Lemma 16. With probability at least 1 − \(\delta/3\), for any \(x \in X\) and \(y \in X\), if \(\|x - y\|_1 \leq 1.5\varepsilon\), then \(\exists i \in [t], h_i(x) = h_i(y)\).

Proof. Fix arbitrary \(x \in X\) and \(y \in X\) such that \(\|x - y\|_1 \leq 1.5\varepsilon\). Consider \(i \in [t]\). We have:

\[
\Pr[h_i(x) \neq h_i(y)] \leq \sum_{j \in [D]} \Pr\left[\frac{\|x^{(j)} + \eta_i\|}{2\varepsilon} \neq \frac{\|y^{(j)} + \eta_i\|}{2\varepsilon}\right]
\]

\[
= \|x - y\|_1/(2\varepsilon) \leq 3/4.
\]

Thus, \(\mathbb{E}\left[\|i \in [t] \mid h_i(x) = h_i(y)\|\right] \geq 25 \log(n/\delta)\). By Chernoff bound, with probability at least 1 − \(\delta/(3n^2)\), \(\exists i \in [t], h_i(x) = h_i(y)\).

By taking union bound over all pairs \(x \in X\) and \(y \in X\), with probability at least 1 − \(\delta/3\), for any \(x \in X\) and \(y \in X\), if \(\|x - y\|_1 \leq 1.5\varepsilon\), \(\exists i \in [t]\) such that \(h_i(x) = h_i(y)\).

A direct corollary of above lemma is the following:

Corollary 17. With probability at least 1 − \(\delta/3\),

1. for any \(x \in C\) and \(y \in C\), if \(\|x - y\|_1 \leq 1.5\varepsilon\), then \(x\) and \(y\) are in the same connected component in \(G\);
2. for any \(x \in X \setminus C\) and \(y \in C\), if \(\|x - y\|_1 \leq 1.5\varepsilon\), then \(\exists y' \in C\), \(x\) and \(y'\) are in the same connected component in \(G\).
Theorem 18. For each connected component $C$ in $L_f(\lambda)$, there is a unique connected component $\tilde{C}$ in $G$ such that

$$d_{Haus}(C, \tilde{C}) \leq 4 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}.$$ 

Proof. Let $C$ be a connected component in $L_f(\lambda)$. Let $Q = X \cap B(C, \frac{\lambda}{2})$. According to Lemma 12, for $x \in C$, we can always find $x' \in Q$ such that $||x - x'||_2 \leq \frac{\lambda}{2}$. Next, let us show that $Q$ is connected in $G$. Consider two points $x', y' \in Q$, we can find a curve $\rho \subset C$ such that $d(x', \rho), d(y', \rho) \leq \varepsilon/2$. We can find a sequence of points $u_0, u_1, \ldots, u_m$ on $\rho$ such that $d(u_{i-1}, u_i) \leq \varepsilon/2$ and $d(u_0, x')$, $d(u_m, y') \leq \varepsilon/2$. According to Lemma 12, $\forall i \in \{0, 1, \ldots, m\}, \exists u_i' \in C$ such that $||u_i - u_i'||_2 \leq \varepsilon/2$. Notice that $\forall i \in [m], ||u_i' - u_{i-1}||_2 \leq 1.5\varepsilon$, and $||u_0' - x'||_2, ||u_m' - y'||_2 \leq 1.5\varepsilon$. By Corollary 17, $x', u_0'$ are connected in $G$, $\forall i \in [m], u_i', u_{i-1}'$ are connected in $G$, and $y', u_m'$ are connected in $G$. Thus $Q$ is connected in $G$.

Let $\tilde{C}$ be the connected component in $G$ containing $Q$. We have $\sup_{x \in C} d(x, \tilde{C}) \leq \varepsilon/2 \leq 4 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}$.

