Abstract

Given an \( n \)-point metric space \((X,d)\) where each point belongs to one of \( m = O(1) \) different categories or groups and a set of integers \( k_1, \ldots, k_m \), the fair Max-Min diversification problem is to select \( k_i \) points belonging to category \( i \in [m] \), such that the minimum pairwise distance between selected points is maximized. The problem was introduced by Moumoulidou et al. [ICDT 2021] and is motivated by the need to down-sample large data sets in various applications so that the derived sample achieves a balance over diversity, i.e., the minimum distance between a pair of selected points, and fairness, i.e., ensuring enough points of each category are included. We prove the following results:

1. We first consider general metric spaces. We present a randomized polynomial time algorithm that returns a factor 2-approximation to the diversity but only satisfies the fairness constraints in expectation. Building upon this result, we present a 6-approximation that is guaranteed to satisfy the fairness constraints up to a factor \( 1 - \epsilon \) for any constant \( \epsilon \). We also present a linear time algorithm returning an \( m + 1 \) approximation with exact fairness. The best previous result was a \( 3m - 1 \) approximation.

2. We then focus on Euclidean metrics. We first show that the problem can be solved exactly in one dimension. For constant dimensions, categories and any constant \( \epsilon > 0 \), we present a \( 1 + \epsilon \) approximation algorithm that runs in \( O(nk) + 2^{O(k)} \) time where \( k = k_1 + \ldots + k_m \). We can improve the running time to \( O(nk) + \text{poly}(k) \) at the expense of only picking \( (1 - \epsilon)k_i \) points from category \( i \in [m] \).

Finally, we present algorithms suitable to processing massive data sets including single-pass data stream algorithms and composable coresets for the distributed processing.
1 Introduction

Given a universe of $n$ elements $X$ and a metric distance function $d : X \times X \to \mathbb{R}^+_0$, the Max-Min diversification problem seeks to select a $k$-sized subset $S$ of $X$ such that the minimum distance between the points in $S$ is maximized [23, 48]. Intuitively, the goal is to maximize the dissimilarity across all the selected points while $k$ is typically much smaller than $n$. A considerable amount of work in the database community has addressed the diversity maximization problem in the context of query result diversification [28, 32, 52], efficient indexing schemes for result diversification [6, 29, 54], nearest neighbor search [1], ranking schemes [8, 47], and recommendation systems [2, 15].

Recently, Moumoulidou et al. [46] introduced the fair variant of the Max-Min diversification problem. Specifically, the assumption is that the universe of elements $X$ is partitioned into $m = O(1)$ disjoint categories or groups. Then, the aim is to construct a diverse set of points where each group is sufficiently represented. To this end, the input of the problem includes non-negative integers $k_1, \ldots, k_m$ and the goal now is to select a subset $S$ using $k_i$ representatives from each group such that the minimum distance across all points is maximized. As a concrete example, consider a query over a maps service for finding restaurants around Manhattan at NYC. Then the goal is to present the user with a diversified set of restaurant locations while representing different cuisines in the sample.

In this work, we improve currently known approximation results for fair Max-Min diversification. This includes improving the approximation factor in the most general case of the problem; significantly decreasing the approximation factor if we slightly relax the fairness constraints; and reducing the approximation factors to arbitrarily close to 1 when the underlying metric is Euclidean. Before presenting our results, we review related work.

1.1 Related Work

The problem of unconstrained diversity maximization, i.e., when the number of groups $m = 1$, is well-studied in the context of facility location, information retrieval, web search and recommendation systems [8, 16, 23, 32, 35, 37, 41, 45, 47, 48, 52]. We refer the interested readers to the following surveys related to the diversification literature [30, 31].

Among popular diversification models are the distance-based models. In these models, the diversity of a set of points is modeled via some function defined over pairwise distances. Max-Sum (also known as remote-clique) and Max-Min (also known as remote-edge or $p$-dispersion) are two of the most well-established distance-based diversification models [42]. In Max-Sum, diversity is defined as the sum of the pairwise distances of points selected in a set, while in Max-Min the diversity of a set is equal to the minimum pairwise distance. For both problems, there are known 2-approximation algorithms, which yield the best approximation guarantee that can be achieved for both problems [12, 15, 48]. There are also recent works on distance-based diversity maximization models in the streaming, distributed, and sliding-window models [7, 13, 19, 42].

Contrary to unconstrained diversity maximization, the problem of fair diversity maximization is less studied. To the best of our knowledge, there is a known 2-approximation local search algorithm for fair Max-Sum diversification [2, 14, 15] where fairness is modeled via partition matroids [49]. Recent work also extends the local search approach to distances of negative type [22]. Another recently studied objective called Sum-Min [12] is defined as the sum of distances of all points to their closest point in the set. Bhaskara et al. [12] present an 8-approximation algorithm for Sum-Min under partition matroid constraints.
The most relevant result to our work is due to Moumoulidou et al. [46] that introduced the fair variant for the Max-Min diversification problem that we also study. The proposed fairness objectives have been widely studied by prior work [11, 20, 21, 24, 25, 34, 43, 44, 50, 53, 55, 56, 57], and are based on the definition of group fairness and statistical parity [33]. It is worth noting that there are other definitions for fairness, like individual or causal fairness [36], but these are not the focus of our work. Moumoulidou et al. [46] designed a polynomial time algorithm that achieved a $3m-1$-approximation for fair Max-Min diversification. There is also a recent line of work for designing (composable) coresets for various distance-based diversification objectives in the fairness setting [17, 18]. Coresets are small subsets of the original data that contain a good approximate solution and are typically used for speed up purposes or designing streaming and distributed algorithms. Prior efforts leave as an open question the construction of coresets for the fair variant of the Max-Min diversification objective.

1.2 Our Results

We present results for both the cases of general metrics and Euclidean metrics.

1. **General Metrics.** In Section 3.1, we present a randomized polynomial time algorithm that returns a factor 2-approximation to the diversity but only satisfies the fairness constraints in expectation, i.e., for each $i \in [m]$, the output is expected to include at least $k_i$ points from $X_i$. In Section 3.2, we present a 6-approximation that is guaranteed to include $(1 - \epsilon)k_i$ points in each group $i \in [m]$ assuming each $k_i = \Omega(\epsilon^{-2} \log m)$. Both these results are based on randomized rounding of a linear program. Finally, in Section 3.3 we present a linear time algorithm returning an $m + 1$ approximation with perfect fairness. This is an improvement over the previously known $3m - 1$ approximation [46]. We also present an example that shows that the analysis presented in Moumoulidou et al. [46] cannot be improved to obtain a better approximation factor. In Section 3.4, we present a hardness of approximation result arguing that we cannot get an approximation factor better than 2, even allowing for multiplicative approximations in fairness constraints.

2. **Euclidean Metrics.** If the points can be embedded in low dimensional space $\mathbb{R}^D$ (e.g., if the points correspond to geographical locations) and the distances correspond to Euclidean distances then we can significantly improve the approximation factors of our algorithms. In Section 4.1, we show that the problem can be solved exactly for $D = 1$. For constant dimensions, groups, we then present a $1 + \epsilon$ approximation algorithm that runs in $O(nk) + 2^D(k)$ time where $k = k_1 + k_2 + \ldots + k_m$. In Section 4.3, we show how to improve the running time to $O(nk) + \text{poly}(k)$ at the expense of only picking $(1 - \epsilon)k_i$ points from group $i \in [m]$. All these results are based on a new coreset construction.

In Sections 5.1 and 5.2, we present algorithms suitable to processing massive data sets including single-pass data stream algorithms and composable coresets for distributed processing.

2 **Background and Preliminaries**

2.1 **Fair Max-Min Diversification**

We formally define the problem of fair Max-Min diversification recently introduced in [46].

\begin{definition} [Fair Max-Min] Let $(X, d)$ be a metric space where $X = \bigcup_{i=1}^{m} X_i$ is a universe of $n$ elements partitioned into $m$ non-overlapping groups and $d : X \times X \to \mathbb{R}_0^+$ is a metric distance function. Then $\forall u, v \in X$, $d$ satisfies the following properties: (1) $d(u, v) = 0$
iff \( u = v \) (identity), (2) \( d(u, v) = d(v, u) \) (symmetry), and (3) \( d(u, v) \leq d(u, w) + d(w, v) \) (triangle inequality). Further, let \( k_1, k_2, \ldots, k_m \) be non-negative integers with \( k_i \leq |X_i| \), \( \forall i \in [m] \). The problem of fair Max-Min diversification is now defined as follows:

\[
\begin{align*}
\text{maximize} & \quad S \subseteq X \\
\text{subject to} & \quad \min_{u,v \in S, u \neq v} d(u,v) \\
& \quad |S \cap X_i| = k_i, \forall i \in [m] \quad \text{(fairness constraints)}
\end{align*}
\]

The aim is to select a subset \( S \subseteq X \) of points that maximizes the minimum pairwise distance across the points in \( S \) while being constrained to include \( k_i \) points from group \( i \). Throughout the paper we refer to the diversity of a set \( S \) as \( \text{div}(S) = \min_{u,v \in S, u \neq v} d(u,v) \).

