Composable core-sets for determinant maximization: A simple near-optimal algorithm

Composable Core-sets for Determinant Maximization: A Simple Near-Optimal Algorithm *

Piotr Indyk
MIT
indyk@mit.edu

Sepideh Mahabadi
TTIC
mahabadi@ttic.edu

Shayan Oveis Gharan
University of Washington
shayan@cs.washington.edu

Alireza Rezaei
University of Washington
arezaei@cs.washington.edu

Abstract

"Composable core-sets" are an efficient framework for solving optimization problems in massive data models. In this work, we consider efficient construction of composable core-sets for the *determinant maximization* problem. This can also be cast as the MAP inference task for *determinantal point processes*, that have recently gained a lot of interest for modeling diversity and fairness. The problem was recently studied in [IMGR18], where they designed composable core-sets with the optimal approximation bound of $\tilde{O}(k)^k$. On the other hand, the more practical *Greedy* algorithm has been previously used in similar contexts. In this work, first we provide a theoretical approximation guarantee of $O(C^{k^2})$ for the Greedy algorithm in the context of composable core-sets; Further, we propose to use a *Local Search* based algorithm that while being still practical, achieves a nearly optimal approximation bound of $O(k)^{2k}$; Finally, we implement all three algorithms and show the effectiveness of our proposed algorithm on standard data sets.

1 Introduction

Given a set of vectors $P \subset \mathbb{R}^d$ and an integer $1 \leq k \leq d$, the goal of the determinant maximization problem is to find a subset $S = \{v_1, \ldots, v_k\}$ of P such that the determinant of the Gram matrix of the vectors in S is maximized. Geometrically, this determinant is equal to the volume squared of the parallelepiped spanned by the points in S. This problem and its variants have been studied extensively over the last decade. To this date, the best approximation factor is due to a work of Nikolov [Nik15] which gives a factor of e^k, and it is known that the exponential dependence on k is unavoidable [CMI13].

The determinant of a subset of points is used as a measure of diversity in many applications where a small but diverse subset of objects must be selected as a representative of a larger population [MJK17, GCGS14, KT+12, CGGS15, KT11, YFZ+16, LCYO16]; recently, this has been further applied to model fairness [CDKV16]. The determinant maximization problem can also be rephrased as the maximum a posteriori probability (MAP) estimator for *determinantal point processes* (DPPs). DPPs are probabilistic

*This is an equal contribution paper. This paper has appeared in the 36th International Conference on Machine Learning (ICML), 2019

models of diversity in which every subset of k objects is assigned a probability proportional to the determinant of its corresponding Gram matrix. DPPs have found several applications in machine learning over the last few years [KT+12, MJK17, GCGS14, YFZ+16]. In this context, the determinant maximization problem corresponds to the task of finding the most diverse subset of items.

Many of these applications need to handle large amounts of data and consequently the problem has been considered in massive data models of computation [MJK17, WIB14, PJG+14, MKSK13, MKBK15, MZ15, BENW15]. One strong such model that we consider in this work, is *composable core-set* [IMMM14] which is an efficient summary of a data set with the composability property: union of summaries of multiple data sets should provably result in a good summary for the union of the data sets. More precisely, in the context of the determinant maximization, a mapping function c that maps any point set to one of its subsets is called an α-composable core-set if it satisfies the following condition: given any integer m and any collection of point sets $P_1, \cdots, P_m \subset \mathbb{R}^d$,

$$\text{MAXDET}_k(\bigcup_{i=1}^{m} c(P_i)) \geq \frac{1}{\alpha} \cdot \text{MAXDET}_k(\bigcup_{i=1}^{m} P_i)$$

where we use MAXDET_k to denote the optimum of the determinant maximization problem for parameter k. We also say c is a core-set of size t if for any $P \subset \mathbb{R}^d$, $|c(P)| \leq t$. If designed for a task, composable core-sets will further imply efficient streaming and distributed algorithms for the same task. This has lead to recent interest in composable core-sets model since its introduction [MZ15, AK17, IMGR18].

An almost optimal approximate composable core-set. In [IMGR18], the authors designed composable core-sets of size $O(k \log k)$ with approximation guarantee of $\tilde{O}(k)^k$ for the determinant maximization problem. Moreover, they showed that the best approximation one can achieve is $\Omega(k^{k-o(k)})$ for any polynomial size core-sets, proving that their algorithm is almost optimal. However, its complexity makes it less appealing in practice. First of all, the algorithm requires an explicit representation of the point set, which is not present for many DPP applications; a common case is that the DPP kernel is given by an oracle which returns the inner product between the points; in this setting, the algorithm needs to construct the associated gram matrix, and use SVD decomposition to recover the point set, making the time and memory quadratic in the size of the point-set. Secondly, even in the point set setting, the algorithm is not efficient for large inputs as it requires solving $O(kn)$ linear programs, where n is size of the point set.

In this paper, we focus on two simple to implement algorithms which are typically exploited in practice, namely the Greedy and the Local-Search algorithms. We study these algorithms from theoretical and experimental points of view for the composable core-set problem with respect to the determinant maximization objective, and we compare their performance with the algorithm of [IMGR18], which we refer to as the LP-based algorithm.

1.1 Our Contributions

Greedy algorithm. The greedy algorithm for determinant maximization proceeds in k iterations and at each iteration it picks the point that maximizes the volume of the parallelepiped formed by the set of points picked so far. [ÇMI09] has studied the approximation of the greedy algorithm in the standard setting. In the context of submodular maximization over large data sets, variants of this algorithm have been studied [MKSK13]. One can view the greedy algorithm as a heuristic for constructing a core-set of size k. To the best of our knowledge, the previous analysis of this algorithm does not provide any multiplicative approximation guarantee in the context of composable core-sets.[1]

[1] For more details, see related work.

Our first result shows the first multiplicative approximation factor for composable core-sets on the determinant maximization objective achieved by the Greedy algorithm.

Theorem 1.1. *Given a set of points $P \subset \mathbb{R}^d$, the Greedy algorithm achieves a $O(C^{k^2})$-composable core-set of size k for the determinant maximization problem, where C is a constant.*

The Local Search algorithm. Our main contribution is to propose to use the Local Search algorithm for constructing a composable core-set for the task of determinant maximization. The algorithm starts with the solution of the Greedy algorithm and at each iteration, swaps in a point that is not in the core-set with a point that is already in the core-set, as long as this operation increases the volume of the set of picked points. While still being simple, as we show, this algorithm achieves a near-optimal approximation guarantee.

Theorem 1.2. *Given a set of points $P \subset \mathbb{R}^d$, the Local Search algorithm achieves an $O(k)^{2k}$-composable core-set of size k for the determinant maximization problem.*

Directional height. Both of our theoretical results use a modular framework: In Section 3, we introduce a new geometric notion defined for a point set called *directional height*, which is closely related to the width of a point set defined in [AHPV05]. We show that core-sets for preserving the directional height of a point set in fact provide core-sets for the determinant maximization problem. Finally, we show that running either the Greedy (Section 5) or Local Search (Section 4) algorithms on a point set obtain composable core-sets for its directional height. We believe that this new notion might find applications elsewhere.

Experimental resutls. Finally, we implement all three algorithms and compare their performances on two real data sets: MNIST[LBBH98] data set and GENES data set, previously used in [BQK+14, LJS15] in the context of DPPs. Our empirical results show that in more than 87% percent of the cases, the solution reported by the Local Search algorithm improves over the Greedy algorithm. The average improvement varies from 1% to up to 23% depending on the data set and the settings of other parameters such as k. We further show that although Local Search picks fewer points than the tight approximation algorithm of [IMGR18] (k vs. upto $O(k \log k)$), its performance is better and it runs faster.