Consider an arbitrary point $x' \in \tilde{C}$. There must be a core point $y' \in \tilde{C} \cap C$ such that $||x' - y'||_2 \leq 2\varepsilon\sqrt{D}$. We can find a sequence of core points $v_0', v_1', \ldots, v_s'$ such that $v_0' = y'$, $v_s' \in Q$ and $\forall i \in [s], v_i', v_{i-1}'$ are connected in $G$. By Lemma 10, we can find a sequence of points $v_0, v_1, \ldots, v_s \in L_f(\lambda)$ such that $\forall i \in \{0, 1, \ldots, s\}, d(v_i, v_i') \leq 2 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta} \leq \frac{\varepsilon}{C_1}$ and $v_s \in C$. By Corollary 9, $v_0, v_1, \ldots, v_s$ must be connected in $L_f(\lambda)$. Thus, $v_0 \in C$. It implies that $d(y', C) \leq 2 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}$. Then, we have:

$$d(x', \tilde{C}) \leq d(y', C) + 2\varepsilon\sqrt{D} \leq 2 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta} + 2\varepsilon\sqrt{D} \leq 4 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}.$$ 

Thus, $\sup_{x \in \tilde{C}} d(x', \tilde{C}) \leq 4 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}$.

\hfill $\Box$

Removal of False Clusters

In theory, to remove false clusters, we need additional steps. We need to connect connected components with larger $\varepsilon$. This is known as pruning false clusters in the literature (see (Kpotufe and Von Luxburg 2011; Jiang 2017; Jang and Jiang 2018)).

1. Run Algorithm 3 with $k$ and $\varepsilon = \left( \frac{\lambda}{\varepsilon_0(1 - 2C_{\delta,n}/\sqrt{k})} \right)^{1/D}$. Let $\mathcal{G}$ be returned clusters. Let $C$ be the corresponding core points.

2. Let $\varepsilon' = 8 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}$.

3. Construct $t$ independent hash functions $h_1', h_2', \ldots, h_t' : \mathbb{R}^D \rightarrow \mathbb{Z}^D$, where $h_i'$ is constructed as the following: choose $\eta_i' \in [0, 2\varepsilon']$ uniformly at random, and $\forall x \in \mathbb{R}^D$, let

$$h_i'(x) := \left\lfloor \frac{x + \eta_i'}{2\varepsilon'} \right\rfloor.$$

4. Construct a graph $G'$: for $i \in [t]$ and for each subset $S$ of points with the same value of $h_i'(\cdot)$, create a star connecting every points in this subset. Let $G'$ be the connected component of $G'$.

5. Let $\tilde{G}$ be the clusters obtained by merging clusters from $\mathcal{G}$ which are subsets of the same cluster in $G'$.

Lemma 19. For $x, y \in X$, if $x, y$ are in the same connected component in $G$, $x, y$ are in the same connected component in $G'$.

Proof. If $x, y$ are connected in $G$, there is a path $x = u_0, u_1, \ldots, u_m = y$ in $G$ where $\forall i \in [m], d(u_{i-1}, u_i) \leq 2\sqrt{D}\varepsilon \leq \varepsilon'$. According to Corollary 17, there is a path $u_0, u_1, \ldots, u_m$ in $G'$.\hfill $\Box$

Theorem 20. For each connected component $\tilde{C}$ in $\tilde{G}$ which is not an isolated vertex, there is a unique connected component $\tilde{C}$ in $L_f(\lambda)$ such that

$$d_{Haus}(C, \tilde{C}) \leq 4 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}.$$ 

Proof. Let $x'$ be an arbitrary point in $\tilde{C}$. There is a core point $y' \in C$ such that $||x' - y'||_2 \leq 2\sqrt{D}\varepsilon$. By Lemma 10, there is a point $y \in L_f(\lambda)$ such that $||y' - y||_2 \leq 2 \left( \frac{\lambda}{C_1} \cdot 10C_{\delta,n}/\sqrt{k} \right)^{1/\beta}$. Let $C$ be the connected component in $L_f(\lambda)$
According to Lemma 10, we can find a sequence of points \( u_0, u_1, \cdots, u_m \) in \( \mathcal{C} \) such that \( d(u_i, u_{i+1}) \leq 2\sqrt{D}\varepsilon \), and \( \forall i \in [m], d(u_{i-1}, u_i') \leq 2\sqrt{D}\varepsilon' \). By triangle inequality, we have \( \forall i \in [m], d(u_{i-1}, u_i) \leq 6\sqrt{D}\varepsilon' \leq \left( \frac{L_f(\lambda)}{C_1} \right)^{1/\beta} \). By Corollary 9, we know that \( u_0, u_1, \cdots, u_m \) are in \( \mathcal{C} \). Thus, \( d(a', \mathcal{C}) \leq 4 \left( \frac{\lambda}{C_1} \right)^{1/\beta} \) which implies that

$$
\sup_{a' \in C} d(a', \mathcal{C}) \leq 4 \left( \frac{\lambda}{C_1} \right)^{1/\beta} .
$$

