Guaranteed Recovery of Unambiguous Clusters

Kayvon Mazooji and Ilan Shomorony
Electrical and Computer Engineering
University of Illinois Urbana-Champaign
Urbana IL, United States
Email: mazooji2@illinois.edu, ilans@illinois.edu

***Abstract*—Clustering is often a challenging problem because of the inherent ambiguity in what the "correct" clustering should be. Even when the number of clusters K is known, this ambiguity often still exists, particularly when there is variation in density among different clusters, and clusters have multiple relatively separated regions of high density. In this paper we propose an information-theoretic characterization of when a K-clustering is ambiguous, and design an algorithm that recovers the clustering whenever it is unambiguous. This characterization formalizes the situation when two high density regions within a cluster are separable enough that they look more like two distinct clusters than two truly distinct clusters in the clustering. The algorithm first identifies K partial clusters (or "seeds") using a density-based approach, and then adds unclustered points to the initial K partial clusters in a greedy manner to form a complete clustering. We implement and test a version of the algorithm that is modified to effectively handle overlapping clusters, and observe that it requires little parameter selection and displays improved performance on many datasets compared to widely used algorithms for non-convex cluster recovery.**

I. Introduction

There is often significant ambiguity in what the "correct" clustering is for a dataset. Even in the case when the number of clusters K is known, this ambiguity often still exists because a cluster can have multiple relatively separated regions of high density, and clusters can have very different densities. These difficulties cause many algorithms to incorrectly separate a true cluster into multiple clusters, while merging true clusters or failing to detect sufficiently prominent true clusters of low density. These issues are compounded by the well-known fact that many clustering algorithms are ineffective at identifying non-convex clusters, which are present in many applications including image segmentation [1], geospatial data [2, 3], and time series data [4].

In this paper, we try to overcome these difficulties from an information theoretic perspective by proposing a necessary condition for a K-clustering C to be unambiguous, along with an algorithm that recovers C whenever this condition holds. This condition characterizes the situation where two high density regions within the same cluster in C look more like two distinct clusters than two truly distinct clusters in C. Our framework yields the provable recovery of unambiguous K-clusterings that can have

- clusters with arbitrarily many relatively separated regions of high density
- arbitrary variation in density among different clusters
- arbitrary variation in density within clusters
- arbitrarily shaped clusters.

Our approach is information theoretic in the sense that our algorithm guarantees recovery of a K-clustering whenever the dataset warrants an unambiguous K-clustering. This is in contrast to algorithms like K-means that are motivated by the optimization of a clustering quality measure.

The clustering algorithm we propose is somewhat complex, but has a computationally efficient runtime. It works by first finding a small subset of points called a "seed" from each cluster, and then expanding the seeds to form clusters. We implement and test a version of the algorithm that is modified to handle overlapping clusters well, and observe that it requires little parameter selection, and delivers improved performance on artificial and benchmark datasets compared to widely used algorithms for non-convex cluster recovery.

We begin by discussing related work and introducing background information and notation. We then present our main theoretical results, followed by experiments comparing our algorithm to existing algorithms.

II. Related Work

The most widely used paradigms for finding non-convex clusters are spectral clustering, and density-based clustering. Spectral clustering algorithms perform dimensionality reduction on the data points, and run a simple clustering algorithm such as K-means to cluster the low dimensional data [5]. Density-based clustering algorithms find regions of high point density, and then output a set of high density clusters according to some criterion [6–9]. While spectral clustering is very well studied from a theoretical standpoint, most widely used density-based algorithms do not come with theoretical guarantees on when a clustering is recoverable.

The first widely-used density-based clustering algorithm was DBSCAN [6]. Since then, many algorithms have been designed to improve upon various aspects of DBSCAN [7–14]. In addition to DBSCAN, the other most widely used density-based clustering algorithms are OPTICS [7] and HDBSCAN [8], which were both designed to improve upon DBSCAN's ability to output clusters of varying density. On the theoretical side, there has been a recent line of work studying λ-density level set estimation using DBSCAN [14–16].

Our algorithm works by first identifying K partial clusters (or "seeds") using a density-based approach, and then adding unclustered points to the initial K partial clusters in a greedy manner to form a complete clustering. To extract

these seeds, we sequentially find seeds of decreasing density. The intuitive idea of sequentially finding disjoint clusters of decreasing density has been employed in various work [11, 12]. However, to the best of our knowledge, our algorithm for finding K seeds and our algorithm for expanding the seeds to form complete clusters have not previously appeared in the literature. Furthermore, we are not aware of any existing mathematical analysis of density-based clustering algorithms that give guarantees similar to those presented in this paper.

III. Preliminaries

We use X to denote a set of data points. The distance between points x and y is denoted by $d(x, y)$. As in many density-based clustering papers, the measure of density at a point $x \in X$ is determined by an integer N_p, and is defined as $1/\epsilon_{N_p}(x)$ where $\epsilon_{N_p}(x)$ is the minimum distance ϵ such that there are N_p points in X at distance at most ϵ from x (including x itself). In other words, $\epsilon_{N_p}(x)$ is the distance from x to its $(N_p - 1)$th nearest neighbor in X. We call $\epsilon_{N_p}(x)$ the sparsity at x.

A cluster is simply a set of points in X. A clustering C of X is a set of disjoint clusters, where every point in X belongs to exactly one cluster (i.e. a partitioning of X into clusters). A K-clustering is a clustering with K clusters. A partial clustering of X is a set of disjoint clusters whose union does not necessarily include all points in X. We say that a clustering C extends (or is an extension of) a partial clustering C' if there exists a bijective function f that maps the clusters in C' to the clusters in C such that for each cluster $c' \in C'$, c' is a non-empty subset of the cluster $f(c') \in C$.

We say a point x_1 is ϵ-connected to a point x_t if there exists some sequence of points $x_1, x_2, ..., x_t$ such that x_{i+1} is distance at most ϵ from x_i for $1 < i < t - 1$ and $\epsilon_{N_p}(x_i) \leq \epsilon$ for $1 < i < t$. A set of points c is called ϵ-connected if every pair of points in c is ϵ-connected.

For a given ϵ, a set of points c is called an ϵ-cluster if it is a maximal ϵ-connected set of points. In other words, any point that is ϵ-connected to a point in an ϵ-cluster c is included in c. A set of points c is called a maximal cluster if it is an ϵ-cluster for some ϵ. For a given ϵ, it is helpful to consider the graph that is formed where each node corresponds to a data point, and an edge is drawn between two nodes if the corresponding points x_1, x_2 are such that $\epsilon_{N_p}(x_1) \leq \epsilon$, $\epsilon_{N_p}(x_2) \leq \epsilon$, and $d(x_1, x_2) \leq \epsilon$. A set of points is an ϵ-cluster if and only if it corresponds to a connected component in this graph.

The ϵ-cluster centered at a point x is defined as the set of points that are ϵ-connected to x, and is denoted by $c^*(x, \epsilon)$. If $\epsilon < \epsilon_{N_p}(x)$, then $c^*(x, \epsilon) = \{\}$. The sparsity of a set of points c is defined as the minimum ϵ such that c is ϵ-connected, and is denoted by $\epsilon^*(c)$. The ϵ-distance between points x and y is defined as the minimum ϵ such that x is ϵ-connected to y, and is denoted by $\epsilon(x, y)$. The minimum ϵ-distance from a point x to a cluster c is defined as $\epsilon(x, c) = \min_{y \in c} \epsilon(x, y)$. The minimum ϵ-distance from cluster c_1 to cluster c_2 is defined as $\epsilon(c_1, c_2) = \min_{x \in c_1, y \in c_2} \epsilon(x, y)$.

