \geq 4$ and $O(s_{\epsilon,k})$ for $d \leq 3$.*

4 Experiments

We run experiments on real and synthetic data sets to compare the efficiency and the efficacy of Algorithm 1 with the other existing algorithms for the k-RMS problem. The parameter k will be small, thus we use Algorithm 1 and not Algorithm 2 which is for large values of k.
All algorithms are implemented in C++ using a 64-bit machine with four 3600 MHz cores and 16GB of RAM with Ubuntu 14.04.

Algorithms and Implementation. We implement two variations of Algorithm 1, CoresetHS, and HeurHS. Both of the algorithms compute a (k,ϵ)-kernel, $\mathsf{P}' \subseteq \mathsf{P}$ of the input set P and then run the greedy hitting set algorithm considering only points in P'. Our algorithms CoresetHS and HeurHS differ only in the way that they compute a (k,ϵ)-kernel.
We first start by explaining a known practical method for computing a $(1,\epsilon)$-kernel and later we extend it to compute a (k,ϵ)-kernel. A $(1,\epsilon)$-kernel is computed in practice by sampling directions on the unit sphere, in

stages. For a certain number m, we start by taking m points on the unit sphere. Let S be that sample set. As in [1], for each $s \in S$ we add in P' the point $p \in \mathsf{P}$ such that $p = nn(s, \mathsf{P})$, where $nn(s, \mathsf{P})$ is the nearest neighbor of s in the set P. We then check if P' is a $(1, \epsilon)$-kernel by testing using a large set of directions V if the set P' satisfies the conditions of being a $(1, \epsilon)$-kernel. If yes, we stop and output P'. Otherwise, we repeat the sampling process and the nearest-neighbor queries taking $2m$ samples, and so on.

CoresetHS. For CoresetHS, we compute a (k, ϵ)-kernel extending the method of $(1, \epsilon)$-kernels as shown in [2]. CoresetHS runs the algorithm for $(1, \epsilon)$-kernel, k times. More specifically, using m samples on the unit sphere, CoresetHS constructs the set P'_1 with at most m points taking the nearest neighbors in P. Then, it removes P'_1 from P, re-scales the points using the SCALE function (Algorithm 1), and take again m samples to get the next set P'_2. This process is repeated k times, ending with the set $\mathsf{P}' = \bigcup_{i \leq k} \mathsf{P}'_i$. We run the hitting set algorithm on P' constructing a net with m directions and we get the hitting set Q. Using the set V we check if Q is a (k, ϵ)-regret set. If yes, then we return Q. Otherwise, we repeat the process taking $2m$ samples, and so on.

HeurHS. For HeurHS, we show a simpler and practical way to get a (k, ϵ)-kernel, also used in [23]. After taking m samples on the unit sphere, HeurHS finds the $(2k)$-nearest neighbors of each sample in the set P. Let P' be the set with the $(2k)$-nearest neighbors. Notice that it contains at most $2k \cdot m$ points. The algorithm runs the hitting set on P' and finds a set Q. If Q is not a (k, ϵ)-regret set then we repeat the algorithm with $2m$ samples, otherwise, we return the set Q.

We expect that HeurHS would be faster than CoresetHS because the latter re-scales the points k times, and each iteration takes $O(n)$ time. Using the ANN library in C++ for answering k-nearest neighbor queries HeurHS can be implemented very efficiently. On the other hand, CoresetHS may produce smaller regret sets than HeurHS because the way that computes a (k, ϵ)-kernel has nice theoretical guarantees, see [2].

The best known implementation for the k-RMS problem is called HS and is described in [3]. A Greedy algorithm (and its implementation) [11] has also been proposed for the k-RMS problem. Agarwal et al. [3] showed experimentally using real data sets that HS algorithm performs much faster than the Greedy algorithm. Furthermore, the size of the regret sets that the HS algorithm returns is competitive, if not smaller, to the size of the regret set returned by the Greedy algorithm. Similarly, Asudeh et al. [4] showed that their approximation algorithm for the 1-RMS problem, which is essentially the same as HS for $k = 1$, is much faster than the Greedy algorithm presented in [17]. It is clear that Hitting set type of algorithms perform much faster than greedy type algorithms, so we only compare our implementations with HS.

Agarwal et al. [3] showed that their coreset-based algorithm was the fastest over all algorithms for the 1-RMS problem. They used this algorithm for larger k as well. However, coreset-based algorithms produce much larger regret sets than the hitting set based approximation algorithms and are not useful in practice where small regret sets are required.

Datasets. We now describe the datasets used for our experiments.

BB[1]: Each point represents a basketball player with five statistics: points, rebounds, blocks, assists, and fouls. It has 21961 points in 5-d, with 200 points on the skyline. This is a widely used dataset, see for example [3, 4, 11, 13, 14, 22].

ElNino[2]: This dataset contains oceanographic data such as wind speed, water temperature, surface temperature etc., measured by buoys stationed in the Pacific ocean. It has 178080 points in 5-d, with 1183 points on the skyline. It has also been used before, see [3, 11].

AirData[3]: This dataset contains the flight on-time published by the US Department of Transportation. For each flight conducted by the 14 US carriers in January 2015 we keep 7 numerical attributes as in [4]: DepDelay, Taxi-out, Taxi-in, Actual-elapsed-time, Air-time, Distance, Arr-Delay. It has 458311 points in 7-d, with 6439 points on the skyline.

AntiCor: Anti-correlated points (synthetic). This dataset is generated as described in [5]. It has 150000 points in 4-d, with 1685 points on the skyline. It is one of the most used synthetic datasets for skyline, top-k, and regret set problems, see [3, 5, 15, 17, 22].