1.2 Related Work

In a broader scope, determinant maximization is an instance of the (non-monotone) submodular maximization where the logarithm of the determinant is the submodular objective function. There is a long line of work on distributed submodular optimization and its variants [CKT10, BMKK14, MKSK16, KMVV15]. In particular, there has been several efforts to design composable core-sets in various settings of the problem [MKSK13, MZ15, BENW15]; In [MKSK13], authors study the problem for monotone functions, and show the greedy algorithm offers a $min(m, k)$-composable core-set for the problem where m is the number of parts. On the other hand, [IMMM14] shows that it is impossible to go beyond an approximation factor of $\Omega(\frac{\sqrt{k}}{\log k})$ with polynomial size core-sets. Moreover, [MZ15, BENW15] consider a variant of the problem where the data is randomly distributed, and show the greedy algorithm achieves constant factor "randomized" composable core-sets for both monotone and non-monotone functions. However, one can notice that these results can not be directly compared to the current work, as a multiplicative approximation for determinant converts to an additive guarantee for the corresponding submodular function.

As discussed before, the determinant is one way to measure the diversity of a set of items. Diversity maximization with respect to other measures has been also extensively studied in the literature, [HRT97,

GS09, BLY12, BGMS16]. More recently, the problem has received more attention in distributed models of computation, and for several diversity measures constant factor approximation algorithms have been devised [ZGMZ17, IMMM14, CPPU17]. However, these notions are typically defined by considering only the pairwise dissimilarities between the items; for example, the summation of the dissimilarities over all pairs of items in a set can define its diversity.

One can also go further, and study the problem under additional constraints, such as matroid and knapsack constraints. This has been an active line of research in the past few years, and several centralized and distributed algorithms have been designed in this context for submodular optimization [MBK16, LMNS09, LSV10, CGQ15] and in particular determinant maximization [ESV17, NS16].

2 Preliminaries

Throughout the paper, we fix d as the dimension of the ambient space and $k (k \leq d)$ as the size parameter of the determinant maximization problem. We call a subset of $\mathbb{R}^d$ a point set, and use the term point or vector to refer to an element of a point set. For a set of points $S \subset \mathbb{R}^d$ and a point $p \in \mathbb{R}^d$, we write $S + p$ to denote the set $S \cup \{p\}$, and for a point $s \in S$, we write $S - p$ to denote the set $S \setminus \{s\}$.

Let S be a point set of size k. We use $\mathrm{VOL}(S)$ to denote the k-dimensional volume of the parallelepiped spanned by vectors in S. Also, let M_S denote a $k \times d$ matrix where each row represents a point of S. Then, the following relates volume to the determinant $\det(M_S M_S^\intercal) = \mathrm{VOL}^2(S)$. So the determinant maximization problem can also be phrased as *volume maximization*. We use the former, but because of the geometric nature of the arguments, sometimes we switch to the volume notation. For any point set P, we use MAXDET_k to denote the optimal of determinant maximization for P, i.e. $\mathrm{MAXDET}_k(P) = \max_S \det(M_S M_S^\intercal)$, where S ranges over all subsets of size k. MAXVOL_k is defined similarly.

For a point set P, we use $\langle P \rangle$ to refer to the linear subspace spanned by the vectors in P. We also denote the set of all k-dimensional linear subspaces by $\mathcal{H}_k$. For a point p and a subspace $\mathcal{H}$, we use $\mathrm{dist}(p, \mathcal{H})$ to show the Euclidean distance of p from $\mathcal{H}$.

Greedy algorithm for volume maximization. As pointed out before, a widely used algorithm for determinant maximization in the offline setting is a greedy algorithm which given a point set P and a parameter k as the input does the following: start with an empty set $\mathcal{C}$. For k iterations, add $\mathrm{argmax}_{p \in P} \mathrm{dist}(p, \langle \mathcal{C} \rangle)$ to $\mathcal{C}$. The result would be a subset of size k which has the following guarantee.

Theorem 2.1 ([ÇMI09])**.** *Let P be a point set and $\mathcal{C}$ be the output of the greedy algorithm on P. Then* $\mathrm{VOL}(\mathcal{C}) \geq \frac{\mathrm{MAXVOL}_k(P)}{k!}$.

2.1 Core-sets

Core-set is a generic term used for a *small subset* of the data that represents it very well. More formally, for a given optimization task, a core-set is a mapping c from any data set P into one of its subsets such that the solution of the optimization over the core-set $c(P)$ approximates the solution of the optimization over the original data set P. The notion was first introduced in [AHPV04] and many variations of core-sets exist. In this work, we consider the notion of *composable core-sets* defined in [IMMM14].

Definition 2.2 (α-Composable Core-sets)**.** *A function $c(P)$ that maps the input set $P \subset \mathbb{R}^d$ into one of its subsets is called an α-composable core-set for a function $f\colon 2^{\mathbb{R}^d} \to \mathbb{R}$ if, for any collection of sets $P_1, \cdots, P_n \subset \mathbb{R}^d$, we have $f(C) \geq f(P)/\alpha$ where $P = \bigcup_{i \leq n} P_i$ and $C = \bigcup_{i \leq n} c(P_i)$.*

For simplicity, we will often refer to the set $c(P)$ as the core-set for P and use the term "core-set function" with respect to $c(\cdot)$. The *size* of $c(\cdot)$ is defined as the smallest number t such that $c(P) \leq t$ for all sets P (assuming it exists). Unless otherwise stated, we might use the term "core-set" to refer to a composable core-set when clear from the context. Our goal is to find composable core-sets for the determinant maximization problem.

3 k-Directional Height

As pointed out in the introduction, we introduce a new geometric notion called directional height, and reduce the task of finding composable core-sets for determinant maximization to finding core-sets for this new notion.

Definition 3.1 (k-Directional Height)**.** *Let $P \subset \mathbb{R}^d$ be a point set and $\mathcal{H} \in \mathcal{H}_{k-1}$ be a $(k-1)$-dimensional subspace. We define the k-directional height of P with respect to $\mathcal{H}$, denoted by $h(P, \mathcal{H})$, to be the distance of the farthest point in P from $\mathcal{H}$, i.e., $h(P, \mathcal{H}) = \max_{p \in P} \operatorname{dist}(p, \mathcal{H})$.*

The notion is an instance of an *extent measure* defined in [AHPV05]. It is also related to the notion of *directional width* of a point set previously used in [AHPV05], which for a direction vector $v \in \mathbb{R}^d$ is defined to be $\max_{p \in P} \langle v \cdot p \rangle - \min_{p \in P} \langle v \cdot p \rangle$.

Next, we define core-sets with respect to this notion. It is essentially a subset of the point set that approximately preserves the k-directional height of the point set with respect to any subspace in $\mathcal{H}_k$.

Definition 3.2 (α-Approximate Core-set for the k-Directional Height)**.** *Given a point set P, a subset $C \subseteq P$ is a α-approximate core-set for the k-directional height of P, if for any $\mathcal{H} \in \mathcal{H}_{k-1}$, we have $h(C, \mathcal{H}) \geq h(P, \mathcal{H})/\alpha$.*

We also say a mapping $c(.)$ which maps any point set in $\mathbb{R}^d$ to one of its subsets, is an α-approximate core-set for the k-directional height problem, if the above relation holds for any point set P and $c(P)$.

