6

Approximation Stability and Proxy Objectives

Avrim Blum
Toyota Technological Institute at Chicago
avrim@ttic.edu

Abstract

This chapter[1] introduces *approximation stability*, an input condition motivated by the common practice of using the score of a solution under an easy-to-measure objective function as proxy for true solution quality, in problems where the true goal is to find a solution that "looks like" an unknown target solution. An instance is approximation-stable if all solutions with near-optimal values for the proxy objective are close in solution space to the desired target solution, and it turns out that such instances have a number of surprising algorithmic properties. This chapter describes the approximation-stability notion, presents a variety of algorithmic guarantees under this condition, and discusses implications for the use of approximation ratio as a yardstick for problems of solution discovery.

6.1 Introduction and Motivation

Many algorithmic problems, while posed as a task of optimizing a measurable objective function, are motivated by an underlying goal of *approximating a desired (target) solution*. An example would be clustering a dataset of points that represent images of people by optimizing a k-means or k-median objective, when the true goal is to cluster the images based on who is in them.[2] Another example would be searching for a Nash or approximate-Nash equilibrium in a game with the hope that the solution found will approximately predict how people will play. Implicit in formulations such as these is a hope that a solution that optimizes or nearly-optimizes the measurable *proxy* objective (the k-means or k-median score in the case of clustering, or the maximum incentive to deviate in the case of equilibria) will indeed be close to the solution one is hoping to recover.

Approximation-stability formalizes and makes explicit the implicit hoped-for con-

[1] Chapter 6 in "Beyond the Worst-Case Analysis of Algorithms", T. Roughgarden, editor.
[2] For a clustering $C_1, \ldots, C_k$ of a point set S, its k-median score is $\sum_{i=1}^{k} \min_{c_i} \sum_{x \in C_i} d(x, c_i)$. Its k-means score is $\sum_{i=1}^{k} \min_{c_i} \sum_{x \in C_i} d(x, c_i)^2$.

nection above. An instance is approximation-stable if all near-optimal solutions to the proxy objective are indeed close to the desired target solution (see below for a formal definition with parameters). An instance is not approximation-stable if being near-optimal for the proxy objective is *not* a sufficient condition for being close to the target. Any given instance might or might not be approximation-stable. If it is, this motivates use of the proxy objective. If it is not, then it means the motivation for using the proxy objective, at least by itself without additional conditions, is somewhat suspect and perhaps should be re-examined.

The results surveyed in this chapter show the following surprising type of statement for a variety of well-studied objectives: instances satisfying approximation stability at levels that would seem too weak to be helpful, can in fact be solved to high accuracy using structural properties inherent in the stability condition itself. As an example, suppose a clustering instance is stable with respect to the k-median objective in the sense that every clustering whose k-median score is within a factor 1.1 of optimal is ϵ-close to the target solution. For instance, in the case of clustering images by who is in them, this would mean that every 1.1-approximation to the k-median score correctly clusters a $1-\epsilon$ fraction of the images. (In Section 6.2 we will define this as $(1.1, \epsilon)$-approximation stability to the k-median objective). At first glance, this condition seems too weak to be useful since we do not have any efficient algorithms that achieve a 1.1-approximation to the k-median score. The best approximation guarantee known for k-median is roughly a factor of 2.7 (Li and Svensson, 2016), and in fact achieving a 1.1-approximation is known to be NP-hard (Guha and Khuller, 1999; Jain et al., 2002). Nonetheless, as we will see below, we can give a natural, efficient algorithm guaranteed to find a clustering that is $O(\epsilon)$-close to the target in any instance satisfying this condition. Curiously, the k-median problem remains NP-hard to approximate *even on such stable instances*, so the algorithm approximates the solution without necessarily approximating the objective (Balcan et al., 2009b, 2013).[3]

Interesting parameter ranges. We will define an instance to be (c, ϵ) approximation stable for a given objective function if every c-approximation to the objective is ϵ-close to the target solution. Notice that if c is greater than or equal to the best known approximation factor for the objective, then we immediately have an efficient algorithm to find solutions that are ϵ-close to the target for such instances. So, such large values of c are not so interesting. Instead, we will be interested in the case that c is much smaller than the best known approximation factor for the given objective. We will then be asking the question: even though we do not have a c-approximation to the given objective, can we *do as well as if* we had a generic such approximation algorithm, with respect to the goal of finding a solution that is close to the desired target.

[3] If one also makes the assumption that cluster sizes are roughly balanced, then this hardness goes away, and in fact one can give efficient algorithms to approximate k-median to the desired 1.1 factor, and thereby get ϵ-close to the target solution. See Section 6.3.

6.2 Definitions and Discussion

We now formally present the main property studied in this chapter, namely that of (c, ϵ)-approximation stability.

First, consider some optimization problem, such as MAX-SAT or k-median clustering. An optimization problem is defined by an objective function Φ, such as the number of clauses satisfied for the MAX-SAT problem or the k-median cost for the k-median problem. For any given problem instance, there is some space of possible solutions to that instance, and the objective function Φ assigns a score to each one. E.g., for the MAX-SAT problem, an instance would be a formula on n variables, the solution space would be the set $\{0,1\}^n$ of all possible Boolean assignments to the variables, and the objective value for a proposed solution is the number of clauses satisfied. For the k-median problem, an instance would be a set S of n points in some metric space $\mathcal{M} = (X, d)$, the solution space would be the set of all k-clusterings $\{C_1, \ldots, C_k\}$ of S, and the objective value for a proposed solution would be the k-median score $\sum_{i=1}^{k} \min_{c_i \in X} \sum_{x \in C_i} d(x, c_i)$. In addition to the objective score, we are also interested in the distance between solutions in the solution space. So we will assume we are given some natural distance measure $dist(\cdot, \cdot)$ over possible solutions, such as normalized Hamming distance for the case of truth assignments to variables (normalized to the range $[0, 1]$), or the fraction of points that would have to be reassigned in one clustering in order to make it match another clustering, in the case of clustering problems. Lastly, we assume there is some unknown *target* solution we are hoping to get close to, such as a correct clustering of images based on who is in them, or in the case of MAX-SAT that there is some truth assignment that corresponds to "correct" behavior. We then say that an instance satisfies approximation-stability if all near-optimal solutions according to the given objective are close to the target solution according to the given distance measure on solutions. Formally,

Definition 6.1 Consider a problem defined by an objective function Φ, and with distance function $dist$ on its solution space. An instance I satisfies (c, ϵ)-*approximation stability* with respect to an (unknown) target solution y_T if all solutions y of I having $\Phi(I, y)$ within a factor c of the optimal objective value for I satisfy $dist(y, y_T) \leq \epsilon$.

For example, an instance of the k-median problem satisfies (c, ϵ)-approximation stability if all clusterings that have k-median score at most c times that of the optimal k-median clustering agree with the target clustering on at least a $1 - \epsilon$ fraction of points.

It is often helpful to think of approximation-stability in its contrapositive form: any solution that is ϵ-*far* from the target must be expensive, costing more that c times the minimum objective cost. A schematic illustration of approximation-stability is given in Figure 6.1.

