Clustering
Faster DBSCAN and HDBSCAN in Low-Dimensional Euclidean Space

Thomas van Dijk
SS19: Algorithmen für geographische Informationssysteme
3. 7. 2019
Clustering

Clustering is classically the problem of finding a partition of a data set such that elements in the same cell ("cluster") are near each other according to a given distance criterion, while elements in different sets are distant.
Clustering is classically the problem of finding a partition of a data set such that elements in the same cell ("cluster") are near each other according to a given distance criterion, while elements in different sets are distant.

Fundamental problem in data mining, but not uniquely defined.

What are you clustering? What are you trying to do with the data?
Clustering is classically the problem of finding a partition of a data set such that elements in the same cell ("cluster") are near each other according to a given distance criterion, while elements in different sets are distant.

Fundamental problem in data mining, but not uniquely defined.

What are you clustering? What are you trying to do with the data?

Distance: Euclidean? Metric?
Clustering

Clustering is classically the problem of finding a partition of a data set such that elements in the same cell ("cluster") are near each other according to a given distance criterion, while elements in different sets are distant.

Fundamental problem in data mining, but not uniquely defined.

What are you clustering? What are you trying to do with the data?

Distance: Euclidean? Metric?

How many clusters? What can clusters look like?
Sur la division des corps matériels en parties

par

H. STEINHAUS

Présenté le 19 Octobre 1956

Un corps $Q$ est, par définition, une répartition de matière dans l’espace, donnée par une fonction $f(P)$; on appelle cette fonction la \textit{densité} du corps en question; elle est définie pour tous les points $P$ de l’espace; elle est non-négative et mesurable. On suppose que l’ensemble caractérisé, par exemple $F = \{P; f(P) = 0\}$ est borné et de mesure positive.
1. Introduction

The main purpose of this paper is to describe a process for partitioning an $N$-dimensional population into $k$ sets on the basis of a sample. The process, which is called \textit{k-means}, appears to give partitions which are reasonably efficient in the sense of within-class variance. That is, if $p$ is the probability mass function for the population, $S = \{S_1, S_2, \ldots, S_k\}$ is a partition of $E_N$, and $u_i$,
A \textbf{Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise}?

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu

Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
{ester | kriegel | sander | xwxu}@informatik.uni-muenchen.de
Clustering

A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu

Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
{ester | kriegel | sander | xwxu}@informatik.uni-muenchen.de

\[ \geq 8 \times 10^3 \text{ citations} \]

KDD “test of time award” 2014

Open source implementations available in many languages
A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu

Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
{ester | kriegel | sander | xwxu}@informatik.uni-muenchen.de

Clustering

DBSCAN

1996

ing an appropriate value for it. It discovers clusters of arbi-
trary shape. Finally, DBSCAN is efficient even for large spa-
tial databases. The rest of the paper is organized as follows.
Clustering

database 1
Clustering

database 2
Clustering

database 3
DBSCAN: Objectives

1. “Minimal requirements of domain knowledge to determine the input parameters, because appropriate values are often not known in advance when dealing with large databases.”
DBSCAN: Objectives

1. “Minimal requirements of domain knowledge to determine the input parameters, because appropriate values are often not known in advance when dealing with large databases.”

2. “Discovery of clusters with arbitrary shape, because the shape of clusters in spatial databases may be spherical, drawn-out, linear, elongated etc.”
DBSCAN: Objectives

1. “Minimal requirements of domain knowledge to determine the input parameters, because appropriate values are often not known in advance when dealing with large databases.”

2. “Discovery of clusters with arbitrary shape, because the shape of clusters in spatial databases may be spherical, drawn-out, linear, elongated etc.”

3. “Good efficiency on large databases, i.e. on databases of significantly more than just a few thousand objects.”
DBSCAN

**Given:** data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\epsilon$ and $k$. 
Given: data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\varepsilon$ and $k$. 

