Clustering under natural stability assumptions

Pranjal Awasthi
Carnegie Mellon University
pawasthi@cs.cmu.edu

Avrim Blum
Carnegie Mellon University
avrim@cs.cmu.edu

Or Sheffet
Carnegie Mellon University
osheffet@cs.cmu.edu

Abstract

Optimal clustering is a challenging task. Recently, several papers have suggested a new approach to clustering, motivated by examining natural assumptions that arise in practice, or that are made *implicitly* by many standard algorithmic approaches. These assumptions concern various measures of stability of our given clustering instance. The work of Balcan et al [BBG09] refers to stability with respect to approximations of the objective, and gives positive results for inputs such that all $(1+\alpha)$-approximations to the k-median (or k-means) optimal solution are close, as partitions of the data, to the actual desired clustering. A related assumption was considered by Ostrovsky et al. [ORSS06]. In this work we investigate how good these assumptions are in practice and conclude that for most data sets, these stability notions are too strict. As a result we propose a weaker notion of stability which is more practical and at the same time helps us bypass hardness results to get good approximations to k-median and k-means clustering. We also propose using our stability notion to do transfer clustering. Secondly, we also study the stability to perturbations assumption of Bilu and Linial [BL10]. We conclude that datasets often have large subcomponents which are stable to perturbations and propose a heuristic to find such stable components.

1 Introduction

There has been significant research on approximation algorithms for clustering under the natural k-means and k-median objective functions [ARR98, AGK$^+$01, BCR01, CGTS99, KSS04, dlVKKR03]. Unfortunately, in general, these objectives are not only NP-hard to optimize, but they are also NP-hard to approximate to better than certain constant factors [GK98, JMS02]. Recently, in the theoretical computer science community, a new line of research has emerged which tries to bypass these NP-hardness results by focusing on the kind of clustering instances which might arise in practice. These results show that focusing on such "stable" instances can often lead to good and efficient approximation algorithms. The goal of this work is to study to what extent these assumptions on the stability of a clustering instance hold in practice, and when they do hold, how can they help us in efficiently clustering the data. In this work we will be focusing on two classes of stability assumptions, a) Approximation based stability and b) Perturbation based stability.

1.1 Approximation based stability

For many applications such as clustering proteins by function, clustering documents by topic, or clustering images by who or what is in them, our real interest is in getting the right answer. Optimizing objectives such as k-median and k-means are only used as a proxy in helping us reach the right answer. Thus, we only really *care* about approximating these objectives well when data has the property that good approximations to the objective are also close to the desired answer. The work of Balcan, Blum and Gupta [BBG09] uses this idea to bypass approximation hardness results for k-median and k-means objectives. Specifically, they show that for these objectives, for any constant $\alpha > 0$, if data satisfies the property that all $(1+\alpha)$-approximations to the objective are ϵ-close to the desired clustering in terms of how points are partitioned, then one can efficiently get $O(\epsilon)$-close to the desired clustering *even though obtaining a* $1+\alpha$ *approximation to the objective is* NP*-hard* for $\alpha < \frac{1}{\epsilon}$. That is, one can perform nearly as well in terms of distance

to the desired solution as if one could approximate the objective to the NP-hard value. Balcan and Braverman [BB09] extend these results to the min-sum objective as well. While these results are motivated by exploring the foundations of approximation-based clustering, it is natural to ask whether these assumptions actually tend to hold true in practice.

A related assumption was proposed by Ostrovsky et al. [ORSS06]. Specifically, they consider k-means instances where the optimal k-clustering has cost noticeably smaller than the cost of any $(k-1)$-clustering, motivated by the idea that "if a near-optimal k-clustering can be achieved by a partition into fewer than k clusters, then that smaller value of k should be used to cluster the data" [ORSS06]. Under the assumption that the ratio $\mathsf{OPT}(k-1)/\mathsf{OPT}(k) > \max\{100, 1/\alpha^2\}$, Ostrovsky et al. show that one can obtain a $(1 + f(\alpha))$-approximation for k-means, in time polynomial in n and k by using a variant on Lloyd's algorithm.

1.2 Perturbation based stability

The work of Bilu and Linial [BL10] was motivated by the fact that often when using clustering techniques in practice, one does not know exactly how to measure best distance between instances. Thus, unless the optimal on the given distances is correct by pure luck, it likely is correct or nearly so on small perturbations of the given distances as well. Hence they call a clustering instance *perturbation resilient* if the exact optimal solution under some objective function is the desired clustering, and that this is maintained even under bounded perturbations to the distance matrix (i.e., the optimum is stable to such perturbations). Bilu and Linial [BL10] analyze this type of assumption in the context of max-cut clustering and show that one can efficiently cluster instances which are stable up to large perturbations, roughly on the order of $\sqrt{n}$.

1.3 Our Results

In this work we empirically demonstrate that the stability assumptions of Balcan et al. [BBG09] and Ostrovsky et al. [ORSS06] are too strong to hold true in practice. To remedy this, we propose a weaker notion which we call as weak-deletion stability. This generalizes both the works of Balcan et al. and Ostrovsky et al. In addition, we demonstrate that weak-deletion stability often holds true in practice. We also theoretically demonstrate that under this notion of stability, one can get efficient approximation algorithms for the k-median and the k-means problems. We also further study the notion of stability considered by Bilu and Linial [BL10]. They argue that instances of Max-cut which are resilient to perturbations of size $O(n^{1/2})$ are easy to cluster. In addition, they conjecture that instances stable to as little as $O(1)$ perturbations should be solvable in polynomial time. In this paper we prove that this conjecture is true for any center-based clustering objective (such as k-median, k-means, and k-center), i.e., we can efficiently find the optimal clustering assuming only stability to factor of 3 perturbations of the underlying metric. Furthermore, we empirically demonstrate that in practice, clustering instances are not stable to large perturbations(factor 3). But they do have small "stable" subcomponents which are perturbation resilient. We propose and evaluate a heuristic to find such stable components in clustering instances. Finally, we propose using the notion of weak-deletion stability to do transfer clustering.

1.4 Related work

Other notions of stability for clustering have also been considered. The works of Ben-David et al. [BDvLP06, BDPS07] consider a notion of stability where the n data points come from a distribution. In their work, stability refers to the clustering *algorithm*, which is called stable if it outputs similar clusters for any set of m input points (drawn from the distribution). For k-means, the work of Meila [Mei06] discusses the opposite direction – classifying instances where a solution which is close to the target clustering is also a good approximation.

