Chapter 7. Cluster Analysis

Dong-Kyu Chae

PI of the Data Intelligence Lab @HYU
Department of Computer Science & Data Science
Hanyang University

Data
Intelligence
LAB

Data Intelligence LAB

DBSCAN Algorithm

ALGORITHM 1: Pseudocode of Original Sequential DBSCAN Algorithm

outlier or noise


```
Input: DB: Database
Input: ε: Radius
Input: minPts: Density threshold
Input: dist: Distance function
Data: label: Point labels, initially undefined
foreach point p in database DB do                          // Iterate over every point
    if label(p) ≠ undefined then continue                  // Skip processed points
    Neighbors N ← RangeQuery(DB, dist, p, ε)               // Find initial neighbors
    if |N| < minPts then                                   // Non-core points are noise
        label(p) ← Noise
        continue
    c ← next cluster label                                 // Start a new cluster
    label(p) ← c
    Seed set S ← N \ {p}                                   // Expand neighborhood
    foreach q in S do
        if label(q) = Noise then label(q) ← c
        if label(q) ≠ undefined then continue
        Neighbors N ← RangeQuery(DB, dist, q, ε)
        label(q) ← c
        if |N| < minPts then continue                      // Core-point check
        S ← S ∪ N
```

→ not core point 란 뜻

C1→C2, ...

exclude

→ miss given noise → border

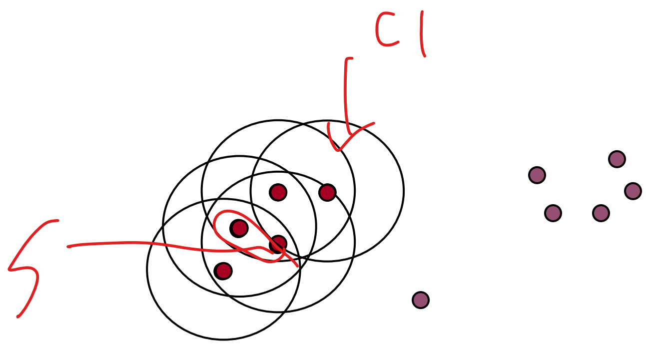
→ border point initially undefined → border

expand the seed set

DBSCAN Algorithm: Example

❑ Parameter

- ε = 2 cm
- *MinPts* = 3

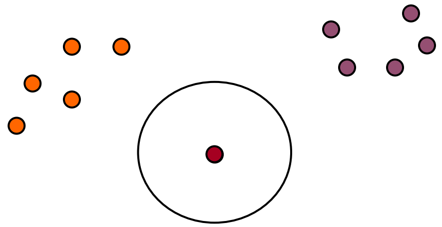


for each $o \in D$ **do**
    if o is not yet marked **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

Parameter

- ε = 2 cm
- $MinPts$ = 3

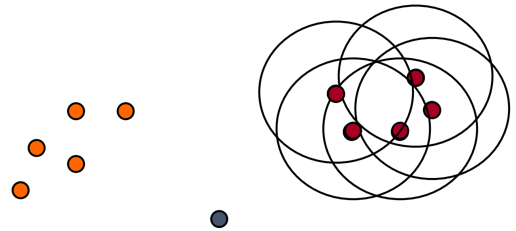


for each $o \in D$ **do**
    if o is not yet marked **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

Parameter

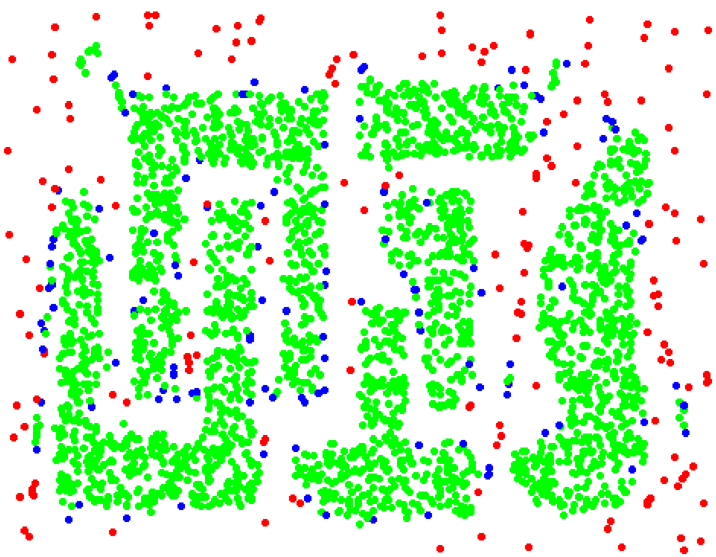
- ε = 2 cm
- *MinPts* = 3



for each $o \in D$ **do**
 if o is not yet marked **then**
  if o is a core-object **then**
   collect all objects density-reachable from o
   and assign them to a new cluster.
  else
   assign o to NOISE