For a dataset X and density parameter N_p, the dendrogram $G = (V, E)$ is a tree structure that gives all ϵ-clusters in X for each $\epsilon \geq 0$. V and E are the sets of nodes and edges in G respectively. Each node $v \in V$ corresponds to an ϵ-cluster for some ϵ, and the clusters corresponding to the children of v form the smallest possible partition of the maximal cluster corresponding to v into maximal clusters. Each value of ϵ specifies a clustering given by all nodes in V corresponding to ϵ-clusters. For a given x and N_p, the dendrogram is unique, and yields a hierarchy of possible clusterings, making it the foundational structure in hierarchical density-based clustering (HDBSCAN) [8].

For a node $v \in V$, $c(v)$ denotes the cluster corresponding to v. Each node v has a real number $\epsilon(v)$ associated with it where $\epsilon(v)$ is the smallest ϵ such that $c(v)$ is a ϵ-cluster. For a leaf node v corresponding to the cluster $\{x\}$, $\epsilon(v) = \epsilon_{N_p}(x)$. A node v_0 has children $v_1, v_2, ..., v_i$ if $c(v_1), c(v_2), ..., c(v_i)$ form the smallest possible partition of $c(v_0)$ composed of maximal clusters, which is guaranteed to be unique. The root node v_r of the resulting tree is such that $c(v_r) = X$.

Any $S \subset V$ such that no node in S is a descendant of another node in S induces a (partial) clustering of X given by $\{c(v) \text{ for } v \in S\}$. Any such clustering is called a dendrogram clustering. In fact, any (partial) clustering that consists of maximal clusters is a dendrogram (partial) clustering since every maximal cluster corresponds to a node in the dendrogram. The ϵ-cut of a dendrogram is the (partial) clustering that includes all clusters corresponding to nodes v such that $\epsilon(v) \leq \epsilon$, and v has no parent v' with $\epsilon(v') \leq \epsilon$. For any integer K, there exists at most one partial clustering given by an ϵ-cut of G that contains K clusters. The (partial) clustering output by DBSCAN for a given N_p and ϵ can be formed by taking the ϵ-cut of G, and adding any unclustered point x to the cluster c that minimizes $d(x, c)$ if $d(x, c) \leq \epsilon$ where $d(x, c) = \min_{y \in c} d(x, y)$.

IV. Results

We first define two conditions that should be satisfied for a K-clustering C of X to be unambiguous. The first condition is called weak separability, and the second is called local maximum separability. We then design an algorithm that is guaranteed to recover C if these two conditions are satisfied.

A. Weak Separability

The simplest density-based notion of cluster separability is what we call weak separability.

Definition 1 *For a given N_p, C is called weakly separable if for each $c \in C$, c is ϵ-connected for some $\epsilon < \min_{c' \in C,\ c' \neq c} \epsilon(c, c')$.*

If a clustering C is not weakly separable, there is ambiguity in what the correct clustering should be in the sense that there must exist a cluster $c \in C$ such that any ϵ-cluster containing c must include at least one point from a distinct cluster $c' \in C$. Figures 1 and 2 illustrate weak separability.

Interestingly, a dataset X often has more than one weakly separable K-clustering. For example, let $N_p = 2$ and $C = [\{1, 3, 5, 7.02, 9.02, 11.02\}, \{17, 18, 19, 20\}, \{22.01, 23.01, 24.01, 25.01\}]$. Clearly, C is weakly separable. However, $C' = [\{1, 3, 5\}, \{7.02, 9.02, 11.02\}, \{17, 18, 19, 20, 22.01, 23.01, 24.01, 25.01\}]$ is also a weakly separable 3-clustering. Intuitively, C is the correct clustering because the spacing between points is nearly uniform within each cluster in C, while there is a large relative gap in the middle of the third cluster in C'. It is indeed very common that the intuitively correct clustering is weakly separable, but is not the unique weakly separable clustering. In fact, it follows from Lemma 1 that any set of maximal clusters that partition X (a dendrogram clustering) is weakly separable.

Lemma 1 *For a given N_p, C is weakly separable if and only if it is a dendrogram clustering.*

Proof: Suppose C is a dendrogram clustering, but is not weakly separable. Let $G = (V, E)$ be the dendrogram. Then there exists some $v, v' \in V$ such that $c(v), c(v') \in C$, we have that $\epsilon(v) \geq \epsilon(c(v), c(v'))$, then $c(v)$ would include points from $c(v')$ since $c(v)$ is maximal. This contradicts the definition of a clustering.

If C is weakly separable, then each cluster in C is maximal. This follows because if a cluster c is not maximal, there exists a point $x \notin c$ that is ϵ-connected to c such that $\epsilon \leq \epsilon^*(c)$, which contradicts weak separability. Every maximal cluster has a corresponding node in G, so C is a dendrogram clustering. ■

Corollary 1 *For a given N_p, X has a unique weakly-separable K-clustering if and only if exactly one dendrogram K-clustering exists.*

If there are two weakly separable clusterings for X, it is impossible to guarantee that we recover one of them if our only criteria is to find a weakly separable clustering. Because there usually does not exist a unique weakly separable K-clustering even when an intuitively correct K-clustering exists, weak separability alone as a sufficient condition for clustering recoverability is not adequate.

B. Local Maximum Separability

To define local maximum separability, several new definitions are required. We call a point $x \in X$ a local maximum if $\epsilon_{N_p}(y) \geq \epsilon_{N_p}(x)$ for $y \in X$ such that $d(x, y) \leq \epsilon_{N_p}(x)$. For a cluster $c \in C$, let X_c^* denote the set of all points x in c such that $\epsilon_{N_p}(x) = \min_{y \in c} \epsilon_{N_p}(y)$. In other words, X_c^* denotes the set of all points x in c that have highest density among points in c.

For a given density parameter N_p, the relative separability of a point $x \in X$ to a point $y \in X$ is the value of A such that $A \cdot \epsilon_{N_p}(x) = \epsilon(x, y)$, and is denoted by $A(x, y)$. Similarly, the relative separability of a point $x \in X$ to a cluster c is given by $\min_{y \in c} A(x, y)$, and is denoted by $A(x, c)$.

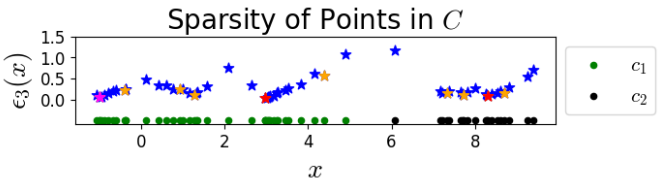

Fig. 1: A plot showing the sparsity value $\epsilon_3(x)$ of each point $x \in X$ in a 60 point one-dimensional clustering $C = [c_1, c_2]$ that is weakly separable and LM-separable for $N_p = 3$, implying that recovery is guaranteed. The orange, red, and pink stars correspond to local maxima. The red stars correspond to points in X_c^* for $c \in C$. The pink star corresponds to the local maximum $x \in X$ such that $\min_{y \in c(x):\, \epsilon_{N_p}(y) \leq \epsilon_{N_p}(x)} A(x, y) = A^\ell(C)$ (i.e. the local maximum that looks the most separated from its cluster). C is weakly separable because $\epsilon^*(c_1) = 0.74$ and $\epsilon^*(c_1) = 1.14$, while $\min_{c, c' \in C} \epsilon(c, c') = 1.18$. C is LM-separable because $A^\ell(C) = 8.95$ while $\min_{c, c' \in C} \min_{z \in X_c^*} A(z, c') = 15.35$.

Definition 2 *For a given N_p and C, we use $A^\ell(C)$ to denote the minimum $A \in \mathbb{R}$ such that*

$$\min_{y \in c(x):\, \epsilon_{N_p}(y) \leq \epsilon_{N_p}(x)} A(x, y) \leq A,$$

for every local maximum $x \in X$ where there exists a $y \in c(x)$ such that $\epsilon_{N_p}(y) \leq \epsilon_{N_p}(x)$. C is called local maximum separable (LM-separable) if

$$A^\ell(C) < \min_{c, c' \in C} \min_{z \in X_c^*} A(z, c').$$

In specific, if C has only one local maximum per cluster, then C is trivially LM-separable.