2 Notation and Preliminaries

We are given a set S of n points in a finite metric space or R^n, and we denote $d : S \times S \to \mathbb{R}_{\geq 0}$ as the distance function. Φ denotes the objective function we want to optimize over the metric. To minimize Φ we partition the n points into k disjoint subsets and assign a center c_i for each subset. In this work we will consider two objectives: k-median and k-means. For k-means, Φ is measured by $\sum_{i=1}^{k} \sum_{x \in C_i} d^2(x, c_i)$. Similarly, for k-median, Φ is measured by $\sum_{i=1}^{k} \sum_{x \in C_i} d(x, c_i)$. The optimal clustering w.r.t. Φ is denoted as $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_k^*\}$, and its cost is denoted as OPT. Clearly, in an optimal solution, we can output a list of k points as centers, $\{c_1^*, c_2^*, \ldots, c_k^*\}$, and assign each x

to its nearest center. Alternatively, given a k-partition $\{C_1^*, C_2^*, \ldots, C_k^*\}$, we can find the best point c_i^* to serve as the least-costly center for every cluster. We use $\mathcal{C}^*$ to denote both the optimal k-partition, and the optimal list of k centers. Given $\mathcal{C}^*$, we denote OPT_i as the contribution of the cluster i to OPT, that is $\mathsf{OPT}_i = \sum_{x \in C_i^*} d^2(x, c_i^*)$ for k-means and $\mathsf{OPT}_i = \sum_{x \in C_i^*} d(x, c_i^*)$ for k-median.

Given two clusterings $\mathcal{C}$ and $\mathcal{C}'$, define their distance as $\mathrm{dist}(\mathcal{C}, \mathcal{C}') = \min_{\pi \in S_k} \sum_{i=1}^{k} |\mathcal{C}_i \setminus C'_{\pi(i)}|$; i.e., the number of points clustered differently under the optimal matching between $\mathcal{C}'$ and $\mathcal{C}$.

We use $\mathcal{C}^{\mathcal{T}}$ to denote the target or desired clustering for a given application. We will often imagine that $\mathcal{C}^* = \mathcal{C}^{\mathcal{T}}$ (and without labeled data, the algorithm may as well assume this) but in reality this will often not be the case. In that case we let $\epsilon^* = \mathrm{dist}(\mathcal{C}^*, \mathcal{C}^{\mathcal{T}})$.

3 Stability assumptions of Balcan et al. and Ostrovsky et al.

The clustering instances considered by Balcan et al. [BBG09] have the property that *any* k-partition which is a $(1+\alpha)$-approximation of OPT, yields a clustering which is ϵ-close to the target clustering. They call this the $(1+\alpha, \epsilon)$-approximation property. For such instances they have the following results:

Theorem 3.1 (BBG 09, large clusters case) *If an instance of k-median clustering satisfies the $(1+\alpha, \epsilon)$-approximation property and every cluster in the target clustering ($\mathcal{C}^{\mathcal{T}}$) is of size at least $(4 + \frac{15}{\alpha})\epsilon n + 2$, then one can find a clustering which is ϵ-close to the target clustering in polynomial time.*

Theorem 3.2 (BBG 09, general case) *If an instance of k-median/k-means clustering satisfies the $(1+\alpha, \epsilon)$-approximation property then we can find a clustering which is $O(\frac{\epsilon}{\alpha})$-close to the target clustering in polynomial time.*

The work of Ostrovsky, Rabani, Schulman and Swamy [ORSS06] is motivated by considering separation conditions. They view the optimal k-clustering as the desired clustering not only in the sense that its cost is minimal among all possible k-clusterings, but also if it is considerably better then the cost of the optimal $(k-1)$-clustering. Denoting by $\mathsf{OPT}(k)$ (resp. $\mathsf{OPT}(k-1)$) the cost of the optimal k-clustering (resp. $(k-1)$-clustering), and introducing a parameter $\alpha > 0$, we define a clustering instance to be $(1+\alpha)$-*ORSS separable* if

$$\frac{\mathsf{OPT}(k-1)}{\mathsf{OPT}(k)} > 1 + \alpha$$

.

For such instances, Ostrovsky et al. have the following result:

Theorem 3.3 (ORSS 06) *If an instance of k-means clustering is $(1+\alpha)$-ORSS separable for $\alpha = max(100, \frac{1}{\epsilon^2})$, then one can get a $(1 + O(\epsilon^2))$-approximation to the k-means objective in polynomial time.*

4 Verifying the assumptions

In this section we study how well the assumptions of Balcan et al. [BBG09] and Ostrovsky et al. [ORSS06] hold true in practice. We will focus on k-median clustering for this section. In order to test whether a dataset is ORSS-separable or not, we need to compute the optimal k-median and the optimal $k-1$ median clustering of the given dataset. Since, in general this problem is NP-hard, we formulate it as an integer program and solve it using the CPLEX solver. This works reasonably efficiently for small datasets.

Verifying the two BBG assumptions (Theorems 3.1 and 3.2) is more difficult. Notice that in order to test whether a given instance satisfies the $(1+\alpha, \epsilon)$-approximation property, one must verify that "any" given k-partitioning of the dataset which is a $(1+\alpha)$-approximation to the optimal k-median score, is in fact ϵ-close to the target clustering $\mathcal{C}^{\mathcal{T}}$. This approach seems impractical and hence we are going to test some weaker conditions as suggested in [SYvZ10]. These conditions are implied by BBG type assumptions, so if we notice that datasets do not satisfy these weaker conditions then we can be sure that the original BBG assumptions do not either. Below we define two such conditions, one is implied by the $(1+\alpha, \epsilon)$-approximation property for the large clusters case (Theorem 3.1) and the other is implied by the general case (Theorem 3.2).