Point Types Marked by DBSCAN

Original Points

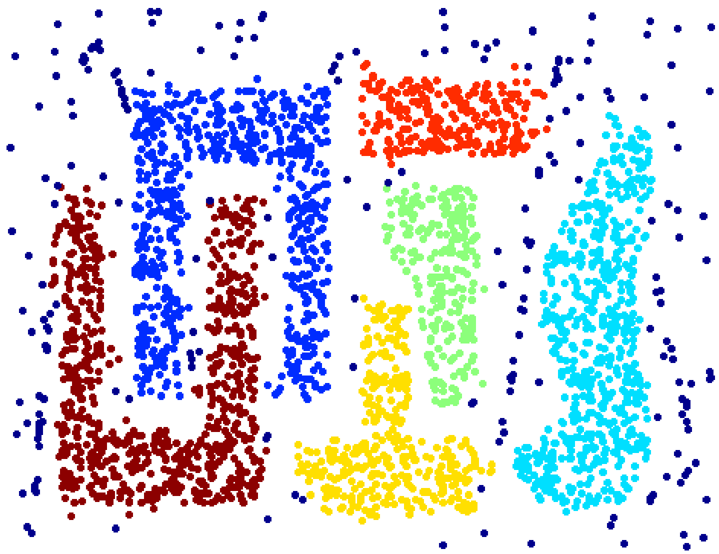


Point types: core, green
border and outliers

$\varepsilon = 10$, MinPts = 4

When DBSCAN Works Well

Original Points

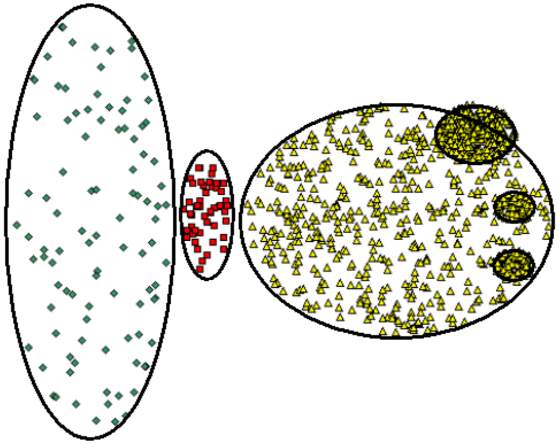


Clusters

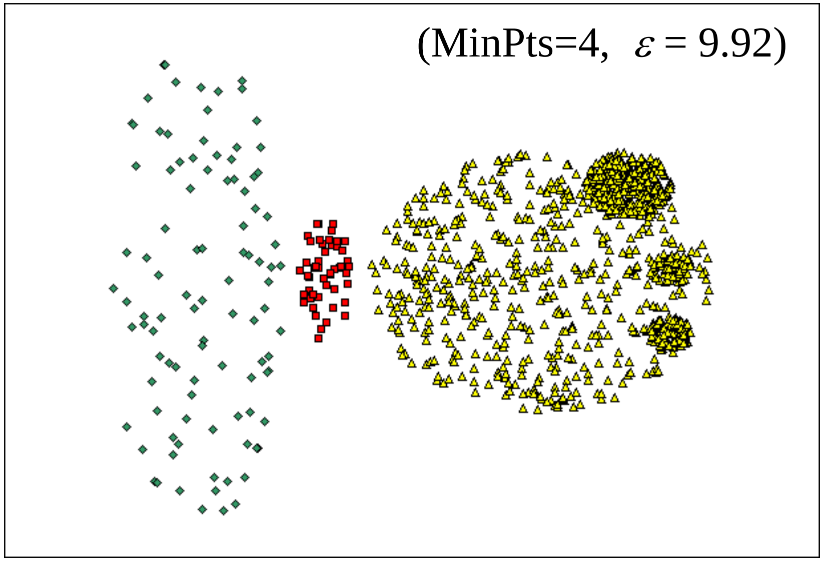
- Resistant to Noise
- Can handle clusters of different shapes and sizes