LM-separability specifies that for every local maximum $x \in X$ where there exists a $y \in c(x)$ whose density is at least as high as that of x, the minimum relative separability of x to such a y is smaller than the relative separability of the highest density point in any cluster to another cluster. This is a condition that should hold for a clustering to unambiguous because if there exists a local maximum x where the minimum relative separability to such a point $y \in c(x)$ is higher than the relative separability of a highest density point $z \in X_c^*$ for some $c \in C$ to another $c' \in C$ (and therefore to a local maximum in c'), then in a sense, x looks more like it belongs to a separate cluster from y than z looks like it belongs to a separate cluster from c', and the correct K-clustering of X is ambiguous. Figures 1 and 2 illustrate LM-separability.

Note that LM-separability does not imply weak separability. Let $N_p = 2$, and consider the clustering $C = [\{7, 8, 10, 13, 21\}, \{17, 25, 27\}]$. C is clearly not weakly separable because the clusters overlap, but is LM-separable, as $7, 8, 25, 27$ are the only local maxima.

Our main result states that if C is weakly separable and LM-separable for a given density parameter N_p, then it is the unique weakly separable, LM-separable clustering for N_p, and can be reconstructed efficiently.

Theorem 1 *If C is weakly separable and LM-separable for N_p, then C is the unique weakly separable, LM-separable clustering for N_p, and can be found in $O(|X|^3 \log(|X|))$ time.*

For a given N_p, it is in general not possible to recover the unique weakly separable, LM-separable clustering by simply finding an extension of the (partial) clustering given by the ϵ-cut of the dendrogram that contains K clusters (if there exists such an ϵ) as proved in Lemma 2. Since DBSCAN follows this approach, it is not sufficient for finding the unique weakly separable, LM separable clustering.

Lemma 2 *There exists a weakly separable, LM-separable clustering for some N_p that does not extend any (partial) clustering given by an ϵ-cut of the dendrogram.*

Proof: Recall that there can only be one ϵ-cut of the dendrogram that gives a (partial) K-clustering. Let $N_p = 3$, and suppose that $C = [\{1, 3, 5, 7.02, 9.02, 11.02\}, \{17, 18, 19, 20\}, \{22.01, 23.01, 24.01, 25.01\}]$. This is clearly weakly separable and LM-separable. The only ϵ-cut that gives a (partial) clustering with three clusters is chosen by setting $\epsilon = 2.01$, and is given by $C' = [\{3\}, \{9.02\}, \{17, 18, 19, 20, 22.01, 23.01, 24.01, 25.01\}]$. The only weakly separable clustering that is an extension of C' is $[\{1, 3, 5\}, \{7.02, 9.02, 11.02\}, \{17, 18, 19, 20, 22.01, 23.01, 24.01, 25.01\}]$. ■

In the clustering C used to prove Lemma 2, the distance separating the points $1, 3, 5$ from the points $7.02, 9.02, 11.02$ within the first cluster is 2.02, which is larger than the distance of 2.01 separating the clusters $\{17, 18, 19, 20\}$ and $\{22.01, 23.01, 24.01, 25.01\}$. However, 2.02 is very similar to the distance of 2 separating the other pairs of adjacent points in the first cluster, while 2.01 is very large compared to the distance of 1 separating the adjacent points in the second cluster. C is therefore a more natural clustering than C' in a sense, but an ϵ-cut is unable to capture C because it only considers absolute distances when separating clusters, without taking cluster density into account.

In the appendix we discuss a stronger notion of separability called strong separability which implies weak separability and LM-separability. Lemma 2 holds for strong separability as well.

V. Algorithm and Proof of Theorem 1

For density parameter N_p, we will prove that if C is weakly separable and LM-separable, then Algorithm 1 returns C, thus proving that C is the unique weakly separable and LM-separable $|C|$-clustering for N_p. We then prove that the algorithm can be implemented to run in $O(|X|^3 \log(|X|))$ time.

Algorithm 2 is an auxiliary algorithm used to define Algorithm 1. Algorithm 2 works by outputting a partial K-clustering of X that can then be postprocessed to form a K-clustering. For Algorithm 2, A governs the maximum variation in density within a partial cluster, M governs the minimum size of a partial cluster, and D governs the maximum variation in density among different partial clusters. We use $C_g(X, N_p, A, M, D)$ to denote the partial clustering of X that is output by Algorithm 2 with parameters N_p, A, M, D.

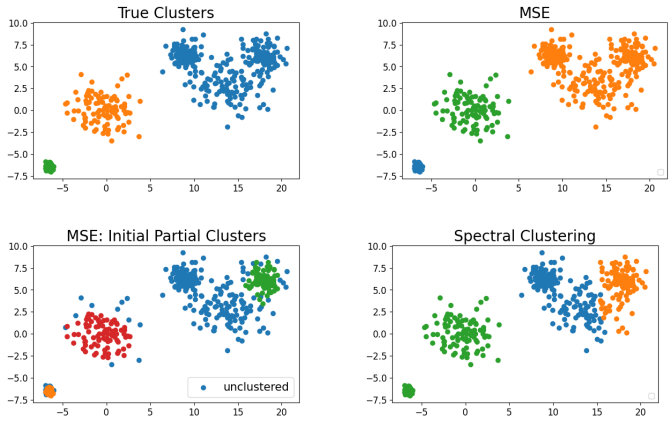

Fig. 2: A clustering with 500 points that is weakly separable and LM-separable, but not strongly separable for $N_p = 5$, implying that recovery by our algorithm ("MSE") is guaranteed. We compare against spectral clustering (scikit-learn) with the 4-nearest neighbors affinity matrix because for $N_p = 5$, our algorithm uses the 4 nearest neighbors of a point to calculate density.

Algorithm 1 Minimal Seed Expansion

Input: X, N_p, M, D, K
Output: $\hat{C}$
1: $C' \leftarrow$ min-A partial clustering of X for N_p, M, D, K
2: $\hat{C} \leftarrow$ output of Algorithm 3 with parameters X, C', N_p

At a high level, Algorithm 2 forms a partial clustering of X by greedily selecting the highest density unclustered point in X, and creating a maximal cluster centered at the point that satisfies the partial cluster constraints set by the parameters A, M, D. Note that if there are multiple candidate points for x^* in a greedy step of Algorithm 2, the final output is not affected by which candidate is assigned to x^*. Thus, for parameters X, N_p, A, M, D, the clustering $C_g(X, N_p, A, M, D)$ output by Algorithm 2 is unique.

Definition 3 *The min-A clustering of X for N_p, M, D, K is given by $C_g(X, N_p, A_{min}, M, D)$ where*

$$A_{min} = \min\{A : |C_g(X, N_p, A, M, D)| = K\}.$$

For parameters N_p, M, D, K, the min-A clustering is denoted by $C_m(X, N_p, M, D, K)$. $A_{min} \geq 1$ since $c^*(x, \epsilon) = \{\}$ if $\epsilon < \epsilon_{N_p}(x)$.

Algorithm 1 accepts N_p, M, D along with a number of clusters K as input, and begins by finding the min-A partial clustering for N_p, M, D, K. The min-A partial clustering is then passed to Algorithm 3 to produce the final output clustering.