Definition 4.1 (weak-$(1+\alpha,\epsilon)$ property for large clusters) *Define ϵ^* to be the error of the optimal k-median solution ($\mathcal{C}^*$) with respect to the target clustering ($\mathcal{C}^{\mathcal{T}}$). For any point x, define $w(x)$ = distance of x from its cluster center in the optimal clustering. Also, define $w_2(x)$ = distance of x from the second closest center. Then we have that at most $(\epsilon - \epsilon^*)n$ of the points on which C and C^* agree, satisfy the property that $w_2(x) - w(x) < \frac{\alpha \mathsf{OPT}}{\epsilon n}$.*

Definition 4.2 (weak-$(1+\alpha,\epsilon)$ property for general case) *At most $6\epsilon n$ of the points, satisfy the property that $w_2(x) - w(x) < \frac{\alpha \mathsf{OPT}}{2\epsilon n}$.*

Theorem 4.3 (BBG 09) *If an instance of k-median clustering satisfies the $(1+\alpha,\epsilon)$-approximation property, then Definition 4.2 also hold true for the instance. In addition, if every cluster in the target clustering is of size at least $2\epsilon n$, then Definition 4.1 also holds.*

These weaker properties are easier to test. To verify the weak-$((1+\alpha,\epsilon))$-property for large clusters, we set $\epsilon_i = i/n$, for $i = 1, 2, \ldots, n$, where n is the total number of points. For each value of ϵ_i we compute α_i such that the conditions in Definition 4.1 hold. For each value of (ϵ_i, α_i), we report the minimum cluster size $min_i\, b_i = (5 + 10/\alpha_i)\epsilon_i n + 2$ which is needed for the BBG algorithm to work. Similarly, for the weak-property in the general case, we set $\epsilon_i = 6i/n$, and compute the corresponding α_i. In this case we report the minimum error ($min_i\, 25\epsilon_i + 40\epsilon_i/\alpha_i$) guaranteed by the BBG algorithm.

Here we show the results of testing the ORSS separability and the weak-$((1+\alpha,\epsilon))$ properties for 6 datasets from the UCI repository[1]. Since, we are computing the optimal k-median and $k-1$ median solutions, we restrict ourselves to small datasets. The results are shown in Table 1. Here, the parameter S denotes the minimum cluster size required by the weak $(1+\alpha,\epsilon)$-property for the large clusters case. The parameter E refers to the minimum error bound guaranteed by the algorithm for the general case.

dataset	**n**	**k**	**minimum cluster size**	$\frac{\mathsf{OPT}(k)}{\mathsf{OPT}(k-1)}$	**minimum cluster size required**	**minimum error bound guaranteed**
iris	150	3	50	1.317	306.48	4220.65
wine	178	3	48	1.123	1509.5	1956.05
digits*	537	3	177	1.149	3399.4	2152.19
satellite*	836	3	159	1.344	2675	4568.03
image-segmentation*	1200	4	330	1.17	2619.4	4432.1
letter-recognition*	1129	5	132	1.18	4901.9	5608.88

Table 1: A table showing the extent to which the various properties hold on 6 datasets. The columns can be interpreted as follows: name of the dataset(* refers to the fact that we used a sample of the original dataset), number of points, number of clusters, minimum cluster size in the target clustering, $(1+\alpha)$ factor for ORSS-separability, minimum cluster size needed for the weak property for large cluster case, the minimum error bound guaranteed (in percentage) by the BBG algorithm for the general case.

As can be seen the separation factor of 100 required by ORSS separability is quite a strong condition for our datasets. Also for all the datasets we tested on, the minimum cluster size required to satisfy the weak-$((1+\alpha,\epsilon))$-property for large clusters is much higher than the size of any target cluster. This shows that the $(1+\alpha,\epsilon)$-property for large clusters does not hold for our datasets. In addition we also notice that although the weak property for general case is satisfied, the error guarantee of the BBG algorithm is vacuous. These results suggest that the proposed notions of stability are too strong to be useful in practice.

5 A weaker notion of stability

The results of the previous section suggest that we need to look for weaker notions of stability which are satisfied in practice and at the same time have enough properties such that we can cluster the datasets which satisfy those properties in a better way. In this section we propose such a notion which we call weak-deletion stability.

[1] http://archive.ics.uci.edu/ml/index.html

Definition 5.1 *Let $\{c_1^*, c_2^*, \ldots, c_k^*\}$ denote the centers in the optimal k-median/k-means solution. Let OPT denote the optimal k-median/k-means cost and let OPT^{-i} denote the cost of the clustering obtained by using $(k-1)$ of the optimal centers excluding c_i^*. We say that the k-median/k-means instance satisfies $(1+\alpha)$ weak-deletion stability if for any i, it holds that*

$$\mathsf{OPT}^{-i} > (1+\alpha)\mathsf{OPT}$$

We show that both the stability notions considered in [ORSS06] and in [BBG09] are in fact special cases of weak-deletion stability.

Claim 5.2 *Any $(1+\alpha)$-ORSS separable k-median/k-means instance also satisfies $(1+\alpha)$ weak-deletion stability.*

Claim 5.3 *A k-median/k-means clustering instance that satisfies the $(1+\alpha, \epsilon)$-property, and in which all clusters in the target clustering have size greater than ϵn, also satisfies $(1+\alpha)$ weak-deletion stability.*

For the proof of these claims see [ABS10b]. In particular, the main result in [ABS10b] is that unlike in [ORSS06], small constant values of alpha can still be useful (see Theorem 5.4). For the datasets considered in the previous section, Table 2 shows the extent to which they satisfy weak-deletion stability. These results show that $(1+\alpha)$ weak-deletion stability is in fact satisfied by clustering instances arising in practice, for some constant $\alpha > 0$.

dataset	**n**	**k**	**minimum cluster size**	$(1+\alpha)$ **for weak-deletion stability**
iris	150	3	50	1.3762
wine	178	3	48	1.1405
digits*	537	3	177	1.1679
satellite*	836	3	159	1.4037
image-segmentation*	1200	4	330	1.2121
letter-recognition*	1129	5	132	1.1835

Table 2: A table showing the extent to which the various datasets satisfy $(1+\alpha)$ weak deletion stability.

We show that for instances satisfying weak deletion (for constant α), one can get any constant factor approximation to the k-median and the k-means objectives in polynomial time. Hence, we have the following theorem. For a proof see [ABS10b].

Theorem 5.4 *There exists an algorithm which for any $(1+\alpha)$-weakly stable k-median instance, and for any $\epsilon > 0$, outputs a clustering whose k-median cost is atmost $(1+\epsilon)\mathsf{OPT}$. Furthermore, the running time of the algorithm is $O(n^{\frac{1}{\alpha\epsilon}} k^{O(\frac{1}{\beta})} n^3)$.*

A similar algorithm also exists for weakly stable instances of k-means clustering in the Euclidean space and runs in time $O(n^{\frac{1}{\alpha^2\epsilon^2}} k^{O(\frac{1}{\beta})} n^3)$. Below we briefly describe the main intuition behind the algorithm.

