Density-based Approaches

- Why Density-Based Clustering methods?
  - Discover clusters of arbitrary shape.
  - Clusters – Dense regions of objects separated by regions of low density
- DBSCAN – the first density based clustering
- OPTICS – density based cluster-ordering
- DENCLUE – a general density-based description of cluster and clustering

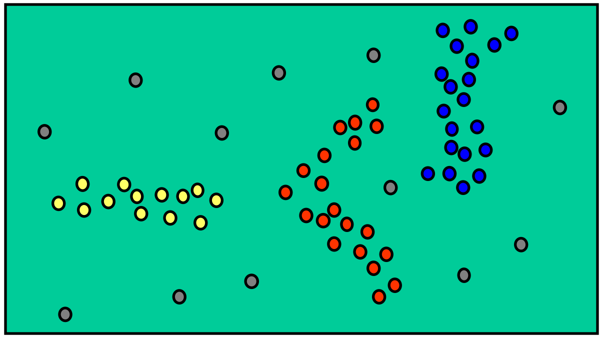
DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Proposed by Ester, Kriegel, Sander, and Xu (KDD96)
- Relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points.
- Discovers clusters of arbitrary shape in spatial databases with noise

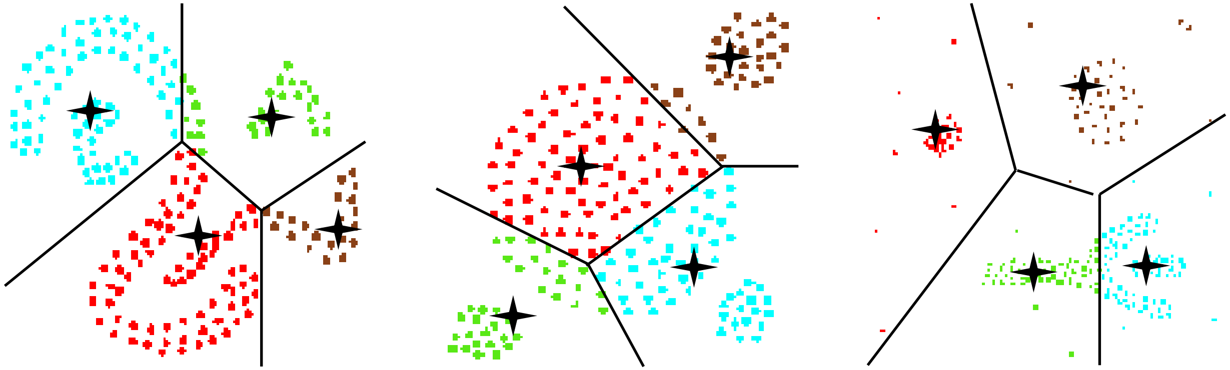
Density-Based Clustering

- *Basic Idea*:

  Clusters are dense regions in the data space, separated by regions of lower object density



Why Density-Based Clustering?



Results of a *k*-medoid algorithm for *k*=4

Different density-based approaches exist (see Textbook & Papers)
Here we discuss the ideas underlying the DBSCAN algorithm

Density Based Clustering: Basic Concept

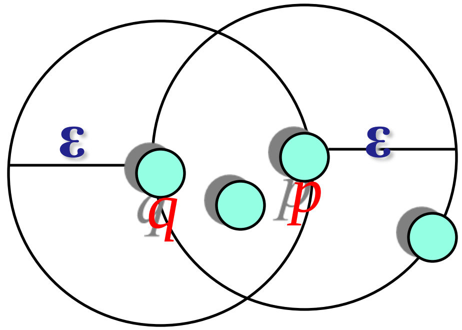
- **Intuition for the formalization of the basic idea**
  - **For any point in a cluster, the local point density around that point has to exceed some threshold**
  - **The set of points from one cluster is spatially connected**
- **Local point density at a point *p* defined by two parameters**
  - ε – **radius for the neighborhood of point p:** $N_\varepsilon(p) := \{q \text{ in data set } D \mid dist(p, q) \leq \varepsilon\}$
  - *MinPts* – **minimum number of points in the given neighbourhood** $N(p)$

ε-Neighborhood

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p):\{q \mid d(p,q) \le \varepsilon\}$$

- "High density" - ε-Neighborhood of an object contains at least *MinPts* of objects.