Let \( S^* = \bigcup_{i=1}^m S_i^* \) be the set of points that obtains the optimal diversity score denoted by \( \text{div}(S^*) = \ell^* \). We say a subset of points \( S \) is an \( \alpha \) approximation if \( \text{div}(S) \geq \ell^*/\alpha \) and achieves \( \beta \) fairness if \( |S \cap X_i| \geq \beta k_i \) for all \( i \in [m] \). When \( \beta = 1 \), we say subset achieves perfect fairness.

Fair Max-Min is an NP-hard problem for which the best known polynomial time algorithms are: a 4-approximation algorithm that only works for \( m = 2 \) groups and a \( 3m - 1 \)-approximation algorithm that yields the best guarantees for any \( m \geq 3 \) [46]. The best approximation factor one can hope for in general metric spaces is a 2-approximation guarantee. This claim easily follows since when \( m = 1 \), the problem is just the Max-Min diversification problem where it is known that no polynomial time algorithm with an approximation factor better than 2 exists if \( P \neq NP \) [48]. We use \( \text{poly}(\cdot) \) to describe polynomial time algorithms using the context dependent parameters.

2.2 Low Doubling Dimension Spaces

Our results for low dimensional Euclidean metrics use the fact that such metrics have low doubling dimension. Our work in this direction is inspired by work on diversity maximization by Ceccarello et al. [17, 18, 19]. We define a ball of radius \( r \) centered at \( p \in X \) as the set of all points in \( X \) within distance strictly less than \( r \) from \( p \). We use the notation: \( B(p, r) = \{q \in X \mid d(p, q) < r\} \).

**Definition 2 (Doubling Dimension).** Let \((X, d)\) be a metric space. The doubling dimension of \( X \) is the smallest integer \( \lambda \) such that any ball \( B(p, r) \) of radius \( r \) around a point \( p \in X \) can be covered using at most \((r/r')^\lambda\) balls of radius \( r' \). The Euclidean metric on \( \mathbb{R}^D \) has doubling dimension \( O(D) \) [10, 19, 40].

2.3 Coresets

Coresets are powerful theoretical tools for designing efficient optimization algorithms in the presence of massive datasets in sequential, streaming or distributed environments [4, 42]. At a high level, coresets are carefully chosen subsets of the original universe of elements that contain an approximate solution to the optimal solution for the optimization problem at hand. A coreset for fair Max-Min diversification is defined as follows:

**Definition 3 (Coreset for Fair Max-Min).** A set \( T \subseteq X \) is an \( \alpha \)-coreset if there exists a subset \( T' \subseteq T \) with \( |T' \cap X_i| = k_i \forall i \in [m] \) and \( \text{div}(T') \geq \ell^*/\alpha \).

Note that optimally solving Fair Max-Min on \( T \), a set typically much smaller in size than \( X \), yields an \( \alpha \)-approximation factor. Further, the notion of coresets is useful for designing algorithms in the distributed setting using the composability property. Composable
coresets closely relate to the notion of mergeable summaries \([5, 42]\) while the assumption is that the universe of elements \(X\) is partitioned into \(L\) subsets (e.g., processing sites). Then the goal is to process each subset independently and extract a \textit{local} coreset such that in the union of these local coresets, there is an approximate solution for the optimization problem at hand. Specifically, for \textsc{Fair Max-Min} a composable coreset is defined as follows:

\begin{definition}[Composable coreset for \textsc{Fair Max-Min}] A function \(c(X)\) that maps a set of elements to a subset of these elements computes an \(\alpha\)-composable coreset for some \(\alpha \geq 1\), if for any partitioning\(^1\) of \(X = \bigcup_j Y_j\) and \(T = \bigcup_j c(Y_j)\), there exists a set \(T' \subseteq T\) with \(|T' \cap X_i| = k_i\) for all \(i \in [m]\) such that \(\text{div}(T') \geq \ell^*/\alpha\).
\end{definition}

3 General Metrics

In this section, we present algorithms for \textsc{Fair Max-Min} with an arbitrary metric. Our first two algorithms are based on rounding a suitable linear program. In Section 3.3 we present a linear time algorithm returning an \(m + 1\) approximation with perfect fairness. Finally, in Section 3.4, we give hardness of approximation results for \textsc{Fair Max-Min}.

3.1 2-Approx with Expected Fairness

In this section and others, we assume a guess \(\gamma\) on the optimal diversity value for \textsc{Fair Max-Min}. Note there are at most \((n^2)\) possible values for the optimal diversity corresponding to the set of distances between pairs of points. Hence, trying all these guesses only increases the running time by a factor \(O(n^2)\). Assuming the ratio between the largest and smallest distance is \(\text{poly}(n)\), this can be reduced to \(O(\epsilon^{-1} \log n)\) at the expense of introducing an additional factor of \(1 + \epsilon\) in the approximation. This follows by the standard technique of only considering guesses that are powers of \((1 + \epsilon)\) [39].

\textbf{Fair Max-Min LP.} Let \(X = \{p_1, \ldots, p_n\}\). For every point \(p_j \in X\), we have a variable \(x_j\). We represent the fairness constraint for every group \(i \in [m]\) using constraint (1). Additionally, for every point \(p \in X\), we add the constraint (2) that includes at most one point in a ball of radius \(\gamma/2\) centered at \(p\). This ensures that the selected points are separated by a distance of at least \(\gamma/2\). Using constraint (3), we allow \(x_p\) to take any value between \(0\) and \(1\). If \(\gamma \leq \ell^*\), observe that the optimal solution for \textsc{Fair Max-Min} is a feasible solution for this LP.

\begin{align*}
\sum_{p_j \in X_i} x_j & \geq k_i \quad \forall i \in [m]. \\
\sum_{p \in B(p, \gamma/2)} x_p & \leq 1 \quad \forall p \in X. \\
x_j & \geq 0 \quad \forall j \in [n].
\end{align*}

Let \(x^*_j\) denote the optimal solution of the linear program stated above. Let \(n' = |\{j : x^*_j > 0\}|\) and without loss of generality suppose \(x^*_j > 0\) for all \(j \in [n']\). We obtain an integral solution using a randomized rounding algorithm, in which we generate a random ordering based on sampling without replacement, such that a point \(p_j\) is selected as the next point in the ordering with probability proportional to \(x^*_j\). This allows us to show (see Lemma 5) that

\(^1\) The notion of composable coresets can also be extended when \(X\) is not divided into disjoint subsets but this is not the focus of our work.
the rounding scheme returns a set \( S \) with at least \( k_i \) points in expectation from each group \( i \in \{0, \ldots, n-1\} \) (satisfying constraint (1) in expectation). Further, our rounding scheme selects at most one point from each ball of radius \( \gamma/2 \) (satisfying constraint (2)). Since for any \( \gamma \leq \ell^{*} \) there is a set \( S \) that satisfies the properties discussed above, selecting the set \( S \) for the largest guess \( \gamma \) results in a 2-approximation for the diversity score.

**Randomized Rounding.** We generate a random ordering \( \sigma \) of \([n']\) where \( \sigma(t) \) is randomly chosen from \( R_t = [n'] \setminus \{ \sigma(1), \ldots, \sigma(t-1) \} \) such that for \( j \in R_t \), \( \Pr[\sigma(t) = j] = \frac{x_j^*}{\sum_{i \in R_t} x_i^*} \).

After generating the ordering \( \sigma \), we construct the output set \( S \) by including the point \( p_j \) in \( S \) iff \( \sigma(j) \leq \sigma(t) \) for all \( p_t \in B(p_j, \gamma/2) \). Note that all points in the output are at least distance \( \gamma/2 \) apart.

