Min-Sum Clustering of Protein Sequences with Limited Distance Information

Konstantin Voevodski[1], Maria-Florina Balcan[2], Heiko Röglin[3], Shang-Hua Teng[4], and Yu Xia[5]

[1] Department of Computer Science, Boston University, Boston, MA 02215, USA
[2] College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA
[3] Department of Computer Science, University of Bonn, Bonn, Germany
[4] Computer Science Department, University of Southern California, Los Angeles, CA 90089, USA
[5] Bioinformatics Program and Department of Chemistry, Boston University, Boston, MA 02215, USA

Abstract. We study the problem of efficiently clustering protein sequences in a limited information setting. We assume that we do not know the distances between the sequences in advance, and must query them during the execution of the algorithm. Our goal is to find an accurate clustering using few queries. We model the problem as a point set S with an unknown metric d on S, and assume that we have access to *one versus all* distance queries that given a point $s \in S$ return the distances between s and all other points. Our one versus all query represents an efficient sequence database search program such as BLAST, which compares an input sequence to an entire data set. Given a natural assumption about the approximation stability of the *min-sum* objective function for clustering, we design a provably accurate clustering algorithm that uses few one versus all queries. In our empirical study we show that our method compares favorably to well-established clustering algorithms when we compare computationally derived clusterings to gold-standard manual classifications.

1 Introduction

Biology is an information-driven science, and the size of available data continues to expand at a remarkable rate. The growth of biological sequence databases has been particularly impressive. For example, the size of GenBank, a biological sequence repository, has doubled every 18 months from 1982 to 2007. It has become important to develop computational techniques that can handle such large amounts of data. Clustering is very useful for exploring relationships between protein sequences. However, most clustering algorithms require distances between all pairs of points as input, which is infeasible to obtain for very large protein sequence data sets. Even with a *one versus all* distance query such as BLAST (Basic Local Alignment Search Tool) [AGM$^+$90], which efficiently compares a sequence to an entire database of sequences, it may not be possible to use it n times to construct the entire pairwise distance matrix, where n is the size of the data set. In this work we present a clustering algorithm that gives an accurate clustering using only $O(k \log k)$ queries, where k is the number of clusters.

We analyze the correctness of our algorithm under a natural assumption about the data, namely the (c, ϵ) approximation stability property of [BBG09]. Balcan et al. assume that there is some relevant "target" clustering C_T, and optimizing a particular objective function for clustering (such as min-sum) gives clusterings that are structurally close to C_T. More precisely, they assume that any c-approximation of the objective is ϵ-close to C_T, where the distance between two clusterings is the fraction of misclassified points under the optimum matching between the two sets of clusters. Our contribution is designing an algorithm that given the (c, ϵ)-property for the *min-sum* objective produces an accurate clustering using only $O(k \log k)$ *one versus all* distance queries, and has a runtime of $O(k \log(k) n \log(n))$. We conduct an empirical study that compares computationally derived clusterings to those given by gold-standard classifications of protein evolutionary relatedness. We show that our method compares favorably to well-established clustering algorithms in terms of accuracy. Moreover, our algorithm easily scales to massive data sets that cannot be handled by traditional algorithms.

The algorithm presented here is related to the one presented in [VBR$^+$10]. The *Landmark-Clustering* algorithm presented there gives an accurate clustering if the instance satisfies the (c, ϵ)-property for the k-median objective. However, if the property is satisfied for the *min-sum* objective the structure of the clustering instance is quite different, and the algorithm given in [VBR$^+$10] fails to find an accurate clustering in such cases. Indeed, the analysis presented here is also quite different. The min-sum objective is also considerably harder to approximate. For k-median the best approximation guarantee is $(3 + \epsilon)$ given by [AGK$^+$04]. For the min-sum objective when the number of clusters is arbitrary there is an $O(\delta^{-1} \log^{1+\delta} n)$-approximation algorithm with running time $n^{O(1/\delta)}$ for any $\delta > 0$ due to [BCR01]. In addition, min-sum clustering satisfies the *consistency* property of Kleinberg [Kle03,ZBD09], while k-median does not [Kle03]. The min-sum objective is also more flexible because the optimum clustering is not always a Voronoi decomposition (unlike the optimum k-median clustering).

There are also several other clustering algorithms that are applicable in our limited information setting [AV07,AJM09,MOP01,CS07]. However, because all of these methods seek to approximate an objective function they will not necessarily produce an accurate clustering in our model if the (c, ϵ)-property holds for values of c for which finding a c-approximation is NP-hard. Other than [VBR$^+$10] we are not aware of any results providing both provably accurate algorithms and strong query complexity guarantees in such a model.

2 Preliminaries

Given a metric space $M = (X, d)$ with point set X, an unknown distance function d satisfying the triangle inequality, and a set of points $S \subseteq X$, we would like to find a k-clustering C that partitions the points in S into k sets $C_1, \ldots, C_k$ by using *one versus all* distance queries.

The *min-sum* objective function for clustering is to minimize $\Phi(C) = \sum_{i=1}^{k} \sum_{x,y \in C_i} d(x, y)$. We reduce the min-sum clustering problem to the related *balanced k-median* problem. The balanced k-median objective function seeks to minimize $\Psi(C) = \sum_{i=1}^{k} |C_i| \sum_{x \in C_i} d(x, c_i)$, where c_i is the median of cluster C_i, which is the point $y \in C_i$ that minimizes $\sum_{x \in C_i} d(x, y)$. As pointed out in [BCR01], in metric spaces the two objective functions are related to within a factor of 2: $\Psi(C)/2 \leq \Phi(C) \leq \Psi(C)$. For any objective function Ω we use OPT_Ω to denote its optimum value.