The above notion of core-sets for k-directional height is similar to the notion of ϵ-kernels defined in [AHPV05] for the directional width of a point set.

We connect it to composable core-sets for determinant maximization by the following lemma.

Lemma 3.3. *Let $P_1, \ldots, P_m \in \mathbb{R}^d$ be an arbitrary collection of point sets, and for any i, let $c(P_i)$ be an α-approximate core-set for the k-directional height for P_i. Then*

$$\mathrm{MAXDET}_k(\bigcup_{i=1}^{m} P_i) \leq \alpha^{2k} \cdot \mathrm{MAXDET}_k(\bigcup_{i=1}^{m} c(P_i)).$$

Proof. Let $W \subset \bigcup_{i=1}^m P_i$ be any subset of size k, and also let $w \in W \setminus \bigcup_{i=1}^m c(P_i)$. We claim that there is a point q in the union of the core-sets such that $\alpha \cdot \mathrm{VOL}(W - w + q) \geq \mathrm{VOL}(W)$. Note that showing this claim is enough to prove the lemma. Since, one can start from the optimum solution which achieves the largest volume on $\bigcup_{i=1}^m P_i$, and for at most k iterations, replace a point outside $\bigcup_{i=1}^m c(P_i)$ by a point inside, while decreasing the volume by a factor of at most α.

So it remains to prove the claim. Let $W = \{w_1, \ldots, w_k\}$, and let $H = \langle w_2, \ldots, w_k \rangle \in \mathcal{H}_{k-1}$ be the plane spanned by $w_2, \ldots, w_k$. By definition, $\mathrm{VOL}(W) = \operatorname{dist}(w_1, H) \cdot \mathrm{VOL}(W - w_1)$. On the other hand,

suppose that $w_1 \in P_i$. Then by our assumption, there exists $q \in c(P_i)$ so that $\mathrm{dist}(q, H) \geq \frac{\mathrm{dist}(w_1,H)}{\alpha}$. Replacing w_1 with q, we get

$$\begin{aligned}\mathrm{VOL}(W - w_1 + q) &= \mathrm{dist}(q, H) \cdot \mathrm{VOL}(W - w_1) \\ &\geq \frac{\mathrm{dist}(w_1, H) \cdot \mathrm{VOL}(W - w_1)}{\alpha} = \frac{\mathrm{VOL}(W)}{\alpha}\end{aligned}$$

which completes the proof. □

Corollary 3.4. *Any mapping which is an α-approximate core-set for k-directional height, is an α^{2k}-composable core-set for the determinant maximization.*

We employ the above corollary to analyze both greedy and local search algorithms in Sections 4 and 5.

4 The Local Search Algorithm

In this section, we describe and analyze the local search algorithm and prove Theorem 1.2. The algorithm is described in Algorithm 1.

To prove Theorem 1.2, we follow a two steps strategy. We first analyze the algorithm for individual point sets, and show that the output is a $(2k)$-approximate core-set for the k-directional height problem, as follows.

Lemma 4.1. *Let P be a set of points and $c(P)$ be the result of running the local search algorithm on P. Then, for any $\mathcal{H} \in \mathcal{H}_{k-1}$,*

$$h(c(P), \mathcal{H}) \geq \frac{h(P, \mathcal{H})}{2k(1+\epsilon)}.$$

Next, we apply Corollary 3.4, which implies that local search gives $(2k(1+\epsilon))^{2k}$-composable core-sets for the determinant maximization. Clearly this completes the proof of the theorem by setting ϵ to a constant.

So proving Theorem 1.2 boils down to showing Lemma 4.1. Before, getting into that, we analyze the running time, and present some remarks about the implementation.

Running time. Let $\mathcal{C}_0$ be the output of the greedy. By Theorem 2.1 $\frac{\mathrm{VOL}(\mathcal{C}_0)}{\mathrm{MAXVOL}_k(P)} \geq \frac{1}{k!}$. The algorithm starts with $\mathcal{C}_0$ and by definition, in any iteration increases the volume by a factor of at least $1+\epsilon$, hence the total number of iterations is $O(\frac{k \log k}{\log(1+\epsilon)})$. Finally, each iteration can be naively executed by iterating over all points in P, forming the corresponding $k \times k$ matrix, and computing the determinant in total time $O(|P| \cdot kd \cdot k^3|P|)$.

We also remark that unlike the algorithm in [IMGR18], our method can also be executed without any changes and additional complexity, when the point set P is not explicitly given in the input; instead, it is presented by an oracle that given two points of P returns their inner product. One can note that in this case the algorithm can be simulated by querying this oracle for at most $O(|P|k)$ times.

4.1 Proof of Lemma 4.1

With no loss of generality, suppose that $\epsilon = 0$, the proof automatically extends to $\epsilon \neq 0$. Therefore, $c(P)$ has the following property: for any $v \in P \setminus c(P)$ and $u \in c(P)$, $\mathrm{VOL}(c(P)) \geq \mathrm{VOL}(c(P) - u + v)$.

Algorithm 1 Local Search Algorithm

Input: A point set $P \subset \mathbb{R}^d$, integer k, and $\epsilon > 0$.
Output: A set $\mathcal{C} \subset P$ of size k.
Initialize $\mathcal{C} = \emptyset$.
for $i = 1$ **to** k **do**
  Add $\text{argmax}_{p \in P \setminus \mathcal{C}} \text{VOL}(\mathcal{C} + p)$ to $\mathcal{C}$.
end for
repeat
  If there are points $q \in P \setminus C$ and $p \in \mathcal{C}$ such that

$$\text{VOL}(\mathcal{C} + q - p) \geq (1 + \epsilon)\text{VOL}(\mathcal{C})$$

  replace p with q.
until No such pair exists.
Return $\mathcal{C}$.

Fix $\mathcal{H} \in \mathcal{H}_{k-1}$, and let $p = \text{argmax}_{p \in P} \text{dist}(p, \mathcal{H})$. Our goal is to show there exists $q \in c(P)$ so that $\text{dist}(q, \mathcal{H}) \geq \frac{\text{dist}(p,\mathcal{H})}{2k}$.

Let $\mathcal{G} = \langle c(P) \rangle$ be the k-dimensional linear subspace spanned by the set of points in the core-set, and let $p_{\mathcal{G}}$ be the projection of p onto this subspace. We proceed with proof by contradiction. Set $\text{dist}(p, \mathcal{H}) = 2x$, and suppose to the contrary that for any $q \in c(P)$, $\text{dist}(q, \mathcal{H}) < \frac{x}{k}$. With this assumption, we prove the two following lemmas.

Lemma 4.2. $\text{dist}(p, p_{\mathcal{G}}) < x$.

Lemma 4.3. $\text{dist}(p_{\mathcal{G}}, \mathcal{H}) < x$.

One can note that, combining the above two lemmas and applying the triangle inequality implies $\text{dist}(p, \mathcal{H}) \leq \text{dist}(p, p_{\mathcal{G}}) + \text{dist}(p_{\mathcal{G}}, \mathcal{H}) < 2x$, which contradicts the assumption $\text{dist}(p, \mathcal{H}) = 2x$ and completes the proof.

Therefore, it only remains to prove the above lemmas. Let us first fix some notation. Let $c(P) = \{q_1, \ldots, q_k\}$ and for any i, let $\mathcal{G}_{\bar{i}}$ denote the $(k-1)$-dimensional subspace spanned by points in $c(P) \setminus \{q_i\}$.