DBSCAN
DBSCAN

**Given:** data points \( X \), distance function \( d(\cdot, \cdot) \), thresholds \( \varepsilon \) and \( k \).

**Def.** The \( \varepsilon \)-**neighborhood** of a point \( p \in X \) is
\[
N_\varepsilon(p) = \{ q \in X \mid d(p, q) \leq \varepsilon \}.
\]
DBSCAN

Given: data points $X$, distance function $d(·, ·)$, thresholds $\varepsilon$ and $k$.

Def. The $\varepsilon$-neighborhood of a point $p \in X$ is

$$N_\varepsilon(p) = \{ q \in X \mid d(p, q) \leq \varepsilon \}.$$ 

Def. A point $p \in X$ is called a core point iff $|N_\varepsilon(p)| \geq k$. 
DBSCAN

Given: data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\epsilon$ and $k$.

Def. The $\epsilon$-neighborhood of a point $p \in X$ is
$$N_\epsilon(p) = \{ q \in X \mid d(p, q) \leq \epsilon \}.$$ 

Def. A point $p \in X$ is called a core point iff $|N_\epsilon(p)| \geq k$.

Def. A point $p \in X$ is directly density-reachable from a point $q$ iff:
$$p \in N_\epsilon(q) \quad |N_\epsilon(q)| \geq k \quad (q \text{ is a core point})$$
DBSCAN

**Given:** data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\varepsilon$ and $k$.

**Def.** The $\varepsilon$-neighborhood of a point $p \in X$ is

$$N_\varepsilon(p) = \{ q \in X \mid d(p, q) \leq \varepsilon \}.$$

**Def.** A point $p \in X$ is called a **core point** iff $|N_\varepsilon(p)| \geq k$.

**Def.** A point $p \in X$ is **directly density-reachable** from a point $q$ iff:

$$p \in N_\varepsilon(q) \quad |N_\varepsilon(q)| \geq k \quad (q \text{ is a core point})$$

Not a symmetric relation!
DBSCAN

Given: data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\varepsilon$ and $k$.

Def. The $\varepsilon$-neighborhood of a point $p \in X$ is
$$N_\varepsilon(p) = \{ q \in X \mid d(p, q) \leq \varepsilon \}.$$

Def. A point $p \in X$ is called a core point iff $|N_\varepsilon(p)| \geq k$.

Def. A point $p \in X$ is directly density-reachable from a point $q$ iff:
$$p \in N_\varepsilon(q) \quad |N_\varepsilon(q)| \geq k \quad (q \text{ is a core point})$$

Not a symmetric relation!

Def. A point $p \in X$ is density reachable from a point $q$ if there exists a chain of direct density-reachability from $q$ to $p$. 
DBSCAN

**Given:** data points $X$, distance function $d(\cdot, \cdot)$, thresholds $\varepsilon$ and $k$.

**Def.** The $\varepsilon$-neighborhood of a point $p \in X$ is 
$$N_\varepsilon(p) = \{ q \in X | d(p, q) \leq \varepsilon \}.$$  

**Def.** A point $p \in X$ is called a **core point** iff $|N_\varepsilon(p)| \geq k$.

**Def.** A point $p \in X$ is **directly density-reachable** from a point $q$ iff:  
$$p \in N_\varepsilon(q) \quad \quad |N_\varepsilon(q)| \geq k \quad (q \text{ is a core point})$$  
Not a symmetric relation!

**Def.** A point $p \in X$ is **density reachable** from a point $q$ if there exists a chain of direct density-reachability from $q$ to $p$.