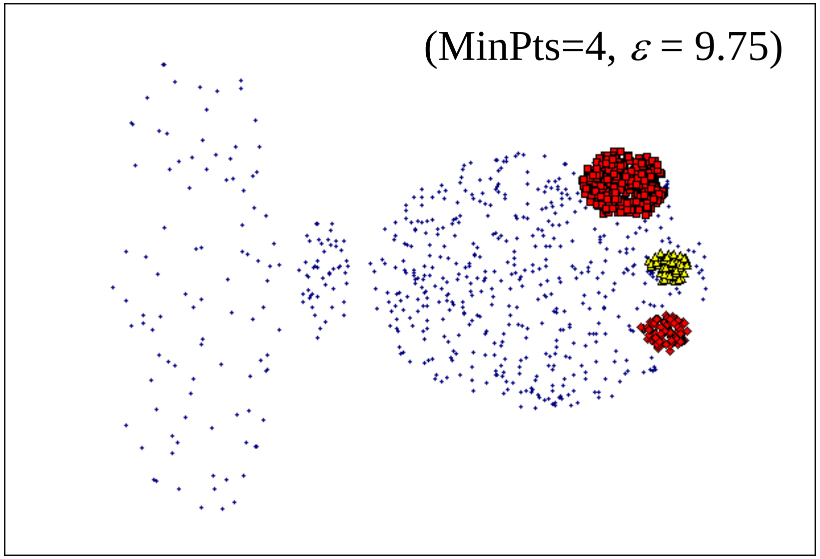
Data Intelligence LAB

DBSCAN: Sensitive to Parameters



Original Points





DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

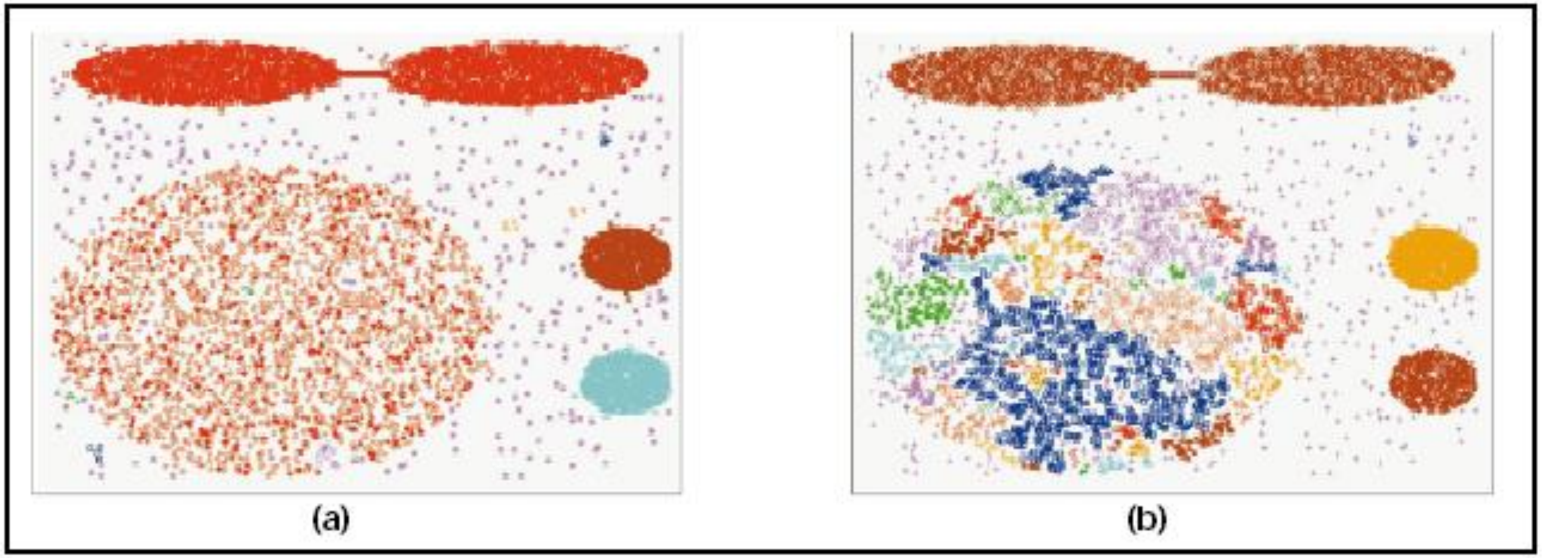
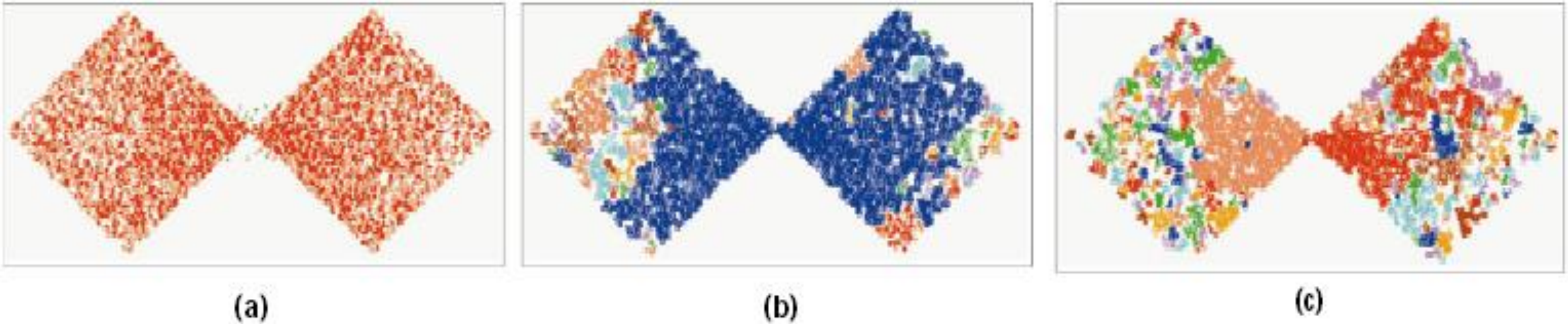