Proof of Lemma 4.2. For $1 \leq i \leq k$, let q'_i be the projection of q_i onto $\mathcal{H}$. We prove that there exists an index $j \leq k$ such that we can write $q'_j = \sum_{i \neq j} \alpha_i q'_i$ where every $\alpha_i \leq 1$. Let r be the rank, i.e., maximum number of independent points of $\mathcal{C}' = \{q'_i | i \leq k\}$ and clearly as $\mathcal{H}$ has dimension $k-1$, we have $r \leq k-1$. Take a subset $S \subset \mathcal{C}'$ of r independent points that have the maximum volume and let q'_j be a point in $\mathcal{C}' \setminus S$ and note that this point should exist as there are k points in the core-set. Thus we can write $q'_j = \sum_{i: q'_i \in S} \alpha_i q'_i$. With an idea similar to the one presented in [ÇMI09], we can prove that the following claim holds.

Claim 4.4. *For any i such that $q'_i \in S$, we have $|\alpha_i| \leq 1$.*

Proof. We prove that if the claim is not true, then $S \setminus \{q'_i\} \cup \{q'_j\}$ has a larger volume than S which contradicts the choice of S. Let $\mathcal{F}$ be the linear subspace passing through $S \setminus \{q'_i\}$. It is easy to see that $\frac{\text{VOL}(S)}{\text{VOL}(S \setminus \{q'_i\} \cup \{q'_j\})} = \frac{\text{dist}(q'_i, \mathcal{F})}{\text{dist}(q'_j, \mathcal{F})}$, meaning that $\text{dist}(q'_i, \mathcal{F}) \geq \text{dist}(q'_j, \mathcal{F})$. However, if $|\alpha_i| > 1$ then since q'_i is the only point in S which in not in $\mathcal{F}$, then $\text{dist}(q'_j, \mathcal{F}) \geq \text{dist}(q'_i, \mathcal{F})$ which is a contradiction. $\square$

Finally, for any $q'_i \notin S$, set the corresponding coefficient $\alpha_i = 0$. So we get that $q'_j = \sum_{i \neq j} \alpha_i q'_i$ where every $|\alpha_i| \leq 1$.

Now take the point $q = \sum_{i \neq j} \alpha_i q_i$. Note that, q'_j is in fact the projection of q onto $\mathcal{H}$. Therefore, using triangle inequality, we have

$$\operatorname{dist}(q'_j, q) = \operatorname{dist}(q, \mathcal{H}) \leq \sum_{i \neq j} |\alpha_i| \operatorname{dist}(q_i, \mathcal{H}) \leq \frac{(k-1)x}{k} \tag{1}$$

Then we get that

$$\begin{aligned}
&\operatorname{dist}(p, p_{\mathcal{G}}) = \operatorname{dist}(p, \mathcal{G}) \\
&\leq \operatorname{dist}(p, \mathcal{G}_{\bar{j}}) \quad \text{as } \mathcal{G}_{\bar{j}} \subset \mathcal{G} \\
&\leq \operatorname{dist}(q_j, \mathcal{G}_{\bar{j}}) \quad \text{by the local search property} \\
&\leq \operatorname{dist}(q_j, q) \quad \text{as } q \in \mathcal{G}_{\bar{j}} \\
&\leq \operatorname{dist}(q_j, q'_j) + \operatorname{dist}(q'_j, q) \quad \text{by triangle inequality} \\
&< x/k + (k-1)x/k \quad \text{by our assumption and Equation 1} \\
&= x
\end{aligned}$$

□

Proof of Lemma 4.3. Again we prove that we can write $p_{\mathcal{G}} = \sum_{i=1}^{k} \alpha_i q_i$ where all $|\alpha_i| \leq 1$. We assume that the set of points q_i are linearly independent, otherwise the points in P have rank less than k and thus the volume is 0. Therefore, we can write $p_{\mathcal{G}} = \sum_{i=1}^{k} \alpha_i q_i$. Note that for any i, we have

$$\begin{aligned}
\operatorname{dist}(p_{\mathcal{G}}, \mathcal{G}_{\bar{i}}) &\leq \operatorname{dist}(p, \mathcal{G}_{\bar{i}}) \\
&\leq \operatorname{dist}(q_i, \mathcal{G}_{\bar{i}}) \quad \text{by the local search property}
\end{aligned}$$

where the first inequality follows since $\mathcal{G}_{\bar{i}}$ is a subspace of $\mathcal{G}$ and $p_{\mathcal{G}}$ is the projection of p onto $\mathcal{G}$. Again, similar to the proof of Claim 4.4, this means that $|\alpha_i| \leq 1$. Therefore, using triangle inequality

$$\begin{aligned}
\operatorname{dist}(p_{\mathcal{G}}, \mathcal{H}) = \operatorname{dist}(\sum_{i=1}^{k} \alpha_i q_i, \mathcal{H}) &\leq \sum_{i=1}^{k} |\alpha_i| \operatorname{dist}(q_i, \mathcal{H}) \\
&< k \times x/k = x.
\end{aligned}$$

□

5 The Greedy Algorithm

In this section we analyze the performance of the greedy algorithm (see section 2) as a composable core-set function for the determinant maximization and prove Theorem 1.1. Our proof plan is similar to the to the analysis of the local search. We analyze the guarantee of the greedy as a core-set mapping for k-directional height, and combining that with Corollary 3.4 we achieve the result. We prove the following.

Lemma 5.1. *Let P be an arbitrary point set and $c(P)$ denote the output of running greedy on P. Then, $c(P)$ is a $(2k) \cdot 3^k$-approximate core-set for the k-directional height of P, i.e. for any $\mathcal{H} \in \mathcal{H}_{k-1}$ we have*

$$h(c(P), \mathcal{H}) \geq \frac{1}{2k \cdot 3^k} \cdot h(P, \mathcal{H})$$

So the greedy is a $(2k \cdot 3^k)$-approximate core-set for k-directional height problem. Combining with Corollary 3.4, we conclude it is also a $(2k \cdot 3^k)^{2k}$ composable core-set for the determinant maximization which proves Theorem 1.1.

5.1 Proof of Lemma 5.1

The proof is similar to the proof of Lemma 4.1. Let $\mathcal{G} = \langle c(P) \rangle$ be the k-dimensional subspace spanned by the output of greedy. Also for a point $p \in P$, define $p_{\mathcal{G}}$ to be its projection onto $\mathcal{G}$. Fix $\mathcal{H} \in \mathcal{H}_{k-1}$, let $h(c(P), \mathcal{H}) = \frac{x}{k}$ for some number x, which in particular implies that for any $q \in c(P)$, $\mathrm{dist}(q, \mathcal{H}) \leq \frac{x}{k}$. Then, our goal is to prove $h(P, \mathcal{H}) \leq 2 \cdot 3^k \cdot x$. We show that by proving the following two lemmas.

Lemma 5.2. *For any $p \in P$, $\mathrm{dist}(p_{\mathcal{G}}, \mathcal{H}) \leq 2^{k-1}x$.*

Lemma 5.3. *For any $p \in P$, $\mathrm{dist}(p, p_{\mathcal{G}}) \leq 3^k x$.*

Clearly, combining them with triangle inequality, we get that for any $p \in P$, $\mathrm{dist}(p, \mathcal{H}) < 2 \cdot 3^k x$, which implies $h(P, \mathcal{H}) \leq 2 \cdot 3^k \cdot x$ and completes the proof. So it remains to prove the lemmas. Let the output of the greedy $c(P)$ be $q_1, \ldots, q_k$ with this order, i.e. q_1 is the first vector selected by the algorithm.