**Lemma 5.** There is an algorithm that returns a set \( S \), such that for all groups \( i \in [m], \) it holds that \( \mathbb{E}[|S \cap X_i|] \geq k_i \). Further all the points selected in \( S \) are at least \( \gamma/2 \) far apart.

**Proof.** Consider the randomized rounding algorithm described in this section. Now, let \( p_j \) be a point with \( x_j^* > 0 \). Define \( A_t \) to be the event \( d(p_{\sigma(t)}, p_j) < \gamma/2 \) and \( d(p_{\sigma(t')}, p_j) \geq \gamma/2 \) for all \( t' < t \). In other words, \( A_t \) is the event that the first point included in \( S \) from the ball \( B(p_j, \gamma/2) \) is the point from the \( t \)-th step (in the ordering \( \sigma \)).

Then,

\[
\Pr[p_j \in S] = \sum_{t=1}^{n'} \Pr[\sigma(t) = j] \Pr[A_t] = \sum_{t=1}^{n'} \frac{x_j^*}{\sum_{p_t \in B(p_j, \gamma/2)} x_t^*} \Pr[A_t] \\
= \frac{x_j^*}{\sum_{p_t \in B(p_j, \gamma/2)} x_t^*} \sum_{t=1}^{n'} \Pr[A_t] \\
\geq \frac{x_j^*}{\sum_{p_t \in B(p_j, \gamma/2)} x_t^*} \geq x_j^*
\]

where the last equality follows because \( \sum_{t=1}^{n'} \Pr[A_t] = 1 \) and the last inequality holds because of constraint (2) in the **Fair Max-Min** LP. Then for \( i \in [m] \), we have \( \mathbb{E}[|S \cap X_i|] \geq \sum_{p_x \in X_i} x_p^* \geq k_i \) where the last inequality follows from constraint (1).

### 3.2 6-Approx with \((1 - \epsilon)\) Fairness

We now present a more involved rounding scheme of the LP given in the previous section that ensures that the selected points contain at least \((1 - \epsilon)k_i \) points in \( X_i \) for each \( i \in [m] \). However, this guarantee comes at the expense of increasing the approximation factor for the diversity score from 2 to 6.

The main idea behind the new rounding scheme stems from the observation that for any \( p_i, p_j \in X_i \), if \( B(p_i, \gamma/2) \) and \( B(p_j, \gamma/2) \) are disjoint, then, in the previous rounding scheme, the event that \( p_i \) is included in the returned solution is independent of the event that \( p_j \) is included. This follows because the relative ordering of the elements in \( \{ \ell : p_t \in B(p_i, \gamma/2) \} \) in \( \sigma \) is independent of the ordering of the elements in \( \{ \ell : p_t \in B(p_j, \gamma/2) \} \) in \( \sigma \). This independence will ultimately allow us to use Chernoff bound to argue concentration of the number of elements chosen from each group \( X_j \) for all \( j \in [m] \).

#### 3.2.1 Randomized Rounding with improved fairness guarantees

We solve the LP in Section 3.1 to get a feasible solution \( \{x_j^*\}_{j \in [n]} \). Next, we transform \( \{x_j^*\}_{j \in [n]} \) into a feasible solution \( \{y_j^*\}_{j \in [n]} \) for the following set of constraints, some of which are no longer linear:
Finally, for all \(0 < y_i \) and \(0 < y_j \)
\[
\sum_{p_j \in X_i} y_j \geq k_i \quad \forall i \in [m].
\]
(1')
\[
\sum_{p \in B(p, \gamma/6)} y_p \leq 1 \quad \forall p \in X.
\]
(2')
\[
y_j \geq 0 \quad \forall j \in [n].
\]
(3')
\[
0 < y_i \implies d(p_i, p_j) \geq \frac{\gamma}{3} \quad \forall p_i, p_j \in X_i, \forall \ell \in [m]
\]
(4')

The constraint (2') ensures that at most one point in a ball of radius \(\gamma/6\) is selected (instead of \(\gamma/2\) used in Section 3.1) and results in an approximation factor of 6. The constraint (4') ensures that points from the same group with non-zero values are separated by at least \(\gamma/3\), which is used to argue \((1 - \epsilon)\) fairness (see Theorem 7). The transformation of \(x^*\) to \(y^*\) can be done by redistributing the values as follows:

(a) For each \(p_j \in X\) with \(x_j^* > 0\) satisfying \(p_j \in X_i\) and \(y_j^*\) value not yet set, we set:
\[
y_j^* \leftarrow \left( \sum_{p_j \in B(p, \gamma/3) \cap X_i} x_j^* \right)
\]
and \(y_j^* \leftarrow 0\) for all \(p \in B(p_j, \gamma/3) \cap (X_i \setminus \{p_j\})\).

(b) Finally, for all \(p_j \in X\) with \(x_j^* = 0\), we set \(y_j^* \leftarrow 0\).

Informally, we are just moving weight to \(p_j\) from points of the same group (as \(p_j\)) that are at a distance strictly less than \(\gamma/3\) from \(p_j\).

\textbf{Lemma 6.} \(\{y_j^*\}_{j \in [n]}\) satisfies Constraints (1'-4').

\textbf{Proof.} Observe that \(\{y_j^*\}_{j \in [n]}\) satisfies the constraint (4'). If a point \(p_j \in X_i\) satisfies \(y_j^* > 0\), then, it means that we set \(y_j^*\) to 0 for every \(p_l \in B(p_j, \gamma/3) \cap (X_i \setminus \{p_j\})\).

Constraint (2') is satisfied because
\[
\sum_{p_l \in B(p_j, \gamma/6)} y_l^* \leq \sum_{p_j \in B(p_j, \gamma/6 + \gamma/3)} x_j^* = \sum_{p_j \in B(p_j, \gamma/2)} x_j^* \leq 1
\]
since \(\{x_j^*\}_{j \in [n]}\) satisfies constraint (2). Constraint (1') is satisfied because \(\sum_{p_j \in X_i} y_j^* = \sum_{p_j \in X_i} x_j^*\) and Constraint (3') is trivially satisfied.

We next pick a random permutation \(\sigma\) as in the previous Section 3.1, but now using the values \(\{y_j^*\}_{j \in [n]}\). We add \(p_j\) to the output \(S\) if \(\sigma(j) \leq \sigma(\ell)\) for all \(p_\ell\) such that \(d(p_\ell, p_j) < \gamma/6\). Note that all points in \(S\) are therefore at least a distance of \(\gamma/6\) apart.

\textbf{Theorem 7.} Assume \(k_i \geq 3\epsilon^{-2}\log(2m)\) for all \(i \in [m]\). There is a poly\((n, k, \delta^{-1})\) time algorithm that returns a subset of points with diversity \(\ell^* / 6\) and includes \((1 - \epsilon)k_i\) points in each group \(i \in [m]\) with probability at least \(1 - \delta\).

\textbf{Proof.} Let \(Y_p = 1\) if the point \(p \in X\) is included in the output \(S\). Fix \(i \in [m]\). The proof of Lemma 5 applied to balls of radius \(\gamma/6\) rather than balls of radius \(\gamma/2\), ensures that for each \(i \in [m]\), \(E[\sum_{p \in X_i} Y_p] \geq k_i\). The fact \(\{Y_p\}_{p \in X_i}\) are fully independent allows us to apply the Chernoff bound and conclude \(\Pr[\sum_{p \in X_i} Y_p \leq (1 - \epsilon)k_i] \leq \exp(-\epsilon^2k_i / 3) \leq 1/(2m)\).

Hence, by an application of the union bound, we ensure that with probability at least \(1/2\), \(|S \cap X_i| \geq (1 - \epsilon)k_i\) for all \(i \in [m]\). Repeating the process \(\log \delta^{-1}\) times ensures that at least one of the trials succeeds with probability at least \(1 - \delta\). □
We now describe the procedure for building a cluster. Let $D$ denote a cluster initialized with a point of group $i \in [m]$. Among the available points $\mathcal{R}$, we include a point $p \in \mathcal{R}$, if it is within a distance of $\frac{2}{m+1}$ to some point $x \in D$, and no other point of the same group is already present in $D$.