In our analysis we assume that S satisfies the (c, ϵ)-property of [BBG09] for the min-sum and balanced k-median objective functions. To formalize the (c, ϵ)-property we need to define a notion of distance between two k-clusterings $C = \{C_1, \ldots, C_k\}$ and $C' = \{C'_1, \ldots, C'_k\}$. As in [BBG09], we define the distance between C and C' as the fraction of points on which they disagree under the optimal matching of clusters in C to clusters in C':

$$\mathrm{dist}(C, C') = \min_{\sigma \in S_k} \frac{1}{n} \sum_{i=1}^{k} |C_i - C'_{\sigma(i)}|,$$

where S_k is the set of bijections $\sigma \colon \{1, \ldots, k\} \to \{1, \ldots, k\}$. Two clusterings C and C' are said to be *ϵ-close* if $\mathrm{dist}(C, C') < \epsilon$.

We assume that there exists some unknown relevant "target" clustering C_T and given a proposed clustering C we define the error of C with respect to C_T as $\mathrm{dist}(C, C_T)$. Our goal is to find a clustering of low error. The (c, ϵ) approximation stability property is defined as follows.

Definition 1. *We say that the instance (S, d) satisfies the (c, ϵ)-property for objective function Ω with respect to the target clustering C_T if any clustering of S that approximates OPT_Ω within a factor of c is ϵ-close to C_T, that is, $\Omega(C) \leq c \cdot \mathrm{OPT}_\Omega \Rightarrow \mathrm{dist}(C, C_T) < \epsilon$.*

We note that because any $(1 + \alpha)$-approximation of the balanced k-median objective is a $2(1 + \alpha)$-approximation of the min-sum objective, it follows that if the clustering instance satisfies the $(2(1 + \alpha), \epsilon)$-property for the min-sum objective, then it satisfies the $(1 + \alpha, \epsilon)$-property for balanced k-median.

3 Algorithm Overview

In this section we present a clustering algorithm that given the $(1+\alpha, \epsilon)$-property for the balanced k-median objective finds an accurate clustering using few distance queries. Our algorithm is outlined in Algorithm 1 (with some implementation details omitted). We start by uniformly at random choosing n' points that we call *landmarks*, where n' is an appropriate number. For each landmark that we choose we use a *one versus all* query to get the distances between this landmark and all other points. These are the only distances used by our procedure.

Our algorithm then expands a ball B_l around each landmark l one point at a time. In each iteration we check whether some ball B_{l^*} passes the test in line 7. Our test considers the size of the ball and its radius, and checks whether their product is greater than the threshold T. If this is the case, we consider all balls that overlap B_{l^*} on any points, and compute a cluster that contains all the points in these balls. Points and landmarks in the cluster are then removed from further consideration.

Algorithm 1 Landmark-Clustering-Min-Sum(S, d, k, n', T)

```
 1: choose a set of landmarks $L$ of size $n'$ uniformly at random from $S$;
 2: $i = 1$, $r = 0$;
 3: while $i \leq k$ do
 4:   for each $l \in L$ do
 5:     $B_l = \{s \in S \mid d(s, l) \leq r\}$;
 6:   end for
 7:   if $\exists l^* \in L : |B_{l^*}| \cdot r > T$ then
 8:     $L' = \{l \in L : B_l \cap B_{l^*} \neq \emptyset\}$;
 9:     $C_i = \{s \in S : s \in B_l$ and $l \in L'\}$;
10:     remove points in $C_i$ from consideration;
11:     $i = i + 1$;
12:   end if
13:   increment $r$ to the next relevant distance;
14: end while
15: return $C = \{C_1, \ldots C_k\}$;
```

A complete description of this algorithm can be found in the next section. We now present our theoretical guarantee for Algorithm 1.

Theorem 1. *Given a metric space $M = (X, d)$, where d is unknown, and a set of points S, if the instance (S, d) satisfies the $(1 + \alpha, \epsilon)$-property for the balanced-k-median objective function, we are given the optimum objective value* OPT*, and each cluster in the target clustering C_T has size at least $(6+240/\alpha)\epsilon n$, then* Landmark-Clustering-Min-Sum*$(S, d, k, n', \frac{\alpha \mathrm{OPT}}{40\epsilon n})$ outputs a clustering that is $O(\epsilon/\alpha)$-close to C_T with probability at least $1 - \delta$. The algorithm uses $n' = \frac{1}{(3+120/\alpha)\epsilon} \ln \frac{k}{\delta}$* one versus all distance queries*, and has a runtime of $O(n'n \log n)$.*

We note that $n' = O(k \ln \frac{k}{\delta})$ if the sizes of the target clusters are balanced. In addition, if we do not know the value of OPT, we can still find an accurate clustering by running Algorithm 1 from line 2 with increasing estimates of T until enough points are clustered. Theorem 2 states that we need to run the algorithm $n'n^2$ times to find a provably accurate clustering in this setting, but in practice much fewer iterations are sufficient if we use larger increments of T. It is not necessary to recompute the landmarks, so the number of distance queries that are required remains the same. We next give some high-level intuition for how our procedures work.

Given our approximation stability assumption, the target clustering must have the structure shown in Figure 1. Each target cluster C_i has a "core" of well-separated points, where any two points in the cluster core are closer than a certain distance d_i to each other, and any point in a different core is farther

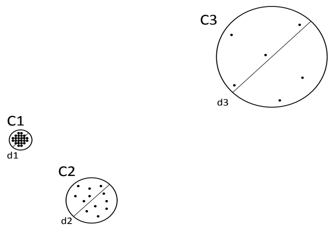

Fig. 1. Cluster cores C_1, C_2 and C_3 are shown with diameters d_1, d_2 and d_3, respectively. The diameters of the cluster cores are inversely proportional to their sizes.

than cd_i, for some constant c. Moreover, the diameters of the cluster cores are inversely proportional to the cluster sizes: there is some constant θ such that $|C_i| \cdot d_i = \theta$ for each cluster C_i. Given this structure, it is possible to classify the points in the cluster cores correctly if we extract the smaller diameter clusters first. In the example in Figure 1, we can extract C_1, followed by C_2 and C_3 if we choose the threshold T correctly and we have selected a landmark from each cluster core. However, if we wait until some ball contains all of C_3, C_1 and C_2 may be merged.