Algorithm 2 Greedy algorithm to find partial clusters

Input: X, N_p, A, M, D
Output: C'
  $C' \leftarrow \emptyset$
  MinExtracted $\leftarrow \infty$
  Tried $\leftarrow \emptyset$
  while $X \setminus$ Tried $\neq \emptyset$ **do**
    $x^* \leftarrow \arg\min_{x \in X \setminus \text{Tried}} \epsilon_{N_p}(x)$
    if $\epsilon_{N_p}(x^*) > D \cdot$ MinExtracted **then**
      break loop
    $c' \leftarrow c^*(x^*, A \cdot \epsilon_{N_p}(x^*))$
    if $|c'| \geq M$ and $c' \cap c = \emptyset \ \ \forall c \in C'$ **then**
      add c' to C'
      if MinExtracted $= \infty$ **then**
        MinExtracted $\leftarrow \epsilon_{N_p}(x^*)$
      $X \leftarrow X \setminus c'$
    else
      Tried $\leftarrow$ Tried $\cup\, x$

Algorithm 3 Greedy algorithm to expand partial clusters

Input: X, C'
Output: $\hat{C}$
  $\hat{C} \leftarrow C'$
  $Y \leftarrow X \setminus (\cup_{c \in C'})$
  while $\cup_{c \in \hat{C}} c \neq X$ **do**
    $(x^*, c^*) \leftarrow \arg\min_{x \in Y, c \in \hat{C}} \epsilon(x, c)$
    $c^* \leftarrow c^* \cup \{x^*\}$
    $Y \leftarrow Y \setminus x^*$

Lemma 3 *For a given N_p, if C is LM-separable, then $|C_g(X, N_p, A^\ell(C), 1, \infty)| = K$.*

Proof: We will prove that all partial clusters output by Algorithm 2 are subsets of distinct clusters in C. Consider the ith partial cluster output by Algorithm 2. The highest density point x that the algorithm uses to build the ith cluster is clearly a local maximum, and can either be from a previously partially reconstructed cluster in C, or can be from a new cluster in C. If x is from a previously partially reconstructed cluster $c' \in C$, then it must include some $z \in X^*_{c'}$ by the definitions of LM-separability and $A^\ell(C)$, and the fact that $\epsilon_{N_p}(x) \geq \epsilon_{N_p}(z)$. This is a contradiction. This along with the fact that $M = 1 \leq N_p$ guarantees that x must be from a new cluster $c'' \in C$ and $x \in X^*_{c''}$. The ith partial cluster $c^*(x, A^\ell(C) \cdot \epsilon_{N_p}(x))$ must only include points from c'' by the definition of LM-separated and $A^\ell(C)$. Because only a new cluster center can be added to the partial clustering in each iteration of the algorithm, a partial K-clustering is output. ■

By Lemma 3, A_{min} is no larger than $A^\ell(C)$.

Lemma 4 *For a given N_p, if C is weakly separable and LM-separable, the min-A clustering $C_m(X, N_p, 1, \infty, |C|)$ is extendable to C.*

Proof: Consider the ith partial cluster output by Algorithm 2. The highest density point x used by algorithm to build the ith cluster must be a local maximum, and can either be from a previously partially reconstructed cluster in C, or can be from a new cluster in C.

If x is from a previously partially reconstructed cluster $c \in C$, the ith partial cluster does not include any points from other true clusters that are not yet part of a previously output partial cluster. This is because if it did include a point from such a cluster $c' \in C$, it would include a point $z \in X^*_{c'}$ by weak separability of C, and since $\epsilon_{N_p}(z) \geq \epsilon_{N_p}(x)$, it would imply that $A_{min} \cdot \epsilon_{N_p}(z) \geq \epsilon(c', c)$ which violates LM-separability of C since $A_{min} \leq A^\ell(C)$.

If the highest density point x is from a new true cluster $c \in C$, by LM-separability of C, only points from c are added to the partial cluster since $A_{min} \leq A^\ell(C)$.

Therefore, the algorithm outputs non-overlapping partial clusters such that each partial cluster chosen does not include any points from true clusters that are not yet part of a previously output partial cluster. Observe that, if no points in a cluster c have been added to previously output partial clusters by the time $w \in X^*_c$ is selected as the highest density unclustered point by the algorithm, then a partial cluster centered at w that is a non-empty subset of c will be output by the algorithm in that step since $M = 1 \leq N_p$, $A_{min} \leq A^\ell(C)$ and C is LM-separable. Therefore, we obtain a partial clustering that contains a partial cluster centered at $w \in X^*_c$ that is a non-empty subset of c for each $c \in C$. This is a min-A partial clustering for K, so it is a partial K-clustering that contains a partial cluster centered at some $w \in X^*_c$ that is a non-empty subset of c for each $c \in C$. Thus, the algorithm outputs a partial K-clustering that is extendable to C. ■

Lemma 5 *For a given N_p, if C is weakly separable, and Algorithm 3 is initialized with a partial clustering $\hat{C}$ that is extendable to C, then Algorithm 3 recovers C.*

Proof: Suppose that at some step in Algorithm 3, a point x is added to a cluster $c' \neq c(x)$ with some ϵ. This implies that at this step, there is no point $y \in c(x)$ that is not yet in $\hat{C}$ that is ϵ-connected to another point $z \in c(x)$ that is already in $\hat{C}$. This is a contradiction to the weak separability of C. ■

Lemma 4 and Lemma 5 imply that if C is weakly separable and LM-separable, then it is the unique weakly separable, LM-separable clustering for X and N_p. As a consequence of Lemmas 6 and 7, Algorithm 1 can be implemented in $O(|X|^3 \log(|X|))$ time.

Lemma 6 *For a given X, N_p, M, D, the min-A clustering can be found in $O(|X|^3 \log(|X|))$ time.*

Proof: We implement the greedy step of Algorithm 2 to have a deterministic rule for deciding whether to choose x^* to be x or y if $\epsilon_{N_p}(x) = \epsilon_{N_p}(y)$ (such as taking the point that comes first in the dataset).

For a given X, N_p, M, D, the number of clusters in $C_g(X, N_p, A, M, D)$ monotonically decreases as A increases

because of the following property. Consider some $a, a' \geq 1$ such that $a > a'$. At the end of the ith iteration of Algorithm 2, the set of remaining candidates for future x^* where $c^*(x^*, A \cdot \epsilon_{N_p}(x^*))$ will not intersection with a previously output cluster in the case when $A = a'$ is a superset of the set of such candidates when $A = a$.

We will prove this by induction. Suppose this property holds for the $(i-1)$th iteration of Algorithm 2, and consider the ith iteration. The point $x_{a'}$ picked to be x^* by the algorithm if $A = a'$ may or may not already be included in a cluster previously output by the algorithm with $A = a$. If not, then $x_{a'}$ will be chosen as x^* for the algorithm with $A = a$ by the inductive hypothesis, so clearly at the end of the ith step, the property still holds since $a > a'$.

If on the other hand, $x_{a'}$ is included in a cluster previously output by the algorithm with $A = a$, then all points x that are $(A \cdot \epsilon_{N_p}(x))$-connected to $x_{a'}$ already cannot be candidates at the beginning of the ith step for the algorithm with $A = a$. Therefore, in this case, at the end of the ith step, the property still holds.

Due to the monotonicity property we have just proved, if we have a set S that includes all A values that lead to distinct clusterings $C_g(X, N_p, A, M, D)$, then we can use binary search on S to find the min-A clustering for X, N_p, M, D. Algorithm 2 runs in $O(|X|^2)$ time. Therefore, if we have such a set S and the sorted list of its elements, this binary search approach to find the min-A clustering runs in $O(\log(|S|) \cdot |X|^2)$ time. Sorting the elements in S for use in binary search runs in $O(|S| \log(|S|))$ time.