5.1 The algorithm

For a given k-means instance, let $\{C_1^*, C_2^*, \ldots, C_k^*\}$ denote the optimal k-means clustering. A cluster C_i^* is called cheap if its contribution in the optimal k-means solution, OPT_i, is no more than a constant fraction of OPT. Given a cheap cluster C_i^* in the optimal k-means clustering, $(1+\alpha)$ weak-deletion stability assures us that any $x \notin C_i^*$ is far from c_i^*, namely, $d^2(x, c_i^*) > \alpha \frac{\mathsf{OPT}}{4|C_i^*|}$. In contrast, the average (squared)distance of $x \in C_i^*$ from c_i^* is $\frac{\mathsf{OPT}_i}{|C_i^*|}$. Thus, if we focus on a cluster whose contribution, OPT_i, is no more than, say, $\frac{\alpha}{400}\mathsf{OPT}$, we have that c_i^* is 10 times closer, on *average*, to the points of C_i^* than to the points outside C_i^*. Furthermore, using the triangle inequality we have that any two "average" points of C_i^* are of (squared)distance at most $\frac{\alpha}{100}\frac{\mathsf{OPT}}{|C_i^*|}$, while the (squared)distance between any such "average" point and any point outside of C_i^* is at least $\frac{81\alpha}{400}\frac{\mathsf{OPT}}{|C_i^*|}$. So, if we manage to correctly guess the size s of a cheap cluster, we can set a radius $r = \Theta\left(\alpha\frac{\mathsf{OPT}}{4s}\right)$ and collect data-points according to the size and intersection of the r-balls around them. Using this observation, the authors propose an algorithm which which iteratively populates a set Q of components. One can the show that for every cheap cluster C_i^*, the set Q will contain a component which contains

the entire inner-ring of C_i^*. Here, the inner ring consists of points which are at a (squared)distance $\leq \alpha \frac{\mathsf{OPT}}{1024|C_i^*|}$ from the center c_i^*. Furthermore, one can also show that the size of Q will be at most $k + O(1)$. Hence, one can do a brute force search to find out these "good" components corresponding to cheap clusters. In order to handle expensive clusters, we use ideas from [KSS04] to sample points which are good substitutes for the centers of the expensive clusters. This sampling part runs in time exponential in the number of clusters. But, since there are only a constant number of such clusters, the overall running time is still polynomial.

Unfortunately, there are two main bottlenecks in making the above algorithm practical: 1) The sampling step to handle expensive clusters is time consuming and 2) although Q contains $k + O(1)$ components, the constant involved is large, and in practice, searching over all possible k of them might not be feasible. We remove these limitations by proposing a heuristic which makes the algorithm faster. The main observation is that the sampling step is only needed for expensive clusters, or in other words, clusters which heavily dominate the cost of the optimal solution. In a lot of datasets arising in practice this does not happen.

Hence, we skip the sampling step and run the core of the algorithm to get $O(k)$ components. Also, in our experiments we observed that the largest k components in Q seem to give the clustering with the lowest cost as well as the best quality. Hence, we replace the brute force guessing of the k components in Q by just taking the largest k components and using the corresponding k centers to induce a clustering. The algorithm, with a running time of $O(n^3)$ is shown below.

1. **Initialization Stage:** Set $\mathcal{Q} \leftarrow \emptyset$.
2. **Population Stage:** For $s = n, n-1, n-2, \ldots, 1$ do:
   (a) Set $r = \frac{\alpha \mathsf{OPT}}{256s}$.
   (b) Remove any point x such that $d^2(x, \mathcal{Q}) < 4r$. (Here, $d^2(x, \mathcal{Q}) = \min_{T \in \mathcal{Q}; y \in T} d^2(x, y)$.)
   (c) For any remaining data point x, denote the set of data points whose distance from x squared is at most r, by $B(x, r)$. Connect any two remaining point a and b if: (i) $d^2(a, b) \leq r$, (ii) $|B(a, r)| > \frac{s}{2}$ and (iii) $|B(b, r)| > \frac{s}{2}$.
   (d) Let T be a connected component of size $> \frac{s}{2}$. Let $B(T)$ be the set of all data points $\{x;\ d^2(x, y) \leq 4r$ for some $y \in T\}$. Then:
      i. Add T to $\mathcal{Q}$. (That is, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{T\}$.)
      ii. Remove the points of $B(T)$ from the instance.
3. **Centers-Retrieving Stage:** Choose the largest k components, $T_1, T_2, \ldots, T_k$, out of $\mathcal{Q}$.
   (a) Compute $c_i = \sum_{x \in T_i \cup B(T_i)} x$.
   (b) Partition all n points according to the nearest point among the k centers of these k components and output the corresponding clustering.

Unfortunately, the $O(n^3)$ running time of this heuristic still makes it a bad choice for large datasets. In Table 3 below we compare our algorithm with some popular algorithms for k-means clustering. We observe that although our algorithm does perform much better than the traditional Lloyd's algorithms, it is not much better than the popular k-means++ algorithm [AV07]. In addition, the k-means++ algorithm is much faster in practice. Hence, inspite of the nice theoretical properties of our algorithm, due to poor scalability, it is not a good choice to cluster large datasets.

6 Stability assumption of Bilu-Linial

Recently, Bilu and Linial [BL10], focusing on the Max-Cut problem, considered clustering instances where the optimal clustering is optimal not only under the given metric, *but also under any bounded multiplicative perturbation of the given metric*. This is motivated by the fact that in practice, distances between data points are typically just the result of some heuristic measure (e.g., edit-distance between strings or Euclidean distance in some feature space) rather than true "semantic distance" between objects. Thus, unless the optimal solution on the given distances is correct by pure luck, it likely is correct or nearly so on small perturbations of the given distances as well. This can also be viewed as a conceptual analog of a *large margin* assumption with respect to clustering objective Φ. Bilu and Linial [BL10] analyze this type of Max-Cut instances, and show that with stability up to quite large perturbations, of multiplicative factors

dataset	Lloyd's(uniform)	Lloyd's(cluster)	kmeans++	our algorithm
wine	37.08	33.15	8.37	6.9
iris	42	38.7	14.67	14
image	38.57	37.62	25.46	22.38
multiple_features	49.2	38.4	27.7	28.6
digits(0 - 3)	3.16	3.16	3.12	3.16
satellite	43.34	41.02	31.93	31.24

Table 3: A table showing the performance of our algorithm on various datasets. Also shown are the error rates (in percentage) achieved by two variants of the Lloyd's heuristic and the k-means++ algorithm proposed by [AV07]. Lloyd's(uniform) corresponds to choosing k initial seed centers uniformly at random. Lloyd's(cluster) performs an initial clustering an a sample of size 10% and uses the centers obtained as the initial seed centers. The preliminary clustering itself is done by choosing random seed centers.

of roughly $O(n^{1/2})$, one can retrieve the optimal Max-Cut in polynomial time. Of course, this is an extremely strong condition. However, they conjecture that stability up to $O(1)$ perturbations should be enough to solve the problem in polynomial time. Here we show that this conjecture is indeed true for k-median and k-means objectives (in fact, any well-behaved center-based objective function). In addition, we analyze the following question: To what extent does this perturbation resilience property hold in existing datasets? We propose a new heuristic for determining whether any subset of a given clustering instance satisfies this property or not.