4 Algorithm Analysis

In this section we give a complete description of our algorithm and present its formal analysis. We describe the structure of the clustering instance that is implied by our approximation stability assumption, and give the proof of Theorem 1. We also state and prove Theorem 2, which concerns what happens when we do not know the optimum objective value OPT and must estimate one of the parameters of our algorithm.

4.1 Algorithm Description

A detailed description of our algorithm is given in Algorithm 2. In order to efficiently expand a ball around each landmark, we first sort all landmark-point pairs (l, s) by $d(l, s)$ (not shown). We then consider these pairs in order of increasing distance (line 7), skipping pairs where l or s have already been clustered; the clustered points are maintained in the set $\bar{S}$.

In each iteration we check whether some ball B_{l^*} passes the test in line 19. Our actual test, which is slightly different than the one presented earlier, considers the size of the ball and the *next largest* landmark-point distance (denoted by r_2), and checks whether their product is greater than the threshold T. If this is the case, we consider all balls that overlap B_{l^*} on any points, and compute a cluster that contains all the points in these balls. Points and landmarks in the cluster are then removed from further consideration by adding the clustered points to $\bar{S}$, and removing the clustered points from any ball.

Our procedure terminates once we find k clusters. If we reach the final landmark-point pair, we stop and report the remaining unclustered points as part of the same cluster (line 12). If the algorithm terminates without partitioning all the points, we assign each remaining point to the cluster containing the closest clustered landmark (not shown). In our analysis we show that if the clustering instance satisfies the $(1+\alpha, \epsilon)$-property for the balanced k-median objective function, our procedure will output exactly k clusters.

The most time-consuming part of our algorithm is sorting all landmark-points pairs, which takes $O(|L|n \log n)$, where n is the size of the data set and L is the set of landmarks. With a simple implementation that uses a hashed set to store the points in each ball, the total cost of computing the clusters and removing clustered points from active balls is at most $O(|L|n)$ each. All other operations take asymptotically less time, so the overall runtime of our procedure is $O(|L|n \log n)$.

Algorithm 2 Landmark-Clustering-Min-Sum(S, d, k, n', T)

```
 1: choose a set of landmarks $L$ of size $n'$ uniformly at random from $S$;
 2: for each $l \in L$ do
 3:     $B_l = \emptyset$;
 4: end for
 5: $i = 1$, $\bar{S} = \emptyset$;
 6: while $i \leq k$ do
 7:     $(l, s)$ = GetNextActivePair();
 8:     $r_1 = d(l, s)$;
 9:     if $((l', s')$ = PeekNextActivePair()$)$ $! =$ null then
10:         $r_2 = d(l', s')$;
11:     else
12:         $C_i = S - \bar{S}$;
13:         break;
14:     end if
15:     $B_l = B_l + \{s\}$;
16:     if $r_1 == r_2$ then
17:         continue;
18:     end if
19:     while $\exists l \in L - \bar{S} : |B_l| > T/r_2$ and $i \leq k$ do
20:         $l^* = \text{argmax}_{l \in L - \bar{S}} |B_l|$;
21:         $L' = \{l \in L - \bar{S} : B_l \cap B_{l^*} \neq \emptyset\}$;
22:         $C_i = \{s \in S : s \in B_l \text{ and } l \in L'\}$;
23:         for each $s \in C_i$ do
24:             $\bar{S} = \bar{S} + \{s\}$;
25:             for each $l \in L$ do
26:                 $B_l = B_l - \{s\}$;
27:             end for
28:         end for
29:         $i = i + 1$;
30:     end while
31: end while
32: return $C = \{C_1, \ldots C_k\}$;
```

4.2 Structure of the Clustering Instance

We next describe the structure of the clustering instance that is implied by our approximation stability assumption. We denote by $C^* = \{C_1^*, \dots, C_k^*\}$ the optimal balanced-k-median clustering with objective value OPT=$\Psi(C^*)$. For each cluster C_i^*, let c_i^* be the median point in the cluster. For $x \in C_i^*$, define $w(x) = |C_i^*| d(x, c_i^*)$ and let $w = \text{avg}_x w(x) = \frac{\text{OPT}}{n}$. Define $w_2(x) = \min_{j \neq i} |C_j^*| d(x, c_j^*)$.

It is proved in [BBG09] that if the instance satisfies the $(1+\alpha, \epsilon)$-property for the balanced k-median objective function and each cluster in C^* has size at least $\max(6, 6/\alpha) \cdot \epsilon n$, then at most 2ϵ-fraction of points $x \in S$ have $w_2(x) < \frac{\alpha w}{4\epsilon}$. In addition, by definition of the average weight w at most $120\epsilon/\alpha$-fraction of points $x \in S$ have $w(x) > \frac{\alpha w}{120\epsilon}$.

We call point x *good* if both $w(x) \leq \frac{\alpha w}{120\epsilon}$ and $w_2(x) \geq \frac{\alpha w}{4\epsilon}$, else x is called bad. Let X_i be the *good* points in the optimal cluster C_i^*, and let $B = S \setminus \cup X_i$ be the bad points. Lemma 1, which is similar to Lemma 14 of [BBG09], proves that the optimum balanced k-median clustering must have the following structure:

1. For all x, y in the same X_i, we have $d(x, y) \leq \frac{\alpha w}{60\epsilon |C_i^*|}$.
2. For $x \in X_i$ and $y \in X_{j \neq i}$, $d(x, y) > \frac{\alpha w}{5\epsilon} / \min(|C_i^*|, |C_j^*|)$.
3. The number of bad points is at most $b = (2 + 120/\alpha)\epsilon n$.

