A Stochastic Alternating Balance k-Means Algorithm for Fair Clustering

Suyun Liu[1][0000−0002−9226−7436] and Luis Nunes Vicente[2][0000−0003−1097−6384]

[1] Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA. sul217@lehigh.edu.

[2] Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA. lnv@lehigh.edu. Support for this author was partially provided by the Centre for Mathematics of the University of Coimbra under grant FCT/MCTES UIDB/MAT/00324/2020.

Abstract. In the application of data clustering to human-centric decision-making systems, such as loan applications and advertisement recommendations, the clustering outcome might discriminate against people across different demographic groups, leading to unfairness. A natural conflict occurs between the cost of clustering (in terms of distance to cluster centers) and the balance representation of all demographic groups across the clusters, leading to a bi-objective optimization problem that is nonconvex and nonsmooth. To determine the complete trade-off between these two competing goals, we design a novel stochastic alternating balance fair k-means (SAfairKM) algorithm, which consists of alternating classical mini-batch k-means updates and group swap updates. The number of k-means updates and the number of swap updates essentially parameterize the weight put on optimizing each objective function. Our numerical experiments show that the proposed SAfairKM algorithm is robust and computationally efficient in constructing well-spread and high-quality Pareto fronts both on synthetic and real datasets.

Keywords: k-means Clustering · Unsupervised Machine Learning · Data Mining · Fairness · Bi-Objective Optimization · Pareto Front.

1 Introduction

Clustering is a fundamental task in data mining and unsupervised machine learning with the goal of partitioning data points into clusters, in such a way that data points in one cluster are very similar and data points in different clusters are quite distinct [16]. It has become a core technique in a huge amount of application fields such as feature engineering, information retrieval, image segmentation, targeted marketing, recommendation systems, and urban planning. Data clustering problems take on many different forms, including partitioning clustering like k-means and k-median, hierarchical clustering, spectral clustering, among many others [8, 16]. Given the increasing impact of automated decision-making systems in our society, there is a growing concern about algorithmic unfairness,

which in the case of clustering may result in discrimination against minority groups. For instance, females may receive proportionally fewer job recommendations with high salary [13] due to their under-representation in the cluster of high salary recommendations. Such demographic features like gender and race are called *sensitive* or *protected* features, which we wish to be fair with respect to.

Related work An extensive literature work studying algorithmic fairness has been focused on developing universal fairness definitions and designing fair algorithms for supervised machine learning problems. Among the broadest representative fairness notions proposed for classification and regression tasks are *disparate impact* [6] (also called *demographic parity* [10]), *equalized odds* [19], and individual fairness [15], based on which the fairness notions in clustering were proposed accordingly. There are a number of classes of fairness definitions proposed and investigated for the clustering task [1, 11, 12, 18, 23, 27]. The most widely used fairness notion is called *balance*. It was proposed by [12], and it has been extended in several subsequent works [7, 20, 30]. As a counterpart of the disparate impact concept in fair supervised machine learning, balance essentially aims at ensuring that the representation of protected groups in each cluster preserves the global proportion of each protected group.

Depending on the stage of clustering in which the fairness requirements are imposed, the prior works on fair clustering are categorized into three families, namely pre-processing, in-processing, and post-processing. A large body of the literature work [5, 12, 20, 30] falls into the pre-processing category. The whole dataset is first decomposed into small subsets named *fairlets*, where the desired balance can be guaranteed. Any resulting solution from classical clustering algorithms using the set of fairlets will then be fair. Chierichetti et al. [12] focused on the case of two demographic groups and formulated explicit combinatorial problems (such as perfect matching and minimum cost flow problems) to decompose the dataset into minimal fair sets defining the fairlets. Their theoretical analysis gave strong guarantees on the quality of the fair clustering solutions for k-center and k-median problems. Following that line of work, Backurs et al. [5] embedded the whole dataset into a hierarchical structure tree and improved the time complexity of the fairlet decomposition step from quadratic to nearly linear time (in the dataset size). Schmidt et al. [30] introduced the notion of fair *coresets* and proposed an efficient streaming fair clustering algorithm for k-means. They introduced a near-linear time algorithm to construct coresets that helps reduce the input data size and hence speeds up any fair clustering algorithm. Huang et al. [20] further boosted the efficiency of coresets construction and made a generalization to multiple non-disjoint demographic groups for both k-means and k-median.

On the contrary, post-processing clustering methods [3, 7, 22, 29] modify the resulting clusters from classical clustering algorithms to improve fairness. For example, Bera et al. [7] proposed a fair re-assignment problem as a linear relaxation of an integer programming model given the clustering results from any vanilla k-means, k-median, or k-center algorithms. They showed how to derive a

(ρ+2)-approximation fair clustering algorithm from any ρ-approximation vanilla clustering algorithm within a theoretical bound of fairness constraints violation. Moreover, their framework works for datasets with multiple and potentially overlapping demographic groups. Lastly, in-processing methods incorporate the fairness constraints into the clustering process [2, 24, 32]. Our approach falls into this category and allows for the determination of the trade-offs between clustering costs and fairness. To our knowledge, the only such other in-processing approach is the one of Ziko et al. [32], where the clustering balance is approximately measured by the KL-divergence and imposed as a penalty term in the fair clustering objective function. The penalty coefficient is then used to control the trade-offs between clustering cost/fairness.