Uniform: Uniform points in the unit hypercube (synthetic). This is the Independent data set described in [5]. It has 200000 points in 6-d, with 2849 points on the skyline.

Sphere: Points on unit sphere (synthetic). 150000 points in 4-d, with 150000 points on the skyline.

Running time. To compare the efficiency of CoresetHS, HeurHS, and HS algorithms, we follow the procedure in [3]. For regret ratio ϵ we measure the time in seconds that each algorithm needs to find a (k, ϵ)-regret set. We run experiments for $k = 5$ and $k = 20$, while ϵ varies over $\{0.005, 0.007, 0.009, 0.02, 0.04, 0.06, 0.08, 0.1\}$.

[1] databasebasketball.com

[2] archive.ics.uci.edu/ml/datasets/El+Nino

[3] transtats.bts.gov/DL_SelectFields.asp?Table_ID=236&DB_Short_Name=OnTime

Since HeurHS, CoresetHS, and HS are randomized algorithms, for each ϵ we show the average running time across 8 executions.

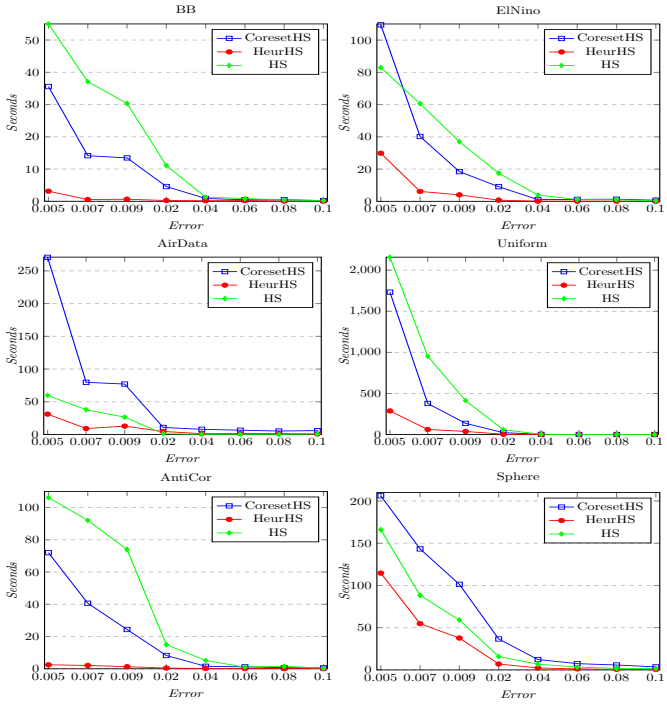

Figure 1: Running time, $k = 5$

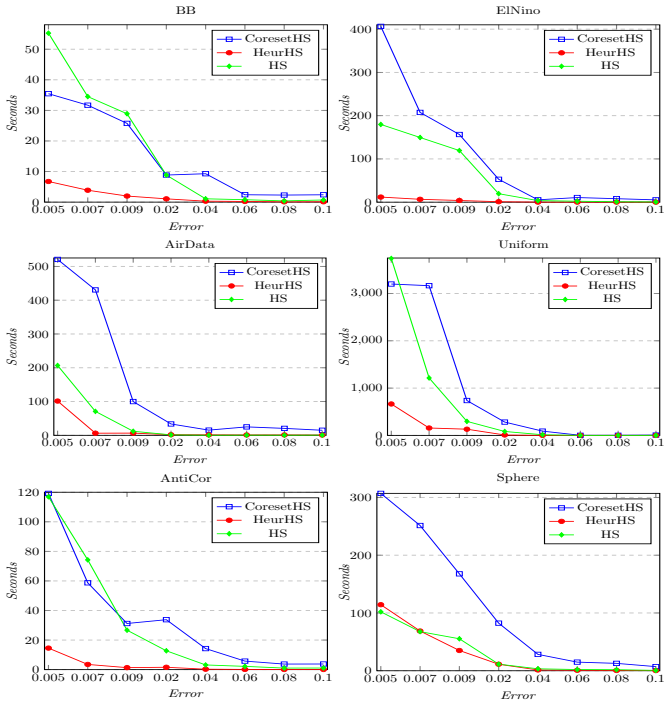

Figure 2: Running time, $k = 20$

Figure 1 shows the running time for all the algorithms over all data sets for $k = 5$. We observe that HeurHS is clearly the fastest over all algorithms for each data set. For some data sets the difference in the running time is remarkable.

For data set BB, HeurHS performs 28 times faster than CoresetHS, and 71 times faster than HS for $\epsilon = 0.007$. On average over all regret ratios (ϵ), HeurHS performs 11 times faster than CoresetHS and 26 times faster than HS. In the synthetic data set Uniform, HeurHS runs 6 (resp. 6.5) times faster than CoresetHS and 7.5 (resp. 17) times faster than HS to get a $(5, 0.005)$-regret set (resp. $(5, 0.02)$-regret set). On average HeurHS is 6.75 times faster than CoresetHS and 7.7 times faster than HS.

We observe a similar behavior for AntiCor data set. Notice that AntiCor is one of the most complicated data sets for finding (k, ϵ) regret sets. The anti-correlated structure of the points in this data set makes the decision of a small representative set a very difficult task. In [3], authors showed that the Greedy algorithm could not find a regret set for such an AntiCor data set even after 2 days of execution. Surprisingly, HeurHS performs very fast for AntiCor data set. On average, HeurHS is 15 times faster than CoresetHS and 30 times faster than HS. Finally, it is worth mentioning that Sphere data set contains the worst case scenario for HeurHS and CoresetHS. The main reason is that all points in Sphere belong in the skyline and the convex hull, and hence the size of (k, ϵ)-kernels is large. Even for this dataset, HeurHS is still the fastest over all algorithms.