4.3 Proof of Theorem 1 and Additional Analysis

We next present the proof of Theorem 1. We give an outline of our arguments, which is followed by the complete proof. We also state and prove Theorem 2.

Proof Outline We first give an outline of our proof of Theorem 1. Our algorithm expands a ball around each landmark, one point at a time, until some ball is large enough. We use r_1 to refer to the current radius of the balls, and r_2 to refer to the next relevant radius (next largest landmark-point distance). To pass the test in line 19, a ball must satisfy $|B_l| > T/r_2$. We choose T such that by the time a ball satisfies the conditional, it must overlap some good set X_i. Moreover, at this time the radius must be large enough for X_i to be entirely contained in some ball; X_i will therefore be part of the cluster computed in line 22. However, the radius is too small for a single ball to overlap different good sets and for two balls overlapping different good sets to share any points. Therefore the computed cluster cannot contain points from any other good set. Points and landmarks in the cluster are then removed from further consideration. The same argument can then be applied again to show that each cluster output by the algorithm entirely contains a single good set. Thus the clustering output by the algorithm agrees with C^* on all the good points, so it must be closer than $b + \epsilon = O(\epsilon/\alpha)$ to C_T.

Complete Proof We next give a detailed proof of Theorem 1.

Proof. Since each cluster in the target clustering has more than $(6 + 240/\alpha)\epsilon n$ points, and the optimal balanced-k-median clustering C^* can differ from the target clustering by fewer than ϵn points, each cluster in C^* must have more than $(5+240/\alpha)\epsilon n$ points. Moreover, by Lemma 1 we may have at most $(2+120/\alpha)\epsilon n$ bad points, and hence each $|X_i| = |C_i^* \setminus B| > (3+120/\alpha)\epsilon n \geq (2+120/\alpha)\epsilon n + 2 = b+2$. We will use $s_{\min}$ to refer to the $(3+120/\alpha)\epsilon n$ quantity.

Our argument assumes that we have chosen at least one landmark from each good set X_i. Lemma 2 argues that after selecting $n' = \frac{n}{s_{\min}} \ln \frac{k}{\delta} = \frac{1}{(3+120/\alpha)\epsilon} \ln \frac{k}{\delta}$ landmarks the probability of this happening is at least $1-\delta$. Moreover, if the target clusters are balanced in size: $\max_{C \in C_T} |C| / \min_{C \in C_T} |C| < c$ for some constant c, because the size of each good set is at least half the size of the corresponding target cluster, it must be the case that $2 s_{\min} c \cdot k \geq n$, so $n/s_{\min} = O(k)$.

Suppose that we order the clusters of C^* such that $|C_1^*| \geq |C_2^*| \geq \ldots |C_k^*|$, and let $n_i = |C_i^*|$. Define $d_i = \frac{\alpha w}{60 \epsilon |C_i^*|}$ and recall that $\max_{x,y \in X_i} d(x,y) \leq d_i$. Note that because there is a landmark in each good set X_i, for radius $r \geq d_i$ there exists some ball containing all of X_i. We use $B_l(r)$ to denote a ball of radius r around landmark l: $B_l(r) : \{s \in S \mid d(s,l) \leq r\}$.

Applying Lemma 3 with all the clusters in C^*, we can see that as long as $r \leq 3d_1$, a ball cannot contain points from more than one good set and balls overlapping different good sets cannot share any points. Also, when $r \leq 3d_1$ and $r < d_i$, a ball $B_l(r)$ containing points from X_i does not satisfy $|B_l(r)| \geq T/r$. To see this, consider that for $r \leq 3d_1$ any ball containing points from X_i has size at most $|C_i^*| + b < \frac{3n_i}{2}$; for $r < d_i$ the size bound $T/r > T/d_i = \frac{\alpha w}{40\epsilon} / \frac{\alpha w}{60 \epsilon |C_i^*|} = \frac{3n_i}{2}$. Finally, when $r = 3d_1$ some ball $B_l(r)$ containing all of X_1 does satisfy $|B_l(r)| \geq T/r$. For $r = 3d_1$ there is some ball containing all of X_1, which must have size at least $|C_1^*| - b \geq n_1/2$. For $r = 3d_1$ the size bound $T/r = n_1/2$, so this ball is large enough to satisfy this conditional. Moreover, for $r \leq 3d_1$ the size bound $T/r \geq n_1/2$. Therefore a ball containing only bad points cannot pass our test for $r \leq 3d_1$ because the number of bad points is at most $b < n_1/2$.

Consider the smallest radius r^* for which some ball $B_{l^*}(r^*)$ satisfies $|B_{l^*}(r^*)| \geq T/r^*$. It must be the case that $r^* \leq 3d_1$, and B_{l^*} overlaps with some good set X_i because we cannot have a ball containing only bad points for $r^* \leq 3d_1$. Moreover, by our previous argument because B_{l^*} contains points from X_i, it must be the case that $r^* \geq d_i$, and therefore some ball contains all the points in X_i. Consider a cluster $\hat{C}$ of all the points in balls that overlap B_{l^*}: $\hat{C} = \{s \in S \mid s \in B_l$ and $B_l \cap B_{l^*} \neq \emptyset\}$, which must include all the points in X_i. In addition, B_{l^*} cannot share any points with balls that overlap other good sets because $r^* \leq 3d_1$, therefore $\hat{C}$ does not contain points from any other good set. Therefore the cluster $\hat{C}$ entirely contains some good set and no points from any other good set.