**3.3 (m + 1)-Approx with Perfect Fairness**

We now describe Fair-Greedy-Flow (Algorithm 1), an $m + 1$-approximation algorithm that ensures perfect fairness. This is an improvement over the previously known $3m − 1$ approximation [46]. We also present an example that shows that the analysis presented in Moumoulidou et al. [46] cannot be improved to obtain a better approximation factor. The analysis for Fair-Greedy-Flow is presented in the extended version of the paper [3].
If there is no such point, the cluster $D$ is complete, and we remove all points from $R$ that are within a distance of $\frac{d}{m+1}$ from some point in $D$. Also, we discard all points of group $i$, i.e., $X_i$ from $R$, as soon as there are at least $k$ distinct clusters in $C$ containing points from $X_i$. We continue this process of iteratively building clusters, until there are points from each group that are part of at least $k$ distinct clusters or if there are no remaining points.

Next, we use an approach similar to [46] and select at most one point from each cluster, satisfying the fairness constraints. We construct a flow network with clusters $D_1, D_2, \cdots, D_t$ in $C$ represented by nodes $v_1, v_2, \cdots, v_t$ and groups represented by nodes $u_1, u_2, \cdots, u_m$. We add an edge with capacity 1 between every pair $u_i$ and $v_j$ if there is a point of group $i$ in cluster $D_j$ for some $j \in [t]$. We create a source node $a$ and add edges with capacity $k_j$ between $a$ and $u_i \forall i \in [m]$. We then create a sink node $b$ and add edges with capacity 1 between $b$ and $v_j \forall j \in [t]$. Finally, we find maximum flow using Ford-Fulkerson algorithm [26]. For each edge $(u_i, v_j)$ with flow equal to 1, we include the point of group $i$ from cluster $D_j$ in our solution. We conclude with the following theorem:

**Theorem 8.** FAIR-GREEDY-FLOW Algorithm returns an $(m+1)(1+\epsilon)$-approximation and achieves perfect fairness for the FAIR MAX-MIN problem using a running time of $O(nkm^3\epsilon^{-1}\log n)$.

We now give a tight example for FAIR-Flow in Moumoulidou et al. [46] and show how FAIR-GREEDY-FLOW yields a better approximation.

**A tight example for Fair-Flow: a $3m-1$ approximation algorithm [46].** Suppose $k = 3$ and we have to select one white and two black points. Here, edges represent the distance across two points, e.g., $d(p_1, p_2) = 1/5$. Note that the optimal solution in this example is the set of points $\{p_1, p_3, p_4\}$ with diversity score equal to 1.

![Graph](image)

FAIR-Flow for a guess $\gamma = 1$, for the black group selects both points since they are at least $d_1 = \frac{m\gamma}{3m+1} = 2/5$ far apart from each other. Similarly, for the white group. Now because there is no pair of points with distance strictly less than $d_2 = \frac{2}{3m-1} = 1/5$, FAIR-Flow constructs four connected components (each with a point). As a result, the points $\{p_1, p_2, p_4\}$ will be selected by the max-flow algorithm and we obtain a set with diversity score equal to 1/5. Note that for this example, FAIR-GREEDY-FLOW returns the set $\{p_1, p_3, p_4\}$ as $p_1$ and $p_2$ are less than 1/3 distance apart. These two points will be in the same cluster and at most one of them can be picked; thus, we guarantee an approximation ratio of 3.

### 3.4 Hardness of Approximation

In this section, we give a hardness of approximation result for the FAIR MAX-MIN problem. Our result is a generalization and improvement over the 2-approximation hardness shown in [46], as we also allow for approximations in fairness constraints.

**Definition 9 (Gap-Clique)$.** Given a constant $\rho \geq 1$, a graph $G$, and an integer $k$, we want to distinguish between the case where a clique exists of size $k$ (the “yes” case) and the case where no clique exists of size $\geq k/\rho$ (the “no” case).

It is known that Gap-Clique $\rho$ is NP-hard for every $\rho \geq 1$ [9]. Now, via a reduction from the Gap-Clique $\rho$ we argue that FAIR MAX-MIN cannot be approximated to a factor better than 2, even allowing for multiplicative approximations in fairness constraints.
Theorem 10. Let $\alpha < 2$ and $\beta > 0$ be constants. Unless $P = NP$, there is no polynomial time algorithm for the Fair Max-Min problem that obtains an $\alpha$-approximation factor for diversity score, and $\beta$ fairness.

Proof. We present a reduction from Gap-Clique$_\rho$, where $\rho = \beta$. For every vertex of the graph $G$, we create a new point, and set of points is denoted by $\mathcal{X}$. For every edge $(u, v)$ in $G$, we set $d(u, v) := 2$. For all other pairs of vertices, we set the distances as 1. Every vertex is assigned the same color, and the corresponding fairness constraint is $|S \cap \mathcal{X}| \geq k$, where $S$ is the set of points whose diversity we are trying to maximize in Fair Max-Min.

Suppose there is a polynomial time algorithm that returns a set $S$, obtains an $\alpha$-approximation for the diversity score, and a $\beta$-approximation for the fairness constraints. We first consider the ‘Yes’ instance in Gap-Clique$_\beta$, i.e., we assume there is a clique of size $k$ in $G$. This implies $\ell' = 2$. As $\alpha < 2$, we have that the set $S$ returned has a diversity score $\geq \ell'/\alpha > 1$. Therefore, $S$ is a clique in $G$ as all other pairwise distances are 1 (from construction). As $S$ is a $\beta$-approximation for the fairness constraint, we have that $|S| \geq k/\beta$. Let us now consider the ‘No’ instance, i.e., there is no clique of size $\geq k/\beta$ in $G$. Therefore, $|S \cap \mathcal{X}| < k/\beta$, as $|S \cap \mathcal{X}|$ is upper bounded by the maximum clique size in $G$. From the above arguments, we have that using our algorithm, we can distinguish the ‘Yes’ and ‘No’ instances of Gap-Clique$_\beta$, which is not possible unless $P = NP$ [9]. Hence, the theorem.

4 Euclidean Metrics

In this section, we assume that the metric space is Euclidean, i.e., we can associate a point $p_i \in \mathbb{R}^D$ with the $i$th entry of $\mathcal{X}$ and $d(p_i, p_j) = \|p_i - p_j\|_2 = \sqrt{\sum_{t \in [D]} (p_i(t) - p_j(t))^2}$. When $D = 1$ we show that the problem can be solved exactly in polynomial time via Dynamic Programming. More generally, when $D = O(1)$ we present a bi-criteria approximation that uses an extension of the dynamic programming approach and properties of low dimensional Euclidean spaces.

4.1 Exact Computation in One Dimension

In this section, we assume the points in the universe $\mathcal{X} = \bigcup_{i=1}^m \mathcal{X}_i$ can be embedded on a line. Specifically, let $\mathcal{X} = \{p_1, \ldots, p_n\}$ where each $p_i \in \mathbb{R}$ and we order the points such that $p_1 \leq p_2 \leq \ldots \leq p_n$. We further assume a guess $\gamma$ on the optimal diversity score for Fair Max-Min and design the dynamic programming algorithm FAIR-LINE (Algorithm 2) that computes an exact solution when $\gamma = \ell^\star$. See the previous section for a discussion on guessing $\gamma$.

Dynamic Programming. Define the dynamic programming table $H \in \{0, 1\}^{(k_1+1) \times \cdots \times (k_m+1) \times n}$ indexed from 0. An entry $H[k_1', k_2', \ldots, k_m', j] \in \{0, 1\}$ is 1 iff there is a subset $S'$ of the first $j$ points on the line with diversity $\gamma$ that contains $k'_i$ points from each group $i \in [m]$. To compute the entries of $H$, we process the points in their order of appearance on the line.

Note that there is a set $S'$ with $k'_i$ points from each group $i$ among the first $j$ points if: (1) there is such a set among the first $j - 1$ points, or (2) point $j$ belongs to group $i$ for some $i \in [m]$, and among the first $j'$ points there is a set with $k'_1, \ldots, k'_i - 1, \ldots, k'_m$ points from the corresponding groups where $j' < j$ is the largest value such that $d(p_j, p_j') \geq \gamma$.