**Def.** A point $p \in X$ is **density connected** to a point $q$ if there exists a (core) point $r$ such that both $p$ and $q$ are density-reachable from $r$. 
DBSCAN example

Legend

\( k = 3 \)
DBSCAN example

Legend

\( k = 3 \)

**Distance** \( \varepsilon \)
DBSCAN example

Legend

\( k = 3 \)

**Distance** \( \varepsilon \)

**Core points**
DBSCAN example

Legend

$ k = 3 $

**Distance** $\varepsilon$

**Core points**
DBSCAN example

Legend

\( k = 3 \)

Distance \( \varepsilon \)

Core points
DBSCAN example

Legend

- \( k = 3 \)
- Distance \( \varepsilon \)
- Core points
DBSCAN example

Legend

\( k = 3 \)

**Distance** \( \varepsilon \)

Core points

Density connected

\[ p \quad q \quad r \]
DBSCAN example

Legend

$k = 3$

Distance $\varepsilon$

Core points

Density connected

DBSCAN clustering

noise point $\rightarrow \bullet$

border point $\leftarrow\bullet$

border point $\rightarrow \bullet$
DBSCAN example

Legend

- \( k = 3 \)
- Distance \( \varepsilon \)
- Core points
- Density connected
- DBSCAN clustering
- DBSCAN* clustering

Distance \( \varepsilon \)

Core points

Density connected

DBSCAN clustering

DBSCAN* clustering

noise point \( \rightarrow \) ●

border point

border point \( \rightarrow \) ●
DBSCAN example

Legend

- $k = 3$
- **Distance** $\varepsilon$
- **Core points**
- **Density connected**

- **DBSCAN** clustering
- **DBSCAN**$^*$ clustering

Runtime

Naive algorithm runs in $O(n^2)$ time.
**DBSCAN example**

**Legend**

- $k = 3$
- **Distance** $\varepsilon$
- **Core points**
- **Density connected**
- DBSCAN clustering
- DBSCAN* clustering

**Runtime**

Naive algorithm runs in $O(n^2)$ time.

“Since the Eps-Neighborhoods are expected to be small compared to the size of the whole data space, the average run time complexity of a single region query is $O(\log n)$. (...) Thus, the average run time complexity of DBSCAN is $O(n \times \log n)$.”
### De Berg, Gunawan, Roeloffzen (2017)

#### Everywhere: $\varepsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>$2D$</th>
<th>$dD$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$\mathcal{O}(n \log n)$</td>
<td>$\mathcal{O}(n^{2-\lfloor d/2 \rfloor + 1 + \gamma})$ $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$\mathcal{O}(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>
De Berg, Gunawan, Roeloffzen (2017)

**Everywhere:** $\varepsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>$dD$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$\Theta(n \log n)$</td>
<td>$\Theta(n^{2 - \lceil d/2 \rceil + 1 + \gamma})$ $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$\Theta(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>
Box graph $g_{box}$
Box graph $G_{\text{box}}$
Box graph $G_{\text{box}}$

A grid-based approach?

- Make a grid
- Side length $\frac{\epsilon}{\sqrt{2}}$
  (Assumes we can round down to a multiple of $\frac{\epsilon}{\sqrt{2}}$)
Box graph $g_{\text{box}}$

A grid-based approach?

Make a grid
Side length $\frac{\varepsilon}{\sqrt{2}}$

(Assumes we can round down to a multiple of $\frac{\varepsilon}{\sqrt{2}}$)

Connectivity within cells?
Box graph $G_{\text{box}}$

A grid-based approach?

- Make a grid
- Side length $\epsilon/\sqrt{2}$
  
  (Assumes we can round down to a multiple of $\epsilon/\sqrt{2}$)

- Connectivity within cells?
- Between points in different cells?
Box graph $G_{\text{box}}$

**A grid-based approach?**

Make a grid
Side length $\varepsilon/\sqrt{2}$

(Assumes we can round down to a multiple of $\varepsilon/\sqrt{2}$)

Connectivity within cells?
Between points in different cells?

Not clear how to get a runtime bound in $n$ without assumption on the distribution.

