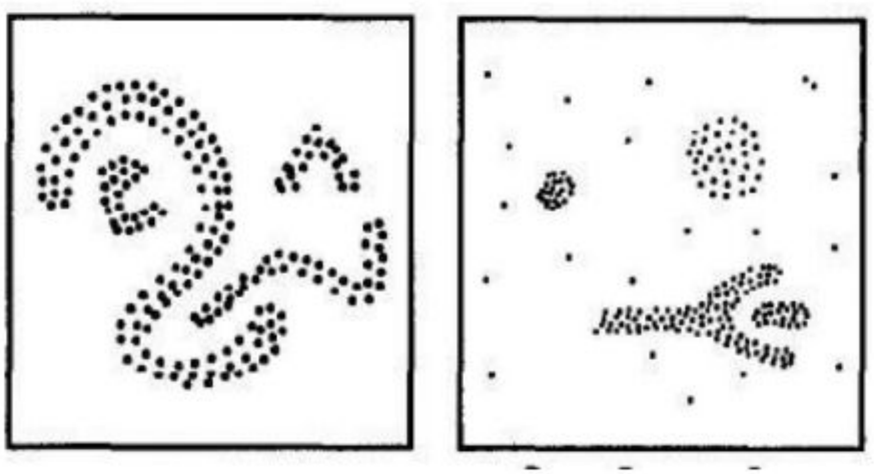
Euclidean DBSCAN

By: Shwetark Patel

Density-based clustering

- Cluster points so that there is a “sparse region” in between clusters
- DBSCAN is a type of density-based clustering



Basic Definitions

- Let B(p,r) be the ball centered at the point p with radius r
- Let dist(p, q) be the Euclidean distance between points p and q
- There are two user supplied inputs to DBSCAN: ϵ and MinPts

Basic Definitions

Definition 2.1. A point $p \in P$ is a **core point** if $B(p, \epsilon)$ covers at least *MinPts* points of P (including p itself).

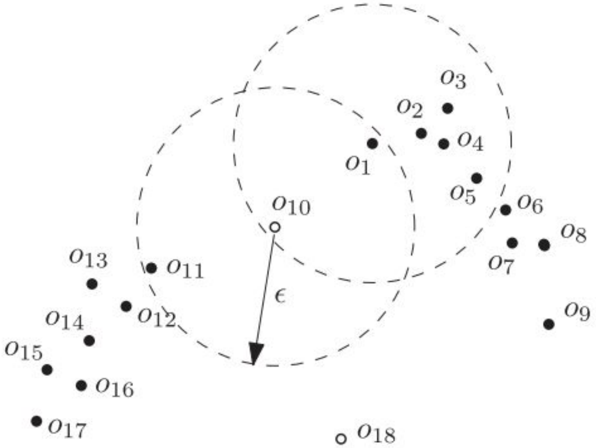


Fig. 2. An example dataset (the two circles have radius ϵ; *MinPts* = 4).

o_1 is a core point

o_{10} is not a core point

Basic Definitions

Definition 2.2. A point $q \in P$ is **density-reachable** from $p \in P$ if there is a sequence of points $p_1, p_2, \ldots, p_t \in P$ (for some integer $t \geq 2$) such that

- $p_1 = p$ and $p_t = q$,
- $p_1, p_2, \ldots, p_{t-1}$ are core points,
- $p_{i+1} \in B(p_i, \epsilon)$ for each $i \in [1, t-1]$.

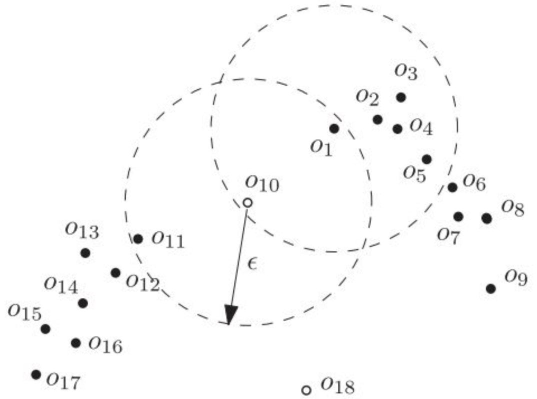


Fig. 2. An example dataset (the two circles have radius ϵ; $MinPts = 4$).

Basic Definitions

Definition 2.3. A **cluster** C is a non-empty subset of P such that

- (Maximality) If a core point $p \in C$, then all the points density-reachable from p also belong to C.
- (Connectivity) For any points $p_1, p_2 \in C$, there is a point $p \in C$ such that both p_1 and p_2 are density-reachable from p.