See FAIR-LINE (Algorithm 2) for the resulting algorithm. For simplicity, the algorithm is written to only determine whether it is possible to pick a subset with diversity $\gamma$ subject to the required fairness constraints. However, the algorithm can be easily extended to construct...
the points can be embedded on a line and requires a running time of $O(d)$ doubling dimension. The proposed approach uses the approach generalizes prior work on constructing efficient coresets for unconstrained Max-Min

In this section, we design efficient coresets for constant dimensions

\begin{algorithm}
\caption{Fair-Line: An exact algorithm for data on a line.}
\label{alg:fair-line}
\begin{algorithmic}[1]
\Statex \textbf{Input:} $\mathcal{X} = \bigcup_{i = 1}^{m} \mathcal{X}_i$: Universe of available points. $k_1, \ldots, k_m \in \mathbb{Z}^+$. $\gamma \in \mathbb{R}^+$: A guess of the optimum fair diversity.
\Statex \textbf{Output:} $k_i$ points in $\mathcal{X}_i$ for $i \in [m]$.
1: Let $n \leftarrow |\bigcup_{i = 1}^{m} \mathcal{X}_i|$ and initialize $H \in \{0, 1\}^{(k_1+1) \times \cdots \times (k_m+1) \times n}$ to 0.
2: Set $H[0, \ldots, 0, 1] \leftarrow 1$, $H[0, \ldots, 1] \leftarrow 1$, and if $p_1 \in \mathcal{X}_i$, $H[0, \ldots, \underbrace{1, \ldots, 1}_\text{index } \ell, \ldots, 0, 1] \leftarrow 1$.
3: for $j = 2$ to $n$ do
4: \hspace{1em} Let $i \in [m]$ satisfy $p_j \in \mathcal{X}_i$.
5: \hspace{1em} Let $j' = \max (\{0\} \cup \{j' \in [n] : p_{j'} + \gamma \leq p_j\})$.
6: \hspace{1em} for $k'_1 \in \{0, \ldots, k_1\}, \ldots, k'_m \in \{0, \ldots, k_m\}$ do
7: \hspace{2em} $H[k'_1, \ldots, k'_m, j] \leftarrow H[k'_1, \ldots, k'_m, j-1]$.
8: \hspace{2em} if $k'_i \geq 1$, $H[k'_1, \ldots, k'_m, j'] \leftarrow H[k'_1, \ldots, k'_m, j-1, \ldots, k'_m, j'] \vee H[k'_1, \ldots, k'_m, j-1]$.
9: \hspace{1em} end for
10: end for
11: return $H[k_1, k_2, \ldots, k_m, n]$.
\end{algorithmic}
\end{algorithm}

a subset of points for every non-zero entry in $H$ by storing a pointer to the choice we made. For an entry $H[k'_1, k'_2, \ldots, k'_m, j] = 1$ that also satisfies $H[k'_1, k'_2, \ldots, k'_m, j-1] = 1$, we store a pointer to that entry. In the second case, if $H[k'_1, k'_2, \ldots, k'_m, j'] = 1$ for some $j'$, we store a pointer to that entry. We construct the solution set using the stored pointers, starting at $H[k_1, k_2, \ldots, k_m, n]$ and backtracking, to indicate which points to add to the solution.

\textbf{Theorem 11.} There is an algorithm that solves the Fair Max-Min problem exactly when the points can be embedded on a line and requires a running time of $O(n^4 \prod_{i=1}^{m} (k_i+1))$.

\textbf{Proof.} We use Fair-Line to identify the exact solution. We observe that any optimal solution can be expressed as a subset of the first $j$ points for some $j \in [n]$. From the construction, if the guess $\gamma \leq \ell^*$ there will always be at least $k_i$ points from group $i$ for all $i \in [m]$ that are all $\gamma$ far apart. Therefore, since the dynamic programming approach finds all the subsets with $k_i$ points per group $i$ for all $j \in [n]$, at least one of the $H[k_1, k_2, \ldots, k_m, j]$ entries will be equal to 1 as required. As discussed previously, we can backtrack and construct the solution set.

\textbf{Running Time.} For a fixed guess $\gamma$, we need to compute $\prod_{i=1}^{m} (k_i+1)$ entries for every point, as every $k'_i$ for $i \in [m]$ takes at most $k_i+1$ values. To compute an entry $H[\cdot, \cdot, \cdot, j]$ using Fair-Line (Algorithm 2), we need to retrieve $O(n)$ distances to find point $j'$ that is at least $\gamma$ far apart from point $j$. Thus, the total running is equal to $O(n^2 \prod_{i=1}^{m} (k_i+1))$ since there are $O(n \prod_{i=1}^{m} (k_i+1))$ entries in $H$ and the computational cost to fill each entry is $O(n)$. As there are $O(n^2)$ distance values the guess $\gamma$ can take, the total running time is $O(n^4 \prod_{i=1}^{m} (k_i+1))$.

\section{4.2 Coresets for Constant Dimensions}

In this section, we design efficient $(1+\epsilon)$-coresets for Fair Max-Min in metric spaces of low doubling dimension (Definition 2). Let $\lambda$ denote the doubling dimension of $\mathcal{X}$. Our approach generalizes prior work on constructing efficient coresets for unconstrained Max-Min diversification [19] to the Fair Max-Min problem.

Specifically, we give the first algorithm for constructing coresets in metric spaces of doubling dimension. The proposed approach uses the GMM algorithm that obtains a factor 2-approximation for the unconstrained Max-Min diversification problem [48, 51].
GMM is a greedy algorithm and works as follows: it starts with an arbitrary point in a set $S$ and in every subsequent step selects the point that is the farthest away from the previously selected points. In fact, readers familiar with the $k$-center clustering problem will recognize that this is the same strategy used by [38]. If $k$ is the size of the subset to be selected and $n$ is the size of the universe of points, it is known that GMM can be implemented in $O(kn)$ time [44, 54].

**Coreset Construction.** First, define $\epsilon' = \epsilon/(1 + \epsilon)$ and note that $\epsilon/2 \leq \epsilon' < 1$ since $\epsilon \in (0, 1]$. The **Coreset Algorithm** constructs coreset $T$ as follows: we run GMM on each group $i \in [m]$ separately to retrieve a set $T_i$ with $O((4/\epsilon')^\lambda k)$ points. The coreset $T$ is equal to the union of the $T_i$ sets for all $i \in [m]$, namely: $T \leftarrow \bigcup_{i=1}^m T_i$, where $T_i \leftarrow \text{GMM}(X_i, (4/\epsilon')^\lambda k)$.

We will show that $T$ contains a set $T'$ with $\text{div}(T') \geq \ell^*/(1 + \epsilon)$ and $k_i$ points from each group $i$. At a high level, the idea is that for each group $i$ there are two cases: (1) either $T_i$ contains a sufficient number of points that are far apart such that even if we had to remove points close to points selected from other groups, we would still have enough points to satisfy fairness, or (2) the optimal points from group $i$ are within small distance from their closest point in $T_i$. In the analysis we show that in both cases we have enough points from each group $i$ to satisfy fairness while these points are at least $\ell^*/(1 + \epsilon)$ far apart. We first prove the following lemma, which we will use later.

**Lemma 12.** Let $S$ be a set of $k' = (4/\epsilon')^\lambda k$ points that are all at least $(\epsilon'/2)\gamma$ far apart. Then, there exists a subset $S' \subset S$ of points that are all at least $\gamma$ far apart and $|S'| \geq k$.

**Proof.** Let $S' = \emptyset$. Add an arbitrary point $x$ from $S$ to $S'$ and remove all points in the ball $B(x, \gamma)$ from $S$. Consider a set of balls of radius $(\epsilon'/4)\gamma$ that cover the removed points. Each of these balls cover at most one removed point since discarded points are at least $(\epsilon'/2)\gamma$ far apart. Hence, the number of balls is at least the number of removed points. But because the doubling dimension is $\lambda$ we know there exists a set of $(4/\epsilon')^\lambda$ balls of radius $(\epsilon'/4)\gamma$ that cover the removed points. Hence, the number of removed points is at most $(4/\epsilon')^\lambda$. Since there were $k' = (4/\epsilon')^\lambda k$ points in $S$, we may continue in this way until we’ve added $k$ points to $S'$. All chosen points are at least $\gamma$ apart as required. □

Our main theorem in this section is as follows:

**Theorem 13.** There is an algorithm that returns a $(1 + \epsilon)$-coreset of size $O((8/\epsilon)^\lambda km)$ in metrics of doubling dimension $\lambda$ with a running time $O((8/\epsilon)^\lambda kmn)$.

**Proof.** We show that the set $\bigcup_{i=1}^m T_i$ constructed by the **Coreset Algorithm** is an $(1 + \epsilon)$-coreset by showing the existence of a set $T' \subseteq \bigcup_{i=1}^m T_i$ with $k_i$ points from each group $i$ and $\text{div}(T') \geq \ell^*/(1 + \epsilon)$.