Be more flexible...
Box graph $G_{\text{box}}$

1. Construct boxes

$\varepsilon$:

$\varepsilon / \sqrt{2}$:
Box graph $G_{\text{box}}$

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 

1. Construct boxes

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 

SORTED
Box graph $G_{\text{box}}$

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.
Box graph $\mathcal{G}_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 

SORTED

$\varepsilon$: 

$\frac{\varepsilon}{\sqrt{2}}$: 

Box graph $G_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \frac{\epsilon}{\sqrt{2}}$. 

\[ \epsilon : \]
\[ \frac{\epsilon}{\sqrt{2}} : \]
Box graph $\mathcal{G}_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.
Box graph $\mathcal{G}_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 

$\varepsilon$: \\
$\frac{\varepsilon}{\sqrt{2}}$: \\
SORTED
Box graph $G_{\text{box}}$

Add points as long as strip width $\leq \epsilon/\sqrt{2}$.
Box graph $G_{\text{box}}$

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$.

1. Construct boxes

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 
Box graph $G_{\text{box}}$

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.

1. Construct boxes

$> \varepsilon/\sqrt{2}$
Box graph $\mathcal{G}_{\text{box}}$

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$.

1. Construct boxes

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$. 

SORTED
Box graph $G_{\text{box}}$

1. **Construct boxes**
   
   Add points as long as strip width $\leq \varepsilon/\sqrt{2}$. 

   $\varepsilon$: 
   $\varepsilon/\sqrt{2}$:
Box graph $\mathcal{G}_{\text{box}}$

1. **Construct boxes**

- Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.
- Per strip: add points to box as long as height $\leq \varepsilon/\sqrt{2}$.
Box graph $\mathcal{G}_{\text{box}}$

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$.

Per strip: add points to box as long as height $\leq \frac{\varepsilon}{\sqrt{2}}$. 

1. Construct boxes
Box graph $G_{\text{box}}$

**1. Construct boxes**

Add points as long as strip width $\leq \frac{\varepsilon}{\sqrt{2}}$.

Per strip: add points to box as long as height $\leq \frac{\varepsilon}{\sqrt{2}}$. 

$\varepsilon$: 

$\frac{\varepsilon}{\sqrt{2}}$: 

\[ > \frac{\varepsilon}{\sqrt{2}} \]
Box graph $G_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \epsilon / \sqrt{2}$.

Per strip: add points to box as long as height $\leq \epsilon / \sqrt{2}$. 

$\varepsilon$: 
$\epsilon / \sqrt{2}$:
Box graph $G_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.

Per strip: add points to box as long as height $\leq \varepsilon/\sqrt{2}$.
Box graph $G_{\text{box}}$

1. Construct boxes

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.

Per strip: add points to box as long as height $\leq \varepsilon/\sqrt{2}$. 

$\varepsilon$:

$\varepsilon/\sqrt{2}$:
Box graph $\mathcal{G}_{\text{box}}$

1. Construct boxes

Add points as long as strip width \( \leq \frac{\varepsilon}{\sqrt{2}} \).

Per strip: add points to box as long as height \( \leq \frac{\varepsilon}{\sqrt{2}} \).

Runtime:

Sort by $x$

$\Theta(n \log n)$
Box graph $\mathcal{G}_{\text{box}}$

1. **Construct boxes**

Add points as long as strip width $\leq \varepsilon/\sqrt{2}$.

Per strip: add points to box as long as height $\leq \varepsilon/\sqrt{2}$.

Runtime:

Sort by $x$

$\Theta( n \log n )$

Sort by $y$ per strip

$\sum_j \Theta( n_j \log n_j )$

Total

$\Theta( n \log n )$
Box graph $\mathcal{G}_{\text{box}}$

**Property of single boxes**

All points within a box...
Box graph $g_{\text{box}}$
Box graph $G_{\text{box}}$

**Property of single boxes**

All points within a box... are in $\varepsilon$-neighbourhood. (Box width & height are each $\leq \varepsilon/\sqrt{2}$.)