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

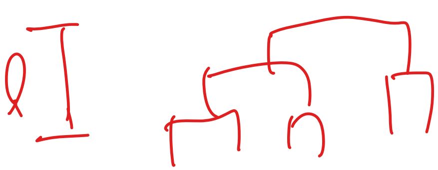


DBSCAN Summary

- **Advantages**
  - Clusters can have arbitrary shape and size
  - Number of clusters is determined automatically
  - Can separate clusters from noise and outliers
- **Disadvantages**
  - Input parameters may be difficult to determine
  - In some situations very sensitive to input parameter setting
- **OPTICS**
  - Based on DBSCAN
  - Does not produce clusters explicitly
  - Rather generate **an ordering of data objects** representing density-based clustering structure

OPTICS: Some Extension from DBSCAN

- **OPTICS: Ordering Points To Identify the Clustering Structure**
  - It aims to answer: *"how to choose proper ε value?"*
  - Produces a linear **order** of objects such that **spatially closest points become neighbors in the ordering**
  - This ordering can produce a **graphical information** equivalent to density-based clustering structure corresponding to a broad range of parameter settings (ε)
  - OPTICS can be seen as a **visualization technique** for clustering, rather than a clustering solution
  - Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure



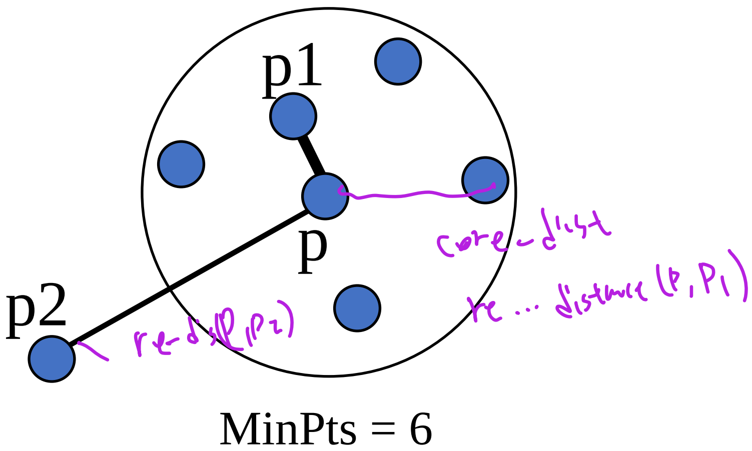
OPTICS: Some Extension from DBSCAN

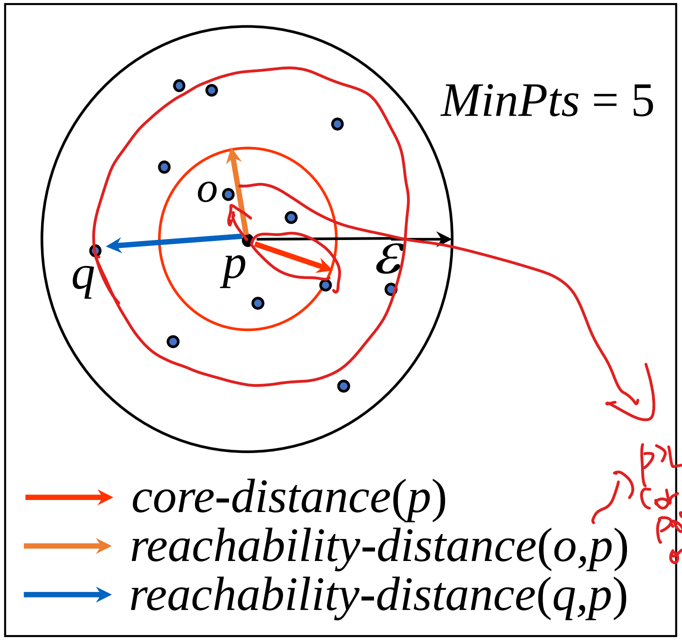
- $$\text{core-dist}_{\varepsilon,MinPts}(p) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(p)| < MinPts \\ MinPts\text{-th smallest distance in } N_\varepsilon(p) & \text{otherwise} \end{cases}$$

  "smallest distance that makes *p* a core"

- $$\text{reachability-dist}_{\varepsilon,MinPts}(o,p) = \begin{cases} \text{UNDEFINED} & \text{if } |N_\varepsilon(p)| < MinPts \\ \max(\text{core-dist}_{\varepsilon,MinPts}(p), \text{dist}(p,o)) & \text{otherwise} \end{cases}$$

  "smallest distance that makes *o* directly density-reachable from *p*"