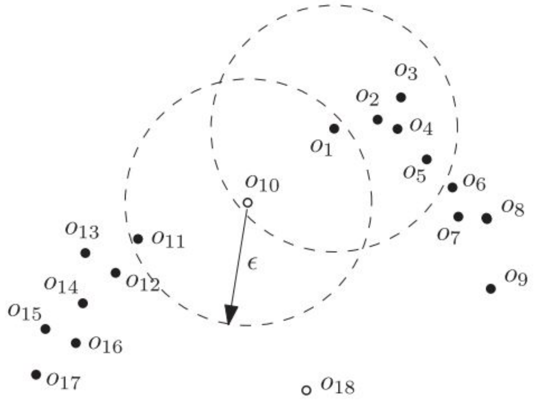


Fig. 2. An example dataset (the two circles have radius ϵ; *MinPts* = 4).

Basic Definitions

Remark. A cluster can contain both core and non-core points. Any non-core point p in a cluster is called a *border point*. Some points may not belong to any clusters at all; they are called *noise points*. In Figure 2, o_{10} is a border point, while o_{18} is noise.

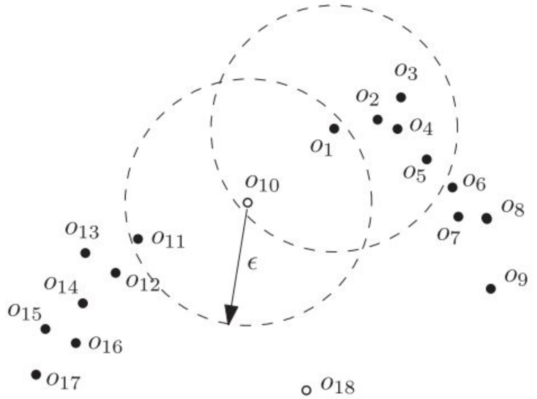


Fig. 2. An example dataset (the two circles have radius ϵ; $MinPts = 4$).

Basic Definitions

PROBLEM 1. *The* **DBSCAN problem** *is to find the unique set* $\mathscr{C}$ *of clusters of* P.

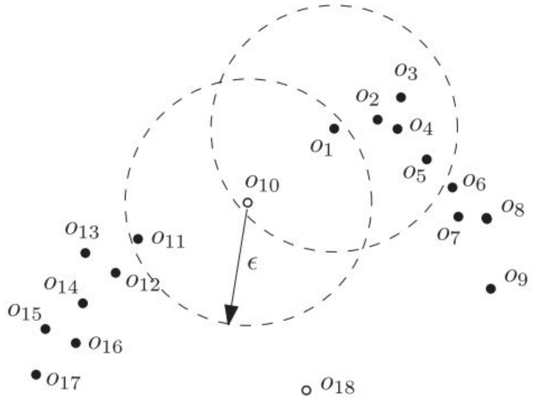


Fig. 2. An example dataset (the two circles have radius ϵ; *MinPts* = 4).

Existing Work

- An O(nlogn) worst case solution to 2D DBSCAN already exists
- However, our algorithm does better on an approximate version of DBSCAN

New DBSCAN Algorithm for d >= 3

- Split up R^d into cells that are hyper-squares with side lengths $\frac{\epsilon}{\sqrt{d}}$
- This ensures that any two points in the same cell are within distance ϵ of each other
- A core cell is a cell that contains at least one core point
- Consider two different cells. They are ϵ-neighbors of each other if the minimum distance between them is less than ϵ.

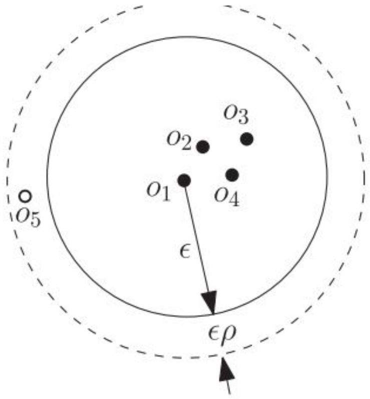
New DBSCAN Algorithm for d >= 3

- Create a graph G where we have a node for each core cell
- Connect two nodes c1 and c2 if the distance between any core point in c1 and any core point in c2 is <= ϵ
- We can do this by going through all ϵ-neighbors of the cell (of which there are a constant number) and using a closest-pairs algorithm
- Compute the connected components of G. Each connected component corresponds to a cluster (excluding border points, which we need to assign by ourselves).