Our contribution The partitioning clustering model, also referred to as the center-based clustering model, consists of selecting a certain number K of centers and assigning data points to their closest centers. In this paper, we will focus on the well-known k-means model, and we will introduce a novel fair clustering algorithm using the balance measure. The main challenge of the fair clustering task comes from the violation of the assignment routine, which then indicates that a data point is no longer necessarily assigned to its closest cluster. The higher the balance level one wants to achieve, the more clustering cost is added to the final clustering. Hence, there exists a natural conflict between the fairness level, when measured in terms of balance, and the classical k-means clustering objective.

We explicitly formulate the trade-offs between the k-means clustering cost and the fairness as a bi-objective optimization problem, where both objectives are written as nonconvex and nonsmooth functions of binary assignment variables defining point assignments in the clustering model (see (2) further below). Our goal is to construct an informative approximation of the Pareto front for the proposed bi-objective fair k-means clustering problem, without exploring exhaustively the binary nature of the assignment variables. The most widely used method in solving general bi-objective optimization problems is the so-called weighted-sum method [17]. There, one considers a set of single objective problems, formed by convex linear combinations of the two functions, and (a portion of) the Pareto front might be approximated by solving the corresponding weighted-sum problems. However, this methodology has no rigorous guarantees due to the nonconvexity of both objective functions. Also, the non-smoothness of the fairness objective poses an additional difficulty to the weighted-sum method, as one function will be smooth and the other one no. Moreover, even ignoring the nonconvexity and non-smoothness issues, the two objectives, namely the clustering cost and the clustering balance, can have significantly different magnitudes. One can hardly preselect a good set of weights corresponding to decision-makers' preferences to capture a well-spread Pareto front.

Therefore, we were motivated to design a novel stochastic alternating balance fair k-means (SAfairKM) algorithm, inspired from the classical mini-batch k-means algorithm, which essentially consists of alternatively taking pure mini-batch k-means updates and swap-based balance improvement updates. In fact,

the number of k-means updates (denoted by n_a) and the number of swap updates (denoted by n_b) play a role similar to the weights in the weighted-sum method, parameterizing the efforts of optimizing each objective. In the pure mini-batch k-means updates, we focus on minimizing the clustering cost. A mini-batch of points is randomly drawn and assigned to their closest clusters, after which the set of centers are updated using mini-batch stochastic gradient descent. In the swap-based balance improvement steps, we aim at increasing the overall clustering balance. For this purpose, we propose a simple swap routine that is guaranteed to increase the overall clustering balance by swapping data points between the minimum balance cluster and a target well-balanced cluster. Similarly to the k-means updates, the set of centers are updated using the batch of data points selected to swap. While the k-means updates reproduce the stochastic gradient descent directions for the clustering cost function, the swap updates can be seen as taking steps along some increasing directions for the clustering balance objective (not necessarily the best ascent direction).

We have evaluated the performance of the proposed SAfairKM algorithm using both synthetic datasets and real datasets. To endow SAfairKM with the capability of constructing a Pareto front in a single run, we use a list of non-dominated points updated at every iteration. The list is randomly generated at the beginning of the process. At every iteration, and for every point in the current list, we apply SAfairKM for all considered pairs of (n_a, n_b). For each pair (n_a, n_b), one does n_a k-means updates and n_b swap updates. At the end of each iteration, we remove from the list all dominated points (those for each there exists another one with higher clustering cost and lower clustering balance). Such a simple mechanism is also beneficial for excluding bad local optima, considering that the two objectives are nonconvex. We will present the full trade-offs between the two conflicting objectives for four synthetic datasets and two real datasets. A numerical comparison with the fair k-means algorithm proposed in [32] further confirms the robustness and efficiency of the proposed algorithm in constructing informative and high-quality trade-offs.

2 The mini-batch k-means algorithm

In the classical k-means problem, one aims to choose K centers (representatives) and to assign a set of points to their closest centers. The k-means objective is the sum of the minimum (squared Euclidean) distance of all points to their corresponding centers. Given a set of N points $P = \{x_p\}_{p=1}^N$, where x_p is the non-sensitive feature vector, the goal of clustering is to assign N points to K clusters identified by K centroids $C = [c_1, \ldots, c_K]^\top$. Let $[M]$ denote the set of positive integers up to M for any $M \in \mathbb{N}$. The k-means clustering problem is formulated as the minimization of a nonsmooth function of the set of centroids:

$$\min \ f_1^{KM}(C) \ = \ \frac{1}{2} \sum_{p=1}^{N} \min_{k \in [K]} \|x_p - c_k\|^2. \tag{1}$$

Since each data point is assigned to the closest cluster, the K cluster centroids are implicitly dependent on the point assignments. Let $s_{p,k} \in \{0,1\}$ be an assignment variable who takes the value 1 if point x_p is assigned to cluster k, and 0 otherwise. For simplicity, we denote s_k, $k \in [K]$, as an N-dimensional assignment vector for cluster k, and $s_p, p \in [N]$, as a K-dimensional assignment vector for point x_p. Let $X \in \mathbb{R}^{N\times d}$ be the data matrix stacking N data points of dimension d and $e_N \in \mathbb{R}^N$ be an all-ones vector. Then one can compute each centroid using $c_k = X^\top s_k / e_N^\top s_k$.

In practice, Lloyd's heuristic algorithm [26], also known as the standard batch k-means algorithm, is the simplest and most popular k-means clustering algorithm, and converges to a local minimum but without worst-case guarantees [21, 31]. The main idea of Lloyd's heuristic is to keep updating the K cluster centroids and assigning the full batch of points to their closest centroids.