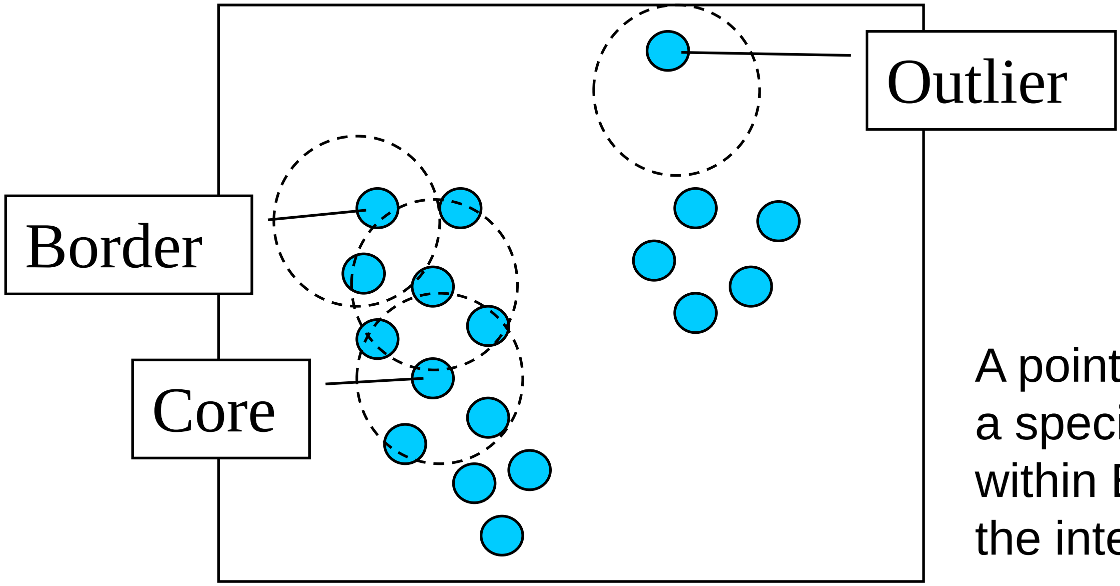


ε-Neighborhood of *p*
ε-Neighborhood of *q*

Density of p is "high" (MinPts = 4)

Density of q is "low" (MinPts = 4)

Core, Border & Outlier



Given ε and *MinPts*, categorize the objects into three exclusive groups.

A point is a core point if it has more than a specified number of points (MinPts) within Eps These are points that are at the interior of a cluster.

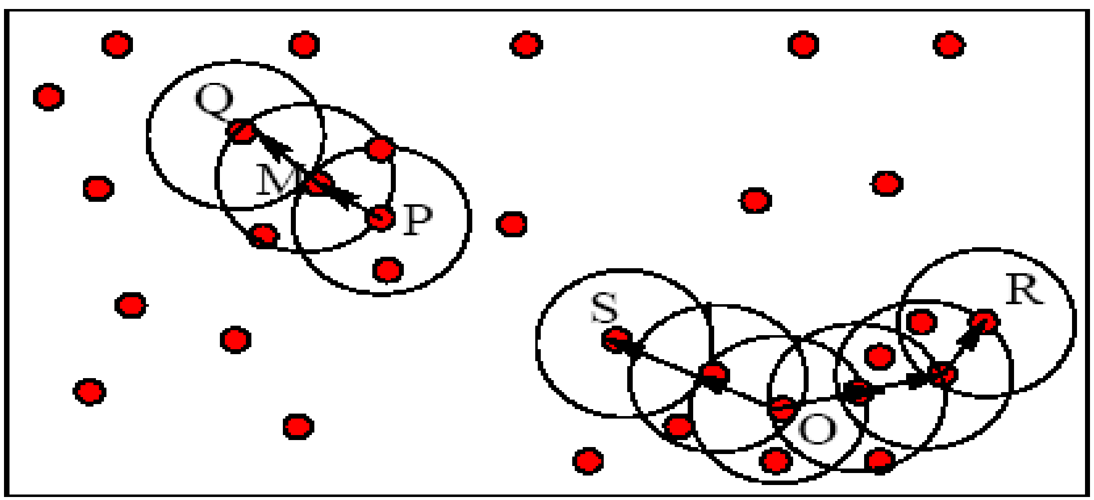
ε = 1unit, MinPts = 5

A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

Example

- M, P, O, and R are core objects since each is in an Eps neighborhood containing at least 3 points