6.1 Main Result

Let us formally define what stability under multiplicative perturbation means.

Definition 6.1 *Given a metric (S, d), and $\alpha > 1$, we say a distance function $d' : S \times S \to \mathbb{R}_{\geq 0}$ is a* α-perturbation *of d, if for any $x, y \in S$ it holds that*

$$d(x, y) \leq d'(x, y) \leq \alpha d(x, y)$$

Note that in this definition, much like in the definition of [BL10], we allow d' to be any non-negative function, and not just a metric. In particular, we allow d' to *not* satisfy the triangle inequality. We now give our main definition and main theorem.

Definition 6.2 *Suppose we have a clustering instance composed of n points residing in a metric (S, d) and an objective function Φ which we wish to optimize. We call the clustering instance* α-perturbation resilient *for Φ if for any d', a α-perturbation of d, the optimal clustering of Φ under (S, d) is point-wise identical (up to relabeling of clusters) to the optimal clustering of Φ under (S, d').*

We prove the following theorem. For a proof see [ABS10a].

Theorem 6.3 *For any $\alpha \geq 3$, there exists a $O(n^2k^2)$-time algorithm that finds the optimal k-median / k-means clustering of α-perturbation resilient instances.*

In fact, the algorithm we propose for k-median and k-means, applies also to 3-perturbation resilient clustering instances for any center-based objective Φ which is *separable*.

Definition 6.4 *A clustering objective is separable if it satisfies the following two conditions:*

- *The objective function value of a given clustering is either the (weighted) sum or the maximum of the individual cluster scores.*
- *Given a proposed cluster, its score can be computed in polynomial time.*

The algorithm builds on the results regarding the min-stability property of Balcan et al [BBV08]. Our algorithm's first step is to run the single-linkage algorithm, and, unlike the canonical single-linkage algorithm, that halts once k clusters remain, we halt only once *all* clusters have been merged into a single cluster. Then, past the Single-Linkage algorithm,

the algorithm's second step is to apply dynamic programming to the hierarchical clustering formed. The overall idea resembles, in spirit, the line of work on general metric embeddings in trees, like the works of Bartal [Bar98] and Abraham et al [ABC$^+$05] and the work of Räcke [RÖ8]. Here too, we reduce the problem of retrieving the optimal Φ clustering from a general instance (where it might be infeasible) to a tree-like instance (where it is poly-time solvable). For a complete analysis see [ABS10a].

We would also like to point out that our poly-time algorithm for k-median instances that are 3-perturbation resilient uses only a weaker property, which we call center-proximity. In contrast to proving that any instance satisfying 3-center proximity can be solved in poly-time, we show that for any $\epsilon > 0$, there exist NP-hard instances of (restricted) k-median which satisfy $(3-\epsilon)$-center proximity. Therefore, our approach, which is based on center-proximity alone, is inherently bounded by the factor of 3.

6.2 Perturbation-Resilience on Average

As in the case of approximation based stability assumptions, we wish to study whether perturbation resilience can be useful in clustering real-life instances. Indeed, our result about $(3-\epsilon)$-center proximity shows that theoretically, there is evidence to believe that one cannot correctly cluster in poly-time all the datapoints of a, say, 2.99-perturbation resilient instance. However, α-perturbations of the underlying metric may be useful, in practice, in providing a level of confidence for the clustering of *sub-instances*, even for $\alpha < 3$. Suppose we are clustering a dataset in search of some target clustering (e.g., clustering webpages according to topic), and furthermore, assume (for this section alone) that k is a small enough constant so that we can find the optimal k-clustering efficiently without any assumptions. Imagine our dataset is such that for a given α, it is *not* α-perturbation resilient, yet it contains a subset of points which are clustered together under all α-perturbations of the instance. This gives a strong empirical evidence that this subset of points should reside in a single cluster. Furthermore, this suggests a framework for defining confidence among the classification of pairs of datapoints: the larger the perturbation we need in order to place x and y in two different clusters, the more we are confident x and y belong to the same cluster!

Additionally, it makes sense to consider stability to random perturbations rather than worst-case. Specifically, we introduce the definition of perturbation resilience *on the average*.

Definition 6.5 *Suppose we are given a clustering instance (S, d) and some subset $S' \subset S$, and also given a probability distribution $\mathcal{D}$ over α-perturbations of d. We say S' is* (ϵ, α)-perturbation resilient on average *if for a d' randomly sampled from $\mathcal{D}$, the probability that the optimal clustering under d and the optimal clustering under d' identify over S' is at least $1-\epsilon$. When $S' = S$ we say our clustering instance is (ϵ, α)-perturbation resilient on average.*

It is clear that in order to verify with confidence $1-\delta$ whether a clustering instance is perturbation-resilient on average, it suffices to i.i.d sample $\left(\frac{\ln(1/\delta)}{\epsilon}\right)$ many perturbations from $\mathcal{D}$ and compare the optimum under d to the optimum under each perturbed metric d'. What may seem surprising at first glance, is the fact that in order to find *all subsets* $S' \subset S$ which are (ϵ, α)-perturbation resilient on average, it suffices to i.i.d sample $\left(\frac{2\ln(n/\delta)}{\epsilon}\right)$ perturbations from $\mathcal{D}$. To see this, observe that for any two data points x, y, we can determine with probability at least $1-(\delta/n^2)$ whether x and y are assigned to the sample cluster in the optimum under both d and d' by a set of $\left(\frac{2\ln(n/\delta)}{\epsilon}\right)$ samples from $\mathcal{D}$. Once we determine whether any pair x, y is perturbation resilient, the following procedure finds all subsets of S that are perturbation resilient. Draw a graph over the points in S, and connect any x and y if they belong to the same cluster of the optimum under all sampled perturbations of d. The subsets that are perturbation resilient are exactly the connected components (in fact, cliques) in this graph. [2]

Recall, we assume that k is sufficiently small so that the optimal k-clustering can be found efficiently, so the above argument gives a simple heuristic for measuring the perturbation-resilience of existing datasets. A similar idea involving random perturbations was explored in [BLRR04]. We apply this heuristic to two well-studied datasets, and discuss our experimental results next.