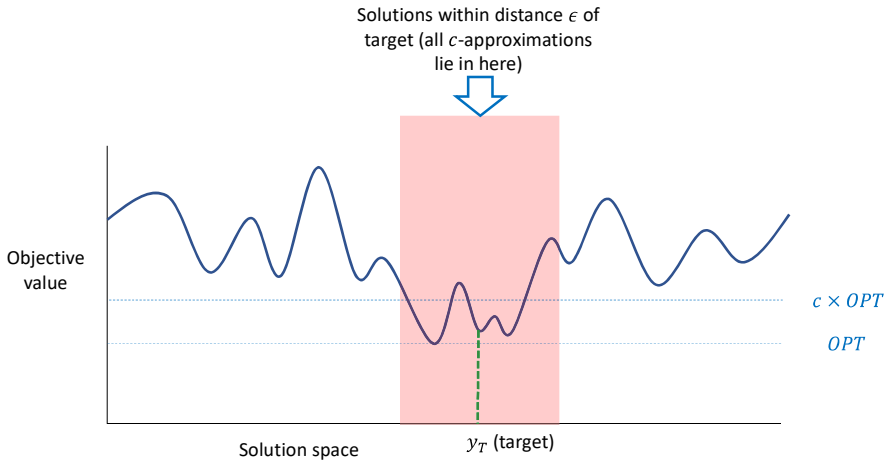

Figure 6.1 A schematic view of approximation stability. Note that the target y_T need not have the optimum objective value, but all near-optimal solutions must be close to y_T.

Removing the target solution. One can also define a nearly identical notion of approximation stability without reference to any target solution, by just asking that all near-optimal solutions be close *to each other*. Specifically, if all near-optimal solutions are within distance ϵ of a target solution then they are all within distance 2ϵ of each other by the triangle inequality, and if all near-optimal solutions are within distance ϵ of each other and the target is also a near-optimal solution, then clearly they are within distance ϵ of the target.

Target versus optimal. Approximation-stability does not require that the target solution be the same as the solution with optimal objective value (see Figure 6.1). For example, for the problem of clustering, we will typically refer to the target clustering as $\mathcal{C}_T$ and the optimal clustering for the objective as $\mathcal{C}^*$. However, it can be helpful to think of the two as equal on first reading.

Determining if an instance is stable. Because approximation-stability refers to distance to an unknown target, there is no way for an algorithm to tell if an instance is indeed approximation-stable. However, if one has an oracle that will report if a solution is "good enough", and if one has an algorithm that finds good solutions on stable instances, then one can just run the algorithm: if the oracle reports success, then one has found a good solution (in which case one probably doesn't care if the instance was actually stable); if it reports failure, then one knows the instance wasn't stable.

Algorithmic structure. For a given stability notion, it is natural to ask what

kinds of algorithms that notion motivates. In the case of clustering, we will see that approximation-stability motivates "ball growing" approaches, where one increases a threshold τ, connecting together all pairs of distance $\leq \tau$, and then forms cluster cores based on dense components in this graph. One can then make a second pass to assign non-core points to clusters based on their distance to the cluster cores from the first pass. In the case of Nash equilibria, approximation-stability does not seem to necessarily motivate new algorithms, but rather leads to improved bounds for existing algorithms that aim to find solutions of small support.

Connection to perturbation stability. Perturbation stability, discussed in Chapter 5, asks that modifying the instance (e.g., changing the distances between datapoints) by up to a factor of c should not change the optimal solution to the given objective (e.g., should not change how points are clustered in the optimal k-median clustering). One can also define a relaxed version of perturbation stability that allows the optimal solution to change by up to an ϵ fraction of points (Balcan and Liang, 2016). This relaxed version has an interesting connection to approximation stability. In particular, for many problems of interest, if one modifies an instance by changing distances by up to a factor of c, then the cost of any given solution changes by at most some function of c (e.g., for k-median clustering, the cost of any given solution would change by at most a factor of c and for k-means clustering, the cost of any given solution would change by at most a factor of c^2). This implies that an optimal solution to a perturbed instance is also a near-optimal solution to the original instance. Thus, the perturbation-stability requirement that optimal solutions to perturbed instances be close to the optimal solution to the original instance is a less stringent version of the approximation-stability requirement that *all* approximately-optimal solutions to the original instance be close to the optimal solution to the original instance (if we associate the optimal solution with the unknown target). On the other hand, while perturbation-stability is a less stringent condition than approximation-stability for the same c, typically one can only achieve positive results for perturbation-stability for factors c that are close to or greater than the best approximation ratios possible, whereas for approximation-stability one aims to get positive results for much smaller parameter values, ideally constants arbitrarily close to 1. So the types of results one can prove about the two stability notions are generally incomparable.

Connection to separability notion of Ostrovsky et al. (2012). The separability notion of Ostrovsky et al. (2012), which is specifically designed for clustering, asks that the optimal objective value for k clusters should be substantially lower (by a sufficiently large constant factor) than the optimal objective value for $k-1$ clusters. E.g., the optimal k-means cost should be substantially less than the optimal $(k-1)$-means cost. Ostrovsky et al. (2012) then show that under this condition, a Lloyd's-style algorithm will succeed in finding a near-optimal solution. They also show that if a clustering instance satisfies this property for a sufficiently large

constant factor, then it also has the property that all near-optimal k-means clusterings are close together, namely approximation-stability. Therefore, algorithms designed for approximation stability (such as given in this chapter) will also succeed under their separability condition. In the other direction, the Ostrovsky et al. (2012) separation condition for a small separation constant is a weaker condition than approximation-stability in the case that all target clusters are large. That is because approximation-stability asks that *all* clusterings that are ϵ-far from the target be more expensive than optimal by at least a factor of c, whereas this condition only asks that clusterings having at most $k-1$ clusters (which are ϵ-far from the target if all target clusters have at least ϵn points) be expensive.

Proxy objectives. An ideal proxy objective would both (a) be something one has reason to believe is optimized by the target and not by any solution far from the target, and (b) be efficiently optimizable. If (b) does not hold but either one has a good approximation algorithm or one has an algorithm under approximation stability, then it would be enough to satisfy a somewhat stronger version of (a) in which solutions far from the target are not even near-optimal. Thus, algorithms that work under approximation stability can help broaden the set of proxy objectives one might reasonably consider for a given problem.

More broadly, a general approach to finding a desired target solution is to identify properties that one believes the target solution should have, and then use them to identify the target or a close approximation to it. In the context of clustering, Balcan et al. (2008) and Daniely et al. (2012) even more broadly consider properties that are not (even in principle) sufficient to uniquely identify the target solution, but do allow for a small set of candidate solutions, that one could then present to a user for further refinement using other criteria. An example of such a property is that most data points x should be closer on average to points in their own target cluster than to points in any other target cluster, by some additive (Balcan et al., 2008) or multiplicative (Daniely et al., 2012) gap γ. Ackerman and Ben-David (2009) consider an intriguing property called "center perturbation clusterability" that is a bit like an inverse of approximation-stability. They consider center-based clusterings (a clustering is defined by k centers, with each datapoint assigned to its nearest center) and ask that all clusterings whose centers are close to the optimal centers should have a cost within a small factor of the optimal cost. One could also hope to *learn* relevant properties from past data, using techniques such as in Chapter 29.

6.3 The k-Median Problem

The k-median problem is a particularly clean objective for studying approximation-stability, and many of the ideas used for it can be extended to other clustering formulations such as k-means and min-sum clustering. So, we focus on it here.

We now show how one can design an efficient clustering algorithm with the guarantee that if an instance is $(1.1, \epsilon)$-approximation stable for the k-median objective, it will find a solution that is $O(\epsilon)$-close to the target, or even ϵ-close to the target if all target clusters are large compared to ϵn. That is, it performs nearly as well (in terms of distance to the target) as would be guaranteed by a generic 1.1-factor approximation to the k-median objective, even though approximating k-median to such a level is NP-hard. More generally, if the instance is $(1+\alpha, \epsilon)$-stable then the algorithm will find a solution that is $O(\epsilon/\alpha)$-close to the target, or even ϵ-close to the target if all target clusters are large compared to $\epsilon n/\alpha$. Note that $1/\epsilon$, $1/\alpha$, and k need not be constants (and in fact, one should not think of k as a constant since we do not want to view an algorithm that "tries all possible k-tuples of centers" as efficient). For example, we might have $\mathcal{C}_T$ consist of $n^{0.1}$ clusters of size $n^{0.9}$, $\epsilon = 1/n^{0.2}$ and $\alpha = 1/n^{0.09}$ (this would correspond to the case of large target clusters in Theorem 6.2).