These facts suggest the following conceptual algorithm for finding a clustering that classifies all the good points correctly: increment r until some ball satisfies $|B_l(r)| \geq T/r$, compute the cluster containing all points in balls that overlap $B_l(r)$, remove these points, and repeat until we find k clusters. We can argue that each cluster output by the algorithm entirely contains some good set and no points from any other good set. Each time we consider the clusters $C \subseteq C^*$ whose good sets have not yet been output, order them by size, and consider the diameters d_i of their good sets. We apply Lemma 3 with C to argue that while $r \leq 3d_1$ the radius is too small for the computed cluster to overlap any of the remaining good sets. As before, we argue that by the time we reach $3d_1$ we must output some cluster. In addition, when $r \leq 3d_1$ we cannot output a cluster of only bad points and whenever we output a cluster overlapping some good set X_i, it must be the case that $r \geq d_i$. Therefore each computed cluster must entirely contain some good set and no points from any other good set. If there are any unclustered points upon the completion of the algorithm, we can assign the remaining points to any cluster. Still, we are able to classify all the good points correctly, so the reported clustering must be closer than $b + \text{dist}(C^*, C_T) < b + \epsilon = O(\epsilon/\alpha)$ to C_T.

It suffices to show that even though our algorithm only considers discrete values of r corresponding to landmark-point distances, the output of our procedure exactly matches the output of the conceptual algorithm described above. Consider the smallest (continuous) radius r^* for which some ball $B_{l_1}(r^*)$ satisfies $|B_{l_1}(r^*)| \geq T/r^*$. We use d_{real} to refer to the largest landmark-point distance that is at most r^*. Clearly, by the time our algorithm reaches $r_1 = d_{real}$ it must be the case that B_{l_1} passes the test on line 19: $|B_{l_1}| > T/r_2$, and this test is not passed by any ball at any prior time. Moreover, B_{l_1} must be the largest ball passing our test at this point because if there is another ball B_{l_2} that also satisfies our test when $r_1 = d_{real}$ it must be the case that $|B_{l_1}| > |B_{l_2}|$ because B_{l_1} satisfies $|B_{l_1}(r)| \geq T/r$ for a smaller r. Finally because there are no landmark-point pairs (l, s) with $r_1 < d(l, s) < r_2$, $B_l(r_1) = B_l(r^*)$ for each landmark $l \in L$. Therefore the cluster that we compute on line 22 for $B_{l_1}(r_1)$ is equivalent to the cluster the conceptual algorithm computes for $B_{l_1}(r^*)$. We can repeat this argument for each cluster output by the conceptual algorithm, showing that Algorithm 2 finds exactly the same clustering.

We note that when there is only one good set left the test in line 19 may not be satisfied anymore if $3d_1 \geq \max_{x,y \in S} d(x, y)$, where d_1 is the diameter of the remaining good set. However, in this case if we exhaust all landmark-points pairs we report the remaining points as part of a single cluster (line 12), which must contain the remaining good set, and possibly some additional bad points that we consider misclassified anyway.

Using a hashed set to keep track of the points in each ball, our procedure can be implemented in time $O(|L| n \log n)$, which is the time necessary to sort all landmark-point pairs by distance. All other operations take asymptotically less time. In particular, over the entire run of the algorithm, the cost of computing the clusters in lines 21-22 is at most $O(n|L|)$, and the cost of removing clustered points from active balls in lines 23-28 is also at most $O(n|L|)$. □

Theorem 2. *If we are not given the optimum objective value* OPT*, then we can still find a clustering that is $O(\epsilon/\alpha)$-close to C_T with probability at least $1 - \delta$ by running Landmark-Clustering-Min-Sum at most $n'n^2$ times with the same set of landmarks, where the number of landmarks $n' = \frac{1}{(3+120/\alpha)\epsilon} \ln \frac{k}{\delta}$ as before.*

Proof. If we are not given the value of OPT then we have to estimate the threshold parameter T for deciding when a cluster develops. Let us use T^* to refer to its correct value ($T^* = \frac{\alpha \mathrm{OPT}}{40\epsilon n}$). We first note that there are at most $n \cdot n|L|$ relevant values of T to try, where L is the set of landmarks. Our test in line 19 checks whether the product of a ball size and a ball radius is larger than T, and there are only n possible ball sizes and $|L|n$ possible values of a ball radius.

Suppose that we choose a set of landmarks L, $|L| = n'$, as before. We then compute all $n'n^2$ relevant values of T and order them in ascending order: $T_i \leq T_{i+1}$ for $1 \leq i < n'n^2$. Then we repeatedly execute Algorithm 2 starting on line 2 with increasing estimates of T. Note that this is equivalent to trying all continuous values of T in ascending order because the execution of the algorithm does not change for any T' such that $T_i \leq T' < T_{i+1}$. In other words, when $T_i \leq T' < T_{i+1}$, the algorithm will give the same exact answer for T_i as it would for T'.

Our procedure stops the first time we cluster at least $n - b$ points, where b is the maximum number of bad points. We give an argument that this gives an accurate clustering with an additional error of b.