In the standard batch k-means algorithm, one can compute the full gradient of the objective function (1) with respect to k-th center by $\nabla_{c_k} f_1^{KM}(C) = \sum_{x_p \in \mathcal{C}_k}(c_k - x_p)$, where $\mathcal{C}_k, k \in [K]$, is the set of points assigned to cluster k. Whenever there exists a tie, namely a point that has the same distance to more than one cluster, one can randomly assign the point to any of such clusters. A full batch gradient descent algorithm would iteratively update the centroids by $c_k^{t+1} - c_k^t = \alpha_k^t \sum_{x_p \in \mathcal{C}_k}(x_p - c_k), \forall k \in [K]$, where $\alpha_k^t > 0$ is the step size. Let N_k^t be the number of points in cluster k at iteration t. It is known that the full batch k-means algorithm with $\alpha_k^t = 1/N_k^t$ converges to a local minimum as fast as Newton's method, with a superlinear rate [9].

The standard batch k-means algorithm is proved to be slow for large datasets. Bottou and Bengio [9] proposed an online stochastic gradient descent (SGD) variant that takes a gradient descent step using one sample at a time. Given a new data point x_p to be assigned, a stochastic gradient descent step would look like $c_k^{t+1} = c_k^t + \alpha_k^t(x_p - c_k^t)$ if x_p is assigned to cluster k. While the SGD variant is computationally cheap for large datasets, it finds solutions of lower quality than the batch algorithm due to the stochasticity. The mini-batch version of the k-means algorithm uses a mini-batch sampling to lower stochastic noise and, in the meanwhile, speed up the convergence. The detailed mini-batch k-means is given in Algorithm 1.

3 A new stochastic alternating balance fair k-means method

3.1 The bi-objective balance k-means formulation

Balance [12] is the most widely used fairness measure in the literature of fair clustering. Consider J disjoint demographic groups. Let V_j represent the set of points in demographic group $j \in [J]$. Then, $v_{p,j}$ takes the value 1 if point $x_p \in V_j$. We denote v_j as an N-dimensional indicator vector for the demographic group $j \in [J]$. The balance of cluster k is formally defined as $b_k = \min_{j\neq j'} v_j^\top s_k / v_{j'}^\top s_k \leq 1, \forall k \in [K]$, which calculates the minimum ratio among different pairs of protected groups. The overall clustering balance is the minimum balance over all

Algorithm 1 Mini-batch k-means algorithm

Input: The set of points P and an integer K.
Output: The set of centers $C = \{c_1, \ldots, c_K\}$.
Randomly select K points as initial centers.
for $t = 0, 1, 2, \ldots$ **do**
  Randomly sample a batch of points B_t.
  for $k = 1, \ldots, K$ **do**
    Identify the set of points $B_t^k \subseteq B_t$ whose closest center is c_k.
    $N_k = N_k + |B_t^k|$.
    $c_k = c_k + \frac{1}{N_k} \sum_{x_p \in B_t^k} (x_p - c_k)$.

clusters, i.e., $b = \min_{k=1}^{K} b_k$. The higher the overall balance, the fairer the clustering.

By the definition of cluster balance given above, the balance function can be easily computed only using the assignment variables. The k-means objective (1) can be rewritten as a function of the assignment variables as well. Hence, one can directly formulate the inherent trade-off between clustering cost and balance as a bi-objective optimization problem, i.e.,

$$\min \quad (f_1(s), -f_2(s)) \quad \text{s.t.} \quad \sum_{k=1}^{K} s_{p,k} = 1, \forall p \in [N], \quad s \in \{0,1\}^{N \times K}, \tag{2}$$

where s is the binary-valued assignment matrix with column vectors $s_k, k \in [K]$, and row vectors $s_p, p \in [N]$, and

$$f_1(s) \;=\; \frac{1}{N} \sum_{k=1}^{K} \sum_{p=1}^{N} s_{p,k} \|x_p - c_k\|^2, \quad \text{with } c_k \;=\; \frac{X^\top s_k}{e_N^\top s_k} = \frac{\sum_{p=1}^{N} x_p s_{p,k}}{\sum_{p=1}^{N} s_{p,k}},$$

$$f_2(s) \;=\; \min_{k \in [K]} \; \min_{\substack{j \neq j' \\ j, j' \in [J]}} \frac{v_j^\top s_k}{v_{j'}^\top s_k}.$$

The two constraints in (2) ensure that one point can only be assigned to one cluster. Note that both objectives are nonconvex functions of the binary assignment variables.

3.2 The stochastic alternating balance fair k-means method

We propose a novel stochastic alternating balance fair k-means clustering algorithm to compute a nondominated solution on the Pareto front. We will use a simple but effective alternating update mechanism, which consists of improving *either* the clustering objective *or* the overall balance, by iteratively updating cluster centers and assignment variables. Specifically, every iteration of the proposed algorithm contains two sets of updates, namely pure k-means updates

and pure swap-based balance improvement steps. The pure k-means updates were introduced in Section 2, and will consist of taking a certain number of stochastic k-means steps. In the balance improvement steps, a certain batch of points is selected and swapped between the minimum balanced cluster and a target well-balanced cluster.