We begin with a formal definition of the k-median problem, state the main results, and then give algorithms and proofs.

6.3.1 Definitions

Let $\mathcal{M} = (X, d)$ be a metric space with point set X and distance function d. A *k-clustering* of a point set $S \subseteq X$ is a partition $\mathcal{C}$ of S into k clusters $\{C_1, \ldots, C_k\}$. The *k-median cost* of a clustering is the total distance of all points to the best "center" of their clustering. That is,

$$\Phi_{kmedian}(\mathcal{C}) = \sum_{i=1}^{k} \min_{c_i \in X} \sum_{x \in C_i} d(x, c_i). \tag{6.1}$$

As mentioned earlier, we define the *distance* $dist(\mathcal{C}, \mathcal{C}')$ between two clusterings of the same point set S as the fraction of points that would need to be reassigned in one of the clusterings to make it equal to the other (up to reindexing of the clusters, since the names of the clusters do not matter). Formally, the distance between $\mathcal{C} = \{C_1, \ldots, C_k\}$ and $\mathcal{C}' = \{C'_1, \ldots, C'_k\}$ is:

$$dist(\mathcal{C}, \mathcal{C}') = \min_{\sigma} \frac{1}{n} \sum_{i=1}^{k} |C_i \setminus C'_{\sigma(i)}|, \tag{6.2}$$

where the minimum is taken over all bijections $\sigma : \{1, \ldots, k\} \to \{1, \ldots, k\}$. This distance is a metric on clusterings (see Exercise 6.1).

We say that two clusterings $\mathcal{C}$ and $\mathcal{C}'$ are ϵ-close if $dist(\mathcal{C}, \mathcal{C}') < \epsilon$. Note that if $\mathcal{C}$ and $\mathcal{C}'$ are ϵ-close and all clusters C_i have size at least $2\epsilon n$, then the bijection σ minimizing $\frac{1}{n}\sum_{i=1}^{k} |C_i \setminus C'_{\sigma(i)}|$ has the property that for all i, $|C_i \cap C'_{\sigma(i)}| > \frac{1}{2}|C_i|$. This implies for instance that such σ is unique, and we can say that $\mathcal{C}$ and $\mathcal{C}'$ *agree* on x if $x \in C_i \cap C'_{\sigma(i)}$ for some i, and $\mathcal{C}$ and $\mathcal{C}'$ *disagree* on x otherwise.

6.3.2 Some Interesting Results

We now present some interesting results known about k-median clustering under approximation stability. We will then go into more detail into the algorithm and proof for one of them.

Theorem 6.2 **k-Median, Large Clusters Case (Balcan et al., 2013):** *There is an efficient algorithm that will produce a clustering that is ϵ-close to the target clustering $\mathcal{C}_T$ whenever the instance satisfies $(1+\alpha, \epsilon)$-approximation-stability for the k-median objective and each cluster in $\mathcal{C}_T$ has size at least $(4 + 15/\alpha)\epsilon n + 2$.*

The proof of Theorem 6.2 by Balcan et al. (2013) focuses on the distance of the clustering produced to the target $\mathcal{C}_T$, though Schalekamp et al. (2010) point out that under the assumptions of the theorem, the algorithm additionally achieves a good k-median approximation as well. So, in this case, the k-median approximation problem itself has become easier under approximation stability. However, interestingly, once we allow small clusters, finding an approximation to the objective becomes as hard as in the general case, and yet we can still find a solution that is close to the target clustering.

Theorem 6.3 **k-Median: General Case (Balcan et al., 2013):** *There is an efficient algorithm that will produce a clustering that is $O(\epsilon+\epsilon/\alpha)$-close to the target clustering $\mathcal{C}_T$ whenever the instance satisfies $(1+\alpha, \epsilon)$-approximation-stability for the k-median objective.*

Theorem 6.4 **Hardness of Approximation (Balcan et al., 2013):** *For k-median, k-means, and min-sum objectives, for any $c > 1$, the problem of finding a c-approximation can be reduced in polynomial time to the problem of finding a c-approximation under (c, ϵ)-approximation-stability. Therefore, a polynomial-time algorithm for finding a c-approximation under (c, ϵ)-approximation stability implies a polynomial-time algorithm for finding a c-approximation in general.*

As noted above, α and ϵ may be sub-constant. However, in the case that $1/\alpha = O(1)$, Awasthi et al. (2010b) give an improvement to Theorem 6.2, needing a minimum cluster size of only ϵn to produce a solution that is ϵ-close to the target. Their result also holds under the separability notion of Ostrovsky et al. (2012) at the $1+\alpha$ separation level, and further was used as a building block by Li and Svensson (2016) in the current best k-median approximation.[4] So it is interesting that results

[4] Li and Svensson (2016) give a bi-criteria algorithm that for some constant c_0 finds a k-clustering $\mathcal{C}_k$ whose k-median cost is not too much greater than the cost of the optimal $k - c_0$ clustering $\mathcal{C}^*_{k-c_0}$. To convert this to a true approximation, one then considers two cases. Case (a) is that $\mathcal{C}^*_{k-c_0}$ is not too much more expensive than the optimal k-clustering C^*_k, in which case the solution $\mathcal{C}_k$ found is itself a good approximation to $\mathcal{C}^*_k$. Case (b) is that there is a large gap between the cost of $\mathcal{C}^*_{k-c_0}$ and the cost of $\mathcal{C}^*_k$. But in that case, running the algorithm of Awasthi et al. (2010b) on values $k, k-1, k-2, ..., k-c_0+1$ is guaranteed to produce at least one low-cost k'-clustering for $k' \leq k$. So, running both procedures guarantees a good approximation.

based on non-worst-case stability notions can also have application to worst-case approximation bounds.

6.3.3 Algorithms and Proofs

We now present the algorithm and main ideas for the proof of Theorem 6.2.

First, a small amount of notation. Given a clustering instance specified by a metric space $\mathcal{M} = (X, d)$ and a set of points $S \subseteq X$, fix an optimal k-median clustering $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$, and let c_i^* be the center point (a.k.a. "median") for C_i^*. Note that $\mathcal{C}^*$ may not be exactly the same as the target $\mathcal{C}_T$. For $x \in S$, define

$$w(x) = \min_i d(x, c_i^*)$$

to be the contribution of x to the k-median objective in $\mathcal{C}^*$ (i.e., x's "weight"). Similarly, let $w_2(x)$ be x's distance to its second-closest center point among $\{c_1^*, c_2^*, \ldots, c_k^*\}$. Also, let OPT denote the k-median cost of $\mathcal{C}^*$ and define

$$w_{avg} = \frac{1}{n}\sum_{i=1}^{n} w(x) = \frac{OPT}{n}.$$

That is, w_{avg} is the average weight of the points. Finally, let $\epsilon^* = dist(\mathcal{C}_T, \mathcal{C}^*)$. By the approximation stability assumption, $\epsilon^* \leq \epsilon$. (The reader may wish to think of $\epsilon^* = 0$ and $\mathcal{C}^* = \mathcal{C}_T$ on first read.)

The way that approximation-stability will be used is via the following key lemma, which gives us two important properties of approximation-stable instances.

Lemma 6.5 *If the instance $(\mathcal{M}, S)$ is $(1+\alpha, \epsilon)$-approximation stable for the k-median objective, then*

a. If each cluster in $\mathcal{C}_T$ has size at least $2\epsilon n$, then less than $(\epsilon - \epsilon^*)n$ points $x \in S$ on which $\mathcal{C}_T$ and $\mathcal{C}^*$ agree have $w_2(x) - w(x) < \frac{\alpha w_{avg}}{\epsilon}$.

b. For any $t > 0$, at most $t(\epsilon n/\alpha)$ points $x \in S$ have $w(x) \geq \frac{\alpha w_{avg}}{t\epsilon}$.