As before, we assume that we have selected at least one landmark from each good set, which happens with probability at least $1 - \delta$. Clearly, if we choose the right threshold T^* the algorithm must cluster at least $n - b$ points because the clustering will contain all the good points. Therefore the first time the algorithm clusters at least $n - b$ points for some estimated threshold T, it must be the case that $T \leq T^*$. Lemma 4 argues that if $T \leq T^*$ and the number of clustered points is at least $n - b$, then the reported partition must be a k-clustering that contains a distinct good set in each cluster. This clustering may exclude up to b points, all of which may be good points. Still, if we arbitrarily assign the remaining points we will get a clustering that is closer than $2b + \epsilon = O(\epsilon/\alpha)$ to C_T. □

Lemma 1. *If the balanced k-median instance satisfies the $(1 + \alpha, \epsilon)$-property and each cluster in C^* has size at least $\max(6, 6/\alpha) \cdot \epsilon n$ we have:*

1. *For all x, y in the same X_i, we have $d(x, y) \leq \frac{\alpha w}{60\epsilon|C_i^*|}$.*
2. *For $x \in X_i$ and $y \in X_{j \neq i}$, $d(x, y) > \frac{\alpha w}{5\epsilon}/\min(|C_i^*|, |C_j^*|)$.*
3. *The number of bad points is at most $b = (2 + 120/\alpha)\epsilon n$.*

Proof. For part 1, since $x, y \in X_i \subseteq C_i^*$ are both good, they are at distance of at most $\frac{\alpha w}{120\epsilon|C_i^*|}$ to c_i^*, and hence at distance of at most $\frac{\alpha w}{60\epsilon|C_i^*|}$ to each other.

For part 2 assume without loss of generality that $|C_i^*| \geq |C_j^*|$. Both $x \in C_i^*$ and $y \in C_j^*$ are good; it follows that $d(y, c_j^*) \leq \frac{\alpha w}{120\epsilon|C_j^*|}$, and $d(x, c_j^*) > \frac{\alpha w}{4\epsilon|C_j^*|}$ because $|C_j^*| d(x, c_j^*) \geq w_2(x) > \frac{\alpha w}{4\epsilon}$. By the triangle inequality it follows that

$$d(x, y) \geq d(x, c_j^*) - d(y, c_j^*) \geq \frac{\alpha w}{\epsilon|C_j^*|}(\frac{1}{4} - \frac{1}{120}) > \frac{\alpha w}{5\epsilon}/\min(|C_i^*|, |C_j^*|),$$

where we use that $|C_j^*| = \min(|C_i^*|, |C_j^*|)$.

Part 3 follows from the maximum number of points that may not satisfy each of the properties of the good points and the union bound. □

Lemma 2. *After selecting $\frac{n}{s}\ln\frac{k}{\delta}$ points uniformly at random, where s is the size of the smallest good set, the probability that we did not choose a point from every good set is smaller than $1-\delta$.*

Proof. We denote by s_i the cardinality of X_i. Observe that the probability of not selecting a point from some good set X_i after $\frac{nc}{s}$ samples is $(1-\frac{s_i}{n})^{\frac{nc}{s}} \leq (1-\frac{s_i}{n})^{\frac{nc}{s_i}} \leq (e^{-\frac{s_i}{n}})^{\frac{nc}{s_i}} = e^{-c}$. By the union bound the probability of not selecting a point from every good set after $\frac{nc}{s}$ samples is at most ke^{-c}, which is equal to δ for $c = \ln\frac{k}{\delta}$. □

Lemma 3. *Given a subset of clusters $C \subseteq C^*$, and the set of the corresponding good sets X, let $s_{\max} = \max_{C_i \in C} |C_i|$ be the size of the largest cluster in C, and $d_{\min} = \frac{\alpha w}{60\epsilon s_{\max}}$. Then for $r \leq 3d_{\min}$, a ball cannot overlap a good set $X_i \in X$ and any other good set, and a ball containing points from a good set $X_i \in X$ cannot share any points with a ball containing points from any other good set.*

Proof. By part 2 of Lemma 1, for $x \in X_i$ and $y \in X_{j \neq i}$ we have

$$d(x,y) > \frac{\alpha w}{5\epsilon} / \min(|C_i^*|, |C_j^*|).$$

It follows that for $x \in X_i \in X$ and $y \in X_{j \neq i}$ we must have $d(x,y) > \frac{\alpha w}{5\epsilon}/\min(|C_i^*|,|C_j^*|) \geq \frac{\alpha w}{5\epsilon}/|C_i^*| > \frac{\alpha w}{5\epsilon}/s_{\max} = 12d_{\min}$, where we use the fact that $|C_i| \leq s_{\max}$. So a point in a good set in X and a point in any other good set must be farther than $12d_{\min}$.

To prove the first part, consider a ball B_l of radius $r \leq 3d_{\min}$ around landmark l. In other words, $B_l = \{s \in S \mid d(s,l) \leq r\}$. If B_l overlaps a good set in $X_i \in X$ and any other good set, then it must contain a point $x \in X_i$ and a point $y \in X_{j \neq i}$. It follows that $d(x,y) \leq d(x,l) + d(l,y) \leq 2r \leq 6d_{\min}$, giving a contradiction.

To prove the second part, consider two balls B_{l_1} and B_{l_2} of radius $r \leq 3d_{\min}$ around landmarks l_1 and l_2. Suppose B_{l_1} and B_{l_2} share at least one point: $B_{l_1} \cap B_{l_2} \neq \emptyset$, and use s^* to refer to this point. It follows that the distance between any point $x \in B_{l_1}$ and $y \in B_{l_2}$ satisfies $d(x,y) \leq d(x,s^*) + d(s^*,y) \leq [d(x,l_1) + d(l_1,s^*)] + [d(s^*,l_2) + d(l_2,y)] \leq 4r \leq 12d_{\min}$.