For every group $i \in [m]$, we define $\hat{T}_i$ to be the maximal prefix of the points added by GMM to form $T_i$ such $\text{div}(\hat{T}_i) \geq (\epsilon'/2)\ell^*$. We first process all the groups for which $|\hat{T}_i| < (4/\epsilon')^\lambda k$, which we call **critical** groups. For all critical groups, any point $p \in X \setminus \hat{T}_i$ is within distance $(\epsilon'/2)\ell^*$ from its closest point $f(p)$ in $\hat{T}_i$, i.e., $d(p, f(p)) < (\epsilon'/2)\ell^*$. As a result, for any pair of optimal points $o_1, o_2$ in critical groups we deduce:

$$d(f(o_1), f(o_2)) \geq d(o_1, o_2) - d(o_1, f(o_1)) - d(o_2, f(o_2))$$
$$> \ell^* - 2 \cdot \epsilon' \ell^*/2 = \ell^*/(1 + \epsilon).$$
We initialize $T' = \bigcup_{i \in \{1, \ldots, m\}} S'_{i}$ where $S'_{i}$ is the set of points in an optimal solution belonging to group $X_{i}$. We now process all non-critical groups $j \in [m]$ in an arbitrary order and remove any point in $T_{j}$ that is less than $\ell^{*}$ apart from some point in $T'$. Then we argue that in the remaining points there is a set of points $T'_{j}$ with $k_{j}$ points that are at least $\ell^{*}$ far apart.

By the doubling dimension property and the fact that all the points in $T'_{j}$ are at least $(\ell^{*}/2)\ell^{*}$ far apart, the removal step described above discards at most $(4/\ell^{*})k \sum_{i \text{ processed}} |T' \cap X_{i}|$ points from $T_{j}$. Consequently, regardless of the order in which we process the non-critical groups, by the time we process $T_{j}$ for some $j \in [m]$, there will be at least $(4/\ell^{*})k - \sum_{i \text{ processed}} (4/\ell^{*})k_i \geq (4/\ell^{*})k_j$ points that are at least $(\ell^{*}/2)\ell^{*}$ apart from each other.

Now by applying Lemma 12 on the points of $T'_{j}$, we conclude that there are at least $k_{j}$ points within $\ell^{*}$ distance from all other points in $T'_{j}$. Then this set of points $T'_{j}$ can be added to $T'$ to satisfy fairness for group $j$. Thus, it holds that $\text{div}(T') \geq \ell^{*}/(1 + \epsilon)$ which implies the claimed approximation factor for coreset $T$. As $\epsilon = \epsilon/(1 + \epsilon) \geq \epsilon/2$, we have $|T| = O((8/\epsilon)k\lambda m)$. Since we use GMM to obtain $T$, the running time of the CORESET algorithm is $O((8/\epsilon)k\lambda m\lambda)$.

From the coreset $T$, we can obtain a $(1 + \epsilon)$-approximation by enumerating over all subsets of $T$ and returning the subset with maximum diversity and perfect fairness. The running time of this algorithm is $O(2^{O(k)} + nk)$, when $m, \lambda$ are constants. In the next section, we describe an algorithm that has a polynomial dependence on $n$ and $k$, obtained at the cost of $(1 - \epsilon)$-fairness.

### 4.3 $(1 + \epsilon)$ Approx with $(1 - \epsilon)$ Fairness

In this section, we describe FAIR-EUCLIDEAN (Algorithm 4) which uses $(1 + \epsilon)$-coresets described in Section 4.2 and returns a subset of points with diversity at least $\ell^{*}/(1 + \epsilon)$ and has $(1 - \epsilon)k_i$ points from each group $i \in [m]$.

First, we discuss FAIR-DP (Algorithm 3), which is a dynamic programming subroutine used in FAIR-EUCLIDEAN. The subroutine will be applied to a collection of $t$ disjoint subsets of $X$: $C = \{C_{1}, C_{2}, \ldots, C_{t}\}$. This collection will be well-separated in the sense that for all $i \neq j$ and $x \in C_{i}$, $y \in C_{j}$ then $d(x, y) \geq \gamma$. Points in the same set can be arbitrarily close together. We design FAIR-DP (Algorithm 3): a dynamic programming algorithm to retrieve a set $F = \bigcup_{i=1}^{m} F_{i} \subseteq C$ with $k_i$ points per group $i$ and $\text{div}(F) \geq \gamma$ if such a set exists in $C$.

#### Dynamic Programming

Define the dynamic programming table $H \in \{0, 1\}^{(k_1+1) \times \ldots \times (k_m+1) \times t}$ indexed from 0. An entry $H[k'_1, k'_2, \ldots, k'_m, j] \in \{0, 1\}$ is 1 iff there is a subset $F'$ among the first $j$ clusters such that $|F' \cap X_{i}| \geq k'_i \forall i \in [m]$ and $\text{div}(F') \geq \gamma$.

To compute the entries of $H$, we process the clusters in $C$ using some fixed ordering. Note that there is a set $F'$ with $k'_i$ points from each group $i$ among the first $j$ clusters if there is a subset $P \subseteq C_{j}$ with $\text{div}(P) \geq \gamma$ and $p'_i$ points from each group $i$; and, among the first $j - 1$ clusters, there is a set with $k'_i - p'_i, k'_2 - p'_2, \ldots, k'_j - p'_j, k'_m - p'_m$ points from each group $i \in [m]$ that are at least $\gamma$ far apart (the function $f$ in FAIR-DP (Algorithm 3) evaluates where there is such a set $P$). We enumerate over all possible subsets of $C_{j}$ to identify the subset $P$. 

Improved Approximation and Scalability for Fair Max-Min Diversification

**Algorithm 3** **FAIR-DP:** A dynamic programming subroutine.

| Input: | $C_1, C_2, \ldots, C_l$: Family of disjoint subsets of $X = \bigcup_{i=1}^m X_i$. $k_1, \ldots, k_m \in \mathbb{Z}^+$. $\gamma \in \mathbb{R}^+$: A guess of the optimum fair diversity. |
|---------------------------------------------|
| Output: | $k_i$ points in $X_i$ for $i \in [m]$. |
|---------------------------------------------|
| 1: Define boolean function $f(p_1', \ldots, p_m', j)$ that evaluates to 1 iff there exists $P \subseteq C_j$ with $\text{div}(P) \geq \gamma$ and $|P \cap X_i| = p_i'$ for all $i \in [n]$. |
| 2: Initialize $H \in \{0,1\}^{(k_1+1)\times \cdots \times (k_m+1) \times t}$ to 0. |
| 3: Set $H[p_1', \ldots, p_m', 1] = f(p_1', \ldots, p_m', 1)$. |
| 4: for $j = 1$ to $t$ do |
| 5: For $k_i' \in \{0, \ldots, k_i\}$ for $i \in [m]$, update the entries in $H$ as: |
| $H[k_1', \ldots, k_m', j] \leftarrow \bigvee_{p_i' \leq k_i'} \bigwedge_{\forall i \in [m]} H[k_1' - p_1', \ldots, k_m' - p_m', j - 1] f(p_1', \ldots, p_m', j)$. |
| 6: end for |
| 7: return $H[k_1, k_2, \ldots, k_m, n]$. |

See **FAIR-DP** (Algorithm 3) for additional details and implementation. For simplicity, the algorithm is written to only determine whether it is possible to pick a subset with diversity $\gamma$ subject to the required fairness constraints. Similar to **FAIR-LINE**, the algorithm can be easily extended to construct a subset of points for every non-zero entry in $H$ by storing a pointer to the choice we made.

**Theorem 14.** If $\gamma = \ell^*$, then, **FAIR-DP** (Algorithm 3) returns a set $S$ that satisfies $\text{div}(S) \geq \ell^*$ and $|S \cap X_i| \geq k_i \forall i \in [m]$ and has a running time of $O(\prod_{i=1}^m (k_i + 1)^{2^m R t})$ where $R = \max\{|C_1|, |C_2|, \ldots, |C_l|\}$.

**Proof.** As $\gamma = \ell^*$, the optimal set of points satisfy the fairness constraints. From the construction in **FAIR-DP**, we will return a set $S$ that has diversity $\ell^*$, and achieves perfect fairness.