Proof To prove Property (a), assume to the contrary. Then one could take $\mathcal{C}^*$ and move $(\epsilon - \epsilon^*)n$ points x on which $\mathcal{C}_T$ and $\mathcal{C}^*$ agree to their second-closest clusters, increasing the objective by at most αOPT. Moreover, this new clustering $\mathcal{C}' = \{C_1', \ldots, C_k'\}$ has distance at least ϵ from $\mathcal{C}_T$, because we begin at distance ϵ^* from $\mathcal{C}_T$ and each move increases this distance by $\frac{1}{n}$ (here we use the fact that because each cluster in $\mathcal{C}_T$ has size at least $2\epsilon n$, the optimal bijection between $\mathcal{C}_T$ and $\mathcal{C}'$ remains the same as the optimal bijection between $\mathcal{C}_T$ and $\mathcal{C}^*$). Hence we have a clustering that is not ϵ-close to $\mathcal{C}_T$ with cost only $(1+\alpha)OPT$, a contradiction.

Property (b) simply follows from the definition of the average weight w_{avg}, and Markov's inequality. □

Note: one can also prove that a slightly weaker version of Property (a) of Lemma 6.5 holds in the case that $\mathcal{C}_T$ may have small clusters. The small clusters case is trickier because reassigning points need not always increase the distance between the clusterings (e.g., think of just swapping all points in two clusters). So, the argument is more involved. See Section 6.3.4.

Let us now use Lemma 6.5 to define the notion of a *critical distance* and of *good* and *bad* points. Specifically,

Definition 6.6 Define the **critical distance** $d_{crit} = \frac{\alpha w_{avg}}{5\epsilon}$; note that this is 1/5 the value in property (a) of Lemma 6.5. Define point $x \in S$ to be **good** if both $w(x) < d_{crit}$ and $w_2(x) - w(x) \geq 5d_{crit}$, else define x to be **bad**. Let $X_i \subseteq C_i^*$ be the good points in the optimal cluster C_i^*, and let $B = S \setminus (\cup X_i)$ be the bad points.

We now show that if an instance is approximation-stable, there cannot be too many bad points:

Proposition 6.7 *If the instance $(\mathcal{M}, S)$ is $(1+\alpha, \epsilon)$-approximation-stable for the k-median objective and each cluster in $\mathcal{C}_T$ has size at least $2\epsilon n$, then $|B| < (1 + 5/\alpha)\epsilon n$.*

Proof By Lemma 6.5(a), the number of points on which $\mathcal{C}^*$ and $\mathcal{C}_T$ agree where $w_2(x) - w(x) < 5d_{crit}$ is at most $(\epsilon - \epsilon^*)n$, and there can be at most $\epsilon^* n$ additional such points where $\mathcal{C}^*$ and $\mathcal{C}_T$ disagree. Setting $t = 5$ in Lemma 6.5(b) bounds the number of points that have $w(x) \geq d_{crit}$ by $(5\epsilon/\alpha)n$. □

Let us now see one way we can use the critical distance and definition of good and bad points to help with clustering. To do this we begin by defining the notion of a *threshold graph*.

Definition 6.8 (Threshold Graph) Define the *τ-threshold graph* $G_\tau = (S, E_\tau)$ to be the graph produced by connecting all pairs $\{x, y\} \in \binom{S}{2}$ with $d(x, y) < \tau$.

Lemma 6.9 (Threshold Graph Lemma) *For a $(1+\alpha, \epsilon)$-approximation-stable instance, the threshold graph G_τ for $\tau = 2d_{crit}$ has the following properties:*

(i) For all x, y in the same X_i, the edge $\{x, y\}$ is in the graph G_τ.

(ii) For $x \in X_i$ and $y \in X_j$ for $j \neq i$, $\{x, y\}$ is not *an edge in G_τ. Moreover, such points x, y do not share any neighbors in G_τ.*

Proof For part (i), since x, y are both good, they are at distance less than d_{crit} to their common cluster center c_i^*, by definition. Hence, by the triangle inequality, the distance $d(x, y)$ satisfies

$$d(x, y) \leq d(x, c_i^*) + d(c_i^*, y) < 2 \times d_{crit} = \tau.$$

For part (ii), note that the distance from any good point x to any other cluster

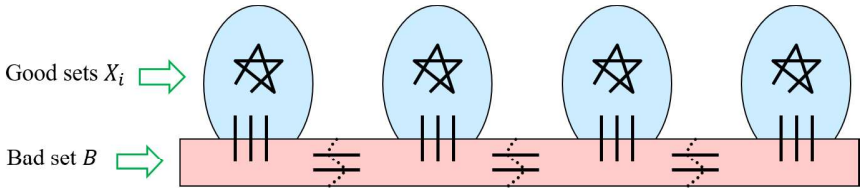

Figure 6.2 The high-level structure of a threshold graph

center, and in particular to y's cluster center c_j^*, is at least $5d_{crit}$. Again by the triangle inequality,

$$d(x,y) \geq d(x,c_j^*) - d(y,c_j^*) \geq 5d_{crit} - d_{crit} = 2\tau.$$

Since each edge in G_τ is between points at distance less than τ, the points x, y cannot share any common neighbors. □

Hence, the graph G_τ for the above value of τ is fairly simple to describe: each X_i forms a clique, and its neighborhood $N_{G_\tau}(X_i) \setminus X_i$ lies entirely in the bad set B with no edges going between X_i and $X_{j\neq i}$, or between X_i and $N_{G_\tau}(X_{j\neq i})$. See Figure 6.2 for an illustration.

We now show how we can use this structure to find a clustering of error at most ϵ. We do this in two steps, beginning with the following lemma.

Lemma 6.10 *There is a deterministic polynomial-time algorithm that given a graph $G = (S, E)$ satisfying properties (i), (ii) of Lemma 6.9 and given an upper bound b on the number of bad points such that each $|X_i| \geq b + 2$, outputs a k-clustering with each X_i contained in a distinct cluster.*

Proof Construct a graph $H = (S, E')$ where we place an edge $\{x, y\} \in E'$ if x and y have at least b common neighbors in G. By property (i), each X_i is a clique of size $\geq b+2$ in G, so each pair $x, y \in X_i$ has at least b common neighbors in G and hence $\{x, y\} \in E'$. Now consider $x \in X_i \cup N_G(X_i)$, and $y \notin X_i \cup N_G(X_i)$: we claim there is no edge between x, y in this new graph H. First, x and y cannot share neighbors that lie in X_i (since $y \notin X_i \cup N_G(X_i)$), nor in some $X_{j\neq i}$ (since $x \notin X_j \cup N_G(X_j)$ by property (ii)). Hence the common neighbors of x, y all lie in B, which has size at most b. Moreover, at least one of x and y must itself belong to B for them to have any common neighbors at all (again by property (ii))—hence, the number of distinct common neighbors is at most $b - 1$, which implies that $\{x, y\} \notin E'$.

Thus each X_i is contained within a distinct component of the graph H. Note that the component containing some X_i may also contain some vertices from B; moreover, there may also be components in H that only contain vertices from B. But since the X_i's are larger than B, we can obtain the claimed clustering by

taking the largest k components in H, and adding the vertices of all other smaller components in H to any of these, using this as the k-clustering. □

We now show how we can use Lemma 6.10 to find a clustering that is ϵ-close to $\mathcal{C}_T$ when all clusters are large. The algorithm will run in two phases: first creating a threshold graph and using the algorithm of Lemma 6.10 to get an initial clustering, and then running a second "Lloyd's-like" step to re-cluster points based on their *median* distances to the initial clusters, which will fix most of the errors from the first step. For simplicity, we begin by assuming that we are given the value of $w_{avg} = \frac{OPT}{n}$, and then we show how this assumption can be removed.