If B_{l_1} overlaps with $X_i \in X$ and B_{l_2} overlaps with $X_{j \neq i}$, and the two balls share at least one point, there must be a pair of points $x \in X_i$ and $y \in X_{j \neq i}$ such that $d(x,y) \leq 12d_{\min}$, giving a contradiction. Therefore if B_{l_1} overlaps with some good set $X_i \in X$ and B_{l_2} overlaps with any other good set, $B_{l_1} \cap B_{l_2} = \emptyset$. □

Lemma 4. *If $T \leq T^* = \frac{\alpha w}{40\epsilon}$ and the number of clustered points is at least $n-b$, then the clustering output by Landmark-Clustering-Min-Sum using the threshold T must be a k-clustering that contains a distinct good set in each cluster.*

Proof. Our argument considers the points that are in each cluster that is output by the algorithm. Let us call a good set *covered* if any of the clusters $C_1, \ldots, C_{i-1}$ found so far contain points from it. We will use $\bar{C}^*$ to refer to the clusters in

C^* whose good sets are not *covered.* It is critical to observe that if $T \leq T^*$ then if C_i contains points from an *uncovered* good set, C_i cannot overlap with any other good set.

To see this, let us order the clusters in $\bar{C}^*$ by decreasing size: $|C_1^*| \geq |C_2^*| \geq \ldots |C_j^*|$, and let $n_i = |C_i^*|$. As before, define $d_i = \frac{\alpha w}{60\epsilon|C_i^*|}$. Applying Lemma 3 with $\bar{C}^*$ we can see that for $r \leq 3d_1$, a ball of radius r cannot overlap a good set in $\bar{C}^*$ and any other good set, and a ball containing points from a good set in $\bar{C}^*$ cannot share any points with a ball containing points from any other good set. Because $T \leq T^*$ we can also argue that by the time we reach $r = 3d_1$ we must output some cluster.

Given this observation, it is clear that the algorithm can cover at most one new good set in each cluster that it outputs. In addition, if a new good set is covered this cluster may not contain points from any other good set. If the algorithm is able to cluster at least $n - b$ points, it must cover every good set because the size of each good set is larger than b. So it must report k clusters where each cluster contains points from a distinct good set. □

5 Experimental Results

We present some preliminary results of testing our *Landmark-Clustering-Min-Sum* algorithm on protein sequence data. Instead of requiring all pairwise similarities between the sequences as input, our algorithm is able to find accurate clusterings by using only a few BLAST calls. For each data set we first build a BLAST database containing all the sequences, and then compare only some of the sequences to the entire database. To compute the distance between two sequences, we invert the bit score corresponding to their alignment, and set the distance to infinity if no significant alignment is found. In practice we find that this distance is almost always a metric, which is consistent with our theoretical assumptions.

In our computational experiments we use data sets created from the Pfam [FMT+10] (version 24.0, October 2009) and SCOP [MBHC95] (version 1.75, June 2009) classification databases. Both of these sources classify proteins by their evolutionary relatedness, therefore we can use their classifications as a ground truth to evaluate the clusterings produced by our algorithm and other methods. These are the same data sets that were used in the [VBR+10] study, therefore we also show the results of the original *Landmark-Clustering* algorithm on these data, and use the same amount of distance information for both algorithms: $30k$ queries for each data set, where k is the number of clusters. In order to run *Landmark-Clustering-Min-Sum* we need to set the parameter T. Because in practice we do not know its correct value, we use increasing estimates of T until we cluster enough of the points in the data set; this procedure is similar to the algorithm for the case when we don't know the optimum objective value OPT and hence don't know T. We set the k parameter using the number of clusters in the ground truth clustering. In order to compare a computationally derived clustering to the one given by the gold-standard classification, we use the distance measure from the theoretical part of our work.

Because our Pfam data sets are so large, we cannot compute the full distance matrix, so we can only compare with methods that use a limited amount of distance information. A natural choice is the following algorithm: uniformly at random choose a set of landmarks L, $|L| = d$; embed each point in a d-dimensional space using distances to L; use k-means clustering in this space (with distances given by the Euclidian norm). This procedure uses exactly d one versus all distance queries, so we can set d equal to the number of queries used by the other algorithms. For SCOP data sets we are able to compute the full distance matrix, so we can compare with a spectral clustering algorithm that has been shown to work very well on these data [PCS06].

From Figure 2 we can see that *Landmark-Clustering-Min-Sum* outperforms k-means in the embedded space on all the Pfam data sets. However, it does not perform better than the original *Landmark-Clustering* algorithm on most of these data sets. When we investigate the structure of the ground truth clusters in these data sets, we see that the diameters of the clusters are roughly the same. When this is the case the original algorithm will find accurate clusterings as well [VBR$^+$10]. Still, *Landmark-Clustering-Min-Sum* tends to give better results when the original algorithm does not work well (data sets 7 and 9).

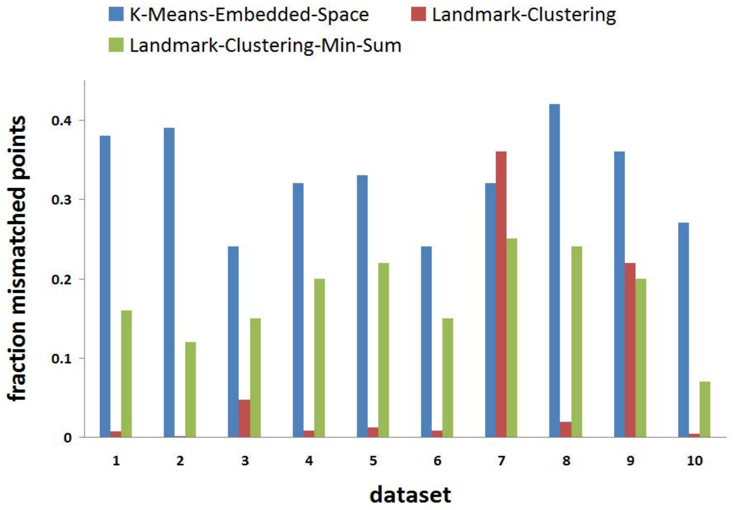

Fig. 2. Comparing the performance of k-means in the embedded space (blue), *Landmark-Clustering* (red), and *Landmark-Clustering-Min-Sum* (green) on 10 data sets from Pfam. Datasets **1-10** are created by uniformly at random choosing 8 families from Pfam of size s, $1000 \leq s \leq 10000$.