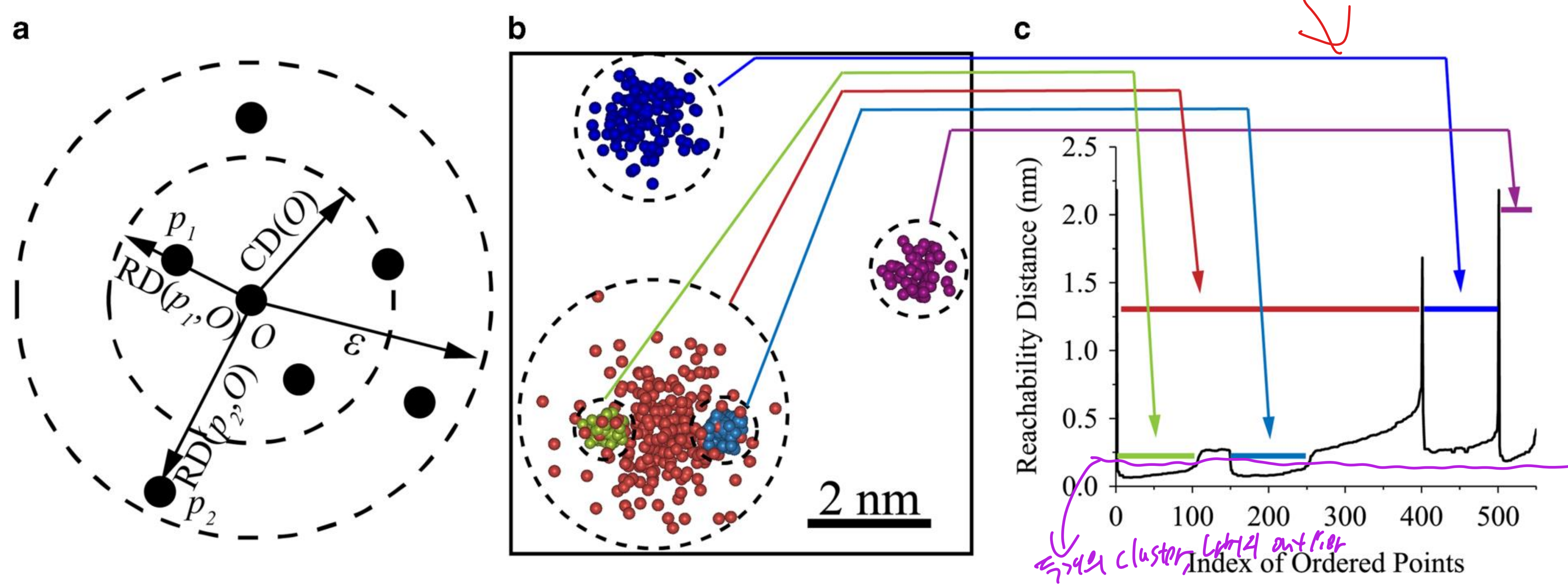
p가 core point 여야 하니까

OPTICS: Some Extension from DBSCAN

detail skip

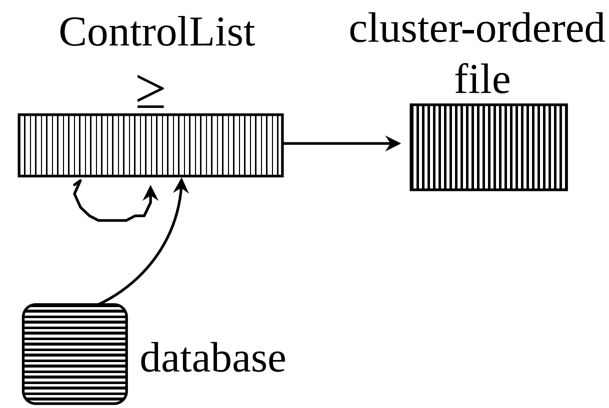
❑ Idea

- ❑ Order data points by ***shortest reachability distance***
- ❑ The clusters show up as **valleys** in the **reachability plot**. The deeper the valley, the denser the cluster (having shorter reachability distances).
- ❑ Extracting clusters from this plot can be done manually by selecting a threshold on the y-axis
- ❑ Needs an initial ε*, but it does not require a careful decision.



The Pseudocode of OPTICS (skip)