As we expected, CoresetHS is not much faster than HS, and in some data sets, AirData, and Sphere, HS runs faster than CoresetHS. As we pointed out, the running time of CoresetHS depends highly on the parameter k and the size of the (k, ϵ)-kernel, and hence it is not always faster than HS.

In Figure 2 we can see the running time for $k = 20$. We observe that as k increases from 5 to 20, the running time of all algorithms increases. However, the running time of HeurHS is not affected as much as the time of CoresetHS and HS. Thus, HeurHS is the fastest even for larger values of k.

Size of regret set generated. An algorithm for k-RMS problem should not be just fast; it also needs to find small regret sets. For example, in [3], coreset-based algorithms perform faster than HS for the 1-RMS problem but compute much larger regret sets. We compare the size of the (k, ϵ)-regret sets found by HeurHS, CoresetHS, and HS algorithms for different values of ϵ.

Figure 3 shows the size of the returned regret sets for $k = 5$. Overall, we can observe that CoresetHS and

HeurHS return slightly smaller regret sets than HS, but the difference is not remarkable. The same observations

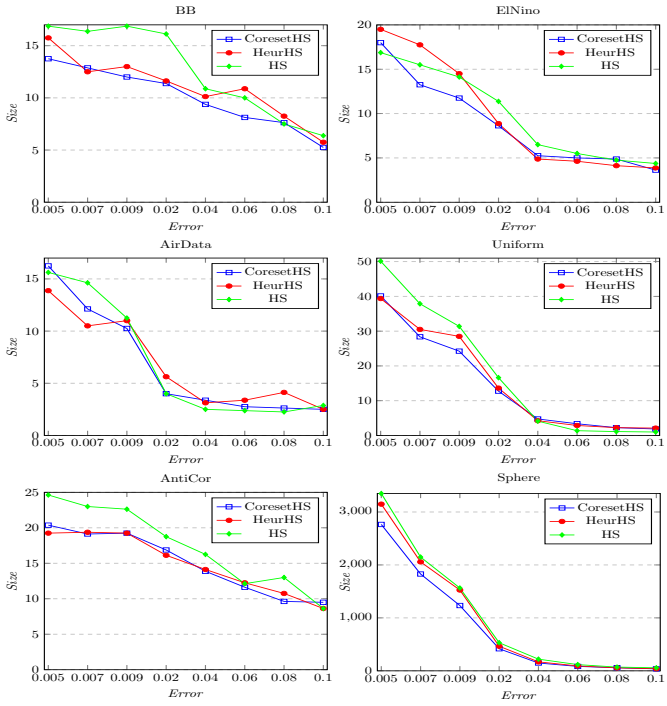

Figure 3: Size of regret sets, $k = 5$

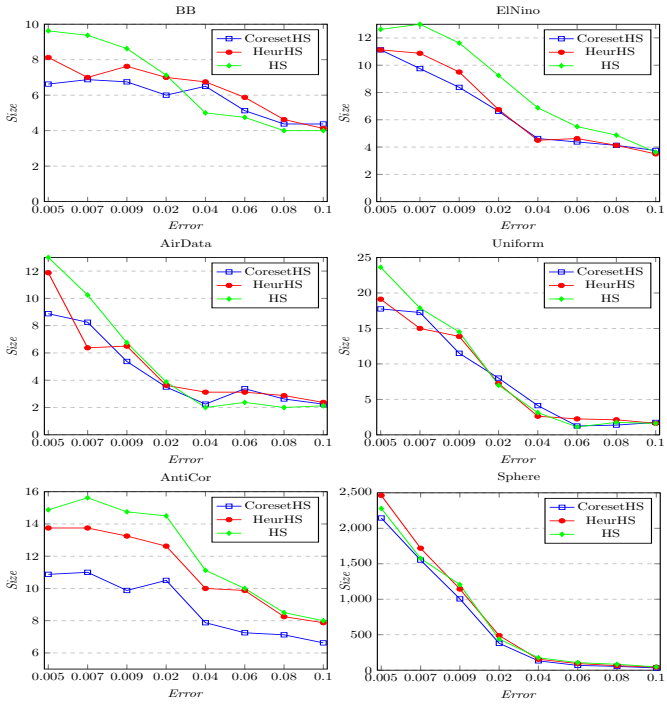

Figure 4: Size of regret sets, $k = 20$

hold when $k = 20$ in Figure 4. Furthermore, as we expected, the size of the regret sets for $k = 20$ is always smaller than the size of the returned regret sets for $k = 5$. We conclude that all the algorithms return comparable sized regret sets.

Summary. We summarize the above discussion into the following conclusion.

> *HeurHS is the best over all algorithms for k-RMS problem. It is more efficient than the other algorithms (CoresetHS and HS), and returns comparable sized regret sets.*

Scalability. We test the scalability of HeurHS, CoresetHS, and HS algorithms as the number of points (n) in a data set increases. For this, we generate two types of synthetic data sets.

First, we generate anti-correlated points in 4 dimensions. (We set the σ parameter equal to 0.05 as described in [5]. This parameter handles the variance of the points.) We fix $\epsilon = 0.01$, and we generate 6 sets of $1000, 10000, 50000, 100000, 500000, 1000000$ points, respectively. Second, we try uniform (independent) points in 5 dimensions. We fix $\epsilon = 0.007$ and we generate 6 data sets similar to the anti-correlated case.