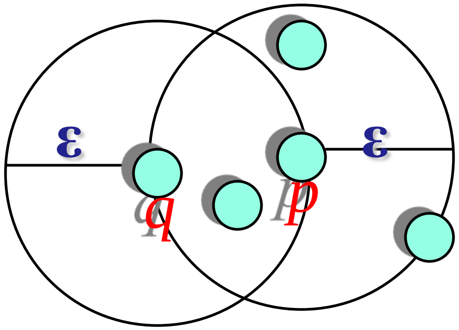


Minpts = 3

Eps=radius of the circles

Density-Reachability

- Directly density-reachable
  - An object q is directly density-reachable from object p if p is a core object and q is in p's ε-neighborhood.

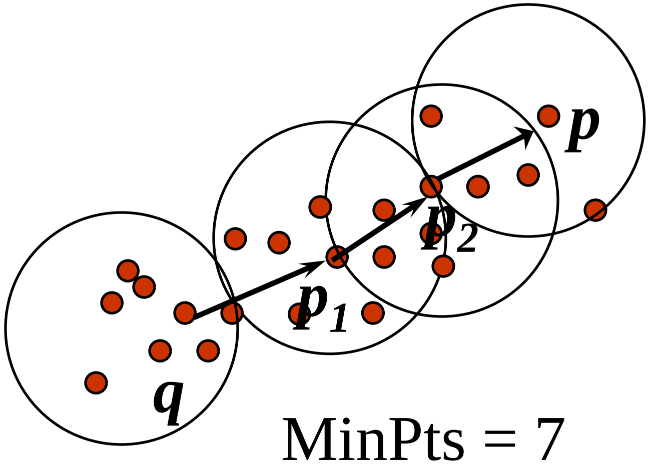


MinPts = 4

- q is directly density-reachable from p
- p is not directly density- reachable from q?
- Density-reachability is asymmetric.

Density-reachability

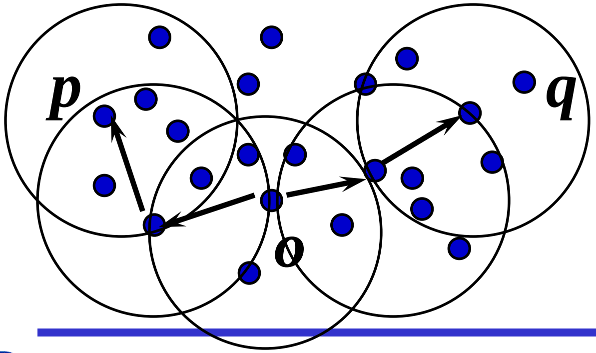
- **Density-Reachable (directly and indirectly):**
  - A point p is directly density-reachable from p2;
  - p2 is directly density-reachable from p1;
  - p1 is directly density-reachable from q;
  - p←p2←p1←q form a chain.



- **p is (indirectly) density-reachable from q**
- **q is not density- reachable from p?**

Density-Connectivity

- **Density-reachable is not symmetric**
  - **not good enough to describe clusters**
- **Density-Connected**
  - **A pair of points p and q are density-connected if they are commonly density-reachable from a point o.**