Balance improvement steps At the current iteration, let $\mathcal{C}_l$ be the cluster with the minimum balance. Then $\mathcal{C}_l$ is the bottleneck cluster that defines the overall clustering balance. Without loss of generality, we assume that $b_l = |\mathcal{C}_l \cap V_1|/|\mathcal{C}_l \cap V_2|$, which then implies that the pair of demographic groups (V_1, V_2) forms a key to improve the balance of cluster $\mathcal{C}_l$, as well as the overall clustering balance. In terms of the assignment variables, we have

$$b \;=\; b_l \;=\; \frac{v_1^\top s_l}{v_2^\top s_l} \;=\; \frac{\sum_{p=1}^N v_{p,1} s_{p,l}}{\sum_{p=1}^N v_{p,2} s_{p,l}}. \tag{3}$$

One way to determine a target well-balanced cluster $\mathcal{C}_h$ is to select it as the one with the maximum ratio between V_1 and V_2, i.e.,

$$h \;\in\; \operatorname{argmax}_{k\in[K]} \; \left\{ v_1^\top s_k / v_2^\top s_k, v_2^\top s_k / v_1^\top s_k \right\}. \tag{4}$$

Another way to determine such a target cluster is to select a cluster $\mathcal{C}_h$ that is closest to $\mathcal{C}_l$, i.e.,

$$h \;\in\; \operatorname{argmin}_{k\in[K], k\neq l} \; \|c_k - c_l\|. \tag{5}$$

We call the target cluster computed by (4) a *global* target and the one selected by (5) a *local* target. Using a global target cluster makes the swap updates more efficient and stable in the sense that the target cluster is only changed when the minimum balanced cluster changes. Instead, swapping according to the local target leads to less increase in clustering costs.

To improve the overall balance, one swaps a point in cluster $\mathcal{C}_l$ belonging to V_2 with a point in cluster $\mathcal{C}_h$ belonging to V_1. Each of these swap updates will guarantee an increase in the overall balance. The detailed stochastic alternating balance fair k-means clustering algorithm is given in Algorithm 2. At each iteration, we alternate between taking k-means updates using a drawn batch of points (denote the batch size by n_a) and "swap" updates using another drawn batch of points (denote the batch size by n_b). The generation of the two batches is independent. The choice of n_a and n_b influences the nondominated point obtained at the end, in terms of the weight put into each objective.

Instead of randomly selecting points to swap in line 11 of Algorithm 2, in our experiments we have used a more accurate swap strategy by increasing the batch size. Basically, we randomly sample a batch of points from $\mathcal{C}_l \cap V_2$ (resp. $\mathcal{C}_h \cap V_1$) and select x_p (resp. x'_p) as the one closest to $\mathcal{C}_h$ (resp. $\mathcal{C}_l$). The batch size could be increased as the algorithm proceeds. Our numerical experiments show that the combination of local target clusters and the increasingly accurate swap strategy result in better numerical performance.

Algorithm 2 Stochastic alternating balance fair k-means clustering (SAfairKM) algorithm

Input: The set of points P, an integer K, and parameters n_a, n_b.
Output: The set of clustering labels $\Delta = \{\delta_1, \ldots, \delta_N\}$, where $\delta_p \in [K]$.
Randomly initialize labels $\{\delta_1, \ldots, \delta_N\}$ and a set of counters $\{N_1, \ldots, N_K\}$. Compute k-means centers $\{c_1, \ldots, c_K\}$ and balances $\{b_1, \ldots, b_K\}$ for all clusters.
for $t = 1, 2, \ldots$ **do**
  Randomly sample a batch of n_a points $B_t \subseteq P$ without replacement.
  for $x_p \in B_t$ **do**
    Decrease the counter $N_{\delta_p} = N_{\delta_p} - 1$ for the previous clustering label.
    Identify its closest center index i_p. Update clustering label $\delta_p = i_p$.
    Increase the counter $N_{\delta_p} = N_{\delta_p} + 1$ and center $c_{\delta_p} = c_{\delta_p} + \frac{1}{N_{\delta_p}}(x_p - c_{\delta_p})$.
  for $r = 1, 2, \ldots, n_b$ **do**
    Identify $\mathcal{C}_l$, $\mathcal{C}_h$, and the pair of demographic groups (V_1, V_2) according to (3) and (5).
    Randomly select points $x_p \in \mathcal{C}_l \cap V_2$ and $x_{p'} \in \mathcal{C}_h \cap V_1$.
    Swap points: set $\delta_p = h$ and $\delta_{p'} = l$.
    Update centers $c_l = c_l + \frac{1}{N_l}(x_{p'} - c_l)$ and $c_h = c_h + \frac{1}{N_h}(x_p - c_h)$.
    Update balance for clusters $\mathcal{C}_l$ and $\mathcal{C}_h$.

One could have converted the bi-objective optimization problem (2) into a weighted-sum function using the weights associated with the decision-maker's preference. However, optimizing such a weighted-sum function hardly reflects the desired trade-off due to significantly different magnitudes of the two objectives. Moreover, the existing k-means algorithm frameworks, including Lloyd's heuristic algorithm, are not capable of directly handling the weighted-sum objective function. In our proposed SAfairKM algorithm, the pair (n_a, n_b) plays a role similar to the weights in the weighted-sum method.