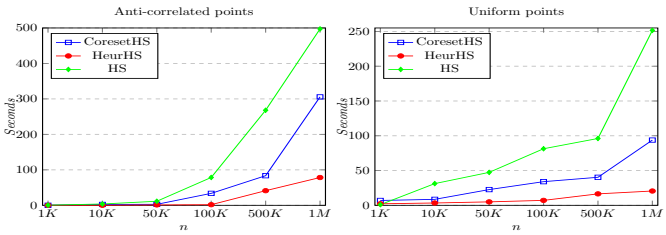

Figure 5: Scalability-Running time, $k = 5$

In Figure 5 we observe that HeurHS is faster than CoresetHS and HS algorithms in all cases. Furthermore, it is clear that as n increases, the running time of HeurHS does not degrade (i.e., increase) as much as the running time of CoresetHS and HS. Similarly, the running time of CoresetHS does not increase as much as that of HS. We conclude:

> *HeurHS is more scalable than CoresetHS and HS. It can be used even with very large data sets.*

5 Conclusion

We presented faster algorithms for the k-RMS problem with theoretical guarantees that improve the running time of the previously known algorithms. By doing simple modifications of our proposed theoretical algorithm, we showed a practical algorithm that works faster than

all previous implementations for the k-RMS problem and finds competitive regret sets. Furthermore, we presented an efficient approximation algorithm for the Top-k RMS problem.

There are still some interesting open problems for finding k-regret minimizing sets. The main question is: can we have an efficient approximation algorithm when the dimension d is not a constant? Notice that we improved the complexity from roughly $O(\frac{n}{\epsilon^{d-1}})$ to $O(n + \frac{1}{\epsilon^{d-1}})$ (skipping polylog(n) factors) but if the dimension d is large then the factor $\frac{1}{\epsilon^{d-1}}$ can be extremely large.

Acknowledgments. We thank Pankaj K. Agarwal, and Subhash Suri for helpful discussions on the problems and the presentation of the paper.

References

[1] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

[2] P. K. Agarwal, S. Har-Peled, and H. Yu. Robust shape fitting via peeling and grating coresets. *Discrete & Computational Geometry*, 39(1-3):38–58, 2008.

[3] P. K. Agarwal, N. Kumar, S. Sintos, and S. Suri. Efficient algorithms for k-regret minimizing sets. In *16th International Symposium on Experimental Algorithms, SEA*, pages 7:1–7:23, 2017.

[4] A. Asudeh, A. Nazi, N. Zhang, and G. Das. Efficient computation of regret-ratio minimizing set: A compact maxima representative. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 821–834. ACM, 2017.

[5] S. Borzsony, D. Kossmann, and K. Stocker. The skyline operator. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 421–430. IEEE, 2001.

[6] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[7] W. Cao, J. Li, H. Wang, K. Wang, R. Wang, R. C.-W. Wong, and W. Zhan. k-regret minimizing set: Efficient algorithms and hardness. In *ICDT 2017-20th International Conference on Database Theory*, 2017.

[8] I. Catallo, E. Ciceri, P. Fraternali, D. Martinenghi, and M. Tagliasacchi. Top-k diversity queries over bounded regions. *ACM Transactions on Database Systems (TODS)*, 38(2):10, 2013.

[9] T. M. Chan. Faster core-set constructions and data stream algorithms in fixed dimensions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 152–159. ACM, 2004.

[10] C. Chekuri, K. L. Clarkson, and S. Har-Peled. On the set multicover problem in geometric settings. *ACM Transactions on Algorithms (TALG)*, 9(1):9, 2012.

[11] S. Chester, A. Thomo, S. Venkatesh, and S. Whitesides. Computing k-regret minimizing sets. *Proceedings of the VLDB Endowment*, 7(5):389–400, 2014.

[12] T. Kessler Faulkner, W. Brackenbury, and A. Lall. K-regret queries with nonlinear utilities. *Proceedings of the VLDB Endowment*, 8(13):2098–2109, 2015.

[13] R. Kulkarni and B. Momin. Parallel skyline computation for frequent queries in distributed environment. In *Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on*, pages 374–380. IEEE, 2016.

[14] R. Kulkarni and B. Momin. Skyline computation for frequent queries in update intensive environment. *Journal of King Saud University-Computer and Information Sciences*, 28(4):447–456, 2016.

[15] M. Morse, J. M. Patel, and W. I. Grosky. Efficient continuous skyline computation. *Information Sciences*, 177(17):3411–3437, 2007.

[16] D. Nanongkai, A. Lall, A. Das Sarma, and K. Makino. Interactive regret minimization. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 109–120. ACM, 2012.

[17] D. Nanongkai, A. D. Sarma, A. Lall, R. J. Lipton, and J. Xu. Regret-minimizing representative databases. *Proceedings of the VLDB Endowment*, 3(1-2):1114–1124, 2010.

[18] P. Peng and R. C.-W. Wong. Geometry approach for k-regret query. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, pages 772–783. IEEE, 2014.

[19] S. Rahul and Y. Tao. Efficient top-k indexing via general reductions. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 277–288. ACM, 2016.

[20] S. Rajagopalan and V. V. Vazirani. Primal-dual rnc approximation algorithms for (multi)-set (multi)-cover and covering integer programs. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pages 322–331. IEEE, 1993.

[21] T. Soma and Y. Yoshida. Regret ratio minimization in multi-objective submodular function maximization. In *AAAI*, pages 905–911, 2017.

[22] A. Vlachou, C. Doulkeridis, Y. Kotidis, and K. Nørvåg. Reverse top-k queries. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 365–376. IEEE, 2010.

[23] A. Yu, P. K. Agarwal, and J. Yang. Processing a large number of continuous preference top-k queries. In *Proceedings of the 2012 ACM SIGMOD international conference on management of data*, pages 397–408. ACM, 2012.