Consider a point x^* selected in a greedy step of Algorithm 2. One set that includes all A values that could lead to different clusterings is given by $T_{x^*} = \{d(y,z)/\epsilon_{N_p}(x^*) : \ y, z \in X\}$ i.e. the set of all distances between points in X divided by $\epsilon_{N_p}(x^*)$. Thus, $S = \cup_{x \in X} T_x$ includes every possible A value that could lead to a different clustering. We have that $|S| = O(|X|^3)$. Thus, the binary search to find the min-A clustering runs in $O(|X|^2 \log(|X|))$ time. Constructing S runs in $O(|X|^3)$ time, and sorting the values of S for binary search runs in $O(|X|^3 \log(|X|))$ time. The total runtime of the approach is therefore $O(|X|^3 \log(|X|))$. ∎

Lemma 7 *For a given $X, N_p, \hat{C}$, Algorithm 3 can be implemented to run in $O(|X|^2)$ time.*

Proof: Observe that for each greedy step of Algorithm 3, any point $x \in X \setminus (\cup_{c \in \hat{C}} c)$ and cluster $c \in \hat{C}$ that minimize the quantity $\epsilon^!(x, c) = \min_{y \in c} \max(d(x,y), \epsilon_{N_p}(x), \epsilon_{N_p}(y))$ can be selected as (x^*, c^*). To see this, consider the set S of unclustered points that are closest to some cluster in terms of ϵ-distance. Denote this minimum ϵ-distance by ϵ'. Clearly, S must include an unclustered point that contains a clustered point in its ϵ'-ball. Thus, at each greedy step of Algorithm 3, we can pick (x, c) that minimizes $\epsilon^!(x, y)$.

To initialize the algorithm, for each unclustered point $x \in X \setminus (\cup_{c \in \hat{C}} c)$, and each clustered point y, we compute $\max(d(x,y), \epsilon_{N_p}(x), \epsilon_{N_p}(y))$ which then allows us to compute $\epsilon^!(x) = \min_{c \in \hat{C}} \epsilon^!(x, c)$ and $c^!(x) = \arg\min_{c \in \hat{C}} \epsilon^!(x, c)$ for every $x \in X \setminus (\cup_{c \in \hat{C}} c)$. This can be done in $O(|X|^2)$ time.

For each greedy step of Algorithm 3, we simply set (x^*, c^*) equal to the (x, c) that minimizes $\arg\min_{c \in \hat{C}} \epsilon^!(x, c)$ in $O(|X|)$ time by checking $(\epsilon^!(x), \ c^!(x))$ for all $x \in X \setminus (\cup_{c \in \hat{C}} c)$. After assigning x^* to c^*, we update $\epsilon^!(x)$ for every remaining unclustered point x by setting $\epsilon^!(x) \leftarrow \min(\epsilon^!(x), \ \max(d(x, x^*), \ \epsilon_{N_p}(x), \ \epsilon_{N_p}(x^*)))$, and setting $c^!(x) \leftarrow c^*$ if the value of $\epsilon^!(x)$ is changed. Each of these greedy steps takes $O(|X|)$ time and there are at most $O(|X|)$ greedy steps. This stage of the algorithm therefore runs in $O(|X|^2)$ time. ∎

VI. Experiments

We compare a modified version of Algorithm 1 for handling overlapping clusters to widely used algorithms for non-convex cluster recovery on a range of datasets. This modified version uses a variation of Algorithm 2 where each time a cluster is output, the points in the cluster are removed from X. This makes the check of whether a new cluster intersects with a previously output cluster inapplicable since only unclustered points remain in X at the beginning of each greedy step. To increase speed, instead of finding the min-A clustering exactly, the implementation approximates the min-A clustering by progressively adjusting A until a K-clustering is output by the modified version of Algorithm 2. To improve performance slightly, we use $N_p = 2$ in Algorithm 3, regardless of the N_p used for finding the initial partial clusters.

We compare our algorithm to K-means, spectral clustering, HDBSCAN, OPTICS, and spectACl [13]. We refer to our algorithm as "MSE" which stands for minimal seed expansion, and implemented it in Python. We use the spectral clustering and K-means implementations from scikit-learn, where the number of clusters K is specified by the user. The affinity matrix in spectral clustering is formed using each point's K_n nearest neighbors, where K_n is by the user. The implementation of K-means uses the "greedy K-means++" algorithm [17, 18]. We use the HDBSCAN implementation from the Python HDBSCAN clustering library, and we run it using the default cluster selection criteria named Excess of Mass (eom). The default setting for HDBSCAN sets the N_p equal to $M + 1$ where M is the minimum possible cluster size set by the user. We refer to this default version as "HDBSCAN." We refer to the version of HDBSCAN where N_p and $M + 1$ are not tied as "HDBSCAN (2)." We use the OPTICS implementation from scikit-learn, where N_p, the minimum cluster size M, and the cluster selection parameter X_i are set by the user. Neither HDBSCAN nor OPTICS uses the number of clusters K. spectACl uses knowledge of K and accepts a parameter ϵ. We use the implementation available on the TU Dortmund website from the authors (https://sfb876.tu-dortmund.de/spectacl/index.html).

The Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) are the most common measures of similarity between an estimated clustering and a ground truth clustering. For each algorithm, we report both measures for each dataset

dataset	# points	# features	# clusters	min. cluster size
Iris	150	4	3	50
Wine	178	13	3	48
Seeds	210	7	3	70
Glass	214	9	7	9
Cancer	556	30	2	212
Digits	1,797	64	10	174
Letter	4,000	16	26	132
MNIST	10,000	784	10	892

TABLE I: benchmark datasets and their properties

tested. The algorithms' performance is compared on the real world benchmark datasets whose properties are in Table I. All of these datasets are available on the UCI server [19]. "Cancer" refers to the Breast Cancer Wisconsin (Diagnostic) dataset. "Digits" refers to the test set of the Optical Recognition of Handwritten Digits dataset. "Letters" refers to the test set of the Letter Recognition dataset. "MNIST" refers to the test set of the MNIST dataset. For the MNIST dataset, we used t-SNE to reduce the dimensionality to two [20]. We also compare the same algorithms for 5 artificial datasets given in Figure 3 which were used in the example titled "Comparing different clustering algorithms on toy datasets" on the scikit-learn website [21]. The properties of the datasets are given in Table IV.

The first set of results for benchmark datasets are reported in Table II. In these experiments, for MSE, we did not optimize N_p, M and D exhaustively. Instead, we set $N_p = 3$, picked M smaller than the true minimum cluster size for each dataset to give a competitive ARI, and used the D value from the set $\{1.5, 2, 20\}$ that gave the best ARI.

For the other algorithms tested, we used grid search over the parameters to maximize ARI. For spectral clustering, we tried all K_n in the range $\{1, 2, ..., 20\}$ and reported the clustering with the best ARI. For HDBSCAN, we tried all N_p in the range $\{2, 3, ..., 20\}$ and reported the clustering with best ARI. For HDBSCAN (2), we used the same M and N_p we used for MSE. For OPTICS, used the same N_p we used for MSE, and tried all X_i values in the set $\{0, 0.05, 0.1, ..., 0.95\}$ and all M values in the set $\{\delta \cdot |X|/K : \delta \in \Delta\}$ where $\Delta = \{0.05, 0.10, 0.15, ..., 1\}$, and reported the clustering with best ARI. For MNIST, we tested $\Delta = \{0.1, 0.2, 0.3, ..., 1\}$ for OPTICS to save time. For spectACl, we tried all ϵ values in the range $\{0, 0.1, 0.2, ..., 100\}$. For some large datasets where the optimal ϵ was clearly less than 20, we tried all ϵ values in the range $\{0, 0.1, 0.2, ..., 20\}$.