Proof of Lemma 5.2. Recall that $q_1, \ldots, q_k$ is the output of greedy. For any $p \in P$ and for any $1 \leq t \leq k$, let $\mathcal{G}_t = \langle q_1, \ldots, q_t \rangle$ and define p^t to be the projection of p onto $\mathcal{G}_t$. We show the lemma using the following claim.

Claim 5.4. *For any $p \in P$ and any $1 \leq t \leq k$, we can write $p^t = \sum_{i=1}^{t} \alpha_i q_i$ so that for each i, $|\alpha_i| \leq 2^{t-1}$.*

Let us first show how the above claim implies the lemma. It follows that we can write $p_{\mathcal{G}} = p^k = \sum_{i=1}^{k} \alpha_i q_i$ where all $|\alpha_i| \leq 2^{k-1}$. Now since for each $i \leq k$, $\mathrm{dist}(q_i, \mathcal{H}) \leq x/k$ by assumption, we have that $\mathrm{dist}(p_{\mathcal{G}}, \mathcal{H}) \leq \sum \alpha_i \mathrm{dist}(q_i, \mathcal{H}) \leq 2^{k-1}x$. Therefore, it suffices to prove the claim.

Proof of Claim 5.4. We use induction on t. To prove the base case of induction, i.e., $t = 1$, note that q_1 is the vector with largest norm in P. Thus we have that $||p^1|| \leq ||q_1||$ and therefore we can write $p^1 = \alpha_1 q_1$ where $|\alpha_1| \leq 1$. Now, lets assume that the hypothesis holds for the first t points; that is, the projection of any point p onto $\mathcal{G}_t$ can be written as $\sum_{j \leq t} \alpha_j q_j$ where $|\alpha_j|$'s are at most 2^{t-1}.

Now, note that by the definition of the greedy algorithm, q_{t+1} is the point with farthest distance from $\mathcal{G}_t$. Therefore, for any point $p \in P \setminus \{q_1, \cdots, q_{t+1}\}$, we know that $\mathrm{dist}(p, \mathcal{G}_t) \leq \mathrm{dist}(q_{t+1}, \mathcal{G}_t)$, and thus, $\mathrm{dist}(p^{t+1}, \mathcal{G}_t) \leq \mathrm{dist}(q_{t+1}, \mathcal{G}_t)$. Therefore if we define q_{t+1}^t to be the projection of q_{t+1} onto $\mathcal{G}_t$, we can write

$$p^{t+1} = \alpha_{t+1} q_{t+1} - \alpha_{t+1} q_{t+1}^t + p^t \quad \text{where } |\alpha_{t+1}| \leq 1.$$

By the hypothesis, we can write $p^t = \sum_{j \leq t} \beta_j q_j$, and $q_{t+1}^t = \sum_{j \leq t} \gamma_j q_j$, where $|\beta_j| \leq 2^{t-1}$, and $|\gamma_j| \leq 2^{t-1}$. Since $|\alpha_{t+1}| \leq 1$, we can write

$$p^{t+1} = \alpha_{t+1} q_{t+1} + \sum_{j \leq t} (\beta_j - \alpha_{t+1}\gamma_j) q_j = \sum_{j \leq t+1} \alpha_j q_j$$

where $|\alpha_j| \leq 2^t$. This completes the proof. □

□

Proof of Lemma 5.3. First, note that for any t, we have $\mathrm{dist}(q_{t+1}, \mathcal{G}_t) \geq \mathrm{dist}(p, \mathcal{G}_{k-1})$. This is because the greedy algorithm has chosen q_k over p in its k-th round which means that $\mathrm{dist}(p, \mathcal{G}_{k-1}) \leq \mathrm{dist}(q_k, \mathcal{G}_{k-1})$,

and by definition of the greedy algorithm for any $i < j$ we have $\mathrm{dist}(q_{i+1}, \mathcal{G}_i) \geq \mathrm{dist}(q_{j+1}, \mathcal{G}_j)$. So it is enough to prove

$$\exists\, 1 \leq t \leq k-1 \text{ s.t. } \mathrm{dist}(q_{t+1}, \mathcal{G}_t) \leq 3^k x \tag{2}$$

For $1 \leq i \leq k$, let q'_i be the projection of q_i onto $\mathcal{H}$. Recall that, we are assuming that for any i, $\mathrm{dist}(q_i, q'_i) < x/k$. To prove (2), we use proof by contradiction, so suppose that for all t, $\mathrm{dist}(q_{t+1}, \mathcal{G}_t) > 3^k x$. We also define $\mathcal{G}'_t$ to be the projection of $\mathcal{G}_t$ on $\mathcal{H}$, i.e., $\mathcal{G}'_t = \langle q'_1, \ldots, q'_t \rangle$. Given these assumptions, we prove the following claim.

Claim 5.5. *For any $1 \leq t \leq k-1$, we can write $\Pi(\mathcal{G}'_t)(q'_{t+1}) = \sum_{i \leq t} \alpha_i q'_i$ where $|\alpha_i| \leq 3^t$, where for a point q and a subspace $\mathcal{A}$, $\Pi(\mathcal{A})(q)$ denotes projection of q onto $\mathcal{A}$.*

Proof. Intuitively, this is similar to Claim 5.4. However, instead of looking at the execution of the algorithm on the points $q_1, \cdots, q_k$, we look at the execution of the algorithm on the projected points $q'_1, \cdots, q'_k$. Since all of these k points are relatively close to the hyperplane $\mathcal{H}$, the distances are not distorted by much and therefore, we can get approximately the same bounds. Formally, we prove the claim by induction on t, and show that for any j s.t. $j > t$, the point $\Pi(\mathcal{G}'_t)(q'_j)$ can be written as the sum $\sum_{i \leq t} \alpha_i q'_i$ such that $|\alpha_i| \leq 3^t$.

Base Case. First, we prove the base case of induction, i.e., $t = 1$. Recall that by our assumption, $||q_1|| > 3^k x$, and thus by triangle inequality, we have that $||q'_1|| \geq ||q_1|| - x/k \geq 3^k x - x/k \geq 2x$. Therefore, since q_1 is the vector with largest norm in P, using triangle inequality again, we have that for any $j > 1$,

$$||q'_j|| \leq ||q_j|| \leq ||q_1|| \leq ||q'_1|| + x/k \leq (1 + \frac{1}{2k})||q'_1||$$

Therefore we can write $\Pi(\mathcal{G}'_1)(q'_j) = \alpha_1 q'_1$ where $|\alpha_1| \leq 2$.