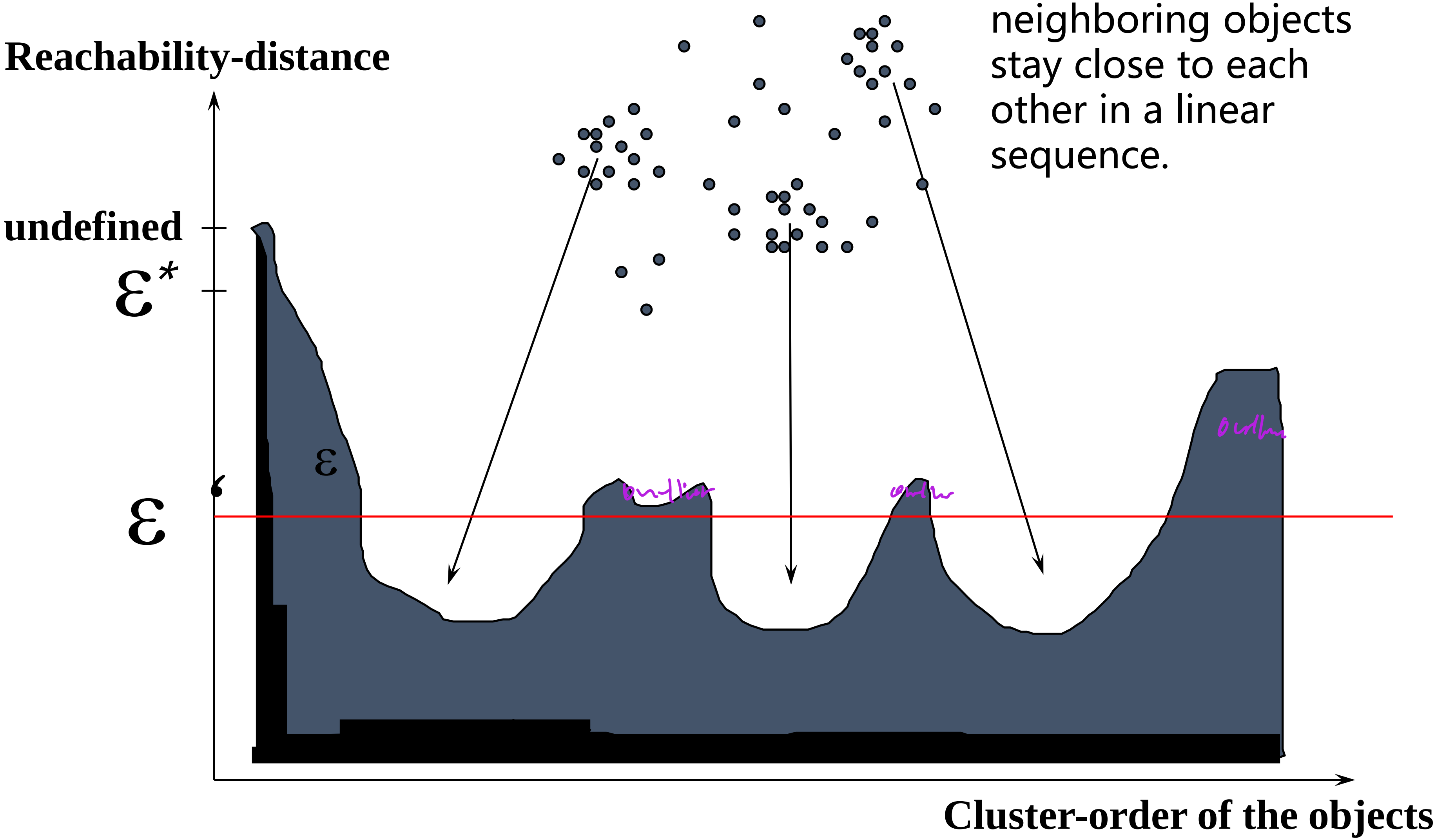
❑ Output: ordered data points

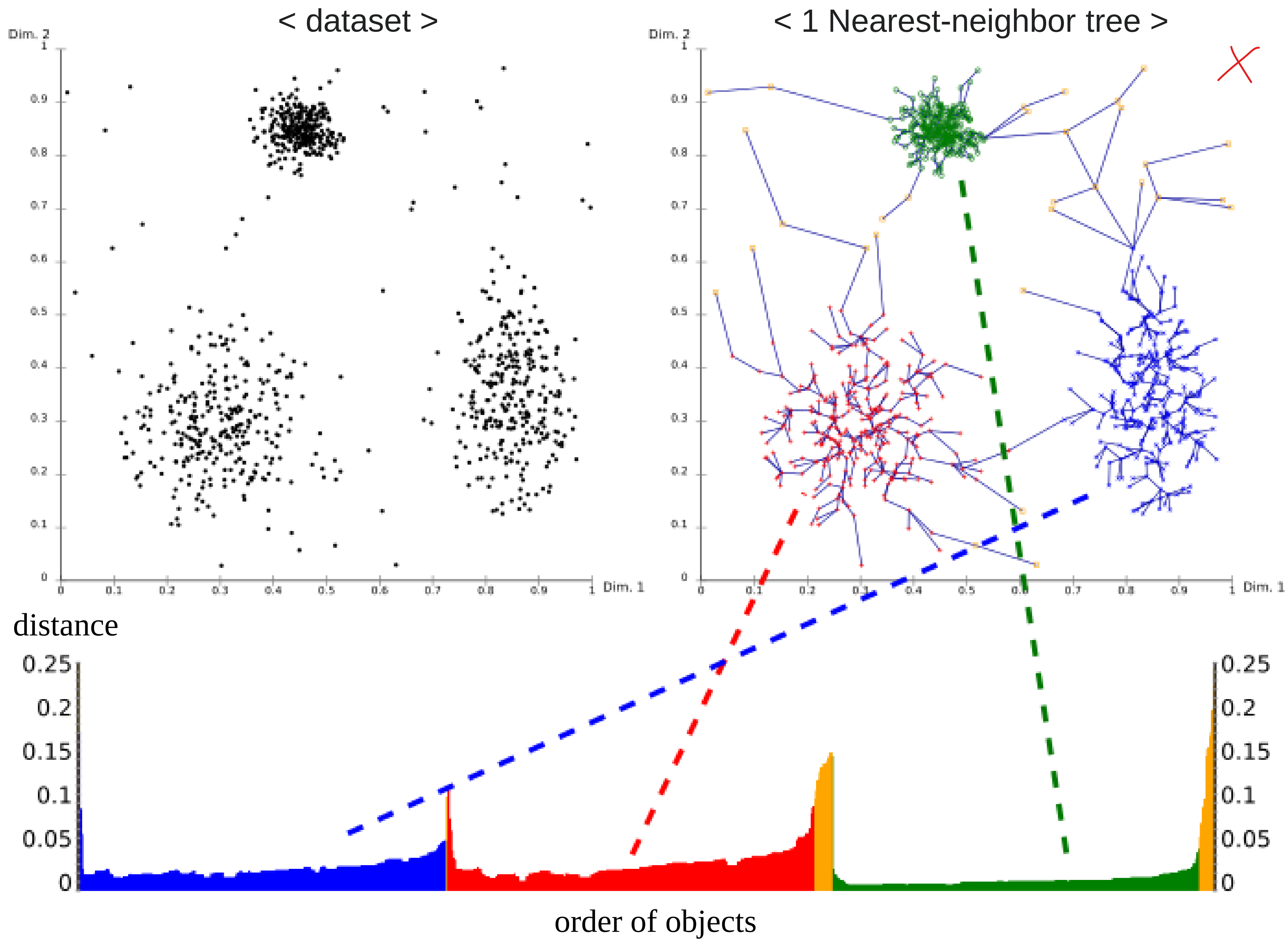


foreach $o \in$ Database
  // initially, o.processed = false for all objects o
  if o.processed = false;
    insert (o, *"undefined"*) into *ControlList;*
  while *ControlList* is not empty
      select first element (o, *r-dist*) from *ControlList*;
      retrieve $N_\varepsilon(o)$ and determine *c_dist= core-distance*(o);
      set o.processed = true;
      write (o, *r_dist*, *c_dist*) to file;
      if o is a core object at any distance $\leq \varepsilon$
        foreach $p \in N_\varepsilon(o)$ not yet processed;
          determine r_dist_p = *reachability-distance*(p, o);
          if (p, _) $\notin$ *ControlList*
            insert (p, r_dist_p) in *ControlList;*
          else if (p, *old_r_dist*) $\in$ *ControlList* **and** $r\_dist_p < old\_r\_dist$
            update (p, r_dist_p) in *ControlList*;