**Running Time.** Consider a value $j \in [t]$. There are $\prod_{i=1}^m (k_i + 1)$ entries in the table $H$ corresponding to this value of $j$. For every $k_i' \in \{0, 1, \ldots, k_i\}$ and every subset $R \subseteq C_j$ where $|R \cap X_i| = p_i' \forall i \in [m]$, we check if there is a valid subset of points satisfying fairness constraints using the condition mentioned in **FAIR-DP**. Since there at most $\prod_{i=1}^m (k_i + 1)$ ways to enumerate the $p_i'$ values (because $p_i' \leq k_i'$), the total time to compute entries corresponding to this $j$ value is $O(\prod_{i=1}^m (k_i + 1)^{2^m R t})$. Therefore, to compute all the entries in $H$ we need $O(\prod_{i=1}^m (k_i + 1)^{2^m R t})$ time.

Now, we describe a $1 + \epsilon$ approximation algorithm for Euclidean metrics called **FAIR-EUCLIDEAN** that achieves $1 - \epsilon$ fairness.

**Overview of Fair-Euclidean.** As part of the input, we construct a $(1+\epsilon)$-coreset $\mathcal{T} = \bigcup_{i=1}^m T_i$ of size $O((8/\epsilon)^k km)$ using the **CORESET** algorithm described in Section 4.2. We further assume a guess $\gamma$ for the optimal diversity score $\ell^*$. Note that the coreset $\mathcal{T}$ is only constructed once and used for different guesses of $\ell^*$.

For a fixed guess $\gamma$, for every group $i \in [m]$, we select a maximal prefix of points $\hat{T}_i \subset T_i$ that are at least $\gamma/4$ far apart and define $\hat{T} = \bigcup_{i=1}^m \hat{T}_i$. Our main idea is to partition $\hat{T}$ and obtain a collection of sets $\mathcal{C} = \{C_1, C_2, \ldots, C_l\}$ separated by at least $\gamma$ distance; thus
Within a range of total length $\lambda$, therefore, for $\alpha$, we can use $q_{\text{Fair-DP}}$ removal step there are least boundaries of any cube. Note that the $(\alpha, \gamma)$-max-min coreset.

Proof. ▶ Lemma 15. Boundaries of a cube. Let $X$ be the probability that a point $q \in T_i$ is not removed from a cube is at least $\lambda \gamma$ from one of its boundaries. Notice that any point that was not removed from a cube is at least $\gamma$ far apart from any other point in a different cube. However, points within the same cube can be arbitrarily close. It is now easy to see that we can use $\text{Fair-DP}$ (Algorithm 3) on $T_i$ to retrieve a sufficient number of points from each group in $[m]$. In the analysis below, we show that with probability at least $1/2$, we are able to find at least $(1 - \epsilon)k_i$ points from each group $i \in [m]$ that are all $\gamma$ far apart.

**Analysis.** Let $S^* = \bigcup_{i=1}^m S_i^* \subset T$ denote the optimal solution for $\text{Fair Max-Min}$ on the coreset $T = \bigcup_{i=1}^m T_i$ with $\text{div}(S^*) \geq \gamma/(1 + \epsilon)$. Note that the optimal solution in $T$ is some subset in $\hat{T}$ (see Theorem 13).

As a first step, we bound the number of optimal points $S_i^*$ from a group $i \in [m]$ that are removed by $\text{Fair-Euclidean}$ because they are within a distance of $\gamma/2$ from one of the boundaries of a cube.

▶ Lemma 15. $\text{Pr}[\forall i \in [m] : |\bigcup_{j \in [t]} C_j \cap S_i^*| \geq (1 - \epsilon)k_i] \geq 1/2$.

Proof. Let $T_i' = \bigcup_{j \in [t]} C_j \cap \hat{T}_i$ be the remaining points in $\hat{T}_i$ that are not close to the boundaries of any cube. Note that the $\text{Fair-Euclidean}$ algorithm succeeds if after the removal step there are least $(1 - \epsilon)k_i$ optimal points from each group $i$ that can be selected by $\text{Fair-DP}$ at the final step of the algorithm while it fails otherwise. Below, we show that the probability it succeeds is at least $1/2$.

We compute the probability that a point $q \in T_i$ is not removed by $\text{Fair-Euclidean}$, i.e., $q \in T_i'$. It is removed if it lies within a distance of $\gamma/2$ from its boundaries in each dimension. Therefore, for $q$ to remain in $T_i'$, the point $p$ selected randomly from $[0, W]^D$ must not fall within a range of total length $\gamma$, in each dimension, which gives us:

$$\Pr[q \notin T_i'] = 1 - \Pr[q \in T_i] = 1 - \left(\frac{W - \gamma}{W}\right)^D \leq \gamma D/W = \epsilon/2m.$$
Fix a specific optimum solution. Define $A_i$ be the number of points removed from this solution that are in group $i$. By Markov’s inequality, $\Pr[A_i \geq k_i \epsilon] \leq \frac{\mathbb{E}[A_i]}{k_i \epsilon} \leq \frac{k_i(2m)}{k_i \epsilon} = \frac{1}{2m}$.

Taking union bound over all groups $i \in [m]$, we can bound the probability of discarding more than $k_i \epsilon$ points from some group $i$, $\Pr[\exists i \in [m] : A_i \geq k_i \epsilon] \leq \sum_{i=1}^{m} \Pr[A_i \geq k_i \epsilon] < 1/2$, and the lemma follows.

\textbf{Fair-DP} depends exponentially on the number of points remaining in each cube (see Theorem 14). Now, we show that the total number of points remaining in each cube does not depend on $n$ or $k$, and depends only on $m, D, \epsilon$.

\textbf{Lemma 16}. $|C_j| \leq m \cdot (8mD^{3/2}/\epsilon^2)^\lambda$ for all $j \in [t]$.

\textbf{Proof}. Consider all points in $C_j$ that belong to group $i$, i.e., $C_j \cap \hat{\mathcal{T}}_i$. From the construction of $\hat{\mathcal{T}}_i \subseteq \mathcal{T}$, we have that every pair of points of the same group is separated by a distance at least $c_\gamma/4$. Therefore, each point can be represented by a ball of radius $c_\gamma/8$, and we want to count the maximum number of non-overlapping balls that can be packed inside the cube $C_j$. Observe that the length of the diagonal of $C_j$ is $W \sqrt{D}$, and the cube lies entirely in the ball of radius $W \sqrt{D}/2$ with center at the middle of the diagonal. We call this cube ball. As Euclidean metrics are doubling metrics, we can cover the cube ball with overlapping balls of radius $c_\gamma/8$ and the number of the balls required is $\left(\frac{W \sqrt{D}/2}{c_\gamma/8}\right)^\lambda$, where $\lambda = O(D)$ is the doubling dimension of $\mathbb{R}^D$.

We can observe that the total volume occupied by the overlapping balls is at least the volume occupied by the non-overlapping balls corresponding to the points and having the same radius. Therefore, we can upper bound the number of points using the total number of non-overlapping balls used to cover the cube ball. As there are $m$ groups, we have that the total number of the points in $C_j$ is: $|C_j| \leq m \cdot \left(\frac{W \sqrt{D}/2}{c_\gamma/8}\right)^\lambda = m \cdot (8mD^{3/2}/\epsilon^2)^\lambda$.

We showed that for a fixed guess $\gamma$, the success probability of \textsc{Fair-Euclidean} is $\geq 1/2$. Note that the only randomization used by \textsc{Fair-Euclidean} is in selecting $\eta$. In order to increase the probability of success to $1 - \delta$ for some small $\delta \in (0, 1)$, we repeatedly select $\eta$ points uniformly at random from $[0, W]^D$ as the corners. For each corner, we obtain a solution using \textsc{Fair-Euclidean}, and we output the solution with the biggest diversity which also satisfies the fairness constraints with a loss of $(1 - \epsilon)$ multiplicative factor. The value of $\eta = \log(1/\delta)$ is selected such that the failure probability is $(1/2)^n < \delta$.