4 Numerical experiments

4.1 Pareto front SAfairKM algorithm

In our implementation[3], to obtain a well-spread Pareto front, we frame the SAfairKM algorithm into a Pareto front version using a list updating mechanism. See Algorithm 3 for a detailed description. In the initialization phase, we specify a sequence of pairs of the number of k-means updates and swap updates $\mathcal{W} = \{(n_a, n_b) : n_a + n_b = n_{\text{total}}, n_a, n_b \in \mathbb{N}_0\}$, and we generate a list of random initial clustering labels $\mathcal{L}_0$. Then we run Algorithm 2 for a certain number of

[3] Our implementation code is available at `https://github.com/sul217/SAfairKM`. All the experiments were conducted on a MacBook Pro Intel Core i5 processor.

iterations ($q = 1$ in our experiments) parallelly for each label in the current list $\mathcal{L}_t$, resulting in a new list of clustering labels $\mathcal{L}_{t+1}$. At the end of each iteration, the list is cleaned up by removing all the dominated points from $\mathcal{L}_{t+1}$. Using this algorithm, the list of nondominated points is refined towards the true real Pareto front. The process can be terminated when either the number of nondominated points is greater than a certain budget (1500 in our experiments) or when the total number of iterations exceeds a certain limit (depending on the size of the dataset).

Algorithm 3 Pareto-Front SAfairKM Algorithm

Generate a list of starting labels $\mathcal{L}_0$. Select parameter $q \in \mathbb{N}$ and a sequence of pairs $\mathcal{W} = \{(n_a, n_b) : n_a + n_b = n_{\text{total}}, n_a, n_b \in \mathbb{N}_0\}$.
for $t = 0, 1, \ldots$ **do**
    Set $\mathcal{L}_{t+1} = \mathcal{L}_t$.
    for each clustering label Δ in the list $\mathcal{L}_{k+1}$ **do**
        for $(n_a, n_b) \in \mathcal{W}$ **do**
            Apply q iterations of Algorithm 2 starting from Δ using the parameters (n_a, n_b).
            Add the final output label to the list $\mathcal{L}_{t+1}$.
    Remove all the dominated points from $\mathcal{L}_{t+1}$: **for** each label Δ in the list $\mathcal{L}_{t+1}$ **do**
    If $\exists\ \Delta' \in \mathcal{L}_{t+1}$ such that $f_1(\Delta') < f_1(\Delta)$ and $f_2(\Delta') > f_2(\Delta)$ hold, remove Δ.

To the best of our knowledge, the only approach in the literature providing a mechanism of controlling trade-offs between the two conflicting objectives was suggested by [32] and briefly described in Appendix A. Their approach (here called VfairKM) consists of solving (6) for different penalty coefficients μ, resulting in a set of solutions from which we then remove dominated solutions to obtain an approximated Pareto front. To ensure a fair comparison, we select a set of penalty coefficients evenly from 0 to an upper bound $\mu_{\max}$, which is determined by pre-experiments such that the corresponding fairness error is less than 0.01 or no longer possibly decreased when further increasing its value. In some cases, we found that VfairKM is not able to produce a fairer clustering outcome when the penalty coefficient is greater than $\mu_{\max}$ due to numerical instability.

In addition, we compare the Pareto fronts computed by SAfairKM with the fair k-means solution obtained from the postprocessing fair assignment approach proposed in [7] (marked as FairAssign). In [7], the fairest clustering solution is computed by first using a standard clustering algorithm and then applying the so-called fair assignment procedure. Such a fair assignment procedure consists of solving a linear programming relaxation of an integer programming problem, followed by an iterative rounding procedure to satisfy the bound constraints $\beta_j \leq |\mathcal{C}_k \cap V_j|/|\mathcal{C}_k| \leq \gamma_j, \forall j \in [J], k \in [K]$, where $\beta_j \in [0, 1]$ and $\gamma_j \in [0, 1]$ are lower and upper fairness bounds respectively. In our case, however, and in order to get the fairest solution, we set $\beta_j = \gamma_j = |V_j|/N$ which is exactly the proportion of demographic group j in the input dataset. Finally, we also present as benchmarks the k-means solutions obtained by both the state-of-

the-art Lloyd's algorithm (denoted as VanillaKM) and the mini-batch k-means algorithm (denoted as MinibatchKM). Both VanillaKM and MinibatchKM were equipped with the well-known k-means++ initialization [4].

4.2 Numerical results

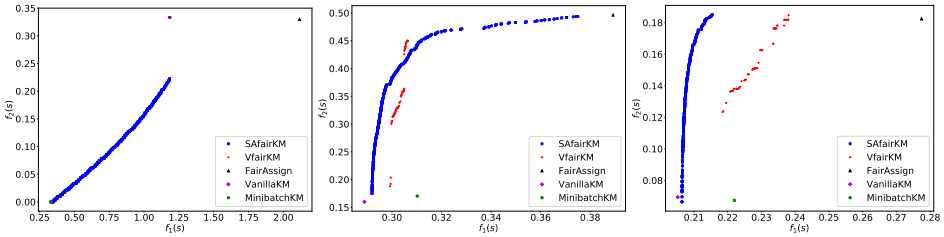
Trade-offs for synthetic datasets We randomly generated four synthetic datasets from Gaussian distributions, and their demographic compositions are given in Figure 2 of Appendix B. Each synthetic dataset has 400 data points in the $\mathbb{R}^2$ space and two demographic groups ($J = 2$) marked by black/circle and purple/triangle.