[2] Alternatively, they are the equivalence-classes under the suitably defined equivalence relation.

7 Measuring Perturbation Resilience on Average

Following the logic described in the introduction (Section 6.2), we employ the aforementioned heuristic over existing datasets, where we try various values of α, and for each value we uniformly i.i.d sample α-perturbations of the given metric. The datasets we used are the IRIS and WINE datasets from the UC Irvine repository [AA07]. IRIS contains $n = 150$ data-points and WINE contains of $n = 178$ data-points, and each dataset is composed of $k = 3$ target clusters. Small datasets were intentionally chosen, so that we may find the optimal 3-median on each dataset under 100 random perturbations. Indeed, in our experiments, we set $\epsilon = 1/6$ and $\delta = 0.01$, so that $\left(\frac{\ln(n^2/\delta)}{\epsilon}\right) \leq 100$.

Figure 1 shows how the number of connected components evolve as the value of the perturbation increases, and how large these components are. As expected, we see that as α increases, the number of components we end up with increases, and the fraction of points that reside in the $k = 3$ largest components decreases. These naive measurements suggests that our datasets have a "cut-off" point between $\alpha = 1.1$ and $\alpha = 1.2$. Indeed, in both IRIS and WINE datasets, the change (both in the number of components and the fraction of the instance the belongs to the top three clusters) is the most noticeable. Hence, for $\alpha < 1.2$ a large fraction of the dataset is resilient to α-perturbation. This approach is simple enough to be extended to other datasets, should one wish to measure to what extent they are perturbation-resilient.

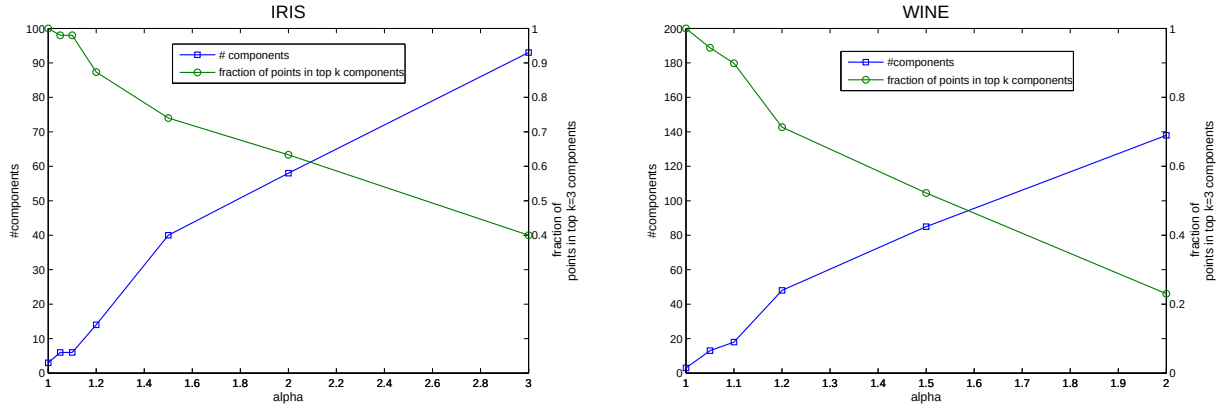

Figure 1: The number of components increases and the fraction of points contained in the 3 largest components decreases, as we consider larger and larger perturbations.

Next, we consider the *purity* of each cluster. Observe that the datasets we use have ground truth, and naturally, our goal is to come up with a clustering that matches the ground truth. However, the 3-median optimum does *not* match the target clustering, but rather contains points from different target clusters. Therefore, in order to measure how well cluster C matches the target clustering, denoted $\{C_1^*, C_2^*, C_3^*\}$, we define the purity of C as

$$\max_i \frac{|C \cap C_i^*|}{|C|}$$

Consider the IRIS dataset for example. The largest cluster in the 3-median optimum of the given metric is 76% pure. Denote the set of data points within this cluster as C. As we start perturbing the metric, we find a component graph in which C is broken into many sub-clusters. For each value of α we consider the largest of these sub-clusters and measure its purity. Not surprisingly, as α increases, the size of the largest sub-cluster decreases. But, as we show in Figure 2, the *purity of the largest cluster* increases as α increases. Indeed, one may expect the purity to increase, as in the extreme case, where α is remarkably big and all components we get contain a single point, they are all 100% pure. However, it is worth noting that even when the size of the largest component is comparable with the size of the target-cluster its purity is almost perfect. (We refer to the WINE dataset under 1.2-perturbations. The largest cluster contains 46 points, out of which 97% are pure.) We also note that the same approach was applied not only the largest

cluster in the 3-median optimum, but also to the other two clusters. However, these clusters' purity was high to begin with (and remained high under larger and larger perturbations).

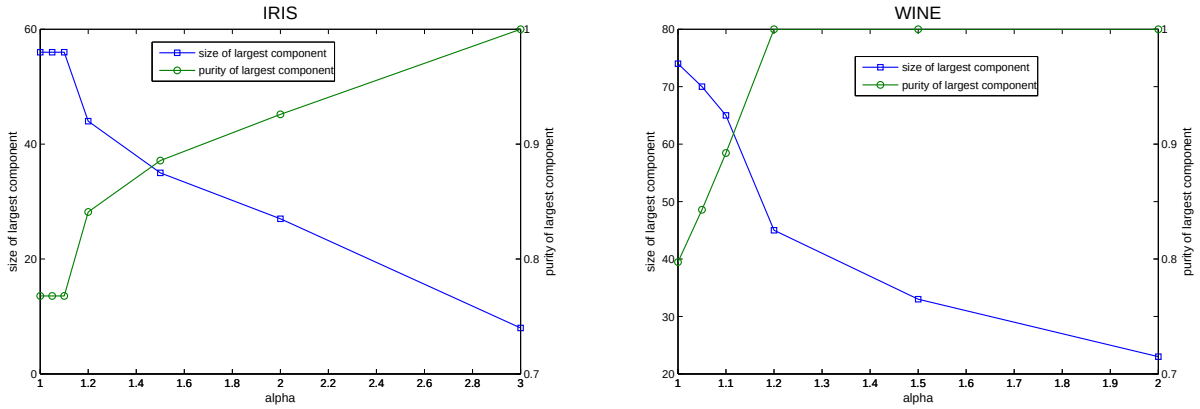

Figure 2: The size of the largest sub-cluster decreases, but its purity increases, as we allow for larger and larger perturbations.