Note that the construction of the coreset $\mathcal{T}$ allows us to reduce the number of guesses on $\ell^*$ from $O(n^2)$ to $O(|\mathcal{T}|^2) = O((8/\epsilon)^{2k^2m^2})$, which are all the pairwise distances in $\mathcal{T}$. Further, the number of clusters (i.e., cubes) in \textsc{Fair-Euclidean} is upper bounded by the size of the coreset $\mathcal{T}$, which does not depend on $n$. The running time of \textsc{Fair-Euclidean} depends on the running time to construct the coreset, which is $O((8/\epsilon)^{\lambda} kmn)$, and the running time of \textsc{Fair-DP} (Algorithm 3) on the cubes $\mathcal{C}$. Since the number of points in each cube is $O(m \cdot (8mD^{3/2}/\epsilon^2)^\lambda)$, we conclude with the following theorem:

\textbf{Theorem 17}. If $\gamma \geq \ell^*/(1 + \epsilon)$, \textsc{Fair-Euclidean} Algorithm returns a set $S$ such that $\text{div}(S) \geq \ell^*/(1 + \epsilon)$ and $|S \cap X_i| \geq k_i(1 - \epsilon) \forall i \in [m]$ with probability at least $1 - \delta$. The running time of the algorithm is:

$$O(n \cdot (8/\epsilon)^{\lambda} km + \sum_{i=1}^{m} (k_i + 1)^2 (8mD^{3/2}/\epsilon^2)^\lambda (8/\epsilon)^{\lambda} km \log |\mathcal{T}| \log(1/\delta)).$$
Proof. The running time of \textsc{Fair-Euclidean} (Algorithm 4) depends on: (1) the running time of constructing the coreset \( T \) which is \( O(nkm(8/\epsilon)^\lambda) \), where \( \lambda \) is the doubling dimension, and (2) the running time of \textsc{Fair-DP} (Algorithm 3) on the clusters for every guess \( \gamma \).

From Theorem 14, we know that \textsc{Fair-DP} has a running time of \( O(\prod_{i=1}^{m}(k_i + 1)2^{2R_i}) \), where \( t \) is the number of clusters and \( R \) is the maximum size across all \( t \) clusters. We upper bound the number of clusters by the coreset size. So, \( t = O((8/\epsilon)^\lambda km) \). From Lemma 16, we have \( R = O(m(8mD^{3/2}/\epsilon^2)^\lambda) \). Combining all the above, the final running time is:

\[
O((8/\epsilon)^\lambda kmn + \log |T| \log(1/\delta) \prod_{i=1}^{m}(k_i + 1)2^{m(2mD^{3/2}/\epsilon^2)^\lambda} (8/\epsilon)^\lambda km).
\]

We can observe that the running time depends doubly exponentially on the doubling dimension, which is not uncommon for diversity maximization in doubling dimension metrics [17, 19].

5 Scalable Implementations

5.1 Data Stream Algorithms

In this section, we present single pass data stream algorithms that obtain the same approximation guarantees as that of sequential algorithms, while using low space. Missing details from this section are presented in the extended version of the paper [3].

5.1.1 Extending Previous Algorithms

First, we describe an algorithm called \( \tau \)-\textsc{GMM} that processes points sequentially, and includes a point in the solution if it is at least the threshold \( \tau \) apart from every point in the current solution set. The set of points returned by \( \tau \)-\textsc{GMM} are all separated by a distance of at least \( \tau \). If \( m = 1 \), then, we can set \( \tau = \ell^*/2 \) (using guessing for \( \ell^* \)), and \( \tau \)-\textsc{GMM} returns a solution set that is also a 2-approximation for the \textsc{Fair Max-Min} problem [27]. \( \tau \)-\textsc{GMM} allows us to extend it to data streaming setting, unlike the \textsc{GMM} algorithm which requires identifying the maximum distance point in each iteration.

Using \( \tau \)-\textsc{GMM} with \( \tau = \ell^*/2 \), we can obtain a 5-coreset for general metrics [46], and \((1+\epsilon)\)-coreset for Euclidean metrics (Section 4.2). Then, on the coreset, we use the randomized rounding algorithm from Section 3.2 and return the solution. This approach gives us the following guarantees:

\begin{itemize}
  \item \textbf{Corollary 18.} There is a \( O(\epsilon^{-1}km\log n) \)-space data stream algorithm that returns a 30\((1+\epsilon)\)-approximation with \((1-\epsilon)\)-fairness for general metrics. For Euclidean metrics, there is a \( O((8/\epsilon)^{\lambda}kmc^{-1}\log n) \)-space data stream algorithm that returns a \( 1+\epsilon \)-approximation with \((1-\epsilon)\)-fairness where \( \lambda \) is the doubling dimension of \( X \subset \mathbb{R}^D \).
\end{itemize}

5.1.2 Improved Result for \( m = 2 \)

In [46], the authors describe an algorithm called \textsc{Fair-Swap} which returns a 4-approximation to the \textsc{Fair Max-Min} problem when the number of groups is \( m = 2 \). The algorithm can be directly extended to a 2-pass streaming algorithm using \( O(k) \) space with the same 4-approximation guarantee. Building upon their work, and using new ideas we obtain a single pass algorithm \textsc{Fair-Stream-2Groups} which uses \( O(k) \) space, and obtains 4-approximation to the \textsc{Fair Max-Min} problem.
The algorithm maintains 3 sets $S, S_1, S_2$ using $\tau$-GMM for all of them. In $S$, we include points in a group-agnostic way (similar to FAIR-SWAP) ignoring the fairness constraints. In $S_1$, we include points only of group 1, and in $S_2$, we include points only of group 2. By setting $\tau = \ell^* / 2$ we maintain the sets $S, S_1$ and $S_2$ such that all points are at least $\ell^*/2$ distance apart in every one of them.

Without loss of generality, suppose $X_1$ satisfies $|S \cap X_1| < k_1$. Our algorithm proceeds by identifying $k_1 - |S \cap X_1|$ additional points from $S_1$ denoted by $Z_1$ by running $\tau$-GMM with $\tau = \ell^*/4$. This ensures that the final set of points from group 1, i.e., $(S \cap X_1) \cup Z_1$ are $\ell^*/4$ apart. By discarding the nearest neighbors of newly added points (i.e., $Z_1$), in $S \cap X_2$, we argue that our algorithm obtains a 4-approximation. We obtain the following guarantees:

▶ Theorem 19. There is a one-pass streaming algorithm that returns a $4(1 + \epsilon)$-approximation for FAIR MAX-MIN problem using $O(kn / \epsilon)$ space.

5.2 Composable Coresets

In this section, we design composable coresets for FAIR MAX-MIN. We assume the points $X$ are partitioned into $L$ disjoint sets. We discuss an algorithm for constructing $(1 + \epsilon)$-composable coresets for Euclidean metrics, and discuss extensions. Missing details are presented in the extended version of the paper [3].

5.2.1 Constructing $(1 + \epsilon)$-composable coresets

We assume the universe of points $X$ is partitioned into a collection of $L$ disjoint sets $Y_1, Y_2, \ldots, Y_L$. As in Section 4.2, we define an $\epsilon' > 0$ value such that $(1 - \epsilon') = 1/(1 + \epsilon)$. We generalize the approach for constructing the coreset $T$ as follows: let $Y_j^i$ denote the points of group $i$ present in $Y_j$ for $i \in [m]$ and $j \in [L]$. Then on each partition $j$ and group $i$, we run GMM to retrieve a diverse set $T_j^i$ with $O((4/\epsilon')^3 k)$, or equivalently $O((8/\epsilon)^3 k)$ points since $\epsilon' \geq \epsilon/2$. The coreset $T$ is defined as:

1. For $j \in [L]$, construct $T_j$: $T_j \leftarrow \bigcup_{i=1}^{m} T_j^i$, where $T_j^i \leftarrow$ GMM$(Y_j^i, (4/\epsilon')^3 k)$
2. $T \leftarrow \bigcup_{j=1}^{L} T_j$

We obtain the following theorem:

▶ Theorem 20. $T$ is a $(1 + \epsilon)$-composable coreset for fair Max-Min diversification of size $O((8/\epsilon)^3 kmnL)$ in metrics of doubling dimension $\lambda$ that can be obtained in $O((8/\epsilon)^3 kmnL)$ time.

For general metrics, using a similar approach, we obtain a 5-composable coreset by extending a recent construction of 5-coreset for the sequential setting [46]. In the extended version of the paper, we also discuss two-pass distributed algorithms for constructing $\alpha$-composable coresets for Euclidean ($\alpha = 1 + \epsilon$) and general metrics ($\alpha = 5$).

6 Conclusion

In this paper, we presented new approximation algorithms that substantially improve upon currently known results for the FAIR MAX-MIN problem both in general and Euclidean metric spaces. There are several interesting directions for future work, including obtaining a 2-approximation for the problem in general metrics or improving the hardness result.
Another direction is to generalize the fairness constraints to arbitrary matroid constraints (the fairness constraints considered in this paper can be expressed via the special case of a partition matroid). While there are results known for related diversity maximization problems under matroid constraints [2, 12, 15], to the best of our knowledge, there are currently no results for Max-Min diversification.

References

Improved Approximation and Scalability for Fair Max-Min Diversification