In boxes with at least $k$ points, ...
Box graph $\mathcal{G}_{\text{box}}$

Property of single boxes

All points within a box... are in $\varepsilon$-neighbourhood.

(Box width & height are each $\leq \varepsilon/\sqrt{2}$.)

In boxes with at least $k$ points, ...
all points are core points.

$k = 4$

$\varepsilon$: 

$\varepsilon/\sqrt{2}$:
Box graph $\mathcal{G}_{\text{box}}$

Property of single boxes

All points within a box... are in $\varepsilon$-neighbourhood.

(Box width & height are each $\leq \varepsilon/\sqrt{2}$.)

In boxes with at least $k$ points, ...
all points are core points.

In boxes with fewer than $k$ points, ...
Box graph $\mathcal{G}_{\text{box}}$

Property of single boxes

All points within a box... are in $\varepsilon$-neighbourhood. (Box width & height are each $\leq \varepsilon/\sqrt{2}$.)

In boxes with at least $k$ points, ... all points are core points.

In boxes with fewer than $k$ points, ... points can be core points.
Box graph $G_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$. 

$\varepsilon$: 

$\varepsilon/\sqrt{2}$:
Box graph $G_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$. 

$\varepsilon$: 

$\varepsilon/\sqrt{2}$:
Box graph $G_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

$\varepsilon$: 

$\varepsilon/\sqrt{2}$:
Box graph $\mathcal{G}_{\text{box}}$

**Property of box pairs**

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $\mathcal{G}_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have?
Box graph $\mathcal{G}_{\text{box}}$

**Property of box pairs**

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $\mathcal{G}_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $\in \Theta(1)$
Box graph $G_{\text{box}}$

**Property of box pairs**

Connect boxes with edge if distance between **boxes** is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $\in \Theta(1)$
Box graph $G_{\text{box}}$:

**Property of box pairs**

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $\in O(1)$
Box graph $G_{\text{box}}$:

Property of box pairs:
Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$:
none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $\in \Theta(1)$
Box graph $G_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $\in O(1)$
Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $22 \in O(1)$
Box graph $\mathcal{G}_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $\mathcal{G}_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $22 \in \Theta(1)$
Box graph $G_{\text{box}}$

**Property of box pairs**

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $22 \in \Theta(1)$
Box graph $G_{\text{box}}$

Property of box pairs

Connect boxes with edge if distance between boxes is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $22 \in \Theta(1)$
Box graph $G_{\text{box}}$

**Property of box pairs**

Connect boxes with edge if distance between **boxes** is at most $\varepsilon$.

Nonneighbours in $G_{\text{box}}$: none of these points are in $\varepsilon$-neighbourhood.

How many neighbours can a box have? $22 \in O(1)$
Box graph $\mathcal{G}_{\text{box}}$

$k = 4$

$\varepsilon/\sqrt{2}$:

2. Find all core points

Already have all core points in “crowded” boxes.
Box graph $G_{\text{box}}$

$k = 4$

$\varepsilon$: 

$\varepsilon / \sqrt{2}$:

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
Box graph $G_{\text{box}}$

$k = 4$

$\frac{\epsilon}{\sqrt{2}}$:

2. Find all core points

Already have all core points in "crowded" boxes.

For all "sparse" boxes: For all neighbour boxes:
Box graph $G_{\text{box}}$

$k = 4$

$\varepsilon / \sqrt{2}$:

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
  For all neighbour boxes:
    ... check all pairs.

Total runtime?
Box graph $G_{\text{box}}$

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
For all neighbour boxes:
... check all pairs.

Total runtime?
Other box is sparse:
Box graph $G_{\text{box}}$

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
   For all neighbour boxes:
     ... check all pairs.

Total runtime?
Other box is sparse:

$O(k^2) = O(1)$
Box graph $G_{box}$

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
For all neighbour boxes:
... check all pairs.