7.1 Empirical Study - Summary

Our empirical study suggests that indeed, assuming that real-life datasets are perfectly resilient to perturbations (i.e. not even a single point gets assigned to a different cluster) is unrealistic. The datasets studied were both "mostly" resilient to very low-level perturbations (past the point of 1.1-perturbations, less than 90% of the instance remained in the same cluster as in the optimal solution). This suggests that real-life datasets are far from being 3-perturbation resilient, and thus suggests a reason for the fact that it is indeed uncommon for the Single-Linkage algorithm to produce a clustering laminar with the target clustering.

However, in contrast to the perturbation resilience of the entire instance, our experiments show that it is not unrealistic to assume that a *large sub-instance* is resilient to perturbations. And, since discovering what subset is resilient to all perturbations may be intractable, we propose our heuristic to discover what subset is resilient to perturbations *on average*. Furthermore, we believe that using this heuristic one is likely to gain additional insights about the underlying structure of the clustering problem. We view this as a subject for future research.

20 newsgroups data set In order to demonstrate qualitatively, how the stable subcomponents get filtered out with increasing α, we look at a subset of the 20 news group corpus which contains binary occurrence data for 100 words across 16242 postings [3]. Here again we face scalability issues since solving for the k-means optimal solution at every step is going to be very time consuming. Instead, we propose to replace the k-means optimal computation step with the k-means++ algorithm. We run our heuristic for $k = 3$-means clustering for each alpha in the range $[1, 1.1, 1.2, 1.5, 2, 3, 3.5, 4]$. Figure 3 shows how the various "pure" subcomponents filter out as α is increased. For example, at $\alpha = 3.5$ we get the cluster $rec.sport$. In the 20 newsgroups dataset, this cluster itself if subdivided into various sub-clusters. Unfortunately, in our experiments we also observed that as we increase α to higher values, these subclusters do not filter out cleanly and we get various small noisy subcomponents.

[3] http://www.cs.nyu.edu/ roweis/data.html

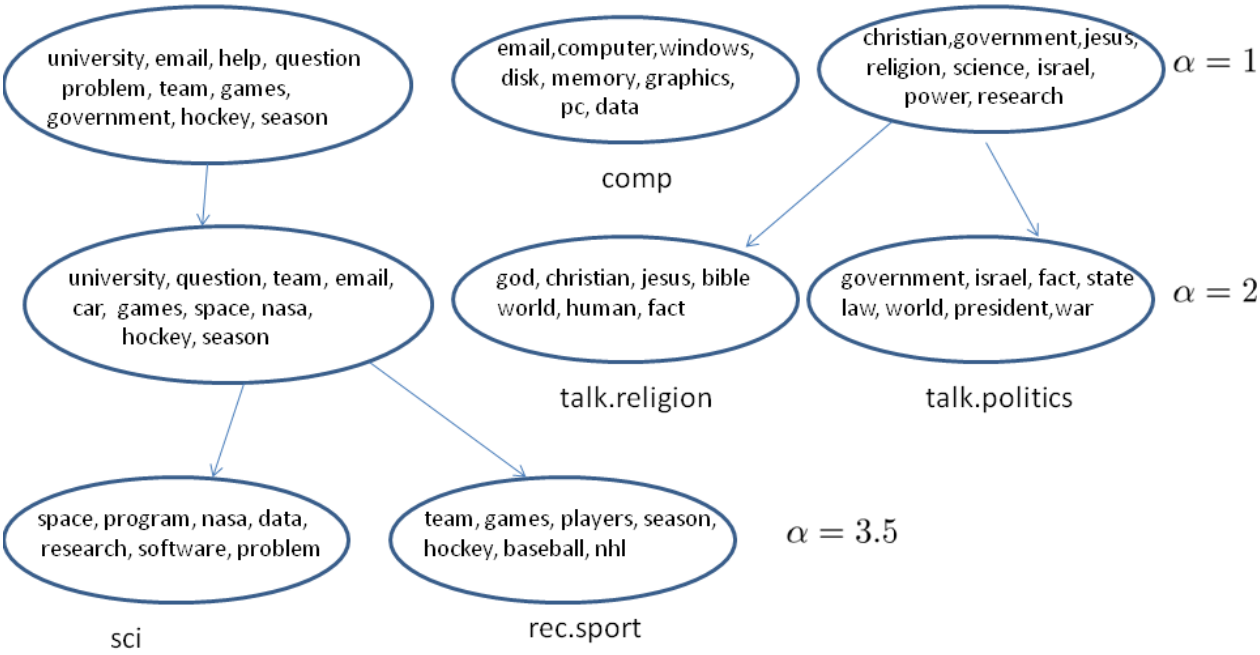

Figure 3: The figure shows how the various subcomponents of the 20newsgroups dataset get filter out as α is increased. For example, at $\alpha = 2$, a noisy top level cluster gets subdivided into subclusters, two of which correspond to the actual clusters $talk.religion$ and $talk.politics$ in the dataset.

8 Transfer clustering

In this section we show that the notion of $(1+\alpha)$ weak deletion stability can be used to approach the problem of transfer clustering. Given a problem for which we know the right answer, the idea is to learn some property of the domain which can then be applied to cluster other related problems. For example consider the digits dataset[4] which has points clustered into 10 different classes $\{0, 1, \ldots, 9\}$. One could take the labeled examples from the first 5 classes and optimize weights for the individual features such that a particular property holds. Then we could use these weights to solve the similar problem of clustering digits belonging to the next 5 classes. There are several properties one could try to optimize for. For example, one candidate property is to learn a set of weights such that the k-means optimal solution is close to the target clustering. Or, may be, one could ask for a set of weights such that the k-means heuristic (Lloyd's algorithm) is close to the target clustering. In spite of being natural candidates, it is not clear at all as to how should one approach towards formulating such a complex optimization problem. This is where the notion of weak-deletion stability is useful. We show that, given a training set, one can set up a linear program to optimize for weights such that the data satisfies $(1+\alpha)$-weak deletion stability. Having done that we know, from Section 5, that there is algorithm which will take this stable input and output a solution of cost at most $(1+\epsilon)\mathsf{OPT}$. If this solution is not close to the target, we can add a constraint into our Linear Program which says that the cost of this solution should $(1+\epsilon)$ be more than the cost of the target. We can then reoptimize for weights and continue until we reach close to the optimal solution.