According to Lemma 18, there is a unique connected component \( \hat{\mathcal{C}} \) in \( \mathcal{G} \) such that \( d_{\text{Haus}}(C, \hat{\mathcal{C}}) \leq 4 \left( \frac{\lambda}{C_1} \right)^{1/\beta} \). Thus, \( \forall a' \in \hat{\mathcal{C}} \), there is a \( b' \in \hat{\mathcal{C}} \) such that \( d(a', b') \leq 8 \left( \frac{\lambda}{C_1} \right)^{1/\beta} \leq \varepsilon' \). According to Corollary 17, \( a', b' \) are in the same connected component in \( \mathcal{G}' \). By combining with Lemma 19, \( \hat{\mathcal{C}} \) is a subset of \( \hat{\mathcal{C}} \), which implies that

$$
\sup_{x \in \mathcal{C}} d(x, \hat{\mathcal{C}}) \leq 4 \left( \frac{\lambda}{C_1} \right)^{1/\beta} .
$$

\[\blacksquare\]

**Missing Details of Section: Experiments**

**Detailed description of the implemented algorithms.** The first version of DBSCAN (DSv1) exactly follows the description of Algorithm 1. The second version of DBSCAN (DSv2) has a slight modification and is described in Algorithm 4. The main difference is that instead of constructing the graph over all points, the modified version only constructs the graph over the core points and assigns each non-core point to the closest core point. According to the analysis of (Jiang 2017; Jang and Jiang 2018), it is easy to verify that the graph over core points has the same density level set estimation guarantee.

**Algorithm 4 Modified DBSCAN**

```plaintext
1: Inputs: \( X \subseteq \mathbb{R}^D, \varepsilon, k \)
2: Initialize core \( \mathcal{C} \leftarrow \emptyset \).
3: Check each point \( x \in X \) if \( |\{y \in X \mid d(x, y) \leq \varepsilon\}| \geq k \), then add \( x \) to the core \( \mathcal{C} \).
4: Construct a graph \( G \) where each vertex corresponds to a point in \( X \).
5: For each core point \( c \in \mathcal{C} \), add an edge in \( G \) between \( c \) and each vertex \( x \in \mathcal{C} \) which satisfies \( d(c, x) \leq \varepsilon \).
6: For each \( x \in X \setminus \mathcal{C} \), add an edge in \( G \) between \( x \) and \( c \in \mathcal{C} \) where \( c \) is the closest core point to \( x \).
7: Return connected components of \( G \).
```

The detailed implementation of DBSCAN++ (Jang and Jiang 2018) is presented in Algorithm 5.\(^6\) For DBSCAN++ with uniform initialization (DS++ unif), \( S \) is a subset of \( m \) uniform samples from \( X \). For DBSCAN++ with k-center initialization (DS++ k-ctr), the choice of \( S \) is described in Algorithm 6.

**Algorithm 5 DBSCAN++**

```plaintext
1: Inputs: \( X \subseteq \mathbb{R}^D, m, \varepsilon, k \)
2: Select a subset \( S \subseteq X \) with \( |S| = m \).
3: Initialize core \( \mathcal{C} \leftarrow \emptyset \).
4: Check each point \( x \in S \) if \( |\{y \in X \mid d(x, y) \leq \varepsilon\}| \geq k \), then add \( x \) to the core \( \mathcal{C} \).
5: Construct a graph \( G \) where each vertex corresponds to a point in \( X \).
6: For each core point \( c \in \mathcal{C} \), add an edge in \( G \) between \( c \) and each vertex \( x \in \mathcal{C} \) which satisfies \( d(c, x) \leq \varepsilon \).
7: For each \( x \in X \setminus \mathcal{C} \), add an edge in \( G \) between \( x \) and \( c \in \mathcal{C} \) where \( c \) is the closest core point to \( x \).
8: Return connected components of \( G \).
```

\(^6\)The details are confirmed by personal communications with the authors of DBSCAN++ (Jang and Jiang 2018).
Algorithm 6 K-Center Initialization

1: Inputs: $X \subset \mathbb{R}^D, m$
2: $S \leftarrow \{x_1\}$.
3: for $i := 1 \rightarrow m - 1$ do
4: $S \leftarrow S \cup \{\arg \max_{x \in X} d(x, S)\}$
5: end for