Inductive step. Now, lets assume that the hypothesis holds for $\mathcal{G}'_t$. In particular this means that we can write $\Pi(\mathcal{G}'_t)(q'_{t+1}) = \sum_{i \leq t} \beta_i q'_i$ where $|\beta_i| \leq 3^t$, and that for a given $j > t+1$, we can write $\Pi(\mathcal{G}'_t)(q'_j) = \sum_{i \leq t} \gamma_i q'_i$ where $|\gamma_i|$'s are at most 3^t. Now let $\ell = \mathrm{dist}(q'_{t+1}, \mathcal{G}'_t)$. By triangle inequality, we get that

$$\begin{aligned}
\mathrm{dist}(q_{t+1}, \mathcal{G}_t) &\leq \mathrm{dist}(q_{t+1}, q'_{t+1}) && (3)\\
&\quad + \mathrm{dist}(q'_{t+1}, \Pi(\mathcal{G}'_t)(q'_{t+1}) + && (4)\\
&\quad \mathrm{dist}(\Pi(\mathcal{G}'_t)(q'_{t+1}), \mathcal{G}_t)\\
&\leq x/k + \ell + \mathrm{dist}(\sum_{i \leq t} \beta_i q'_i, \sum_{i \leq t} \beta_i q_i)\\
&\leq x/k + \ell + \sum_{i \leq t} |\beta_i|\, x/k\\
&\leq \ell + 3^t x. && (5)
\end{aligned}$$

Now we consider two case. If $\ell \leq 3^t x$ then using the above

$$\mathrm{dist}(q_{t+1}, \mathcal{G}_t) \leq 2 \cdot 3^t x \leq 3^k x,$$

which contradicts our assumption of $\mathrm{dist}(q_{t+1}, \mathcal{G}_t) > 3^k x$. Otherwise,

$$\begin{aligned}
\mathrm{dist}(\Pi(\mathcal{G}'_{t+1})(q'_j), \mathcal{G}'_t) &\leq \mathrm{dist}(q'_j, \mathcal{G}'_t) \leq \mathrm{dist}(q_j, \mathcal{G}_t)\\
&\leq \mathrm{dist}(q_{t+1}, \mathcal{G}_t) \leq 2\ell,
\end{aligned}$$

where the last inequality follows from Equation 3 . Therefore, we can write $\Pi(\mathcal{G}'_{t+1})(q'_j) = \alpha_{t+1}q'_{t+1} - \alpha_{t+1}\Pi(\mathcal{G}'_t)(q'_{t+1}) + \Pi(\mathcal{G}_t)(q'_j)$ where $\alpha_{t+1} \leq 2$.

By the hypothesis, we can write $\Pi(\mathcal{G}'_t)(q'_j) = \sum_{i\leq t} \gamma_i q'_i$, where $|\gamma_i| \leq 3^t$. Since $|\alpha_{t+1}| \leq 2$, we can write

$$\begin{aligned}\Pi(\mathcal{G}'_{t+1})(q'_j) &= \alpha_{t+1}q'_{t+1} + \sum_{i\leq t}(\gamma_i - \alpha_{t+1}\beta_i)q'_i \\ &= \sum_{i\leq t+1} \alpha_i q'_i \quad \text{where } |\alpha_i| \leq 3^{t+1}.\end{aligned}$$

This completes the proof of the claim. □

To finish the proof of the lemma, let us show how it follows from the claim. First, note that $q'_1, \ldots, q'_k$ are k points in the $(k-1)$-dimensional space $\mathcal{H}$, so for some t, q'_{t+1} should lie inside $\mathcal{G}'_t$ and we have $\Pi(\mathcal{G}'_t)(q'_{t+1}) = q'_{t+1}$. Fix such t. Define the point $q_\alpha = \sum_{i\leq t} \alpha_i q_i$ where $|\alpha_i| \leq 3^k$ are taken from the above claim which means $q'_{t+1} = \sum_{i\leq t} \alpha_i q'_i$. Note that by definition $q'_{t+1} = \Pi(\mathcal{H})(q_\alpha)$. Therefore,

$$\begin{aligned}\text{dist}(q'_{t+1}, q_\alpha) &= \text{dist}(q_\alpha, \mathcal{H}) &(6)\\ &\leq \sum_{i\leq t} \alpha_i \text{dist}(q_i, \mathcal{H}) \leq 3^k t \cdot x/k. &(7)\end{aligned}$$

Then we get that

$$\begin{aligned}\text{dist}(q_{t+1}, \mathcal{G}_t) &\leq \text{dist}(q_{t+1}, q_\alpha) \quad \text{as } q_\alpha \in \mathcal{G}_t \\ &\leq \text{dist}(q_{t+1}, q'_{t+1}) + \text{dist}(q'_{t+1}, q_\alpha) \\ &\leq x/k + 3^k t \cdot x/k \leq 3^k x\end{aligned}$$

where the second inequality holds because of triangle inequality and the last one from (6) and the fact that $t \leq k-1$. This contradicts our assumption that $\text{dist}(q_{t+1}, \mathcal{G}_t) > 3^k x$, and proves the lemma. □

6 Experiments

In this section, we evaluate the effectiveness of our proposed Local Search algorithm empirically on real data sets. We implement the following three algorithms.

- The Greedy algorithm of Section 5 (GD).
- The Local Search algorithm of Section 4 with accuracy parameter $\epsilon = 10^{-5}$ (LS).
- The LP-based algorithm of [IMGR18] which has almost tight approximation guarantee theoretically (LP). Note that this algorithm might pick up to $O(k \log k)$ points in the core-set.

Data sets. We use two data sets that were also used in [LJS15] in the context of approximating DPPs over large data sets.

- MNIST [LBBH98]: contains a set of 60000 images of hand-written digits, where each image is of size 28 by 28.

- GENES [BQK+14]: contains a set of 10000 genes, where each entry is a feature vector of a gene. The features correspond to shortest path distances of 330 different hubs in the BioGRID gene interaction network. This data set was initially used to identify a diverse set of genes to predict a tumor. Here, we slightly modify it and remove genes that have an unknown value at any coordinate which gives us a data set of size ~ 8000.

Moreover, we apply an RBF kernel on both of these data sets using $\sigma = 6$ for MNIST and $\sigma = 10$ for GENES. These are the same values used in the work of [LJS15].

6.1 Experiment Setup.

We partition the data sets uniformly at random into multiple data sets $P_1, \cdots, P_m$. We use $m = 10$ for the smaller GENES data set, and for the larger MNIST data set we use $m = 50$ and also we use $m = 10$ (equal to the number of digits in the data set). Moreover, since the partitions are random, we repeat every experiment 10 times and take the average in our reported results.

We then use a *core-set construction algorithm* ALG_c to compute core-sets of size k, i.e., $S_1 = \mathrm{ALG}_c(P_1, k)$,$\cdots$,$S_m = \mathrm{ALG}_c(P_m, k)$, for $\mathrm{ALG}_c \in \{\text{GD, LS, LP}\}$. Recall that GD, LS and LP correspond to the Greedy, Local Search and LP-based algorithm of [IMGR18] respectively.

Finally, we take the union of these core-sets $U_{\mathrm{ALG}_c} = S_1 \cup \cdots \cup S_m$ and compute the solutions for U_{ALG_c}. Since computing the optimal solution can take exponential time ($\sim n^k$), we will instead use an *aggregation algorithm* ALG_a (either GD, LS or LP). We will use the notation $\mathrm{ALG}_a/\mathrm{ALG}_c$ to refer to the constructed set of k points, returned by $\mathrm{ALG}_a(U_{\mathrm{ALG}_c}, k)$. For example, GD/LS refers to the set of k points returned by the Greedy algorithm on the union of the core-sets, where each core-set is produced using the Local Search algorithm.

Finally, we vary the value of k from 3 to 20.

6.2 Results

Local Search vs. Greedy as offline algorithms. Our first set of experiments simply compares the quality of Greedy and Local Search as centralized algorithms on whole data sets. We perform this experiment to measure the improvement of Local Search over Greedy in the offline setting. Intuitively, this improvement upper bounds the improvement one can expect in the core-set setting. Figure 1 shows the improvement ratio of the determinant of the solution returned by the Local Search algorithm over the determinant of the solution returned by the Greedy algorithm. On average over all values of k, Local Search improves over Greedy by 13% for GENES data set and 5% for MNIST data set. Figure 2 shows the ratio of the time it takes to run the Local Search and Greedy algorithms as a function of k for both data sets. On average, it takes about 6.5 times more to run the Local Search algorithm.