Theorem 6.11 (Large clusters, known w_{avg}) *There is an efficient algorithm such that if the given clustering instance $(\mathcal{M}, S)$ is $(1+\alpha, \epsilon)$-approximation-stable for the k-median objective and each cluster in $\mathcal{C}_T$ has size at least $(3 + 10/\alpha)\epsilon n + 2$, then given w_{avg} it will find a clustering that is ϵ-close to $\mathcal{C}_T$.*

Proof Let us define $b := (1+5/\alpha)\epsilon n$. By assumption, each cluster in the target clustering has at least $(3+10/\alpha)\epsilon n+2 = 2b+\epsilon n+2$ points. Since the *optimal k-median clustering* $\mathcal{C}^*$ differs from the target clustering by at most $\epsilon^* n \leq \epsilon n$ points, each cluster C_i^* in $\mathcal{C}^*$ must have at least $2b+2$ points. Moreover, by Proposition 6.7(i), the bad points B have $|B| \leq b$, and hence for each i,

$$|X_i| = |C_i^* \setminus B| \geq b + 2.$$

Now, given w_{avg}, we can construct the graph G_τ with $\tau = 2d_{crit}$ (which we can compute from the given value of w_{avg}), and apply Lemma 6.10 to find a k-clustering $\mathcal{C}'$ where each X_i is contained within a distinct cluster. Note that this clustering $\mathcal{C}'$ differs from the optimal clustering $\mathcal{C}^*$ only in the bad points, and hence, $dist(\mathcal{C}', \mathcal{C}_T) \leq \epsilon^* + |B|/n \leq O(\epsilon + \epsilon/\alpha)$. However, our goal is to get ϵ-close to the target, which we do as follows.

Call a point x "red" if it is a bad point of the type given in part (a) of Lemma 6.5 (i.e., $w_2(x) - w(x) < 5d_{crit}$), "yellow" if it is not red but is a bad point of the type given in part (b) of Lemma 6.5 with $t = 5$ (i.e., $w(x) \geq d_{crit}$), and "green" otherwise. So, the green points are those in the sets X_i, and we have partitioned the bad set B into red points and yellow points. Let $\mathcal{C}' = \{C'_1, \ldots, C'_k\}$ and recall that $\mathcal{C}'$ agrees with $\mathcal{C}^*$ on the green points, so without loss of generality we may assume $X_i \subseteq C'_i$. We now construct a new clustering $\mathcal{C}''$ that agrees with $\mathcal{C}^*$ on both the green and yellow points. Specifically, for each point x and each cluster C'_j, compute the median distance $d_{median}(x, C'_j)$ between x and all points in C'_j; then insert x into the cluster C''_i for $i = \text{argmin}_j d_{median}(x, C'_j)$. Since each non-red point x satisfies $w_2(x) - w(x) \geq 5d_{crit}$, and all green points g satisfy $w(g) < d_{crit}$, this means that any non-red point x must satisfy the following two conditions: (1) for a green point g_1 in the *same* cluster as x in $\mathcal{C}^*$ we have

$$d(x, g_1) \leq w(x) + d_{crit},$$

and (2) for a green point g_2 in a *different* cluster than x in $\mathcal{C}^*$ we have

$$d(x, g_2) \geq w_2(x) - d_{crit} \geq w(x) + 4d_{crit}.$$

Therefore, $d(x, g_1) < d(x, g_2)$. Since each cluster in $\mathcal{C}'$ has a strict majority of green points (even with point x removed) all of which are clustered as in $\mathcal{C}^*$, this means that for a non-red point x, the median distance to points in its correct cluster with respect to $\mathcal{C}^*$ is less than the median distance to points in any incorrect cluster. Thus, $\mathcal{C}''$ agrees with $\mathcal{C}^*$ on all non-red points. Therefore, every point where $\mathcal{C}''$ and $\mathcal{C}_T$ disagree must be either (i) a point where $\mathcal{C}^*$ and $\mathcal{C}_T$ disagree or (ii) a red point where $\mathcal{C}^*$ and $\mathcal{C}_T$ agree. Since there are at most $\epsilon^* n$ of the former and at most $(\epsilon - \epsilon^*)n$ of the latter by Lemma 6.5, this implies $dist(\mathcal{C}'', \mathcal{C}_T) \leq \epsilon$ as desired. For convenience, the above procedure is given as Algorithm 1 below. □

Algorithm 1 k-median Algorithm: Large Clusters (given a guess w of w_{avg})

Input: w, $\epsilon \leq 1$, $\alpha > 0$, k.

Step 1: Construct the τ-threshold graph G_τ with $\tau = 2d_{crit} = \frac{1}{5}\frac{\alpha w}{\epsilon}$.

Step 2: Apply the algorithm of Lemma 6.10 to find an initial clustering $\mathcal{C}'$. Specifically, construct graph H by connecting x, y if they share at least $b = (1 + 5/\alpha)\epsilon n$ neighbors in G_τ and let $C'_1, \ldots, C'_k$ be the k largest components of H.

Step 3: Produce clustering $\mathcal{C}''$ by reclustering according to smallest median distance in $\mathcal{C}'$. That is, $\mathcal{C}''_i = \{x : i = \text{argmin}_j d_{median}(x, C'_j)\}$.

Step 4: Output the k clusters $\mathcal{C}''_1, \ldots, \mathcal{C}''_k$.

We now extend the above argument to the case where we are not given the value of w_{avg}.

Theorem 6.12 (Large Clusters, unknown w_{avg}) *There is an efficient algorithm such that if the given instance $(\mathcal{M}, S)$ is $(1 + \alpha, \epsilon)$-approximation-stable for the k-median objective, and each cluster in $\mathcal{C}_T$ has size at least $(4 + 15/\alpha)\epsilon n + 2$, the algorithm will produce a clustering that is ϵ-close to $\mathcal{C}_T$.*

Proof The algorithm for the case that we are not given the value w_{avg} is the following: we run Steps 1 and 2 of Algorithm 1 repeatedly for different guesses w of w_{avg}, starting with $w = 0$ (so the graph G_τ is empty) and at each step increasing w to the next value such that G_τ contains at least one new edge (so we have at most n^2 different guesses to try). If the current value of w causes the k largest components of H to miss more than $b := (1 + 5/\alpha)\epsilon n$ points, or if any of these components has size $\leq b$, then we discard this guess w, and try again with the next larger guess for w. Otherwise, we run Algorithm 1 to completion and let $\mathcal{C}''$ be the clustering produced.

Note that we still might have $w < w_{avg}$, but this just implies that the resulting graphs G_τ and H can only have fewer edges than the corresponding graphs for the correct w_{avg}. Hence, some of the X_i's might not have fully formed into connected components in H. However, if the k largest components together miss at most b points, then this implies we must have at least one component for each X_i, and therefore exactly one component for each X_i. So, we never misclassify the good points lying in these largest components. We might misclassify all the bad points (at most b of these), and might fail to cluster at most b of the points in the actual X_i's (i.e., those not lying in the largest k components), but this nonetheless guarantees that each cluster $\mathcal{C}'_i$ contains at least $|X_i| - b \geq b + 2$ correctly clustered green points (with respect to $\mathcal{C}^*$) and at most b misclassified points. Therefore, as shown in the proof of Theorem 6.11, the resulting clustering $\mathcal{C}''$ will correctly cluster all non-red points as in $\mathcal{C}^*$ and so is at distance at most $(\epsilon - \epsilon^*) + \epsilon^* = \epsilon$ from $\mathcal{C}_T$. For convenience, this procedure is given as Algorithm 2 below. □

Algorithm 2 k-median Algorithm: Large Clusters (unknown w_{avg})

Input: $\epsilon \leq 1$, $\alpha > 0$, k.