Total runtime?
Other box is sparse:
$O(k^2) = O(1)$

Other box is crowded:
Box graph $\mathcal{G}_{\text{box}}$

$k = 4$

$\epsilon$:
$\frac{\epsilon}{\sqrt{2}}$:

2. Find all core points

Already have all core points in “crowded” boxes.

For all “sparse” boxes:
  For all neighbour boxes:
    ... check all pairs.

Total runtime?

Other box is sparse:
  $O(k^2) = O(1)$

Other box is crowded:
  Charge to crowded box
### 2. Find all core points

- Already have all core points in “crowded” boxes.

- For all “sparse” boxes:
  - For all neighbour boxes:
    - ... check all pairs.

- Total runtime?
  - Other box is sparse:
    - \( O(k^2) = O(1) \)
  - Other box is crowded:
    - Charge to crowded box
    - Point in crowded box checked \( \leq 22k \) times.
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the **the same cluster**?
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the \textbf{the same cluster}?
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the the same cluster?

Bichromatic Closest Pair
Box graph $G_{\text{box}}$

**Pairs of crowded boxes**

These are all core points.

Are the **the same cluster**?

**Bichromatic Closest Pair**

In Euclidean 2D?
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the the same cluster?

Bichromatic Closest Pair

In Euclidean 2D?
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.
Are the the same cluster?
BICHROMATIC CLOSEST PAIR
In Euclidean 2D?
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the the same cluster?

Bichromatic Closest Pair

In Euclidean 2D?

Delaunay triangulation has this edge.
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the **same cluster**?

BICHROMATIC CLOSEST PAIR

In Euclidean 2D?
Delaunay triangulation has this edge.
$\Theta(n \log n)$
Box graph $G_{\text{box}}$

Pairs of crowded boxes

These are all core points.

Are the **the same cluster**?

**BICHROMATIC CLOSEST PAIR**

In Euclidean 2D?

Delaunay triangulation has this edge.

$O(n \log n)$

Charge to edges in $G_{\text{box}}$
Results

**Everywhere:** $\varepsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>$d=2$</th>
<th>$d\neq 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$O(n \log n)$</td>
<td>$O(n^{2-\lceil d/2 \rceil + 1 + \gamma})$; $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$O(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>
Results

**Everywhere:** $\epsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>dD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$\Theta(n \log n)$</td>
<td>$\Theta(n^{2-\left\lceil d/2\right\rceil+1+\gamma})$, $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$\Theta(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

1. Construct $g_{\text{box}}$
2. Find core points
3. Merge clusters
(4. Assign border points.)
## Results

**Everywhere:** $\epsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>dD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$\Theta(n \log n)$</td>
<td>$\Theta(n^{2 - \lceil d/2 \rceil + 1} + \gamma)$, $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$\Theta(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

1. Construct $g_{\text{box}}$

2. Find core points

3. Merge clusters

(4. Assign border points.)

BICHROMATIC CLOSEST POINT instead of Delauney triangulation.
### Results

**Everywhere:** $\varepsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>dD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DBSCAN</strong></td>
<td>$\Theta(n \log n)$ expected</td>
<td>$\Theta(n^{2-\lceil d/2 \rceil + 1 + \gamma}) \quad \gamma &gt; 0$</td>
</tr>
<tr>
<td><strong>HDBSCAN</strong></td>
<td>$\Theta(n \log n)$</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

1. Construct $g_{\text{box}}$

2. Find core points

3. Merge clusters

4. Assign border points.

(Bichromatic Closest Point instead of Delauney triangulation.

(Agarwal, Edelsbrunner, Schwarzkopf, 1991)
## Results

### Everywhere: ε free, k fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>dD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$O(\ n \log \ n \ )$</td>
<td>$O(\ n^2 - \lceil d/2 \rceil + 1 + \gamma ) \ \gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$O(\ n \log \ n \ ) \ \text{expected}$</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

1. Construct $g_{\text{box}}$
2. Find core points
3. Merge clusters
4. Assign border points.

Instead of Delauney triangulation, **BICHROMATIC CLOSEST POINT** instead of Delauney triangulation.