For MSE, and HDBSCAN (2), N_p was set to 3 for all datasets and M was set to 35 for Iris, 30 for Wine, 50 for Seeds, 3 for Glass, 100 for Cancer and Digits, 70 for Letters, and 600 for MNIST. For MSE, D was set to 20 for Iris, Wine, Glass, and Cancer, 2 for Seeds, Letters and MNIST, and 1.5 for Digits. For spectral clustering, in order of dataset appearance in the Table II, the chosen K_n values are $4, 6, 18, 3, 5, 4, 19, 11$. For HDBSCAN, in order of appearance in the Table II, the chosen N_p values are $3, 20, 6, 3, 6, 4, 3, 14$. For OPTICS, in order of appearance in the Table II, the chosen

dataset	algorithm	ARI	NMI	$\lvert\hat{C}\rvert$
Iris	MSE	**0.886**	**0.871**	3
	Spectral	0.835	0.833	3
	K-means	0.716	0.742	3
	HDBSCAN	0.568	0.734	2
	HDBSCAN (2)	0.568	0.734	2
	OPTICS	0.732	0.753	4
	spectACl	0.653	0.682	3
Wine	MSE	**0.439**	0.430	3
	Spectral	0.401	0.395	3
	K-means	0.371	0.429	3
	HDBSCAN	0.292	0.379	2
	HDBSCAN (2)	0.291	0.403	3
	OPTICS	0.418	0.405	3
	spectACl	0.427	**0.462**	3
Seeds	MSE	**0.725**	0.682	3
	Spectral	0.657	0.660	3
	K-means	0.717	**0.695**	3
	HDBSCAN	0.336	0.468	5
	HDBSCAN (2)	0.409	0.469	3
	OPTICS	0.551	0.573	3
	spectACl	0.631	0.610	3
Glass	MSE	0.232	0.379	7
	Spectral	0.202	0.367	7
	K-means	0.216	0.388	7
	HDBSCAN	0.277	0.446	6
	HDBSCAN (2)	0.216	0.381	8
	OPTICS	**0.280**	**0.459**	3
	spectACl	0.252	0.381	7
Cancer	MSE	**0.743**	**0.628**	2
	Spectral	0.583	0.487	2
	K-means	0.491	0.465	2
	HDBSCAN	0.625	0.487	4
	HDBSCAN (2)	0.000	0.000	1
	OPTICS	0.737	0.620	2
	spectACl	0.707	0.586	2
Digits	MSE	**0.864**	**0.898**	10
	Spectral	0.781	0.892	10
	K-means	0.615	0.731	10
	HDBSCAN	0.575	0.770	22
	HDBSCAN (2)	0.559	0.762	9
	OPTICS	0.585	0.777	8
	spectACl	0.564	0.750	10
Letters	MSE	**0.193**	0.447	26
	Spectral	0.098	0.408	26
	K-means	0.130	0.356	26
	HDBSCAN	0.023	**0.536**	463
	HDBSCAN (2)	0.003	0.064	3
	OPTICS	0.058	0.392	46
	spectACl	0.092	0.264	26
MNIST	MSE	**0.854**	**0.854**	10
	Spectral	0.632	0.746	10
	K-means	0.656	0.751	10
	HDBSCAN	0.712	0.795	11
	HDBSCAN (2)	0.603	0.766	8
	OPTICS	0.623	0.770	8
	spectACl	0.817	0.831	10

TABLE II: $|\hat{C}|$ is the number of clusters output by the algorithm. For HDBSCAN and OPTICS, the set of noise points counts as a cluster.

dataset	algorithm	ARI	NMI	$\lvert\hat{C}\rvert$
Iris	MSE	**0.886**	**0.871**	3
	MSE (auto)	0.835	0.833	3
Wine	MSE	**0.439**	**0.430**	3
	MSE (auto)	0.359	0.420	3
Seeds	MSE	0.725	0.682	3
	MSE (auto)	0.725	0.682	3
Glass	MSE	0.232	0.379	7
	MSE (auto)	0.232	0.379	7
Cancer	MSE	**0.743**	**0.628**	2
	MSE (auto)	0.694	0.595	2
Digits	MSE	**0.864**	**0.898**	10
	MSE (auto)	0.819	0.879	10
Letters	MSE	0.193	0.447	26
	MSE (auto)	0.193	0.447	26

TABLE III: $\lvert\hat{C}\rvert$ is the number of clusters output by the algorithm.

M values are $38, 47, 56, 20, 57, 162, 54, 800$, and the chosen X_i values are $0, 0, 0, 0.15, 0.05, 0, 0, 0.05$. For spectACl, in order of appearance in the Table II, the chosen ϵ values are $3.3, 44.3, 2.0, 2.3, 77.2, 33.2, 8.9, 9.9$.

The second set of results for benchmark datasets are reported in Table III. MSE (auto) refers to MSE where M values and D values are optimized to give the best Calinski-Harabasz score, which is an internal clustering metric, meaning that it does not use the ground truth labels to asses clustering quality. For all of these experiments, we set $N_p = 3$. We chose the M from $\{\delta \cdot |X|/K \ : \ \delta \in \Delta\}$ where $\Delta = [0.025, 0.05, 0.075, ..., 0.975]$ and chose D from $[1.5, 2, 20]$. We chose from these values of M because we know the number of clusters K, and we choose from these values of D because we have observed that these values work well in general. In Table III, we also reported the statistics for the MSE clusterings from Table II for comparison. We observe that in general, the decrease in clustering quality is not large on these datasets. The average decrease in ARI is 0.032, and the average decrease in NMI is 0.014. The average percent decrease in ARI is 5.1% and the average percent decrease in NMI is 2.0%. We did not test on MNIST due to the slower speed of the algorithm on this dataset. Note that we only tested this approach for this one internal clustering quality measure, and the results may improve if another internal clustering quality measure (e.g. DBCV score [22]) is used to optimize the clustering. Additionally, N_p can also be optimized using this approach.

The results for artificial datasets in Figure 3 are reported in Table V. For MSE and HDBSCAN (2), N_p was set to 3 for all datasets, and M was set to 60 for all datasets in Table V. For MSE, D was set to 10 for Varied Variance Blobs, and 2 for all other datasets in Table V. For spectral clustering, in order of appearance in Table V, the chosen K_n values are $5, 5, 18, 6, 15$. For HDBSCAN, in order of appearance in Table V, the chosen N_p values are $3, 6, 9, 9, 14$. For OPTICS, N_p was set to 3, and in order of appearance in the Table V, the chosen M values are $225, 225, 133, 167, 167$, and the chosen X_i values are $0.0, 0.0, 0.05, 0.0, 0.1$ For spectACl, in order of appearance

dataset	# points	# features	# clusters	min. cluster size
Two Circles	500	2	2	250
Two Moons	500	2	2	250
Fixed Var. Blobs	500	2	3	166
Anisotropic	500	2	3	166
Varied Var. Blobs	500	2	3	166

TABLE IV: artificial datasets and their properties

dataset	algorithm	ARI	NMI	$\lvert\hat{C}\rvert$
Two Circles	MSE	1.000	1.000	2
	Spectral	1.000	1.000	2
	K-means	-0.002	0.000	2
	HDBSCAN	1.000	1.000	2
	HDBSCAN (2)	1.000	1.000	2
	OPTICS	1.000	1.000	2
	spectACl	1.000	1.000	2
Two Moons	MSE	1.000	1.000	2
	Spectral	1.000	1.000	2
	K-means	0.233	0.176	2
	HDBSCAN	1.000	1.000	2
	HDBSCAN (2)	0.699	0.737	4
	OPTICS	1.000	1.000	2
	spectACl	1.000	1.000	2
Fixed Variance Blobs	MSE	0.964	0.948	3
	Spectral	**0.976**	**0.961**	3
	K-means	0.970	0.954	3
	HDBSCAN	0.867	0.847	4
	HDBSCAN (2)	0.568	0.729	3
	OPTICS	0.726	0.734	3
	spectACl	0.964	0.942	3
Anisotropic	MSE	**1.000**	**1.000**	3
	Spectral	0.994	0.989	3
	K-means	0.555	0.593	3
	HDBSCAN	0.917	0.888	4
	HDBSCAN (2)	0.991	0.983	4
	OPTICS	**1.000**	**1.000**	3
	spectACl	0.994	0.989	3
Varied Variance Blobs	MSE	0.896	0.867	3
	Spectral	0.896	0.873	3
	K-means	0.787	0.778	3
	HDBSCAN	0.808	0.809	4
	HDBSCAN (2)	0.844	0.817	4
	OPTICS	0.924	0.894	3
	spectACl	**0.930**	**0.897**	3

TABLE V: $\lvert\hat{C}\rvert$ is the number of clusters output by the algorithm. For HDBSCAN and OPTICS, the set of noise points counts as a cluster.

in Table V, the chosen ϵ values are $0.2, 0.2, 1.2, 0.8, 1.3$.