Using the list update mechanism (described by Algorithm 3), we are able to obtain a well-spread Pareto front with comparable quality for each of the synthetic datasets. Recall that we are minimizing the clustering cost and maximizing the clustering balance. The closer the Pareto front is to the upper left corner, the higher its quality. In particular, Figure 1 (a) gives the approximated Pareto front for the Syn_unequal_ds2 dataset with $K = 2$, which confirms the natural conflict between the clustering cost and the clustering balance. One can see that the VfairKM algorithm is not able to output any trade-off information as it always finds the fairest solution regardless of the value of μ. Due to the special composition of this dataset, the Pareto front generated by SAfairKM is disconnected (the point around $(1.25, 0.35)$ is both VfairKM and SAfairKM). Results for the other three synthetic datasets are given by Figures 3-6 in Appendix B. For all the synthetic datasets, the left end point on the Pareto front given by SAfairKM is consistent with the solution of VanillaKM. On the right end of the Pareto fronts, the fair solution given by FairAssign is dominated by the fairest solution identified by our approach.

Trade-offs for real datasets Two real datasets *Adult* [25] and *Bank* [28] are taken from the UCI machine learning repository [14]. The *Adult* dataset contains $32,561$ samples. Each instance is characterized by 12 nonsensitive features (including age, education, hours-per-week, capital-gain, and capital-loss, etc.). For the clustering purpose, only five numerical features among the 12 features are kept. The demographic proportion of the *Adult* dataset is $[0.67, 0.33]$ in terms of gender ($J = 2$), which corresponds to a dataset balance of 0.49. The *Bank* dataset contains $41,108$ data samples. Six nonsensitive numerical features (age, duration, number of contacts performed, consumer price index, number of employees, and daily indicator) are selected for the clustering task. Its demographic composition in terms of marital status ($J = 3$) is $[0.11, 0.28, 0.61]$, and hence the best clustering balance one can achieve is 0.185.

For the purpose of a faster comparison, we randomly select a subsample of size 5000 from the original datasets and set the number of clusters to $K = 10$. The resulting solutions from the five algorithms are given in Figure 1 (b)-(c). For both datasets, SAfairKM is able to produce more spread-out Pareto fronts which capture a larger range of balance, and hence provide more complete trade-offs between the two conflicting goals. In terms of Pareto front quality (meaning

dominance of one over the other), SAfairKM also performs better than VfairKM. In fact, we can see from Figure 1 (b)-(c) that the Pareto fronts generated by SAfairKM dominate most of the solutions given by VfairKM and FairAssign. Also, the left end of the Pareto front generated by SAfairKM is much closer to the solution given by VanillaKM than VfairKM. The Pareto fronts corresponding to $K = 5$ are also given in Figure 7 of Appendix B. Overall, SAfairKM results in a Pareto front of higher spread and slightly lower quality than VfairKM for the Adult dataset, while the Pareto front output from SAfairKM has better spread and higher quality for the Bank dataset.



(a) Syn_unequal_ds2 ($K = 2$). (b) Adult ($K = 10$). (c) Bank ($K = 10$).

Fig. 1: Pareto fronts: SAfairKM: 400 iterations for Syn_unequal_ds2, 2500 iterations for Adult, and 8000 iterations for Bank, 30 starting labels, and 4 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 0$ for Syn_unequal_ds2, $\mu_{\max} = 3260$ for Adult, and $\mu_{\max} = 2440$ for Bank.

Performance in terms of spread and quality of Pareto fronts SAfairKM is able to generate more spread-out and higher-quality Pareto fronts regardless of the data distribution (see the trade-off results for the four synthetic datasets). The robustness partially comes from the list update mechanism which establishes a connection among parallel runs starting from different initial points and pairs (n_a, n_b), and thus helps escape from bad local optima.

Table 1: Average CPU times per nondominated solution.

Dataset	SAfairKM	VfairKM	Dataset	SAfairKM	VfairKM
Syn_equal_ds1	0.80	1.06	Adult ($K = 10$)	18.52	40.43
Syn_unequal_ds1	0.81	0.98	Bank ($K = 10$)	59.12	76.29
Syn_equal_ds2	0.70	1.29	Adult ($K = 5$)	14.88	15.08
Syn_unequal_ds2	0.80	0.10	Bank ($K = 5$)	11.97	50.31

Performance in terms of computational time Since the two algorithms (SAfairKM and VfairKM) generally produce Pareto fronts of different cardinalities, we evaluate their computational efforts by the average CPU time spent per computed nondominated solution (see Table 1). Our algorithm was shown to be clearly more computationally efficient than VfairKM.

5 Concluding remarks

We have investigated the natural conflict between the k-means clustering cost and the clustering balance from the perspective of bi-objective optimization, for which we designed a novel stochastic alternating algorithm (SAfairKM). A Pareto front version of SAfairKM has efficiently computed well-spread and high-quality trade-offs, when compared to an existing approach based on a penalization of fairness.

Note that a balance improvement routine for the SAfairKM algorithm could be derived to handle more than one demographic group. One might formulate a multi-objective problem with the clustering cost being one objective and the balance corresponding to each protected attribute (e.g., race and gender) written as separate objectives. The balance measured using each attribute can be improved via alternating swap updates with respect to each balance objective.