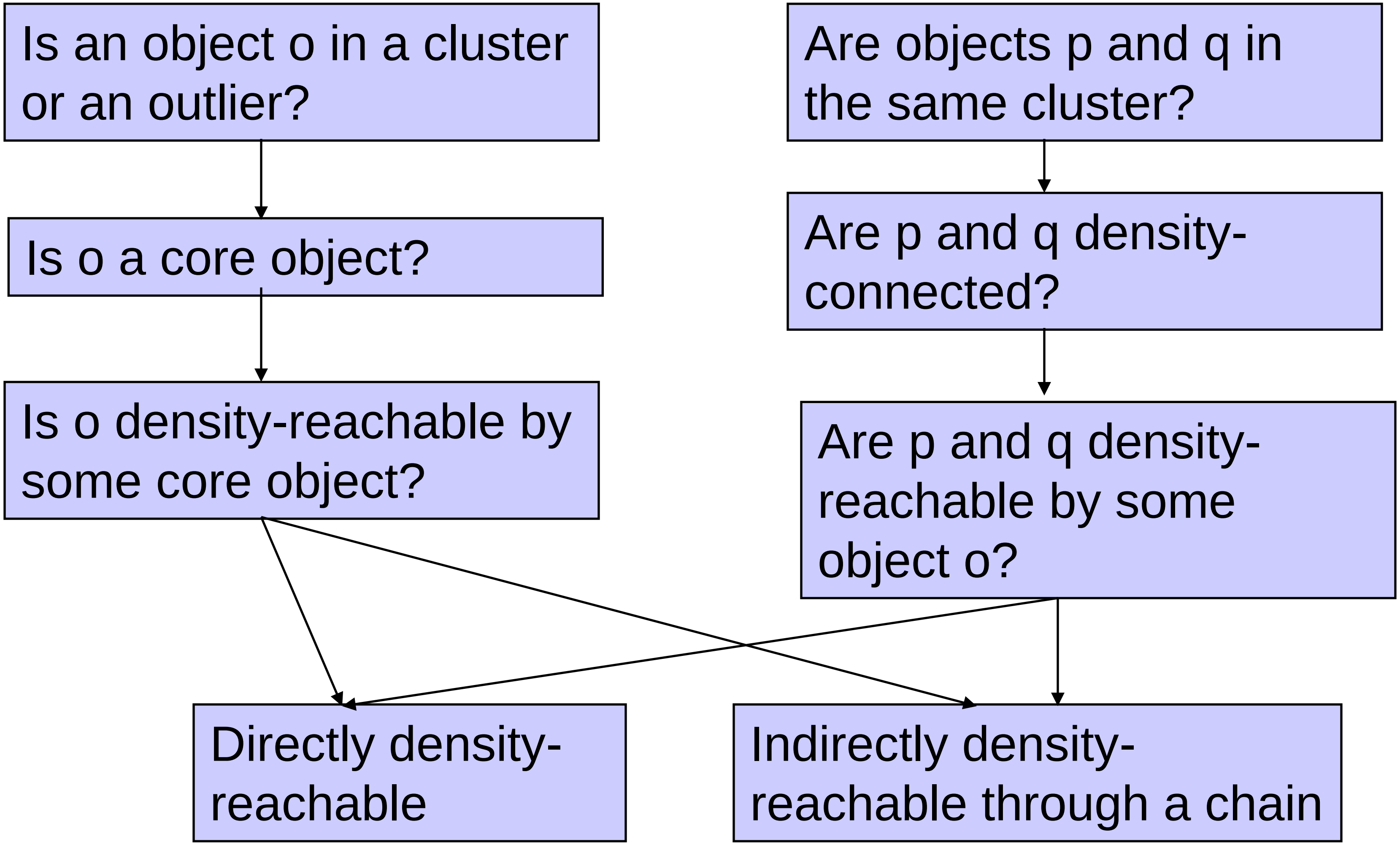
- **Density-connectivity is symmetric**

Formal Description of Cluster

- Given a data set D, parameter ε and threshold MinPts.
- A cluster C is a subset of objects satisfying two criteria:
  - *Connected:* $\forall$ p,q $\in$C: p and q are density-connected.
  - *Maximal:* $\forall$ p,q: if p $\in$C and q is density-reachable from p, then q $\in$C. **(avoid redundancy)**

P is a core object.

Review of Concepts



DBSCAN Algorithm

```
Input: The data set D

Parameter: ε, MinPts

For each object p in D
    if p is a core object and not processed then
        C = retrieve all objects density-reachable from p
        mark all objects in C as processed
        report C as a cluster
    else mark p as outlier
    end if

End For
```

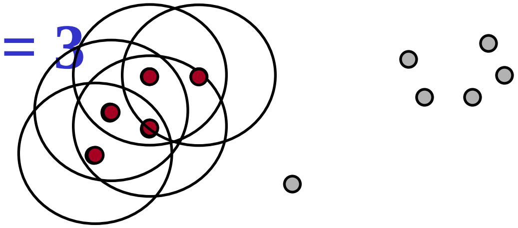
DBScan Algorithm

DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* wrt *Eps* and *MinPts*.
- If *p* is a core point, a cluster is formed.
- If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