Observe that with a lower bound on the minimum cluster size (M), and some knowledge of the maximum difference in maximum density among different clusters (D), MSE gives better ARI than the other algorithms on all benchmark datasets except Glass, and gives competitive ARIs for all artificial datasets. On the Glass dataset, OPTICS gives the best ARI, but outputs an incorrect number of clusters. The NMI values obtained by our algorithm are also competitive. Compared to the other algorithms, MSE performs well more consistently across the datasets tested, despite the fact that the parameters were not optimized in an exhaustive manner. As an example, in the case of the Digits dataset, it is clear in Figure 4 that our algorithm does a significantly better job of recovering the Digits clustering than the other algorithms. While our

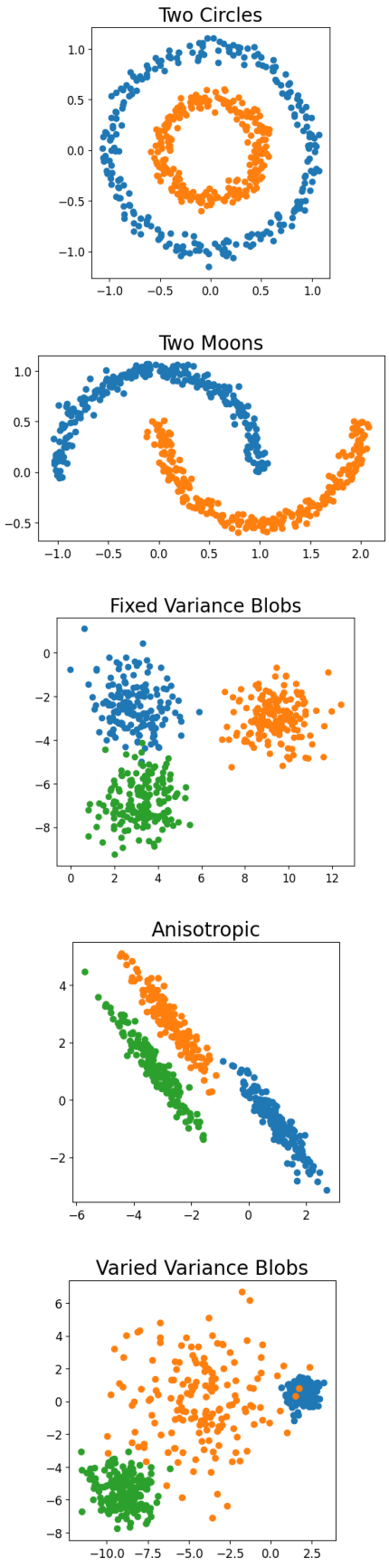

Fig. 3: Artificial datasets used in Table V .

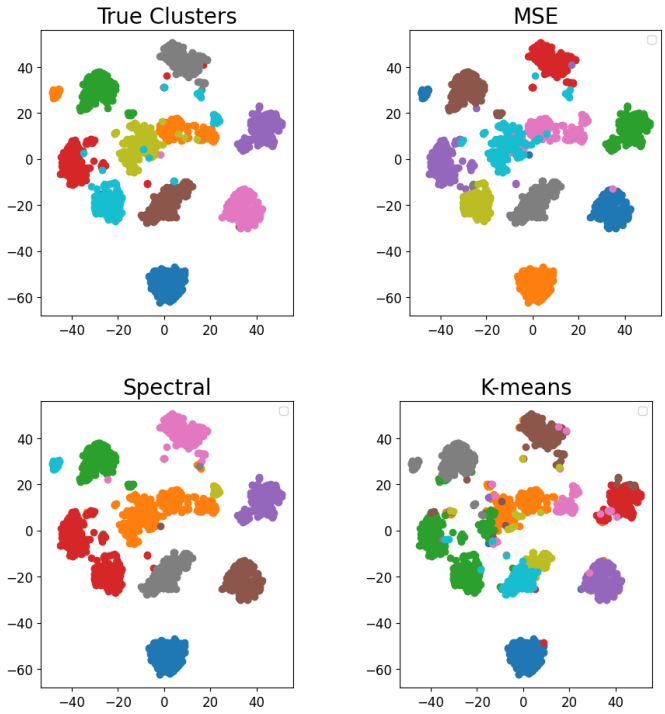

Fig. 4: t-SNE plots for the Digits dataset.

implementation of MSE was not optimized for speed, MSE clustered the MNIST dataset in three minutes on a laptop computer with 32 threads and 64 GB of RAM. MSE ran much faster on the other datasets.

VII. Conclusion

In this paper, we propose an information-theoretic characterization of when a K-clustering is ambiguous, and design an algorithm called Minimal Seed Expansion (MSE) that provably recovers the clustering whenever it is unambiguous. This characterization formalizes the situation when two high density regions within a cluster are separable enough that they look more like two distinct clusters than two truly distinct clusters in the clustering. We then implement and test a version of MSE that is modified to effectively handle overlapping clusters, and observe that it displays improved performance on many datasets without its parameters exhaustively optimized for each dataset tested. This improvement is in comparison to widely used algorithms for non-convex cluster recovery whose parameters are optimized using grid search independently for every dataset tested. MSE also performs well more consistently than all other algorithms tested in these experiments. We also optimized the parameters for MSE using grid search to maximize the Calinski-Harabasz score (an internal clustering quality measure), and observed little decrease in ARI and NMI compared to the case where parameters were chosen to give competitive ARI. This suggests that MSE can also be used effectively without manual parameter tuning.

References

[1] J. Hou, H. Gao, and X. Li, "Dsets-dbscan: A parameter-free clustering algorithm," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3182–3193, 2016.

[2] D. Birant and A. Kut, "St-dbscan: An algorithm for clustering spatial–temporal data," *Data & knowledge engineering*, vol. 60, no. 1, pp. 208–221, 2007.

[3] M. Stonebraker, J. Frew, K. Gardels, and J. Meredith, "The sequoia 2000 storage benchmark," *ACM SIGMOD Record*, vol. 22, no. 2, pp. 2–11, 1993.

[4] R. Ding, Q. Wang, Y. Dang, Q. Fu, H. Zhang, and D. Zhang, "Yading: Fast clustering of large-scale time series data," *Proceedings of the VLDB Endowment*, vol. 8, no. 5, pp. 473–484, 2015.

[5] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[6] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, pp. 226–231, 1996.

[7] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *ACM Sigmod record*, vol. 28, no. 2, pp. 49–60, 1999.

[8] L. McInnes, J. Healy, S. Astels, *et al.*, "hdbscan: Hierarchical density based clustering.," *J. Open Source Softw.*, vol. 2, no. 11, p. 205, 2017.

[9] P. Bhattacharjee and P. Mitra, "A survey of density based clustering algorithms," *Frontiers of Computer Science*, vol. 15, pp. 1–27, 2021.

[10] Y. Zhu, K. M. Ting, and M. J. Carman, "Density-ratio based clustering for discovering clusters with varying densities," *Pattern Recognition*, vol. 60, pp. 983–997, 2016.

[11] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar, "A density based algorithm for discovering density varied clusters in large spatial databases," *International Journal of Computer Applications*, vol. 3, no. 6, pp. 1–4, 2010.

[12] Z. Wang, Z. Ye, Y. Du, Y. Mao, Y. Liu, Z. Wu, and J. Wang, "Amd-dbscan: An adaptive multi-density dbscan for datasets of extremely variable density," in *2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, IEEE, 2022.

[13] S. Hess, W. Duivesteijn, P. Honysz, and K. Morik, "The spectacl of nonconvex clustering: A spectral approach to density-based clustering," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 3788–3795, 2019.

[14] J. Jang and H. Jiang, "Dbscan++: Towards fast and scalable density clustering," in *International conference on machine learning*, pp. 3019–3029, PMLR, 2019.