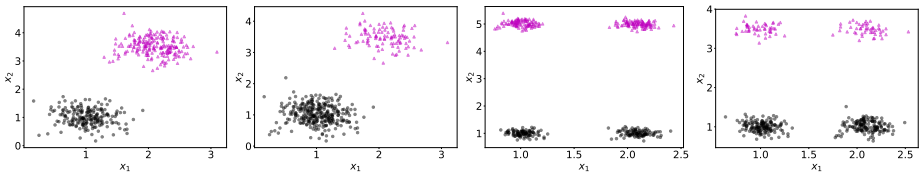
A Description of an existing approach for comparison

The authors in [32] considered the fairness error computed by the Kullback-Leibler (KL)-divergence, and added it as a penalized term to the classical clustering objective. When using the k-means clustering cost, the resulting problem takes the form:

$$\min \ f_1(s) + \mu \sum_{k=1}^{N} \mathcal{D}_{KL}(U \| \mathbb{P}_k) \quad \text{s.t.} \ \sum_{k=1}^{K} s_{p,k} = 1, \forall p \in [N], \tag{6}$$

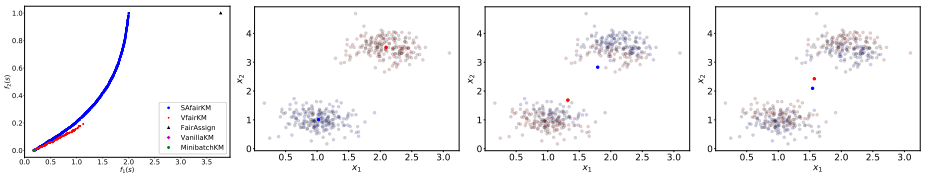
where $\mathcal{D}_{KL}$ is the KL divergence between the desired demographic proportion $U = [u_j, j \in [J]]$ (usually specified by the demographic composition of the whole dataset) and the marginal probability $\mathbb{P}_k = [\mathbb{P}(j|k) = s_k^\top v_j / {e_N}^\top s_k, j \in [J]]$. The penalty coefficient μ associated with the fairness error is the tool to control the trade-offs between the clustering cost and the clustering balance. To solve problem (6) for a fixed $\mu \geq 0$, the authors in [32] have developed an optimization scheme based on a concave-convex decomposition of the fairness term.

B More numerical results



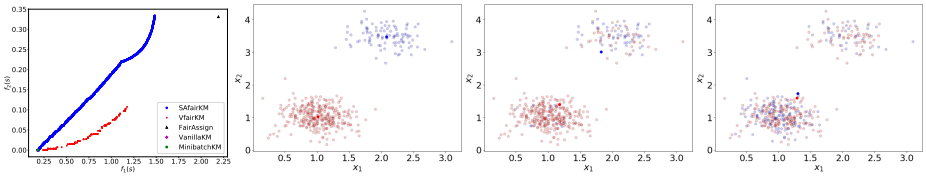
(a) Syn_equal_ds1: $|V_1| : |V_2| = 1 : 1$. (b) Syn_unequal_ds1: $|V_1| : |V_2| = 1 : 3$. (c) Syn_equal_ds2: $|V_1| : |V_2| = 1 : 1$. (d) Syn_unequal_ds2: $|V_1| : |V_2| = 1 : 3$.

Fig. 2: Demographic composition of four synthetic datasets.



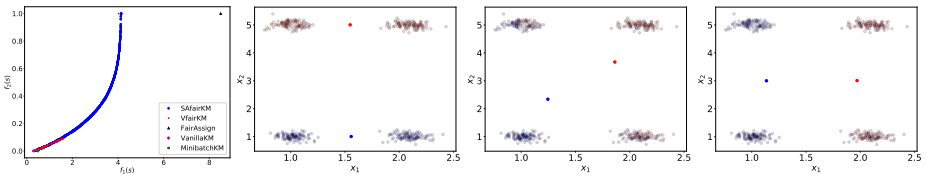
(a) Pareto front. (b) Clust. balance $b = 0$. (c) Clust. balance $b = 0.5$. (d) Clust. balance $b = 1$.

Fig. 3: Syn_equal_ds1 data: SAfairKM: 400 iterations, 10 starting labels, and 3 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 202$.



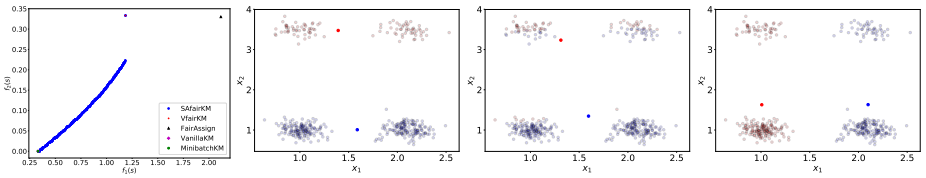
(a) Pareto front. (b) Clust. balance $b = 0$. (c) Clust. balance $b = 0.16$. (d) Clust. balance $b = 0.33$.

Fig. 4: Syn_unequal_ds1 data: SAfairKM: 400 iterations, 10 starting labels, and 3 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 223$.



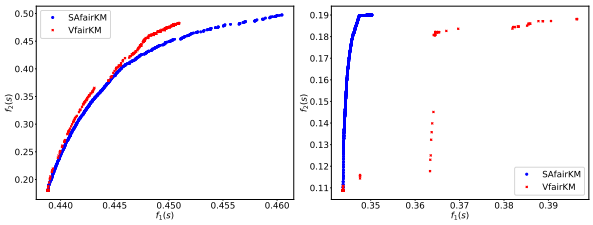
(a) Pareto front. (b) Clust. balance $b = 0$. (c) Clust. balance $b = 0.5$. (d) Clust. balance $b = 1$.

Fig. 5: Syn_equal_ds2 data: SAfairKM: 400 iterations, 10 starting labels, and 3 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 60$.



(a) Pareto front. (b) Clust. balance $b = 0$. (c) Clust. balance $b = 0.17$. (d) Clust. balance $b = 0.33$.