Given a k-means instance $\{x_1, x_2, \ldots x_n\}$ of n training points in R^d which are clustered into k target clusters $\{C_1, C_2, \ldots, C_k\}$, and a value of α the following linear program can be used to find out the weights that make

[4] http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

the instance $(1+\alpha)$-weakly deletion stable. Below we denote $x(k)$ to denote the k^{th} feature value for point x.

$$\begin{aligned} & Minimize \sum_{i=1}^{d} |w_i| \\ & subject\ \ to: \\ \sum_{x \in C_i} \sum_{k=1}^{d} [w_k(x(k) - c_j(k))^2 - w_k(x(k) - c_i(k))^2] & \leq \alpha \mathsf{OPT}, \ \ \forall i, j \neq i \\ \sum_{i=1}^{k} \sum_{x \in C_i} \sum_{k=1}^{d} [w_k(x(k) - c_i(k))^2] & = \mathsf{OPT} \\ w_i & \geq 0, \ \ \forall i \end{aligned}$$

Given a training set, the weight estimation procedure is as follows:

Input: α, ϵ.

Output: feature weight w_i's.

1. Solve the linear program to get weights such that the target is $(1+\alpha)$-weakly stable. If no solution is found, report failure.
2. Run the algorithm from Section 5 with parameters α, and ϵ.
3. If the solution output by the algorithm is ϵ-close to the target, output w_i's and stop.
4. Otherwise, add a constraint to the LP that the cost of the solution output by the algorithm is more than $(1+\epsilon)$ times the cost of the target.
5. Go back to step 1.

It is clear that if there exists a weighting of the features where a $(1+\epsilon)$-approximation to k-means is also close to the target, then the estimation procedure will find such a weighting. In order to make this estimation procedure practical, we replace the costly algorithm from Section 5 with the fast k-means++ algorithm.

We ran this heuristic against several datasets. As shown in Table 4, we got mixed results. One some of the datasets, after a few iterations, the LP fails to find a feasible solution to the weights. Even replacing the hard constraints in the LP by introducing slack variables does not help in improving the accuracy on the test set. We believe that this is due to the fact that we are trying to find a linear weighting of the features which might not be the right representation. Hence, what is needed is to use a principled way to "kernelize" the approach. However, we do not have a clear idea as to how to achieve this.

dataset	**# clusters in the training set**	**# clusters in the test set**	**test error before optimization**	**test error after optimization**
segmentation	3	4	$41 \pm 1.7\%$	$37 \pm 1.56\%$
abalone	5	5	$67 \pm 1.38\%$	fails to find a solution
letter	5	5	$45.8 \pm 1.42\%$	$49.76 \pm 1.35\%$
20 newsgroups	3	3	$54.8 \pm 2.6\%$	fails to find a solution

Table 4: The table shows the change in the test set accuracy (in percentage) after learning a linear weighting of the features on the training set.

9 Conclusions and Open Problems

This work raises several interesting questions. We showed that for data satisfying weak-deletion stability, one can find very good approximation to the k-median and the k-means objectives. However our algorithm suffers from scalability issues which makes it impractical for even modestly sized datasets. It would be interesting to come up with a version of the algorithm which is fast (comparable to kmeans++ or Lloyd's algorithm) and still retains some of the nice theoretical properties of the original algorithm. In section 6.2 we proposed a heuristic to detect if the dataset contains large subcomponents which are perturbation resilient on average. Here again our heuristic suffers from scalability issues and it would be interesting to come up with a fast method for this problem. An interesting algorithmic question is whether one can efficiently (in polynomial time) detect subcomponents which are perturbation resilient in the worst case, rather than on average. Here one could assume that k is a constant. Even then it is not clear how to solve this problem efficiently. Finally, We think that the idea of transfer clustering is quite interesting and further research can give us good results.

References

[AA07] D.J. Newman A. Asuncion. UCI machine learning repository, 2007.

[ABC+05] Ittai Abraham, Yair Bartal, T-H. Hubert Chan, Kedar Dhamdhere, Anupam Gupta, Jon Kleinberg, Ofer Neiman, and Aleksandrs Slivkins. Metric embeddings with relaxed guarantees. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05. IEEE Computer Society, 2005.

[ABS10a] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *CoRR*, abs/1009.3594, 2010.

[ABS10b] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Stability yields a PTAS for k-median and k-means clustering. In *FOCS*, 2010.

[AGK+01] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *Proc. 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001.

[ARR98] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for euclidean k-medians and related problems. In *Proc. 30th Annual ACM Symposium on Theory of Computing (STOC)*, 1998.

[AV07] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.

[Bar98] Yair Bartal. On approximating arbitrary metrices by tree metrics. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98. ACM, 1998.

[BB09] Maria-Florina Balcan and Mark Braverman. Finding low error clusterings. In *COLT*, 2009.

[BBG09] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proc. 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2009.

[BBV08] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proc. 40th annual ACM Symposium on Theory of Computing (STOC)*, 2008.

[BCR01] Yair Bartal, Moses Charikar, and Danny Raz. Approximating min-sum k-clustering in metric spaces. In *STOC*, 2001.

[BDPS07] Shai Ben-David, Dávid Pál, and Hans-Ulrich Simon. Stability of k-means clustering. In *COLT*, pages 20–34, 2007.

[BDvLP06] Shai Ben-David, Ulrike von Luxburg, and David Pal. A sober look at clustering stability. In *COLT*, volume 4005 of *Lecture Notes in Computer Science*, pages 5–19. Springer, 2006.

[BL10] Yonatan Bilu and Nati Linial. Are stable instances easy? 1st Symposium on Innovations in Computer Science (ICS), 2010.

[BLRR04] Avrim Blum, John Lafferty, Mugizi Robert Rwebangira, and Rajashekar Reddy. Semi-supervised learning using randomized mincuts. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, 2004.

[CGTS99] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proc. 31st Annual ACM Symposium on Theory of Computing (STOC)*, 1999.

[dlVKKR03] W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation schemes for clustering problems. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 2003.

[GK98] Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms. In *Journal of Algorithms*, pages 649–657, 1998.

[JMS02] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems (extended abstract). In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 731–740, 2002.

[KSS04] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time (1+ ϵ) -approximation algorithm for k-means clustering in any dimensions. In *Proc. 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[Mei06] Marina Meilă. The uniqueness of a good optimum for k-means. In *Proc. 23rd International Conference on Machine Learning (ICML)*, pages 625–632, 2006.

[ORSS06] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. In *Proc. 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 165–176, 2006.

[RÖ8] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, STOC '08. ACM, 2008.

[SYvZ10] Frans Schalekamp, Michael Yu, and Anke van Zuylen. Clustering with or without the approximation. In *Proceedings of the 16th annual international conference on Computing and combinatorics*, COCOON'10, 2010.