DBSCAN Algorithm: Example

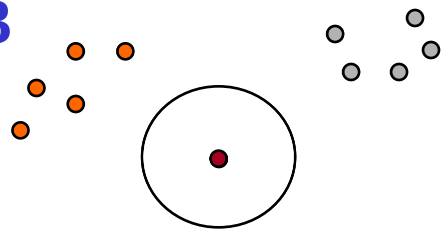
- Parameter
  - $\varepsilon = 2$ cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **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

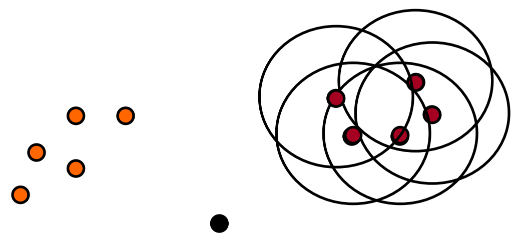
- Parameter
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
  if o is not yet classified **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

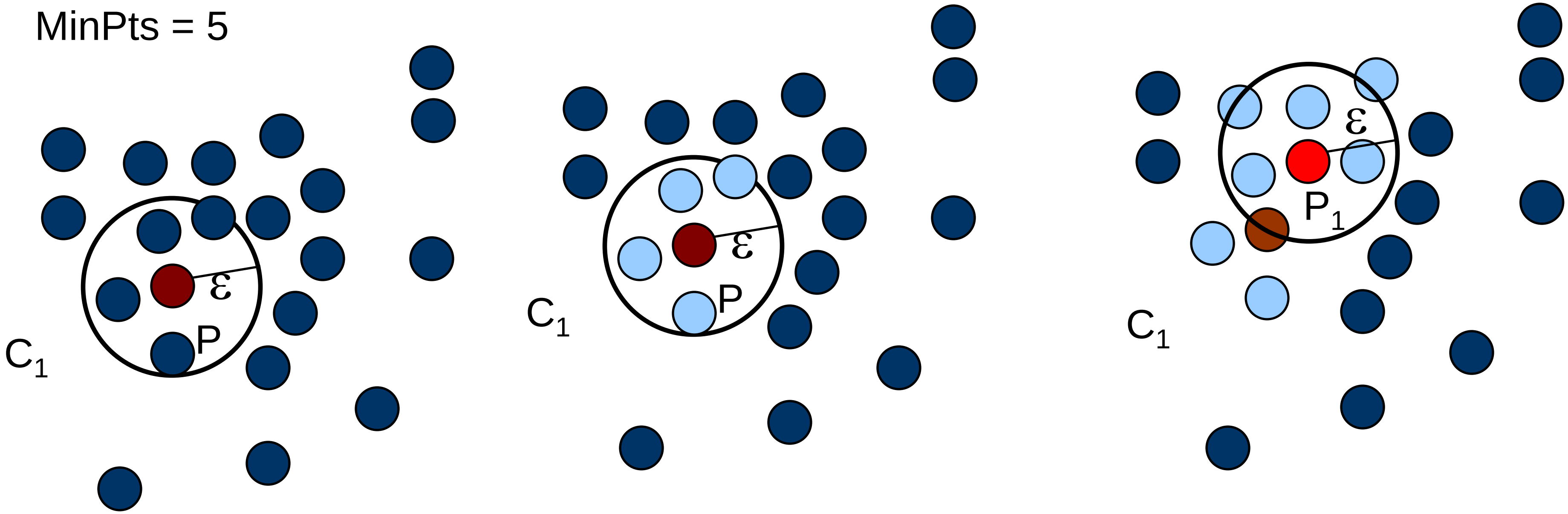
DBSCAN Algorithm: Example

- **Parameter**
  - $\varepsilon = 2$ cm
  - *MinPts* = 3