[15] H. Jiang, "Density level set estimation on manifolds with dbscan," in *International Conference on Machine Learning*, pp. 1684–1693, PMLR, 2017.

[16] H. Esfandiari, V. Mirrokni, and P. Zhong, "Almost linear time density level set estimation via dbscan," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 7349–7357, 2021.

[17] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy, "The effectiveness of lloyd-type methods for the k-means problem," *Journal of the ACM (JACM)*, vol. 59, no. 6, pp. 1–22, 2013.

[18] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," SODA '07, (USA), p. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[19] D. Dua and C. Graff, "Uci machine learning repository," 2017.

[20] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[21] "Comparing different clustering algorithms on toy datasets." https://scikit-learn.org/dev/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py. Accessed: 2024-10-03.

[22] D. Moulavi, P. A. Jaskowiak, R. J. G. B. Campello, A. Zimek, and J. Sander, "Density-based clustering validation," in *Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24-26, 2014* (M. J. Zaki, Z. Obradovic, P. Tan, A. Banerjee, C. Kamath, and S. Parthasarathy, eds.), pp. 839–847, SIAM, 2014.

VIII. Appendix

A. Strong Separability

A similar, yet stronger condition than weak separability is what we call strong separability.

Definition 4 *For a given N_p, C is called strongly separable if there exists some $A \in \mathbb{R}$ such that one of the following equivalent conditions holds.*

1) *For each $c \in C$, c is $(A \cdot \min_{x \in c} \epsilon_{N_p}(x))$-connected, and $A \cdot \min_{x \in c} \epsilon_{N_p}(x) < \min_{c' \in C,\, c' \neq c} \epsilon(c, c')$.*
2) *For each $c \in C$, $c^*(x, A \cdot \epsilon_{N_p}(x)) = c$ for any $x \in X_c^*$*

The minimum such A is denoted by $A^*(C)$.

The definition specifies for each cluster c, an ϵ relative to the sparsity of the maximum density point in c, such that no cluster can be ϵ-connected to c, yet all points in c must be ϵ-connected. The condition is therefore naturally satisfied by clusterings that have arbitrarily shaped clusters with arbitrarily many relatively separated regions of high density and arbitrary variation in density among different clusters, so long as the clusters are separated enough relative to the sparsity values of their respective maximum density points. Unlike weak separability, if C is strongly separable for N_p, then it is the unique strongly separable clustering for N_p and can be found efficiently by Theorem 2, which follows immediately from Theorem 1 and Lemmas 9 and 10.

Theorem 2 *If C is strongly separable for a given N_p, then C is the unique strongly separable clustering for N_p, and can be found in $O(|X|^3 \log(|X|))$ time.*

Perhaps a more intuitive condition that implies strong separability is given in the following lemma, which for each cluster $c \in C$, bounds the maximum variation in density among points in C. Let $\alpha(c) = \max_{x \in c} \epsilon_{N_p}(x) / \min_{y \in c} \epsilon_{N_p}(y)$ be the ratio of the sparsity of the minimum density point in X to that of the maximum density point in X.

Lemma 8 *For a given N_p, if there exists some $A \in \mathbb{R}$ such that for every $c \in C$, $\alpha(c) \leq A$, c is $(\max_{x \in c} \epsilon_{N_p}(x))$-connected, and $A \cdot \min_{x \in c} \epsilon_{N_p}(x) < \min_{c' \in C,\, c' \neq c} \epsilon(c, c')$, then C is strongly separable.*

Proof: The fact that for each $c \in C$, $\alpha(c) \leq A$ and c is $(\max_{x \in c} \epsilon_{N_p}(x))$-connected implies that for each $c \in C$, c is $(A \cdot \min_{x \in c} \epsilon_{N_p}(x))$-connected. ∎

If C is strongly separable, then it is weakly separable as proved in Lemma 9. However, strong separability of C does not imply that C is the unique weakly separable $|C|$-clustering for N_p. Furthermore, there exist weakly separable clusterings that are not strongly separable. For example, for $N_p = 2$, $C = [\{1, 3, 5\}, \{8, 10, 11, 13\}]$ is weakly separable but not strongly separable since the points in the second cluster imply that $A^*(C) \geq 2$, but $c^*(3, 2 \cdot \epsilon_{N_p}(3)) = c^*(3, 4) = X$. Intuitively, C is the correct 2-clustering, thus showing that strong separability as a sufficient condition for clustering recovery is still not general enough. Therefore, we proceed to introduce a more inclusive sufficient condition for recoverability in the next subsection. Figure 2 gives a larger example of a weakly separable clustering that is not strongly separable.

Lemma 9 *For a given N_p, if C is strongly separable, then it is weakly separable.*

Proof: This follows immediately from the first equivalent definition of strong separability. ∎

LM-separability is more natural than strong separability because LM-separability and weak separability hold precisely when a clustering is unambiguous. In fact, strong separability implies LM-separability.

Lemma 10 *For a given N_p, if C is strongly separable, then it is LM-separable.*

Proof: If there is only one local maximum per cluster, then this holds trivially. Suppose C is strongly separable but there exists a local maximum x such that $\min_{y \in c(x):\, \epsilon_{N_p}(y) < \epsilon_{N_p}(x)} A(x, y) \geq \min_{c, c' \in C} \min_{z \in X_c^*} A(z, c')$. Since $\epsilon_{N_p}(x) > \epsilon_{N_p}(w)$ for all $w \in X_{c(x)}^*$, this implies that $A(w, x) \geq \min_{c, c' \in C} \min_{z \in X_c^*} A(z, c')$ for all $w \in X_{c(x)}^*$, which in turn implies that $A^*(C) \geq \min_{c, c' \in C} \min_{z \in X_c^*} A(z, c')$. Thus, for the c, c', z that minimize $\min_{c, c' \in C} \min_{z \in X_c^*} A(z, c')$, we have that $c^*(z,\ A^*(C) \cdot \epsilon_{N_p}(z))$ contains at least one point from c'. This is a contradiction to strong separability. ∎

Furthermore, weak separability together with LM-separability does not imply strong separability. Let $N_p = 2$, and consider the clustering $C = [\{7, 8, 10, 13\}, \{17, 19, 21\}]$. C is weakly separable, and is LM-separable as $7, 8, 17, 19, 21$ are the only local maxima. C is not strongly separable because the first cluster implies that $A^*(C) \geq 3$, but $c^*(19, 3 \cdot \epsilon_{N_p}(19)) = c^*(19, 6) = X$. Figure 2 shows a larger example of a clustering that is not strongly separable, but is weakly separable and LM-separable, therefore guaranteeing recoverability by Theorem 1.

For a given N_p, it is in general not possible to recover the strongly separable clustering by simply finding an extension of the (partial) clustering given by the ϵ-cut of the dendrogram that contains K clusters (if there exists such an ϵ) as proved in Lemma 11. Since DBSCAN follows this approach, it is not sufficient for finding the strongly separable clustering.

Lemma 11 *There exists a strongly separable clustering for some N_p that does not extend any (partial) clustering given by an ϵ-cut of the dendrogram.*

Proof: Recall that there can only be one ϵ-cut of the dendrogram that gives a (partial) K-clustering. Let $N_p = 3$, and suppose that $C = [\{1, 3, 5, 7.02, 9.02, 11.02\}, \{17, 18, 19, 20\}, \{22.01, 23.01, 24.01, 25.01\}]$. This is a strongly separable clustering and $A^*(C) = 2$. The only ϵ-cut that gives a (partial) clustering with three clusters is chosen by setting $\epsilon = 2.01$, and is given by $C' = [\{3\}, \{9.02\}, \{17, 18, 19, 20, 22.01, 23.01, 24.01, 25.01\}]$. The only weakly separable clustering that is an extension of C' is $[\{1, 3, 5\}, \{7.02, 9.02, 11.02\}, \{17, 18, 19, 20, 22.01, 23.01, 24.01, 25.01\}]$. ∎