Local Search vs. Greedy as core-sets. In our second experiment, we use Greedy algorithm for aggregation, i.e., $\mathrm{ALG}_a = GD$, and compare GD/LS with GD/GD. Figure 3 shows the improvement of local search over greedy as a core-set construction algorithm. The graph is drawn as a function of k, and for each k, the improvement ratio is an average over all 10 runs, and shown for all data sets (including GENES, MNIST with partition number $m = 10$, and MNIST with $m = 50$).

On average this improvement is 9.6%, 2.5% and 1.9% for GENES, MNIST10 and MNIST50 respectively. Moreover, in 87% of all 180 runs of this experiment, Local Search performed better than Greedy, and for some instances, this improvement was up to 58%. Finally, this improvement comes at a cost of increased running time. Figure 4 shows average ratio of the time to construct core-sets using Local Search vs. Greedy.

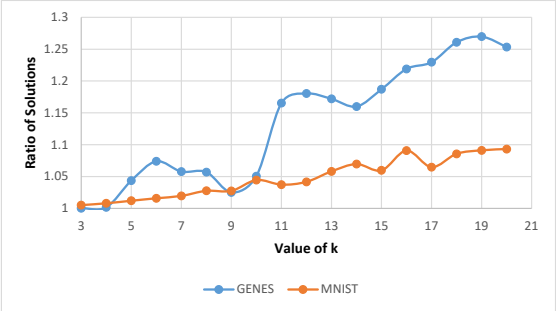

Figure 1: Average improvement of Local Search over Greedy as a function of k.

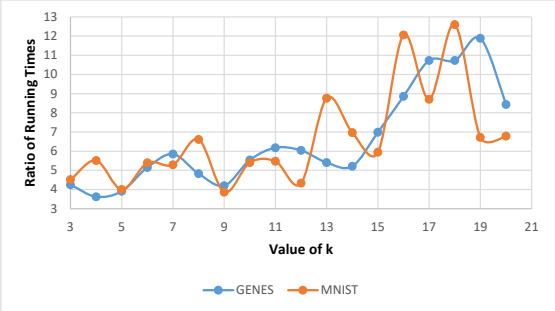

Figure 2: Average ratio of the run time of Local Search over Greedy as a function of k.

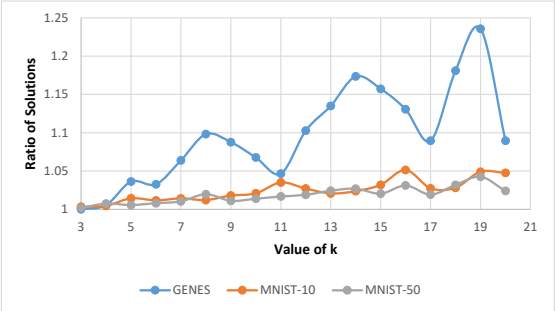

Figure 3: Average improvement of Local Search core-set over Greedy core-set as a function of k.

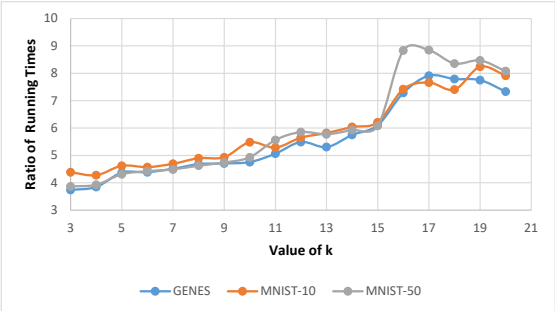

Figure 4: Average ratio of the run time of Local Search over Greedy as a function of k.

Local Search vs. Greedy - identical algorithms. We also consider the setting where the core-set construction algorithm is the same as the aggregation algorithm. This mimics the approach of [MKSK13], who proposed to use the greedy algorithm on each machine to achieve a small solution; then each machine sends this solution to a single machine that further runs the greedy algorithm on the union of these solutions and reports the result.

In this paper show that if instead of Greedy, we use Local Search in *both steps*, the solution will improve significantly. Using our notation, here we are comparing LS/LS vs. GD/GD. Figure 5 shows the improvement as a function of k, taken average over all 10 runs.

On average the improvement is 23%, 5.5% and 6.0% for GENES, MNIST10 and MNIST50 respectively. Moreover, in only 1 out of 180 runs the Greedy perfomed better than Local Search. The improvement could go as high as 67.7%.

Comparing Local Search vs. the LP-based algorithm. In this section, we compare the performance of the Local Search algorithm and the LP-based algorithm of [IMGR18] for constructing core-sets, i.e., we compare GD/LS with GD/LP. Figure 6 shows how much Local Search improves over the LP-based algorithm. On average this improvement is 7.3%, 1.8% and 1.4% for GENES, MNIST10 and MNIST50 respectively. Moreover, in 78% of all runs, Local Search performed better than Lp-based algorithm, and this improvement can go upto 63%. Figure 7 shows the average ratio of the time to construct core-sets using the LP-based algorithm vs. Local Search. As it is clear from the graphs, our proposed Local Search algorithm performs better than even the LP-based algorithm which has almost tight approximation guarantees: while picking

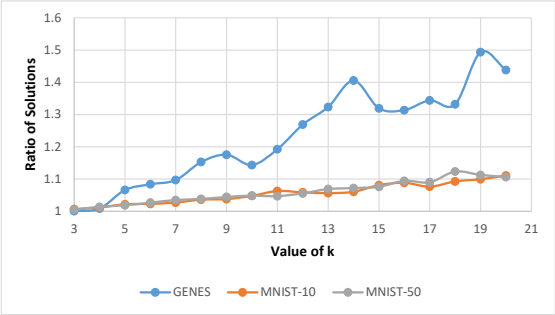

Figure 5: Average improvement of Local Search over Greedy as a function of k, in the identical algorithms setting.

fewer points in the core-set, in most cases it finds a better solution and runs faster.

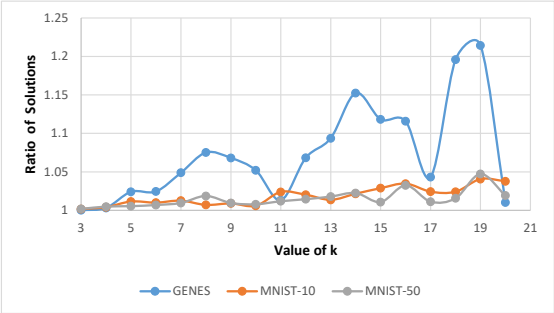

Figure 6: Average improvement of Local Search over LP-based algorithm for constructing core-sets as a function of k.

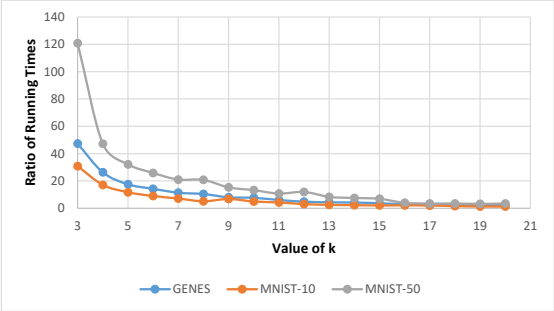

Figure 7: Average ratio of the run time of the optimal algorithm over local search as a function of k.