A Proof of Lemma 2.1

The following elementary lemma is required to prove Lemma 2.1 and we omit the proof.

LEMMA A.1. *Let $A \subseteq \mathbb{R}^d$ be a set of points, conv(A) the convex hull of A, and o the origin. It holds that $o \in conv(A)$ iff $\forall u \in \mathbb{R}^d$, $\exists a \in A$ such that $\langle u, a\rangle \geq 0$.*

The next lemma is our main observation.

LEMMA A.2. *For any point $p \in \mathsf{P}$, there is a set $A_p \subseteq \mathsf{P}_1$, with $|A_p| \leq d+1$ such that $\omega_1(u, A_p) \geq (1-\gamma/3)\omega(u,p)$ for all $u \in \mathbb{X}$.*

Proof. We fix a point $p \in \mathsf{P}$ and a point $r \in \mathsf{P}_1$. If $p \in \mathsf{P}_1$, then $A_p = \{p\}$ and the result follows. Next, we consider that $p \notin \mathsf{P}_1$, and hence $r \neq p$. The inequality $\omega(u,r) \geq (1-\gamma/3)\omega(u,p)$ defines a halfspace h^+_{pr} that is defined by a hyperplane h_{pr} that passes through the origin. Indeed, if p_i, r_i, u_i denote the i-th coordinate of point p, point r and vector u, respectively, then we have $\omega(u,r) \geq (1-\gamma/3)\omega(u,p) \Leftrightarrow \langle u, r\rangle \geq (1-\gamma/3)\langle u, p\rangle \Leftrightarrow \langle u, r-(1-\gamma/3)p\rangle \geq 0 \Leftrightarrow \sum_{i=1}^d u_i[r_i-(1-\gamma/3)p_i] \geq 0$. Hence, h_{pr} has the equation $\sum_{i=1}^d u_i[r_i-(1-\gamma/3)p_i] = 0$, and the halfspace h^+_{pr} is defined as the inequality $\sum_{i=1}^d u_i[q_i-(1-\gamma/3)p_i] \geq 0$.

For any direction $u \in \mathbb{X}$ there is a halfspace h^+_{pq} such that $u \in h^+_{pq}$ (by $u \in h^+_{pq}$ we mean that any point along direction u belongs in h^+_{pq}). Indeed, if $p = \varphi_1(u, \mathsf{P})$ then there is a point $q \in \mathsf{P}_1$ such that $\omega(u,q) \geq (1-\gamma/3)\omega(u,p)$, because P_1 is a $(1,\gamma/3)$-kernel, and hence $u \in h^+_{pq}$. If $p \neq \varphi_1(u,\mathsf{P})$ then there is a point $q \in \mathsf{P}_1$ such that $\omega(u,q) \geq (1-\gamma/3)\omega_1(u,\mathsf{P}) \geq (1-\gamma/3)\omega(u,p)$, and hence, again, $u \in h^+_{pq}$.

Let $H_p = \{h^+_{pq} \mid q \in \mathsf{P}_1\}$ be the set of $O(\frac{1}{\gamma^{(d-1)/2}})$ halfspaces. Assume for now that there exists a subset $S_p \subset H_p$ of at most $d+1$ hafspaces such that, for any direction u, there is a halfspace $h^+ \in S_p$ that contains u, i.e., $u \in h^+$. Let $A_p = \{q \mid h^+_{pq} \in S_p\}$. We argue that A_p is the set that we are looking for. Take any direction u and let $u \in h^+_{pq}$, where $h^+_{pq} \in S_p$. It holds that $q \in A_p$ and $u \in h^+_{pq} \Leftrightarrow \omega(u,q) \geq (1-\gamma/3)\omega(u,p)$, and hence the result follows.

To complete the proof, it remains to show the existence of such a set $S_p \subset H_p$. let $\bar{H}_p$ be the set of hyperplanes that define the halfspaces in H_p. For each $h \in \bar{H}_p$ let v_h be the (unit) vector normal to h that lies in the defined halfspace h^+, and let V be the set of all such vectors, i.e., $V = \{v_h \mid h^+ \in H_p\}$. Let P_V be the points defined by the unit vectors in V. We add d points $T = \{t_1, \ldots, t_d\}$ in P_V such that t_i has its i-th coordinate equal to -1 and all the remaining coordinates equal to 0. Notice that for each direction $u \in \mathbb{R}^d$ there is a point $q \in P_V$ such that $\langle u, q\rangle \geq 0$. Indeed, if $u \notin \mathbb{X}$, then there is at least a point $t_i \in T$ such that $\langle u, t_i\rangle \geq 0$ and hence $q = t_i$. If $u \in \mathbb{X}$, we showed above that there is always a point $q \in P_V$ such that $u \in h^+_{pq}$ and hence $\langle u, q\rangle \geq 0$. From Lemma A.1 we know that the origin o lies in the convex hull of P_V. From the well known Caratheodory's theorem, there is a subset of points $C \subset P_V$, with $|C| \leq d+1$, such that $o \in conv(C)$. Let V_C be the corresponding unit vectors of the points C, and let S_p be the corresponding halfspaces of the vectors in V_C. If $t_i \in C$, let $h^+_{pt_i}$ be the halfspace normal to t_i that contains the point t_i and passes through the origin. Let h^+_T be the set of all halfspaces $h^+_{pt_i}$. We know that $|S_p| \leq d+1$ and from Lemma A.1 we can conclude that for any direction $u \in \mathbb{R}^d$, there is a halfspace $h^+ \in S_p$ such that $u \in h^+$. Notice that if $u \in \mathbb{X}$ then $\langle u, t_i\rangle \leq 0$ for each $1 \leq i \leq d$, and hence for each $u \in \mathbb{X}$ there is a halfspace $h^+ \in S_p \backslash h^+_T$ such that $u \in h^+_{pq}$. So, each direction $u \in \mathbb{X}$ is contained in a halfspace in $S_p \setminus h^+_T$, and $|S_p| \leq d+1$. The result follows.