Figure 3 shows the results of our computational experiments on the SCOP data sets. We can see that the three algorithms are comparable in performance here. These results are encouraging because the spectral clustering algorithm significantly outperforms other clustering algorithms on these data [PCS06].

Moreover, the spectral algorithm needs the full distance matrix as input and takes much longer to run. When we examine the structure of the SCOP data sets, we find that the diameters of the ground truth clusters vary considerably, which resembles the structure implied by our approximation stability assumption, assuming that the target clusters vary in size. Still, most of the time the product of the cluster sizes and their diameters varies, so it does not quite look like what we assume in the theoretical part of this work.

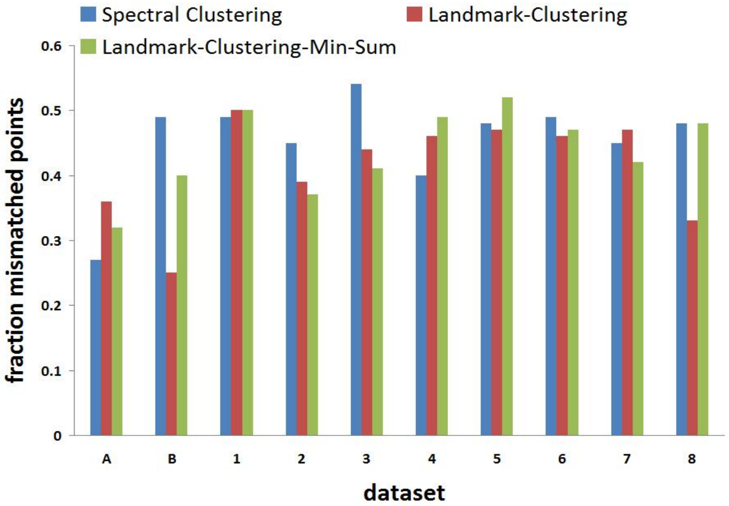

Fig. 3. Comparing the performance of spectral clustering (blue), *Landmark-Clustering* (red), and *Landmark-Clustering-Min-Sum* (green) on 10 data sets from SCOP. Data sets **A** and **B** are the two main examples from [PCS06], the other data sets (**1-8**) are created by uniformly at random choosing 8 superfamilies from SCOP of size s, $20 \leq s \leq 200$.

We plan to conduct further studies to find data where clusters have different scale and there is an inverse relationship between cluster sizes and their diameters. This may be the case for data that have many outliers, and the correct clustering groups sets of outliers together rather than assigns them to arbitrary clusters. The algorithm presented here will consider these sets to be large diameter, small cardinality clusters. More generally, the algorithm presented here is more robust because it will give an answer no matter what the structure of the data is like, whereas the original *Landmark-Clustering* algorithm often fails to find a clustering if there are no well-defined clusters in the data. The *Landmark-Clustering-Min-Sum* algorithm presented here also has fewer hyperparameters and is easier to use in practice when we do not know much about the data.

6 Conclusion

We present a new algorithm that clusters protein sequences in a limited information setting. Instead of requiring all pairwise distances between the sequences as input, we can find an accurate clustering using few BLAST calls. We show that our algorithm produces accurate clusterings when compared to gold-standard classifications, and we expect it to work even better on data who structure more closely resembles our theoretical assumptions.

7 Acknowledgments

This work was supported in part by NSF grant CCF-0953192 and a Microsoft Research Faculty Fellowship.

References

[AGK+04] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3), 2004.

[AGM+90] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–410, 1990.

[AJM09] N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. In *Proc. of 23rd Conference on Neural Information Processing Systems (NIPS)*, 2009.

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

[BBG09] M. F. Balcan, A. Blum, and A. Gupta. Approximate clustering without the approximation. In *Proc. of 20th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2009.

[BCR01] Y. Bartal, M. Charikar, and D. Raz. Approximating min-sum k-clustering in metric spaces. In *Proc. of 33rd ACM Symp. on Theory of Computing (STOC)*, 2001.

[CS07] A. Czumaj and C. Sohler. Sublinear-time approximation algorithms for clustering via random sampling. *Random Struct. Algorithms*, 30(1-2):226–256, 2007.

[FMT+10] R.D. Finn, J. Mistry, J. Tate, P. Coggill, A. Heger, J.E. Pollington, O.L. Gavin, P. Gunesekaran, G. Ceric, K. Forslund, L. Holm, E.L. Sonnhammer, S.R. Eddy, and A. Bateman. The pfam protein families database. *Nucleic Acids Res.*, 38:D211–222, 2010.

[Kle03] J. Kleinberg. An impossibility theorem for clustering. In *Proc. of 17th Conference on Neural Information Processing Systems (NIPS)*, 2003.

[MBHC95] A.G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247:536–540, 1995.

[MOP01] N. Mishra, D. Oblinger, and L Pitt. Sublinear time approximate clustering. In *Proc. of 12th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2001.

[PCS06] A. Paccanaro, J. A. Casbon, and M. A. S. Saqi. Spectral clustering of protein sequences. *Nucleic Acids Res.*, 34(5):1571–1580, 2006.

[VBR+10] K. Voevodski, M. F. Balcan, H. Röglin, S. Teng, and Y. Xia. Efficient clustering with limited distance information. In *Proc. of 26th Conference on Uncertainty in Artifcial Intelligence (UAI)*, 2010.

[ZBD09] R. B. Zadeh and S. Ben-David. A uniqueness theorem for clustering. In *Proc. of 25th Conference on Uncertainty in Artifcial Intelligence (UAI)*, 2009.