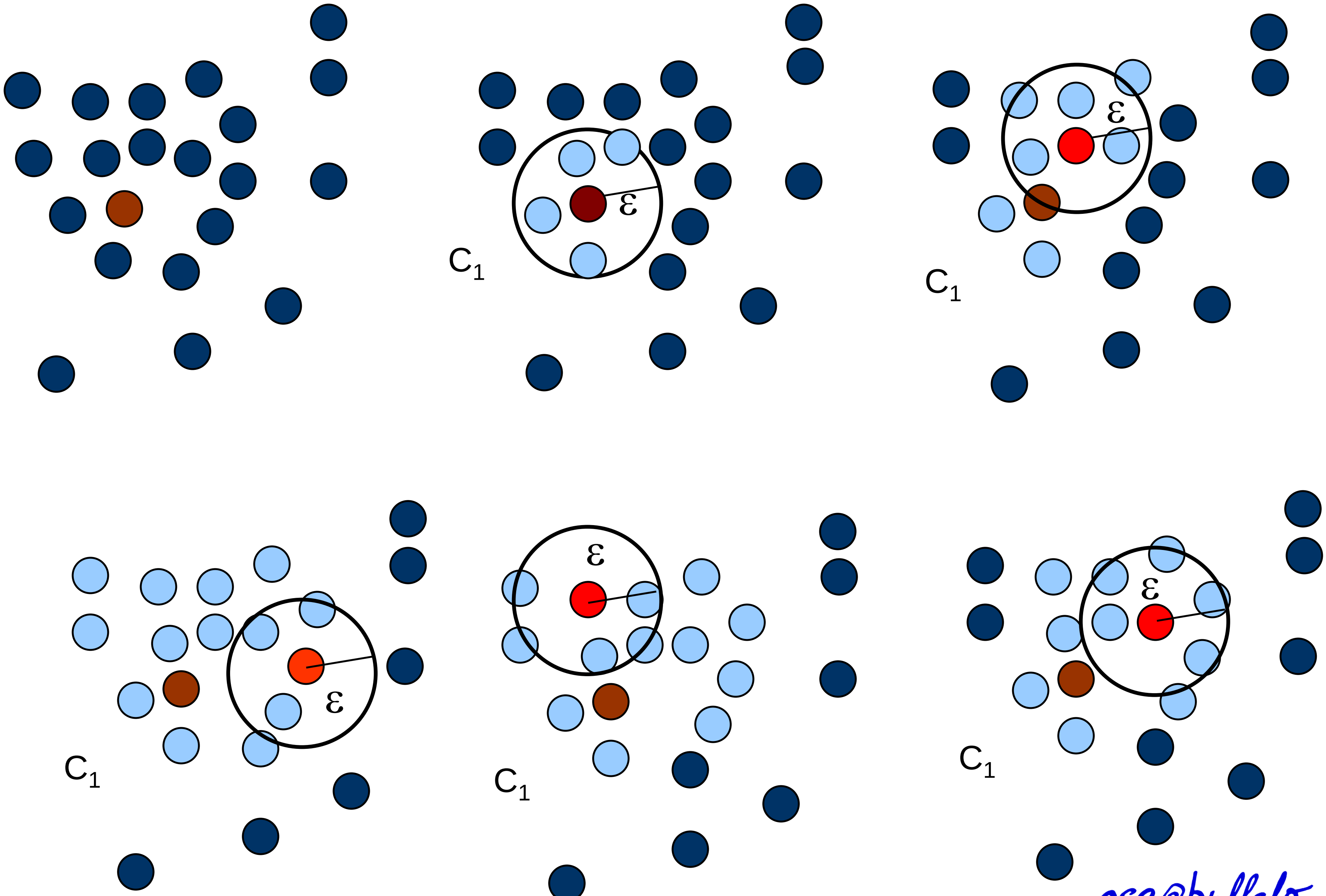
for each $o \in D$ **do**
  if o is not yet classified **then**
    if o is a core-object **then**
      collect all objects density-reachable from o
      and assign them to a new cluster.
    else
      assign o to NOISE

MinPts = 5



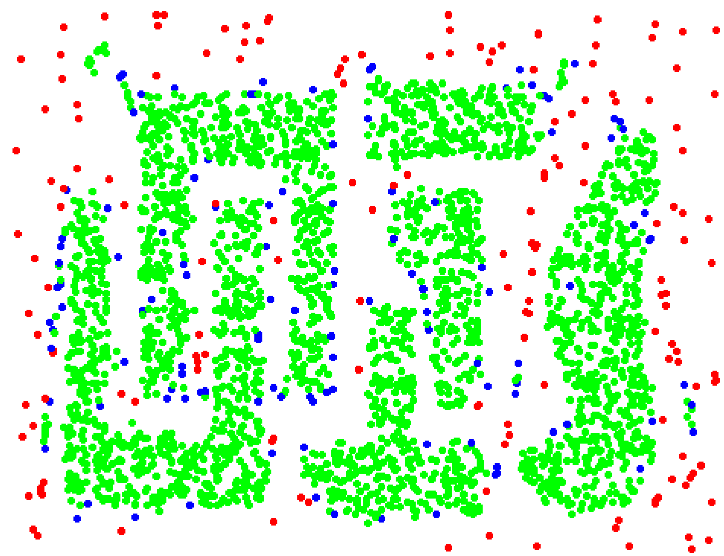
1. Check the ε-neighborhood of p;
2. If p has less than MinPts neighbors then mark p as outlier and continue with the next object
3. Otherwise mark p as processed and put all the neighbors in cluster C