Approximate DBSCAN

Definition 4.1. A point $q \in P$ is **ρ-approximate density-reachable** from $p \in P$ if there is a sequence of points $p_1, p_2, \ldots, p_t \in P$ (for some integer $t \geq 2$) such that

- $p_1 = p$ and $p_t = q$,
- $p_1, p_2, \ldots, p_{t-1}$ are core points, and
- $p_{i+1} \in B(p_i, \epsilon(1+\rho))$ for each $i \in [1, t-1]$.

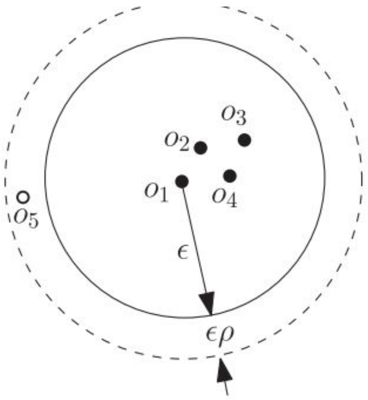


Approximate DBSCAN

Definition 4.2. A **ρ-approximate cluster** C is a non-empty subset of P such that

—(Maximality) If a core point $p \in C$, then all the points density-reachable from p also belong to C.

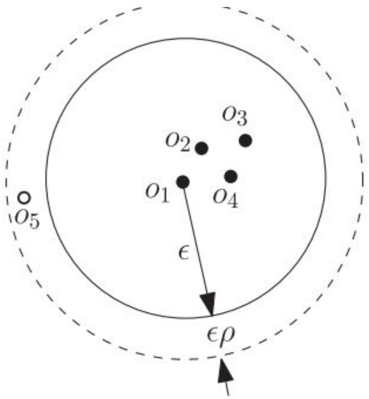
—(ρ-Approximate Connectivity) For any points $p_1, p_2 \in C$, there exists a point $p \in C$ such that both p_1 and p_2 are ρ-approximate density-reachable from p.



o_1, o_2, o_3, o_4 and o_1, o_2, o_3, o_4, o_5 are both valid clusters

Approximate DBSCAN

Problem 2. *The* **ρ-approximate DBSCAN problem** *is to find a set $\mathscr{C}$ of ρ-approximate clusters of P such that every core point of P appears in exactly one ρ-approximate cluster.*

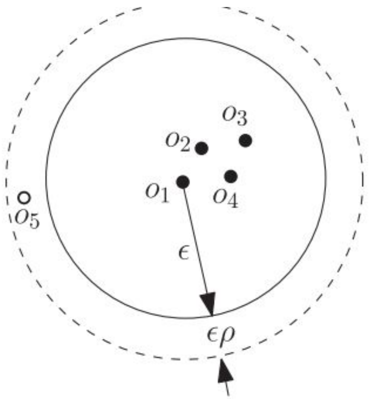


Sandwich Theorem

- Let A be the set of clusters after running DBSCAN with (ϵ, MinPts)
- Let B be the set of clusters after running DBSCAN with (ϵ * (1 + p), MinPts)
- Let C be the set of clusters after running p-Approximate DBSCAN with (ϵ, MinPts)
- Our claim is that C is a result somewhere in between A and B

Approximate Range Counting

Let P still be a set of n points in $\mathbb{R}^d$ where d is a constant. Given any point $q \in \mathbb{R}^d$, a distance threshold $\epsilon > 0$ and an arbitrarily small constant $\rho > 0$, an *approximate range count query* returns an integer that is guaranteed to be between $|B(q, \epsilon) \cap P|$ and $|B(q, \epsilon(1 + \rho)) \cap P|$. For example, in Figure 5, given $q = o_1$, a query may return either 4 or 5.



Approximate Range Counting

LEMMA 4.5. *For any fixed ϵ and ρ, there is a structure of $O(n)$ space that can be built in $O(n)$ expected time, and answers any approximate range count query in $O(1)$ expected time.*

p-Approximate DBSCAN

THEOREM 4.6. *There is a ρ-approximate DBSCAN algorithm that terminates in $O(n)$ expected time, regardless of the value of ϵ, the constant approximation ratio ρ, and the fixed dimensionality d.*

p-Approximate DBSCAN

Algorithm. Our ρ-approximate algorithm differs from the exact algorithm we proposed in Section 3.2 *only* in the definition and computation of the graph G. We re-define $G = (V, E)$ as follows:

- As before, each vertex in V is a core cell of the grid T (remember that the algorithm of Section 3.2 imposes a grid T on $\mathbb{R}^d$, where a cell is a core cell if it covers at least one core point).
- Given two different core cells c_1, c_2, whether E has an edge between c_1 and c_2 obeys the rules below:
  - yes, if there exist core points p_1, p_2 in c_1, c_2, respectively, such that $dist(p_1, p_2) \leq \epsilon$;
  - no, if no core point in c_1 is within distance $\epsilon(1 + \rho)$ from any core point in c_2;
  - *don't care*, in all the other cases.