(Agarwal, Edelsbrunner, Schwarzkopf, 1991)
HDBSCAN

Use \textsc{DBSCAN}* and sweep $\varepsilon$ from 0 to $\infty$. 
HDBSCAN

Use DBSCAN* and sweep $\epsilon$ from 0 to $\infty$.

Initially all points are noise; eventually everything is one cluster.
Three types of “events.”
HDBSCAN

Use DBSCAN* and sweep $\epsilon$ from 0 to $\infty$.
Initially all points are noise; eventually everything is one cluster.

Three types of “events.”
- Noise point becomes core point. Call this value $d_{\text{core}}(p)$.
HDBSCAN

Use DBSCAN* and sweep $\varepsilon$ from 0 to $\infty$.

Initially all points are noise; eventually everything is one cluster.

Three types of “events.”

- Noise point becomes core point. Call this value $d_{\text{core}}(p)$.
- New cluster forms.
HDBSCAN

Use DBSCAN* and sweep $\varepsilon$ from 0 to $\infty$.

Initially all points are noise; eventually everything is one cluster.

Three types of “events.”

- Noise point becomes core point. Call this value $d_{\text{core}}(p)$.
- New cluster forms.
- Two clusters merge
HDBSCAN

Use DBSCAN and sweep $\varepsilon$ from 0 to $\infty$.

Initially all points are noise; eventually everything is one cluster.

Three types of “events.”

- Noise point becomes core point. Call this value $d_{\text{core}}(p)$.
- New cluster forms.
- Two clusters merge

Events only happen when $\varepsilon = d(p, q)$ for some $p, q$. 
HDBSCAN

Use DBSCAN* and sweep $\epsilon$ from 0 to $\infty$.

Initially all points are noise; eventually everything is one cluster.

Three types of “events.”

• Noise point becomes core point. Call this value $d_{\text{core}}(p)$.

• New cluster forms.

• Two clusters merge

Events only happen when $\epsilon = d(p, q)$ for some $p, q$.

Store this tree structure of cluster creation and merges: HDBSCAN.
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$. 
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$. Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$.

Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.

**Def.** Let $d_{\text{mr}}(p, q) = \max\{ d_{\text{core}}(p), d_{\text{core}}(q), d(p, q) \}$. 
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

- 
- 

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$. Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.

**Def.** Let $d_{\text{mr}}(p, q) = \max\{ d_{\text{core}}(p), d_{\text{core}}(q), d(p, q) \}$.

**Def.** Mutual reachability graph $G_{\text{mr}}$: complete, edge weights $d_{\text{mr}}$. 
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

- 

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$.

Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.

**Def.** Let $d_{\text{mr}}(p, q) = \max\{ d_{\text{core}}(p), d_{\text{core}}(q), d(p, q) \}$.

**Def.** Mutual reachability graph $G_{\text{mr}}$: complete, edge weights $d_{\text{mr}}$.

**Algorithm:** 1. Compute $d_{\text{core}}$ for all points.
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

- Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$.
- Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.

**Def.** Let $d_{\text{mr}}(p, q) = \max\{ d_{\text{core}}(p), d_{\text{core}}(q), d(p, q) \}$.

**Def.** Mutual reachability graph $G_{\text{mr}}$: complete, edge weights $d_{\text{mr}}$.

**Algorithm:**
1. Compute $d_{\text{core}}$ for all points.
2. Construct $G_{\text{mr}}$ and compute a minimum spanning tree $\mathcal{T}$. 
Mutual reachability

Starting at which value of $\varepsilon$ will these points be in the same cluster?

- 

Both need to be core point, so at least $d_{\text{core}}(p)$ and $d_{\text{core}}(q)$.