Fig. 6: Syn_unequal_ds2 data: SAfairKM: 400 iterations, 10 starting labels, and 3 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 0$.



(a) Adult. (b) Bank.

Fig. 7: Pareto fronts for $K = 5$: SAfairKM: 2500 iterations for Adult and 1500 iterations for Bank, 30 starting labels, and 4 pairs of (n_a, n_b); VfairKM: $\mu_{\max} = 6190$ for Adult and $\mu_{\max} = 4790$ for Bank.

References

1. Abbasi, M., Bhaskara, A., Venkatasubramanian, S.: Fair clustering via equitable group representations. In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. pp. 504–514 (2021)
2. Abraham, S.S., Sundaram, S.S.: Fairness in clustering with multiple sensitive attributes. arXiv preprint arXiv:1910.05113 (2019)
3. Ahmadian, S., Epasto, A., Kumar, R., Mahdian, M.: Clustering without over-representation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 267–275 (2019)
4. Arthur, D., Vassilvitskii, S.: k-means++ the advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. pp. 1027–1035 (2007)
5. Backurs, A., Indyk, P., Onak, K., Schieber, B., Vakilian, A., Wagner, T.: Scalable fair clustering. In: International Conference on Machine Learning. pp. 405–413. PMLR (2019)
6. Barocas, S., Selbst, A.D.: Big data's disparate impact. California Law Review p. 671 (2016)
7. Bera, S., Chakrabarty, D., Flores, N., Negahbani, M.: Fair algorithms for clustering. In: Advances in Neural Information Processing Systems. pp. 4954–4965 (2019)
8. Berkhin, P.: A survey of clustering data mining techniques. In: Grouping multidimensional data, pp. 25–71. Springer (2006)
9. Bottou, L., Bengio, Y.: Convergence properties of the k-means algorithms. In: Advances in neural information processing systems. pp. 585–592 (1995)
10. Calders, T., Kamiran, F., Pechenizkiy, M.: Building classifiers with independency constraints. In: 2009 IEEE International Conference on Data Mining Workshops. pp. 13–18. IEEE (2009)
11. Chen, X., Fain, B., Lyu, L., Munagala, K.: Proportionally fair clustering. In: International Conference on Machine Learning. pp. 1032–1041 (2019)
12. Chierichetti, F., Kuma, R., Lattanzi, S., Vassilvitskii, S.: Fair clustering through fairlets. In: Advances in Neural Information Processing Systems. pp. 5029–5037 (2017)
13. Datta, A., Tschantz, M.C., Datta, A.: Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. Proceedings on privacy enhancing technologies **2015**, 92–112 (2015)

14. Dua, D., Graff, C.: UCI Machine Learning Repository (2017), `http://archive.ics.uci.edu/ml`
15. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd innovations in theoretical computer science conference. pp. 214–226. ACM (2012)
16. Gan, G., Ma, C., Wu, J.: Data clustering: theory, algorithms, and applications. SIAM (2020)
17. Gass, S., Saaty, T.: The computational algorithm for the parametric objective function. Nav. Res. Logist. Q. **2**, 39–45 (1955)
18. Ghadiri, M., Samadi, S., Vempala, S.: Socially fair k-means clustering. In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. pp. 438–448 (2021)
19. Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. In: Advances in neural information processing systems. pp. 3315–3323 (2016)
20. Huang, L., Jiang, S., Vishnoi, N.: Coresets for clustering with fairness constraints. In: Advances in Neural Information Processing Systems. pp. 7589–7600 (2019)
21. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: A local search approximation algorithm for k-means clustering. Computational Geometry **28**, 89–112 (2004)
22. Kleindessner, M., Awasthi, P., Morgenstern, J.: Fair k-center clustering for data summarization. In: International Conference on Machine Learning. pp. 3448–3457. PMLR (2019)
23. Kleindessner, M., Awasthi, P., Morgenstern, J.: A notion of individual fairness for clustering. arXiv preprint arXiv:2006.04960 (2020)
24. Kleindessner, M., Samadi, S., Awasthi, P., Morgenstern, J.: Guarantees for spectral clustering with fairness constraints. In: International Conference on Machine Learning. pp. 3458–3467. PMLR (2019)
25. Kohavi, R.: Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. p. 202–207. KDD'96, AAAI Press (1996)
26. Lloyd, S.: Least squares quantization in PCM. IEEE transactions on information theory **28**, 129–137 (1982)
27. Mahabadi, S., Vakilian, A.: Individual fairness for k-clustering. In: Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 6586–6596. PMLR, Virtual (13-18 Jul 2020)
28. Moro, S., Cortez, P., Rita, P.: A data-driven approach to predict the success of bank telemarketing. Decision Support Systems **62**, 22–31 (2014)
29. Rösner, C., Schmidt, M.: Privacy preserving clustering with constraints. In: 45th International Colloquium on Automata, Languages, and Programming. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2018)
30. Schmidt, M., Schwiegelshohn, C., Sohler, C.: Fair coresets and streaming algorithms for fair k-means. In: International Workshop on Approximation and Online Algorithms. pp. 232–251. Springer (2019)
31. Selim, S.Z., Ismail, M.A.: k-means-type algorithms: A generalized convergence theorem and characterization of local optimality. IEEE Transactions on pattern analysis and machine intelligence pp. 81–87 (1984)
32. Ziko, I.M., Granger, E., Yuan, J., Ayed, I.B.: Variational fair clustering. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 11202–11209 (2021)