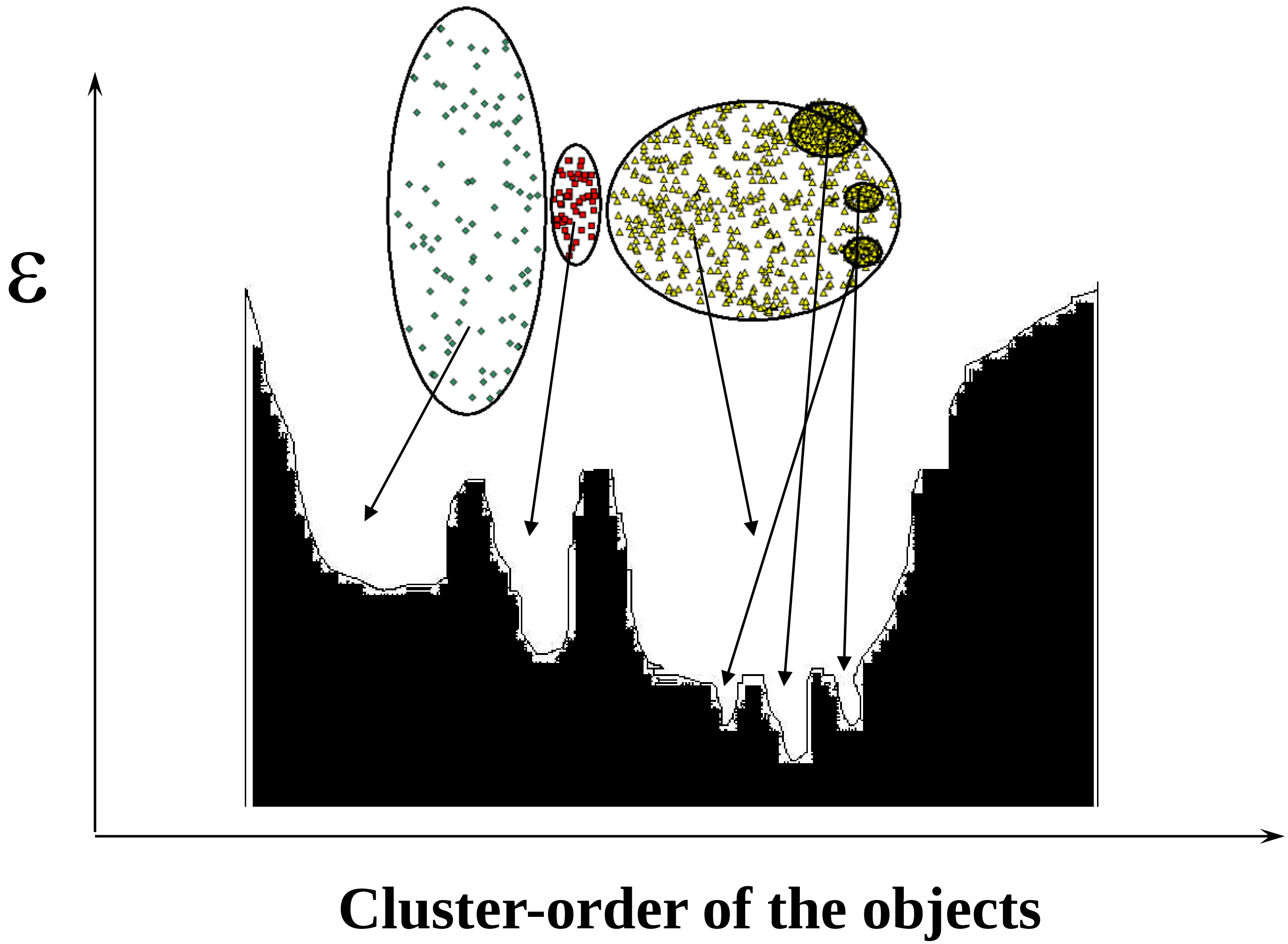
OPTICS: The Reachability Plot

neighboring objects stay close to each other in a linear sequence.



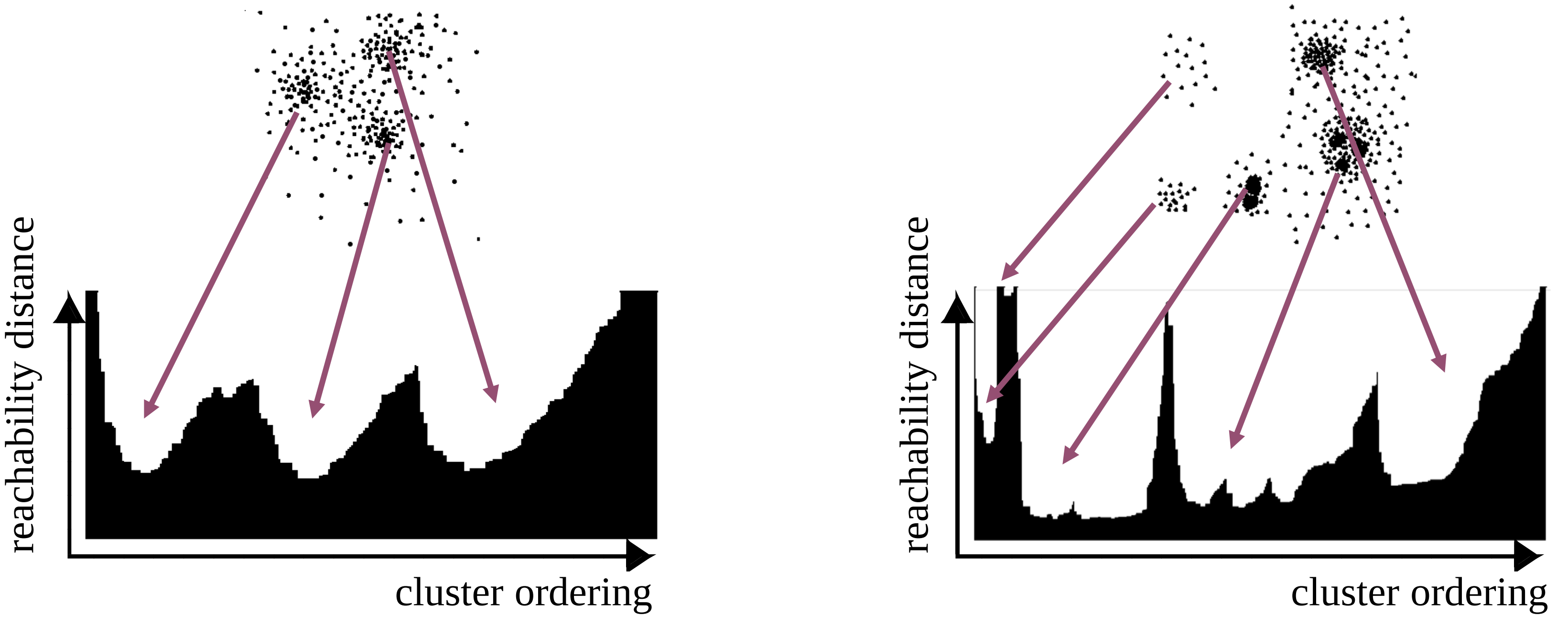

< dataset >
< 1 Nearest-neighbor tree >
Dim. 2
Dim. 1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
distance
0.25
0.2
0.15
0.1
0.05
0
order of objects

OPTICS: The Reachability Plot



OPTICS: The Reachability Plot

- **Represents the density-based clustering structure**
- **Easy to analyze**
- **Independent of the dimension of the data**



DBSCAN VS OPTICS

	DBSCAN	OPTICS
Density	Boolean value (high/low) border or outlier	Numerical value (core distance)
Density-connected	Boolean value (yes/no)	Numerical value (reachability distance)
Searching strategy	random	greedy

Thank You

Data
Intelligence
Lab