For $j = 1, 2, 3 \ldots$ do:

Step 1: Let τ be the jth smallest pairwise distance in S. Construct τ-threshold graph G_τ.

Step 2: Run Step 2 of Algorithm 1 to construct graph H and clusters $C'_1, \ldots, C'_k$.

Step 3: If $\min(|C'_1|, \ldots, |C'_k|) > b$ and $|C'_1 \cup \ldots \cup C'_k| \geq n(1 - \epsilon - 5\epsilon/\alpha)$, run Step 3 of Algorithm 1 and output the clusters $C''_1, \ldots, C''_k$ produced.

6.3.4 Handling small clusters

Small target clusters introduce additional challenges. One is that modifying a clustering $\mathcal{C}$ by reassigning ϵn points into different clusters may no longer produce a clustering $\mathcal{C}'$ that is ϵ-far from $\mathcal{C}$. For example, if two clusters C_i and C_j in $\mathcal{C}$ are both small, then moving all points from C_i into C_j and moving all points from C_j into C_i produces the exact same clustering as at the start. However, it turns out that any set of ϵn reassignments must contain a subset of size at least $\epsilon' n$ for $\epsilon' \geq \epsilon/3$ that indeed create a clustering $\mathcal{C}'$ that is ϵ'-far from the original $\mathcal{C}$ (Balcan et al., 2013). This allows for a slightly weaker analog of Lemma 6.5 to be shown. Another challenge is that in growing the threshold τ, it can be difficult to tell when to stop. In particular, if we grow the threshold until the kth largest cluster produced has more than b points, we may have gone too far — merging two large clusters and producing a high-error solution. However, this can be addressed by first running any constant-factor k-median approximation to get an estimate $\tilde{w}_{avg}$ for w_{avg}, and then using that quantity inside the algorithm. Finally, there may be some clusters

that are dominated by bad points. Nonetheless, this can be handled as well, though we can no longer run the re-clustering phase (Step 3) of algorithm 1, resulting in a solution that is $O(\epsilon + \epsilon/\alpha)$-close to the target rather than ϵ-close. The formal guarantee is in Theorem 6.3.

6.4 k-Means, Min-Sum, and Other Clustering Objectives

Similar results to those presented above for the k-median problem are also known for the k-means and min-sum clustering objectives. The k-means score of a clustering is defined similarly to the k-median score, except we square the distances:

$$\Phi_{kmeans}(\mathcal{C}) = \sum_{i=1}^{k} \min_{c_i \in X} \sum_{x \in C_i} d(x, c_i)^2. \tag{6.3}$$

In min-sum clustering, the objective value is the sum of all pairwise intra-cluster distances.

$$\Phi_{minsum}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{x \in C_i} \sum_{y \in C_i} d(x, y). \tag{6.4}$$

E.g., in a uniform metric space, all clusterings have the same k-median or k-means cost, but the min-sum objective would be optimized by making all clusters equal in size.

For the k-means problem, there is an analogous result to Theorem 6.3:

Theorem 6.13 **k-Means: General Case (Balcan et al., 2013):** *There is an efficient algorithm that will produce a clustering that is $O(\epsilon+\epsilon/\alpha)$-close to the target clustering $\mathcal{C}_T$ whenever the instance satisfies $(1+\alpha, \epsilon)$-approximation-stability for the k-means objective.*

The min-sum objective is more challenging to analyze because the contribution of any given datapoint to the overall cost depends on the size of the cluster it is in. In fact, unlike the k-median and k-means problems that have constant-factor approximation algorithms, the best approximation guarantee known for the min-sum objective is an $O(\log^{1+\delta}(n))$ factor (Bartal et al., 2001).

Balcan et al. (2013) give a bound for min-sum clustering of the form in Theorem 6.13 above but only under the assumption that all target clusters have size at least $c\epsilon n/\alpha$ for a sufficiently large constant c. Balcan and Braverman (2009) extend this to general cluster sizes, so long as one is given up-front a constant-factor approximation to the objective; else their algorithm produces a list of $O(\log \log n)$ solutions such that at least one solution will be $O(\epsilon + \epsilon/\alpha)$-close to the target clustering.

6.5 Clustering Applications

Voevodski et al. (2012) consider clustering applications in computational biology, and show that approximation-stability can be a useful guide in designing algorithms for them, especially when those settings come with additional constraints. Specifically, in the application considered by Voevodski et al. (2012), one is not given the distances between all datapoints up front. Instead, one can make a limited number of one-versus-all queries: proposing a query point and running a procedure that returns its distance to all other points in the dataset. They design an algorithm that, assuming (c, ϵ)-approximation-stability for the k-median objective, finds a clustering that is ϵ-close to the target by using only $O(k)$ such one-versus-all queries in the large cluster case, and furthermore is faster than the algorithm we presented here. They then use their algorithm to cluster biological datasets in the Pfam (Finn et al., 2010) and SCOP (Murzin et al., 1995) databases, where the points are proteins and distances are inversely proportional to their sequence similarity. The Pfam and SCOP databases are used in biology to observe evolutionary relationships between proteins and to find close relatives of particular proteins. Voevodski et al. (2012) show that their algorithms are not only fast on these datasets, but also achieve high accuracy. In particular, for one of these sources they obtain clusterings that almost exactly match the given classification, and for the other, the accuracy of their algorithm is comparable to that of the best known (but slower) algorithms using the full distance matrix.

6.6 Nash Equilibria

We now consider the problem of finding approximate Nash equilibria from the perspective of approximation stability.

Let (R, C) denote a 2-player, n-action bimatrix game. Here, R is the payoff matrix for the row player and C is the payoff matrix for the column player. A (mixed) strategy is a probability distribution over n actions, which we will represent as a column vector. Let Δ_n denote the strategy space, that is, the set of vectors in $[0, 1]^n$ whose entries sum to 1. The goal of each player is to maximize its expected payoff. A pair of strategies (p, q) (p for the row player and q for the column player) is a *Nash equilibrium* if neither player has any incentive to deviate; that is,

- For all $p' \in \Delta_n$, $p'^T Rq \leq p^T Rq$.
- For all $q' \in \Delta_n$, $p^T Cq' \leq p^T Cq$.

A pair of strategies (p, q) is an *approximate* Nash equilibrium if no player has a *large* incentive to deviate. More formally, assume the matrices R, C have all entries in the range $[0, 1]$. We then say that (p, q) is an α-*approximate* equilibrium if

- For all $p' \in \Delta_n$, $p'^T Rq \leq p^T Rq + \alpha$.

- For all $q' \in \Delta_n$, $p^T C q' \leq p^T C q + \alpha$.

We say that (p, q) is a *well-supported* α-approximate Nash equilibrium if only actions whose payoffs are within α of the best-response to the opponent's strategy have positive probability. That is, if i is in the support of p, then $e_i^T R q \geq \max_j e_j^T R q - \alpha$, and similarly if i is in the support of q then $p^T C e_i \geq \max_j p^T C e_j - \alpha$, where e_i is the unit vector with 1 in coordinate i.

Finding approximate equilibria in general $n \times n$ bimatrix games appears to be a challenge computationally. Lipton et al. (2003) show that there always exist α-approximate equilibria with support over at most $O((\log n)/\alpha^2)$ actions, which leads to an $n^{O(\log n/\alpha^2)}$-time algorithm for computing α-approximate equilibria. This is the fastest general algorithm known, and Rubinstein (2016) shows that under the Exponential Time Hypothesis for PPAD, there is no algorithm with running time $n^{O(\log^{1-\delta} n)}$ for any constant $\delta > 0$. The associated structural statement is also known to be existentially tight (Feder et al., 2007). The smallest value of α for which an α-approximate equilibrium is known to be computable in polynomial time is is 0.3393 (Tsaknakis and Spirakis, 2007).