7 Conclusion

In this work, we proposed to use the Local Search algorithm to construct composable core-sets for the determinant maximization problem. From theoretical perspective, we showed that it achieves a near-optimal approximation guarantee. We further analyzed its performance on large real data sets, and showed that most of the times, Local Search performs better than both the almost optimal approximation algorithm, and the widely-used Greedy algorithm. Generally, for larger values of k, the percentage of this improvement has an increasing pattern, however, the amount of this improvement depends on the data set. We also note that here, we used the naive implementation of the Local Search algorithm: one could tune the value of ϵ to further improve the quality of the solution. Finally, we provided a doubly exponential guarantee for the Greedy algorithm, however, our experiments suggest that this bound might be open to improvement.

Acknowledgments

The authors would like to thank Stefanie Jegelka, Chengtao Li, and Suvrit Sra for providing their data sets and source code of experiments from [LJS15].

Piotr Indyk was supported by NSF TRIPODS award No. 1740751 and Simons Investigator Award. Shayan Oveis Gharan and Alireza Rezaei were supported by the NSF grant CCF-1552097 and ONR-YIP grant N00014-17-1-2429.

References

[AHPV04] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

[AHPV05] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52:1–30, 2005.

[AK17] Sepehr Assadi and Sanjeev Khanna. Randomized composable coresets for matching and vertex cover. *arXiv preprint arXiv:1705.08242*, 2017.

[BENW15] Rafael Barbosa, Alina Ene, Huy Nguyen, and Justin Ward. The power of randomization: Distributed submodular maximization on massive datasets. In *International Conference on Machine Learning*, pages 1236–1244, 2015.

[BGMS16] Aditya Bhaskara, Mehrdad Ghadiri, Vahab Mirrokni, and Ola Svensson. Linear relaxations for finding diverse elements in metric spaces. In *Advances in Neural Information Processing Systems*, pages 4098–4106, 2016.

[BLY12] Allan Borodin, Hyun Chul Lee, and Yuli Ye. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, pages 155–166. ACM, 2012.

[BMKK14] Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680. ACM, 2014.

[BQK+14] Nematollah Kayhan Batmanghelich, Gerald Quon, Alex Kulesza, Manolis Kellis, Polina Golland, and Luke Bornn. Diversifying sparsity using variational determinantal point processes. *arXiv preprint arXiv:1411.6307*, 2014.

[CDKV16] L Elisa Celis, Amit Deshpande, Tarun Kathuria, and Nisheeth K Vishnoi. How to be fair and diverse? *arXiv preprint arXiv:1610.07183*, 2016.

[CGGS15] Wei-Lun Chao, Boqing Gong, Kristen Grauman, and Fei Sha. Large-margin determinantal point processes. In *UAI*, pages 191–200, 2015.

[CGQ15] Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming algorithms for submodular function maximization. In *International Colloquium on Automata, Languages, and Programming*, pages 318–330. Springer, 2015.

[CKT10] Flavio Chierichetti, Ravi Kumar, and Andrew Tomkins. Max-cover in map-reduce. In *Proceedings of the 19th international conference on World wide web*, pages 231–240. ACM, 2010.

[ÇMI09] Ali Çivril and Malik Magdon-Ismail. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*, 410(47-49):4801–4811, 2009.

[CMI13] Ali Civril and Malik Magdon-Ismail. Exponential inapproximability of selecting a maximum volume sub-matrix. *Algorithmica*, 65(1):159–176, 2013.

[CPPU17] Matteo Ceccarello, Andrea Pietracaprina, Geppino Pucci, and Eli Upfal. Mapreduce and streaming algorithms for diversity maximization in metric spaces of bounded doubling dimension. *Proceedings of the VLDB Endowment*, 10(5):469–480, 2017.

[ESV17] Javad B Ebrahimi, Damian Straszak, and Nisheeth K Vishnoi. Subdeterminant maximization via nonconvex relaxations and anti-concentration. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 1020–1031. Ieee, 2017.

[GCGS14] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems*, pages 2069–2077, 2014.

[GS09] Sreenivas Gollapudi and Aneesh Sharma. An axiomatic approach for result diversification. In *Proceedings of the 18th international conference on World wide web*, pages 381–390. ACM, 2009.

[HRT97] Refael Hassin, Shlomi Rubinstein, and Arie Tamir. Approximation algorithms for maximum dispersion. *Operations research letters*, 21(3):133–137, 1997.

[IMGR18] Piotr Indyk, Sepideh Mahabadi, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization problems via spectral spanners. *arXiv preprint arXiv:1807.11648*, 2018.

[IMMM14] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 100–108. ACM, 2014.

[KMVV15] Ravi Kumar, Benjamin Moseley, Sergei Vassilvitskii, and Andrea Vattani. Fast greedy algorithms in mapreduce and streaming. *ACM Transactions on Parallel Computing (TOPC)*, 2(3):14, 2015.

[KT11] Alex Kulesza and Ben Taskar. Learning determinantal point processes. 2011.

[KT$^+$12] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LCYO16] Donghoon Lee, Geonho Cha, Ming-Hsuan Yang, and Songhwai Oh. Individualness and determinantal point processes for pedestrian detection. In *European Conference on Computer Vision*, pages 330–346. Springer, 2016.

[LJS15] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Efficient sampling for k-determinantal point processes. *arXiv preprint arXiv:1509.01618*, 2015.

[LMNS09] Jon Lee, Vahab S Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 323–332. ACM, 2009.

[LSV10] Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Mathematics of Operations Research*, 35(4):795–806, 2010.

[MBK16] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, and Amin Karbasi. Fast constrained submodular maximization: Personalized data summarization. In *ICML*, pages 1358–1367, 2016.

[MJK17] Baharan Mirzasoleiman, Stefanie Jegelka, and Andreas Krause. Streaming non-monotone submodular maximization: Personalized video summarization on the fly. *arXiv preprint arXiv:1706.03583*, 2017.

[MKBK15] Baharan Mirzasoleiman, Amin Karbasi, Ashwinkumar Badanidiyuru, and Andreas Krause. Distributed submodular cover: Succinctly summarizing massive data. In *Advances in Neural Information Processing Systems*, pages 2881–2889, 2015.

[MKSK13] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013.

[MKSK16] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization. *The Journal of Machine Learning Research*, 17(1):8330–8373, 2016.

[MZ15] Vahab Mirrokni and Morteza Zadimoghaddam. Randomized composable core-sets for distributed submodular maximization. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 153–162. ACM, 2015.

[Nik15] Aleksandar Nikolov. Randomized rounding for the largest simplex problem. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 861–870. ACM, 2015.

[NS16] Aleksandar Nikolov and Mohit Singh. Maximizing determinants under partition constraints. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 192–201. ACM, 2016.

[PJG+14] Xinghao Pan, Stefanie Jegelka, Joseph E Gonzalez, Joseph K Bradley, and Michael I Jordan. Parallel double greedy submodular maximization. In *Advances in Neural Information Processing Systems*, pages 118–126, 2014.

[WIB14] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Fast multi-stage submodular maximization. In *International conference on machine learning*, pages 1494–1502, 2014.

[YFZ+16] Jin-ge Yao, Feifan Fan, Wayne Xin Zhao, Xiaojun Wan, Edward Y Chang, and Jianguo Xiao. Tweet timeline generation with determinantal point processes. In *AAAI*, pages 3080–3086, 2016.

[ZGMZ17] Sepehr Abbasi Zadeh, Mehrdad Ghadiri, Vahab S Mirrokni, and Morteza Zadimoghaddam. Scalable feature selection via distributed diversity maximization. In *AAAI*, pages 2876–2883, 2017.