p-Approximate DBSCAN

- To generate the edges of a core cell c_1, we examine each ϵ-neighbor cell c_2 of c_1, in turn.
- For every core point p in c_1, do an approximate range count query on the set of core points in c_2.
- If the query returns a non-zero answer, add an edge (c_1,c_2) to G

p-Approximate DBSCAN

- In all the cases where we have to add an edge (there is a point in c_2 within the ϵ-radius circle of the point in c_1), an edge is added
- In all the cases where we can't add an edge (there isn't a point in c_2 within the ϵ(1+p)-radius circle of the point in c_1), an edge is not added

Time complexity

- For each core point of a cell c_1, we issue an approximate range count query for each ϵ-neighbor cell c_2
- There are O(1) ϵ-neighbor cells for each cell c_1, so we issue O(n) approximate range count queries
- Each approximate range count query is O(1) expected time, so O(n) expected time overall

Heuristics

- Instead of maintaining the graph G, just maintain connected components using union find
- Store all non-empty ϵ-neighbors of a cell in a list after computing them for the first time (acts kind of like a cache)

Another interesting result

- For 2D DBSCAN, if n data points have been presorted on each dimension, then 2D DBSCAN can be solved in O(n) time

Performance

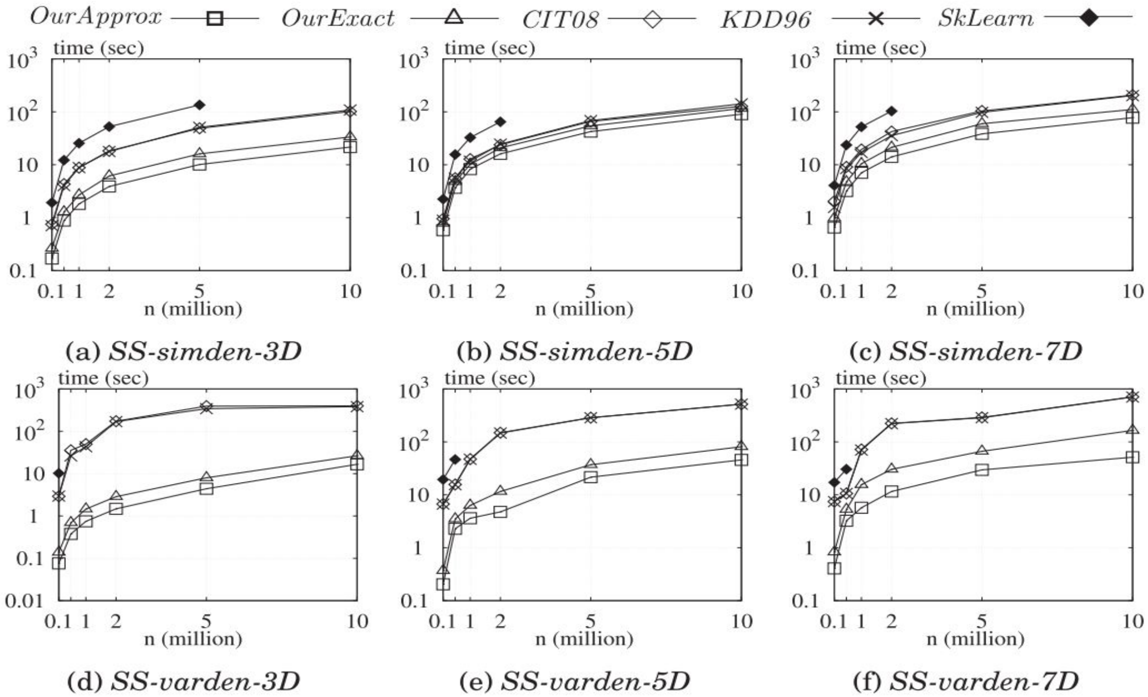


Fig. 19. Running time vs. n ($d \geq 3$).