There is one more step to prove Lemma 2.1.

LEMMA A.3. *The set $\hat{\mu} = \bigcup_{p \in \mu_{\mathsf{N}}} A_p$ is a hitting set for the set system $(\mathsf{P}_1, \mathcal{R}_{\mathsf{N}})$.*

Proof. We show that for each $u \in \mathsf{N}$ there exists a point $q \in \hat{\mu}$ such that $q \in R_u$. We fix a direction u. μ_{N} is the minimum (k,ϵ)-regret set of P in N, so there exists a point $\bar{p} \in \mu_{\mathsf{N}}$ such that $\omega(u,\bar{p}) \geq (1-\epsilon)\omega_k(u,\mathsf{P})$. From Lemma A.2, there exists a set $A_{\bar{p}} \subseteq \mathsf{P}_1$ such that $\omega_1(u, A_{\bar{p}}) \geq (1-\gamma/3)\omega(u,\bar{p})$. We argue that $q = \varphi_1(u, A_{\bar{p}})$. Notice that $A_{\bar{p}} \subseteq \mathsf{P}_1$ and R_u contains all points in P_1 with score greater or equal to $(1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P}_k) \leq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P})$. So, if a point $r \in \mathsf{P}_1$ has score $\omega(u,r) \geq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P})$ then $r \in R_u$. We note that $\omega_1(u, A_{\bar{p}}) \geq (1-\gamma/3)\omega(u,\bar{p}) \geq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P})$, and hence $q \in R_u$.

The subset $\hat{\mu} = \bigcup_{p \in \mu_{\mathsf{N}}} A_p$ is a hitting set of $(\mathsf{P}_1, \mathcal{R}_{\mathsf{N}})$ and contains at most $(d+1)|\mu_{\mathsf{N}}|$ points. This completes the proof of Lemma 2.1.

B Proof of Lemma 2.2

First, we show an interesting connection between the sets S, P. We use the following lemma, proved in [19] (Lemma 1).

LEMMA B.1. *Let E be a set of n ranked elements, and R be a p-sample set of E. Suppose that integer $k \geq 1$ and real value $\delta \in (0,1)$ satisfy $kp \geq 3\ln(3/\delta)$ and $n \geq 4k$. Then, the following hold simultaneously with probability at least $1-\delta$, (1) $|R| > 2kp$, (2) The element with rank $\lceil 2kp \rceil$ in R has rank between k and $4k$ in E.*

We generalize the above lemma in our setting. Consider a direction $u \in \mathbb{R}^d$. Points in P can be considered as

ranked items, $p < q$ if $\omega(u,p) < \omega(u,q)$. It is well known that there are at most n^d distinct orderings of the points for all $u \in \mathbb{X}$. Applying a simple union bound on the result of Lemma B.1, applied to the ordering in each direction, we conclude the following.

LEMMA B.2. *Let $\mathsf{P} \subset \mathbb{R}^d$ be a set of n points. Let S be a p-sample set of P. Suppose that integer $k \geq 1$ and real value $\delta \in (0,1)$ satisfy $kp \geq 3\ln\frac{3n^d}{\delta}$ and $n \geq 4k$. Then the following hold simultaneously with probability at least $1-\delta$, (1) $|S| > 2kp$, (2) for any $u \in \mathbb{X}$, $\omega_{4k}(u,\mathsf{P}) \leq \omega_{\lceil 2kp\rceil}(u,S) \leq \omega_k(u,\mathsf{P})$.*

Lemma B.2 implies that with probability $1-\delta$, the $\lceil 2kp\rceil$-level of S lies between the k-th and the $4k$-th level of P. For the remainder of this section we assume the conditions given in Lemma B.2 do occur. Therefore, the analysis of the approximation factor and the size of Q below, hold true with probability at least $1-\delta$.

We first analyze the approximation factor. For each $u \in \mathsf{N}$ there will be a point in $p \in \mathsf{Q}'$ such that $\omega(u,p) \geq (1-\gamma/3)(1-\epsilon)\omega_{\lceil 2kp\rceil}(u,S') \geq (1-\gamma/3)^2(1-\epsilon)\omega_{\lceil 2kp\rceil}(u,S) \geq (1-\gamma/3)^2(1-\epsilon)\omega_{4k}(u,\mathsf{P})$. The first inequality follows from the definition of R_u, the second inequality follows from the definition of a $(\lceil 2kp\rceil, \gamma/3)$-kernel, and the third inequality holds due to Lemma B.2. Since $\omega_1(u,\mathsf{Q}') \geq (1-\gamma/3)^2(1-\epsilon)\omega_{4k}(u,\mathsf{P})$ for $u \in \mathsf{N}$, following the analysis of [3] we have that $\omega_1(u,\mathsf{Q}) \geq (1-\gamma)(1-\epsilon)\omega_{4k}(u,\mathsf{P})$ for any $u \in \mathbb{X}$.

We now analyze how large $|\mathsf{Q}|$ is compared to s_ϵ. The analysis is similar to the proof of Lemma 2.1 but we include it here for completeness.