Either $\varepsilon \geq d(p, q)$, or they must be connected through other points.

**Def.** Let $d_{\text{mr}}(p, q) = \max\{d_{\text{core}}(p), d_{\text{core}}(q), d(p, q)\}$.

**Def.** Mutual reachability graph $G_{\text{mr}}$: complete, edge weights $d_{\text{mr}}$.

**Algorithm:**  
1. Compute $d_{\text{core}}$ for all points. \(O(n \log n)\) time (Vaidya, 1989)  
2. Construct $G_{\text{mr}}$ and compute a minimum spanning tree $T$.  
2. Construct $G_{mr}$ and compute an MST.

Cannot look at all edges: too slow.
2. Construct $G_{mr}$ and compute an MST.

Cannot look at all edges: too slow.

**Def.** \{p, q\} is a Delaunay edge “iff” there exists a circle with:

- p and q on the boundary
- no points in its interior
2. Construct $G_{mr}$ and compute an MST.

Cannot look at all edges: too slow.

Def. \{p, q\} is a Delaunay edge "iff" there exists a circle with:

- \( p \) and \( q \) on the boundary
- \( \leq k \)
- \( > k \)
- no points in its interior

\( k \)-OD edge
2. Construct $G_{mr}$ and compute an MST.

Cannot look at all edges: too slow.

**Def.** $\{p, q\}$ is a Delaunay edge "iff" there exists a circle with:

- $p$ and $q$ on the boundary
- $\leq k$
- no points in its interior

**Theorem (Gudmundsson, Hammer, v. Kreveld, 2002)** The $k^{th}$-order Delaunay graph has $\Theta(n(k + 1))$ edges and can be computed in $\Theta(n(k + 1) \log n)$ expected time by randomized incremental construction.
2. Construct $G_{mr}$ and compute an MST.

Cannot look at all edges: too slow.

**Def.** \{p, q\} is a Delaunay edge “iff” there exists a circle with:

- p and q on the boundary
- $\leq k$ no points in its interior

**Theorem (Gudmundsson, Hammer, v. Kreveld, 2002)** The $k^{th}$-order Delaunay graph has $O(n(k + 1))$ edges and can be computed in $O(n(k + 1) \log n)$ expected time by randomized incremental construction.

**Claim:** The MST of $G_{mr}$ uses only $k$-OD edges.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.

*Not a $k$-OD edge, so more than $k$ points.*
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.

**Not a $k$-OD edge, so more than $k$ points.**

Pick any point.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:
- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.

Not a $k$-OD edge, so more than $k$ points.

Pick any point.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge $\{p, q\}$, $p$ and $q$ are already in the same cluster, and thus ignores the edge.

Not a $k$-OD edge, so more than $k$ points.

Pick any point.

Recurse until only $k$-OD edges.
The MST of $G_{mr}$ uses only $k$-OD edges.

Consider applying Kruskal’s algorithm to $G_{mr}$:

- Looks at edges in order of increasing cost.
- With weights $d_{mr}$ this corresponds to the HDBSCAN events.

**Claim:** Whenever Kruskal looks at a non-$k$-OD edge \{p, q\}, p and q are already in the same cluster, and thus ignores the edge.

**Not a $k$-OD edge, so more than $k$ points.**

**Pick any point.**

**Recurse until only $k$-OD edges.**

**Kruskal has already considered those edges, so p and q already connected.**
Results

**Everywhere:** $\epsilon$ free, $k$ fixed constant, Euclidean distances

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>dD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSCAN</td>
<td>$\Theta(n \log n)$</td>
<td>$\Theta(n^{2-\lceil d/2 \rceil + 1 + \gamma})$ $\gamma &gt; 0$</td>
</tr>
<tr>
<td>HDBSCAN</td>
<td>$\Theta(n \log n)$ expected</td>
<td>$\times$</td>
</tr>
</tbody>
</table>