Algorithm 7 Detailed Implementation of Near Linear time DBSCAN

1: Inputs: $X \subset \mathbb{R}^D, t, \varepsilon, k$
2: Compute core $C$ by Algorithm 2.
3: Draw $t$ independent hash functions $h_1, h_2, \cdots, h_t : \mathbb{R}^D \rightarrow \mathbb{Z}^D$, where $h_i$ is constructed as the following: choose $\eta_i \in [0, 2\varepsilon]^D$ uniformly at random, and $\forall x \in \mathbb{R}^D$, let $h_i(x) := \left\lfloor \frac{x + \eta_i}{2\varepsilon} \right\rfloor$.
4: Construct $G$: for $i \in [t]$ and for each part $S \subseteq C$ with the same value of $h_i(\cdot)$, choose an arbitrary point $s \in S$, and add an edge in $G$ between $s$ and every $x \in S$ with $d(s, x) \leq \varepsilon$.
5: for $\varepsilon' := \varepsilon, 2\varepsilon, 4\varepsilon, 8\varepsilon, \cdots$ until no isolated non-core point do
6: Draw $t$ independent hash functions $h'_1, h'_2, \cdots, h'_t : \mathbb{R}^D \rightarrow \mathbb{Z}^D$, where $h'_i$ is constructed as the following: choose $\eta'_i \in [0, 2\varepsilon']^D$ uniformly at random, and $\forall x \in \mathbb{R}^D$, let $h'_i(x) := \left\lfloor \frac{x + \eta'_i}{2\varepsilon'} \right\rfloor$.
7: For each isolated non-core point $x \in X \setminus C$, find one arbitrary core point $c \in C$ such that $\exists i \in [t], h'_i(c) = h'_i(x)$. If such point $c$ exists and $d(c, x) \leq \varepsilon'$, connect $x$ to $c$ in $G$.
8: end for
9: Return connected components of $G$.

Further discussion of experimental results. For experiments on real datasets (A)-(K), we do linear search to find best $\varepsilon$ for each compared algorithm. In particular we search 100 times for each dataset. According to (Jang and Jiang 2018), the search range for each dataset is shown in Table 6. The Adjusted Random Index scores and Adjusted Mutual Information scores for different $\varepsilon$ are shown in Figure 2.

Table 6: The range of $\varepsilon$ in the optimal tuning procedure for real datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Range of $\varepsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A)</td>
<td>[0,4]</td>
</tr>
<tr>
<td>(B)</td>
<td>[0,500]</td>
</tr>
<tr>
<td>(C)</td>
<td>[0,100]</td>
</tr>
<tr>
<td>(D)</td>
<td>[0,150]</td>
</tr>
<tr>
<td>(E)</td>
<td>[0,1300]</td>
</tr>
<tr>
<td>(F)</td>
<td>[0,1,6]</td>
</tr>
<tr>
<td>(G)</td>
<td>[0,800]</td>
</tr>
<tr>
<td>(H)</td>
<td>[0,3.5]</td>
</tr>
<tr>
<td>(I)</td>
<td>[0,8]</td>
</tr>
<tr>
<td>(J)</td>
<td>[0,10]</td>
</tr>
<tr>
<td>(K)</td>
<td>[0,45]</td>
</tr>
</tbody>
</table>

As shown by Table 2, the quality of the clusters obtained by the second version of DBSCAN (DSv2) is uniformly better than the quality of the clusters obtained by the original version of DBSCAN (DSv1). This implies that the connected components of the graph over core points usually provide better clusters than the connected components of the graph over all points. Similar to the experimental results presented by (Jang and Jiang 2018), the accuracy of DBSCAN++ is uniformly better than the accuracy of the original version of DBSCAN. But we show that the accuracy of the modified version of DBSCAN is comparable to the accuracy of DBSCAN++. Our near linear time DBSCAN has similar accuracy as modified DBSCAN and DBSCAN++.

As shown in all experiments, KDTree cannot improve the running time of DBSCAN and DBSCAN++ on these datasets due to large overhead.
Figure 2: **The qualities of clustering results over different $\varepsilon$**: For each dataset (A)-(K), the left column corresponds to the Adjusted Random Index scores, and the right column corresponds to the Adjusted Mutual Information scores. $x$-axis corresponds to the tuning range of $\varepsilon$ and $y$-axis corresponds to the scores.