As we defined above, let μ_{N} be the minimum (k,ϵ)-regret set of P for $u \in \mathsf{N}$, and μ be the minimum hitting set of the set system $(\mathsf{P}_1, \mathcal{R}_{\mathsf{N}})$. Fix a direction $u \in \mathsf{N}$. Assume $p \in \mu_{\mathsf{N}}$ such that $\omega(u,p) \geq (1-\epsilon)\omega_k(u,\mathsf{P})$. From Lemma A.2, there is a point $q \in \mathsf{P}_1$ such that $\omega(u,q) \geq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P})$. We claim that $q \in R_u$. Indeed, $q \in \mathsf{P}_1$ and $(1-\gamma/3)(1-\epsilon)\omega_{\lceil 2kp\rceil}(u,S') \leq (1-\gamma/3)(1-\epsilon)\omega_{\lceil 2kp\rceil}(u,S) \leq (1-\gamma/3)(1-\epsilon)\omega_k(u,\mathsf{P})$ and since R_u contains all points in P_1 with score at least $(1-\gamma/3)(1-\epsilon)\omega_{\lceil 2kp\rceil}(u,S')$ we have $q \in R_u$. For each $p \in \mu_{\mathsf{N}}$, there is a set $A_p \subseteq \mathsf{P}_1$ (see proof of Lemma A.2) of at most $d+1$ points such that the union $\bigcup_{p\in\mu_{\mathsf{N}}} A_p$ is a hitting set for the set system $(\mathsf{P}_1, \mathcal{R}_{\mathsf{N}})$. Hence, we have shown that $|\mu| \leq (d+1)|\mu_{\mathsf{N}}| \leq (d+1)s_\epsilon$. Q' is a set with size $O(|\mu|\log|\mu|)$ for $d \geq 4$, so $|\mathsf{Q}'| = O(|\mu_{\mathsf{N}}|\log|\mu_{\mathsf{N}}|) = O(s_\epsilon \log s_\epsilon)$. We also have that $|B| = O(1)$, and hence $|\mathsf{Q}| = O(s_\epsilon \log s_\epsilon)$. For $d \leq 3$, we can conclude that $|\mathsf{Q}| = O(s_\epsilon)$ owing to a better approximation for the hitting set problem.

C Proof of Lemma 3.2

First we need some preparatory lemmas and observations. The proof of the following is an easy adaptation of the proof of Lemma 7 from [3], but we provide it here for completeness.

LEMMA C.1. *Let Q' be a (k,ϵ)-top regret set of P in N, and let B_k be the k-basis of P. Let $\mathsf{Q} = \mathsf{Q}' \cup B_k$. Then, for every direction $u \in \mathbb{U}$, and every integer $1 \leq i \leq k$, there are at least i distinct points $q_1, \ldots, q_i \in \mathsf{Q}$ such that $\omega(u,q_j) \geq (1-\gamma)(1-\epsilon)\omega_i(u,\mathsf{P})$ for every $j = 1,2,\ldots,i$.*

Proof. Fix an integer $i \in [k]$. We first consider the case when $\omega_i(u,\mathsf{P}) \leq \frac{C}{(1-\epsilon)\sqrt{d}}$. In this case, we show that the set B_k is guaranteed to contain i distinct points $q_1,\ldots,q_i$ such that $\omega(u,q_j) \geq \frac{C}{\sqrt{d}}$ for each $1 \leq j \leq i$. Let $u = (u_1,\ldots,u_d)$. Since $\|u\| = 1$ there exists $\ell \leq d$ such that $u_\ell \geq \frac{1}{\sqrt{d}}$. Consider the unit vector $\mathbf{e}_\ell$ along the x_ℓ axis. Notice that $\Phi_k(\mathbf{e}_\ell,\mathsf{P}) \subseteq B_k$ and hence $q_j = \varphi_j(\mathbf{e}_\ell,\mathsf{P}) \in B_k$, for all $1 \leq j \leq k$. Let $q_j = (q_{j1},\ldots,q_{jd})$. By the definition of C, $q_{j\ell} \geq C$, for each $1 \leq j \leq k$, and so $\omega(u,q_j) \geq u_\ell q_{j\ell} = \frac{C}{\sqrt{d}}$, which proves the claim.

Now assume that $\omega_i(u,\mathsf{P}) > \frac{C}{(1-\epsilon)\sqrt{d}}$. Let $\bar{u} \in \mathsf{N}$ be a direction in the net N such that, $(\widehat{u,\bar{u}}) \leq C\gamma/2d$, where $(\widehat{u,\bar{u}})$ is the angle between u and $\bar{u}$. Such a direction exists because N is a $\frac{C\gamma}{2d}$-net on $\mathbb{U}$. Observe that,

$$\|u-\bar{u}\| = \sqrt{2-2\cos((\widehat{u,\bar{u}}))} = 2\sin\left(\frac{(\widehat{u,\bar{u}})}{2}\right) \leq \frac{C\gamma}{2d},$$

where we have used first the cosine rule, the identity $1-\cos\theta = 2\sin^2\left(\frac{\theta}{2}\right)$, as well as the inequality $\sin\theta \leq \theta$ for $\theta \geq 0$ in the final step. Also, observe that for any $p \in \mathsf{P}$ we have,

$$|\omega(u,p) - \omega(\bar{u},p)| \leq \frac{C\gamma}{2\sqrt{d}}. \tag{C.1}$$

This follows because,

$$\begin{aligned}
|\omega(u,p) - \omega(\bar{u},p)| &= |\langle u,p\rangle - \langle \bar{u},p\rangle| \\
&= |\langle u-\bar{u},p\rangle| \\
&\leq \|u-\bar{u}\| \cdot \|p\| \\
&\leq \frac{C\gamma}{2d}\cdot\sqrt{d} \\
&= \frac{C\gamma}{2\sqrt{d}},
\end{aligned}$$

where we have used the Cauchy-Schwarz inequality for the first inequality, the upper bound on $\|u-\bar{u}\|$ derived earlier, along with $\|p\| \leq \sqrt{d}$ (recall P has been scaled so that for every point, any coordinate is at most 1 in value) for the second inequality.