Performance

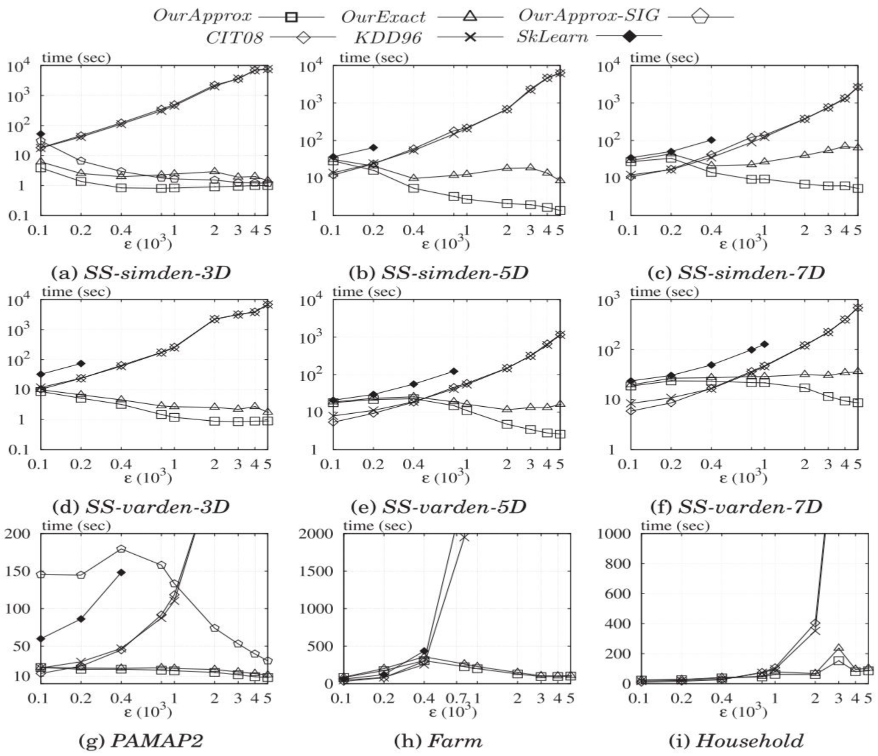


Fig. 18. Running time vs. ϵ ($d \geq 3$).

Performance

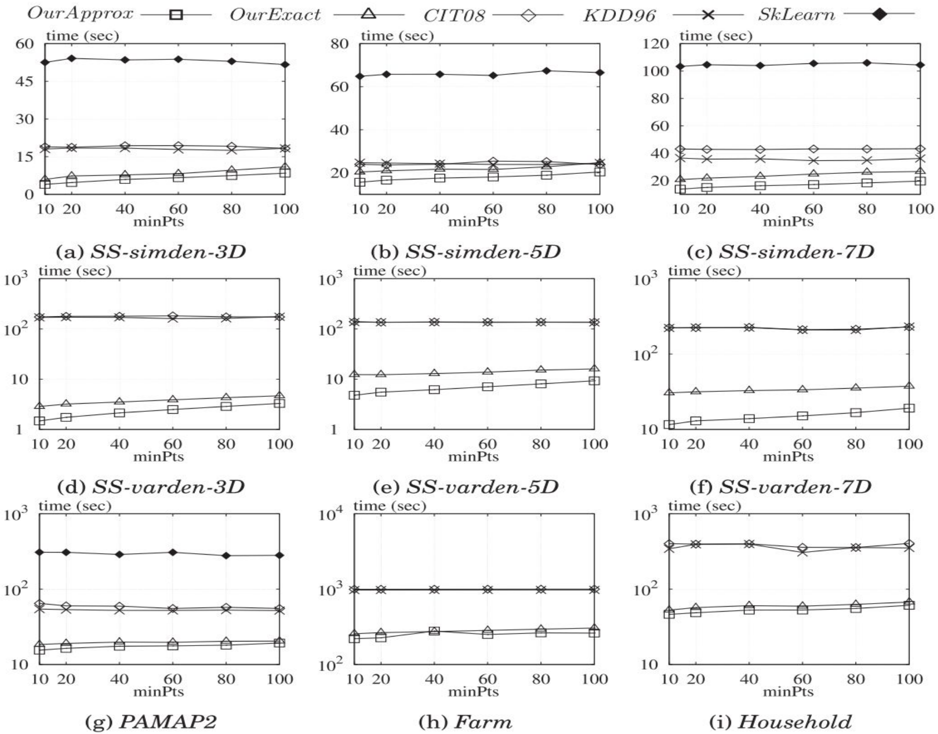


Fig. 21. Running time vs. *MinPts* ($d \geq 3$).

Strengths

- Better time complexity (O(N) expected time) and produces results very similar to regular DBSCAN

Weakness

- We don't have a fixed number of clusters based on ϵ and MinPts, while we do in k-means clustering.

Discussion Questions

- When is it best to use K-means clustering over DBSCAN, and vice versa?
- How does changing ϵ and MinPts generally change our output clusters?