However, one reason we might wish to find an approximate Nash equilibrium is to predict how people will play. If we imagine that people will indeed play an approximate Nash equilibrium, but beyond that we wish to make no additional assumptions on player behavior, then for play to be predictable in principle this requires that all approximate equilibria be close together. That is, if we anticipate people will play an α-approximate equilibrium and wish to predict mixed strategies up to, say, a variation distance ϵ, then we will want the game to satisfy (α, ϵ)-approximation stability.[5]

Awasthi et al. (2010a) show that games satisfying (α, ϵ)-approximation-stability indeed have additional useful structural properties. Specifically, if $\epsilon \leq 2\alpha - 6\alpha^2$, then there must exist an $O(\alpha)$-equilibrium where each player's strategy has support size $O(1/\alpha)$. For constant α this implies a polynomial-time algorithm for computing $O(\alpha)$-equilibria. For general α, ϵ such games must have an α-equilibrium of support size $O((\frac{\epsilon^2}{\alpha^2}) \log(1 + \frac{1}{\epsilon}) \log(n))$; this does not lead to a polynomial-time algorithm, but at least gives a substantial reduction in the dependence on α when $\epsilon = O(\alpha)$, for instance. Note also that α and ϵ need not be constants, so this gives a quasi-polynomial time algorithm for finding, say, $1/poly(n)$-approximate equilibria in games that are sufficiently stable at that value of α. This is especially interesting because it is known to be PPAD-hard to find $1/poly(n)$-approximate equilibria in general games. See Balcan and Braverman (2017) for further discussion.

An example. As a simple example of an approximation-stable game, consider

[5] For concreteness, we define the distance between the strategy pair (p, q) and the strategy pair (p', q') as $\max[d(p, p'), d(q, q')]$ where $d(.,.)$ is variation distance.

the prisoner's dilemma (with payoffs scaled to $[0, 1]$):

$$R = \begin{bmatrix} 0.75 & 0 \\ 1 & 0.25 \end{bmatrix} \qquad C = \begin{bmatrix} 0.75 & 1 \\ 0 & 0.25 \end{bmatrix}$$

Here, the only Nash equilibrium is to both play action 2 (defecting), resulting in a payoff of 0.25 to each, even though both playing action 1 (cooperating) would produce payoff 0.75 to each. This game is (α, ϵ)-approximation stable for $\alpha = \epsilon/4$ for any $\epsilon < 1$ because if any player puts an ϵ probability mass on action 1, then (no matter what the other player is doing) that player will have incentive $\epsilon/4$ to deviate. Further examples of natural approximation-stable games are given in Exercises 6.4 and 6.5.

Approximation stability and perturbation stability. Balcan and Braverman (2017) prove an interesting connection between approximation stability and perturbation-stability (discussed in Chapter 5) for the Nash equilibrium problem. Specifically, they show that if (p, q) is a *well-supported* approximate Nash equilibrium in game (R, C), then there must exist a nearby game (R', C') such that (p, q) is an (exact) Nash equilibrium in (R', C'), and vice-versa. This implies that assuming that all well-supported approximate equilibria are close together is the same as assuming that all exact equilibria in slightly perturbed games are close together. In addition, they extend the above general support-size statement to this assumption, as well as to a reversal of quantifiers when the total number of equilibria is polynomial in n (assuming that for each equilibrium in a perturbed game there exists a close equilibrium in the original game, rather than assuming that there exists an equilibrium in the original game that is close to all equilibria in perturbed games).

6.7 The Big Picture

We now step back and reflect on how approximation stability may be useful and what it can tell us. First, approximation stability allows us to formalize what typically are informal motivations for objective functions that can be measured from the data, when the true goal is to find an unknown target solution. If an algorithm can be designed for approximation-stable instances, it means that this algorithm will perform well on any instances for which that motivation is well-justified, possibly even bypassing approximation-hardness barriers with respect to the true goal.

Second, approximation-stability provides an interesting implication through its contrapositive. Suppose an algorithm designed for approximation-stability does *not* perform well on typical instances from a given domain, e.g., as in the domains considered by Schalekamp et al. (2010). This means that those instances are not approximation-stable, which in turn means that if an algorithm is to perform well on them, it will not be solely due to its ability to achieve a good approximation to

the given objective function. Instead, one should aim to look for other criteria or algorithmic properties, perhaps in concert with performance on the objective. That is, approximation-stability can help motivate and provide guidance in the search for additional theoretical guarantees beyond approximation ratio.

Third, approximation-stability can provide a useful design guide for algorithms in practice. As seen in the work of Voevodski et al. (2012) above on clustering protein sequences from limited information, if one is in a new situation and unsure about what kind of algorithm would be best, asking "can we design an algorithm that operates under our given constraints and would do well if the input was sufficiently approximation-stable?" can help in producing highly practical methods, regardless of whether stability is indeed perfectly satisfied in the instances.

6.8 Open Questions

One problem for which an approximation-stability result would be quite interesting is *sparsest cut*. Given a graph $G = (V, E)$, the sparsest cut problem asks to find the cut $(S, V \setminus S)$ that minimizes $\frac{|E(S,V\setminus S)|}{\min(|S|,|V\setminus S|)}$. This problem is NP-hard and the best approximation known is a factor $O(\sqrt{\log n})$ (Arora et al., 2009). One common motivation for sparsest cut is that nodes might represent objects of two types (say, images of cats and images of dogs), edges might represent similarity, and the hope is that the correct partition (cats on one side and dogs on the other) would in fact be a sparse cut. Notice that this is a problem of recovering a target solution. From this perspective, a natural question then is: suppose an instance satisfies (c, ϵ) approximation-stability for sparsest cut, even for a large constant c. Can one use this to efficiently find a solution that is $O(\epsilon)$-close to the target? If so, then this would be achieving the benefits of a constant-factor approximation even if we do not in general know how to achieve a constant-factor approximation.

Another type of problem for which approximation-stability could be quite interesting to consider is phylogenetic tree reconstruction. Here, the goal is to reconstruct an unknown evolutionary tree for a given set of current objects (species, languages, etc.), by optimizing some quantity over the current objects. Typically, the quantity being optimized is motivated by a specific postulated probabilistic model for how mutations occur. However, it would be interesting to obtain guarantees under non-probabilistic stability assumptions as well.

Finally, the MAX-SAT problem could be interesting to consider in this context. The MAX-SAT problem is sometimes used to model solution discovery problems, including problems of learning and clustering (Berg et al., 2015), so approximation-stability results would be of interest here as well.

6.9 Relaxations

Balcan et al. (2009a) consider a relaxation of (c, ϵ)-approximation-stability that allows for the presence of noisy data: data points for which the (heuristic) distance measure does not reflect cluster membership well, which could cause stability over the full dataset to be violated. Specifically, they define and analyze a notion of (ν, c, ϵ)-approximation-stability, which requires that the data satisfies (c, ϵ) approximation-stability only after a ν fraction of the data points have been removed.

It would also be interesting to consider relaxations that place an assumption only on c-approximations satisfying some additional condition that natural approximation algorithms tend to provide. For example, suppose we require only that c-approximations that are also local optima under some reasonable locality notion must be ϵ-close to the target. Can one extend positive results like those above to weaker assumptions of this form as well?