Let $x_1, x_2, \ldots, x_i \in \mathsf{P}$ be the top-i points along direction u, i.e., $x_i = \varphi_i(u, \mathsf{P})$. Also, let y_i be the top-i point along direction $\bar{u}$. As remarked we can assume, $\omega(u, x_i) \geq \frac{C}{(1-\epsilon)\sqrt{d}}$. We have that,

$$\begin{aligned}\omega(\bar{u}, x_i) &\geq \omega(u, x_i) - \frac{C\gamma}{2\sqrt{d}} \\ &\geq \omega(u, x_i) - \frac{(1-\epsilon)\gamma}{2}\omega(u, x_i) \\ &= \omega(u, x_i)\left(1 - \frac{(1-\epsilon)\gamma}{2}\right).\end{aligned}$$

The first inequality follows by *Equation C.*1, and the second inequality holds since $\omega(u, x_i) > \frac{C}{(1-\epsilon)\sqrt{d}}$. This implies that there are at least i points along $\bar{u}$ whose scores are each at least $\omega(u, x_i)\left(1 - \frac{(1-\epsilon)\gamma}{2}\right)$, and therefore the i-th best score along $\bar{u}$, i.e., $\omega(\bar{u}, y_i)$, is at least $\omega(u, x_i)\left(1 - \frac{(1-\epsilon)\gamma}{2}\right)$. Since Q' is a (k, ε)-top regret set of P in N, there are i distinct points $q_1, q_2, \ldots, q_i \in \mathsf{Q}' \subseteq \mathsf{Q}$ such that $\omega(\bar{u}, q_j) \geq (1-\epsilon)\omega(\bar{u}, y_i)$. We claim that all these points "settle" the direction u as well, up-to the factor $(1-\gamma)(1-\epsilon)$. Indeed,

$$\begin{aligned}\omega(u, q_j) &\geq \omega(\bar{u}, q_j) - \frac{C\gamma}{2\sqrt{d}} \\ &\geq (1-\epsilon)\omega(\bar{u}, y_i) - \frac{C\gamma}{2\sqrt{d}} \\ &\geq (1-\epsilon)\left(1 - \frac{(1-\epsilon)\gamma}{2}\right)\omega(u, x_i) - \frac{C\gamma}{2\sqrt{d}} \\ &\geq (1-\epsilon)\left(1 - \frac{(1-\epsilon)\gamma}{2}\right)\omega(u, x_i) \\ &\qquad - \frac{(1-\epsilon)\gamma}{2}\omega(u, x_i) \\ &= (1-\epsilon)(1-\gamma+\gamma\epsilon/2)\omega(u, x_i) \\ &\geq (1-\gamma)(1-\epsilon)\omega(u, x_i).\end{aligned}$$

This completes the proof.

We need the following graph theoretic lemma.

LEMMA C.2. *Let $G = ([k] \cup V, E)$ be a bipartite graph with the partite sets $[k] = \{1, 2, \ldots, k\}$ and V, such that for each $1 \leq i \leq k$, the degree of i is at least i. Then, there is a matching $M \subseteq E$ that saturates $[k]$.*

Proof. We show that the condition of Hall's marriage theorem are met, thus implying the existence of such a matching. Let $W \subseteq [k]$ be any non-empty subset. Clearly W contains some integer (vertex) ℓ with $\ell \geq |W|$. Then, the set of neighbors of W, i.e., $N_G(W)$, includes at least the neighbors of ℓ which are at least $\ell \geq |W|$ by assumption.

LEMMA C.3. *Let Q' be a (k, ϵ)-top regret set of P in N, and let B_k be the k-basis of P. Let $\mathsf{Q} = \mathsf{Q}' \cup B_k$. Then, for every direction $u \in \mathbb{U}$, there exist k distinct points $q_1, \ldots, q_k \in \mathsf{Q}$ such that for each $1 \leq i \leq k$ we have that $\omega(u, q_i) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$.*

Proof. We construct a bipartite graph $G = ([k] \cup \mathsf{Q}, E)$ with partite sets $[k] = \{1, 2, \ldots, k\}$ and Q. For a vertex $i \in [k]$ we connect it to all $q \in \mathsf{Q}$ such that $\omega(u, q) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$. By Lemma C.1, the degree of vertex i is at least i. As such, by Lemma C.2, there is a matching M in G that saturates $[k]$. Thus, there are points $q_1, q_2, \ldots, q_k$, where q_i is the neighbor of i in M. By definition of G, q_i satisfies $\omega(u, q_i) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$.

We can now show what we want. We need to show that for any direction $u \in \mathbb{U}$ and for any integer $i \in [k]$ it holds that $\omega_i(u, \mathsf{Q}) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$. By Lemma C.3, there are k distinct points $q_1, \ldots, q_k$ such that $\omega(u, q_j) \geq (1-\gamma)(1-\epsilon)\omega_j(u, \mathsf{P})$ for all $1 \leq j \leq k$. Now for the set $X = \{q_1, \ldots, q_k\} \subseteq \mathsf{Q}$ since we have at least i distinct points (the points $q_1, \ldots, q_i$) such that $\omega(u, q_j) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$ for $1 \leq j \leq i$, it follows that $\omega_i(u, X) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$. Thus, $\omega_i(u, \mathsf{Q}) \geq \omega_i(u, X) \geq (1-\gamma)(1-\epsilon)\omega_i(u, \mathsf{P})$, and the result of Lemma 3.2 follows.