6.10 Bibliographic Notes

The notion of approximation-stability for clustering first appeared in Balcan et al. (2008), though its implications were not studied in detail until Balcan et al. (2009b). The terminology used in this chapter follows Balcan et al. (2013). The approximation-stability results for clustering and Nash equilibria described in this chapter are primarily from Balcan et al. (2013); Awasthi et al. (2010a); Balcan and Braverman (2017). Empirical work described on clustering biological sequences is from Voevodski et al. (2012). Other work on approximation-stability, not discussed here, includes work analyzing k-means++ under approximation-stability (Agarwal et al., 2015) and on correlation-clustering (Balcan and Braverman, 2009).

References

Ackerman, Margareta, and Ben-David, Shai. 2009. Clusterability: A theoretical study. Pages 1–8 of: *Artificial Intelligence and Statistics*.

Agarwal, Manu, Jaiswal, Ragesh, and Pal, Arindam. 2015. k-means++ under Approximation Stability. *Theoretical Computer Science*, **588**, 37–51.

Arora, Sanjeev, Rao, Satish, and Vazirani, Umesh. 2009. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM (JACM)*, **56**(2), 5.

Awasthi, Pranjal, Balcan, Maria-Florina, Blum, Avrim, Sheffet, Or, and Vempala, Santosh. 2010a. On Nash-equilibria of approximation-stable games. Pages 78–89 of: *International Symposium on Algorithmic Game Theory*. Springer.

Awasthi, Pranjal, Blum, Avrim, and Sheffet, Or. 2010b. Stability yields a PTAS for k-median and k-means clustering. Pages 309–318 of: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE.

Balcan, Maria-Florina, and Braverman, Mark. 2009. Finding Low Error Clusterings. In: *Proceedings of the 22nd Annual Conference on Learning Theory.*

Balcan, Maria-Florina, and Braverman, Mark. 2017. Nash equilibria in perturbation-stable games. *Theory of Computing*, **13**(1), 1–31.

Balcan, Maria-Florina, and Liang, Yingyu. 2016. Clustering under perturbation resilience. *SIAM Journal on Computing*, **45**(1), 102–155.

Balcan, Maria-Florina, Blum, Avrim, and Vempala, Santosh. 2008. A Discriminative Framework for Clustering via Similarity Functions. In: *Proceedings of the 40th ACM Symposium on Theory of Computing.*

Balcan, Maria-Florina, Röglin, Heiko, and Teng, Shang-Hua. 2009a. Agnostic clustering. Pages 384–398 of: *International Conference on Algorithmic Learning Theory.* Springer.

Balcan, Maria-Florina, Blum, Avrim, and Gupta, Anupam. 2009b. Approximate clustering without the approximation. Pages 1068–1077 of: *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms.* Society for Industrial and Applied Mathematics.

Balcan, Maria-Florina, Blum, Avrim, and Gupta, Anupam. 2013. Clustering under approximation stability. *Journal of the ACM (JACM)*, **60**(2), 8.

Bartal, Yair, Charikar, Moses, and Raz, Danny. 2001. Approximating min-sum k-clustering in metric spaces. In: *Proceedings on 33rd Annual ACM Symposium on Theory of Computing.*

Berg, Jeremias, Hyttinen, Antti, and Järvisalo, Matti. 2015. Applications of MaxSAT in data analysis. In: *Proceedings of the 6th Pragmatics of SAT Workshop (PoS 2015).*

Daniely, Amit, Linial, Nati, and Saks, Michael. 2012. Clustering is difficult only when it does not matter. *arXiv preprint arXiv:1205.4891.*

Feder, Tomas, Nazerzadeh, Hamid, and Saberi, Amin. 2007. Approximating nash equilibria using small-support strategies. Pages 352–354 of: *Proc. 8th ACM-EC.*

Finn, R.D., Mistry, J., Tate, J., Coggill, P., Heger, A., Pollington, J.E., Gavin, O.L., Gunesekaran, P., Ceric, G., Forslund, K., Holm, L., Sonnhammer, E.L., Eddy, S.R., and Bateman, A. 2010. The Pfam protein families database. *Nucleic Acids Research*, **38**, D211–222.

Guha, Sudipto, and Khuller, Samir. 1999. Greedy strikes back: Improved facility location algorithms. *Journal of algorithms*, **31**(1), 228–248.

Jain, Kamal, Mahdian, Mohammad, and Saberi, Amin. 2002. A new greedy approach for facility location problems. Pages 731–740 of: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing.* ACM.

Li, Shi, and Svensson, Ola. 2016. Approximating k-median via pseudo-approximation. *SIAM Journal on Computing*, **45**(2), 530–547.

Lipton, Richard J., Markakis, Evangelos, and Mehta, Aranyak. 2003. Playing large games using simple strategies. Pages 36–41 of: *Proc. 4th ACM-EC.*

Murzin, A.G., Brenner, S. E., Hubbard, T., and Chothia, C. 1995. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, **247**, 536–540.

Ostrovsky, Rafail, Rabani, Yuval, Schulman, Leonard J, and Swamy, Chaitanya. 2012. The effectiveness of Lloyd-type methods for the k-means problem. *Journal of the ACM (JACM)*, **59**(6), 28.

Rubinstein, Aviad. 2016. Settling the complexity of computing approximate two-player Nash equilibria. Pages 258–265 of: *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE.

Schalekamp, Frans, Yu, Michael, and van Zuylen, Anke. 2010. Clustering with or without the Approxiation. In: *Proceedings of the 16th Annual International Computing and Combinatorics Conference*.

Tsaknakis, Haralampos, and Spirakis, Paul G. 2007. An Optimization Approach for Approximate Nash Equilibria. Pages 42–56 of: *WINE*.

Voevodski, Konstantin, Balcan, Maria-Florina, Röglin, Heiko, Teng, Shang-Hua, and Xia, Yu. 2012. Active clustering of biological sequences. *Journal of Machine Learning Research*, **13**(Jan), 203–225.

Exercises

6.1 Prove that Equation 6.2 defining the distance between k-clusterings is a metric. Specifically, show that (a) $dist(\mathcal{C}, \mathcal{C}')$ is symmetric and (b) it satisfies the triangle inequality. Note: the trickier property here is (a).

6.2 What is the expected distance $dist(\mathcal{C}, \mathcal{C}')$ between two *random* k-clusterings $\mathcal{C}, \mathcal{C}'$ of a set of n points, in the limit as $n \to \infty$?

6.3 Consider k-median clustering for $k = 2$. Give an example of a set of points satisfying $(1.4, 0)$ approximation-stability (i.e., all c-approximations for $c \leq 1.4$ are identical to the target clustering) but not $(1.6, 0.3)$ approximation-stability (i.e., there exists a c-approximation for $c \leq 1.6$ that has distance at least 0.3 from the target clustering). Is your example 1.6-perturbation-resilient (Chapter 5)?

6.4 Consider the *matching pennies* game (with payoffs scaled to $[0, 1]$):

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The unique Nash equilibrium of this game is for both players to play $(0.5, 0.5)$, giving each an expected payoff of 0.5. Prove that this game is $(3/16, 1/4)$-approximation stable. That is, for any strategy pair (p, q) such that at least one of p or q puts more than $3/4$ probability on one of its two actions, at least one player must have at least a $3/16$ incentive to deviate (there must be some action they can play in which their expected gain is larger than their current expected gain by at least $3/16$).

6.5 Consider the game of *rock-paper-scissors* (with payoffs scaled to $[0, 1]$):

$$R = \begin{bmatrix} 0.5 & 0 & 1 \\ 1 & 0.5 & 0 \\ 0 & 1 & 0.5 \end{bmatrix} \quad C = \begin{bmatrix} 0.5 & 1 & 0 \\ 0 & 0.5 & 1 \\ 1 & 0 & 0.5 \end{bmatrix}$$

Prove that this game is $(\alpha, 4\alpha)$-approximation stable for any $\alpha < 1/6$.