1. Check the unprocessed objects in C
2. If no core object, return C
3. Otherwise, randomly pick up one core object p_1, mark p_1 as processed, and put all unprocessed neighbors of p_1 in cluster C


ε
C_1
ε
C_1
ε
C_1
ε
C_1
ε
C_1

Example

Original Points

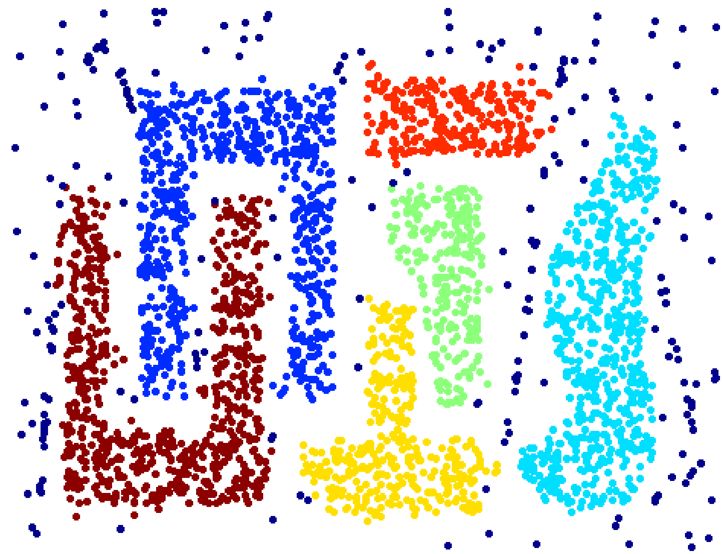


Point types: core, border and outliers

ε = 10, MinPts = 4

When DBSCAN Works Well

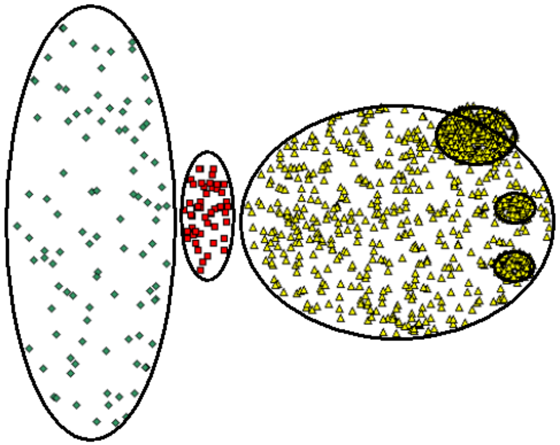
Original Points



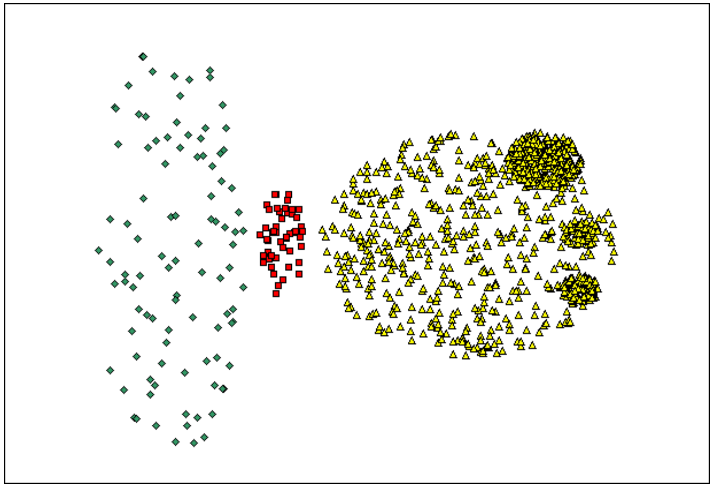
Clusters

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

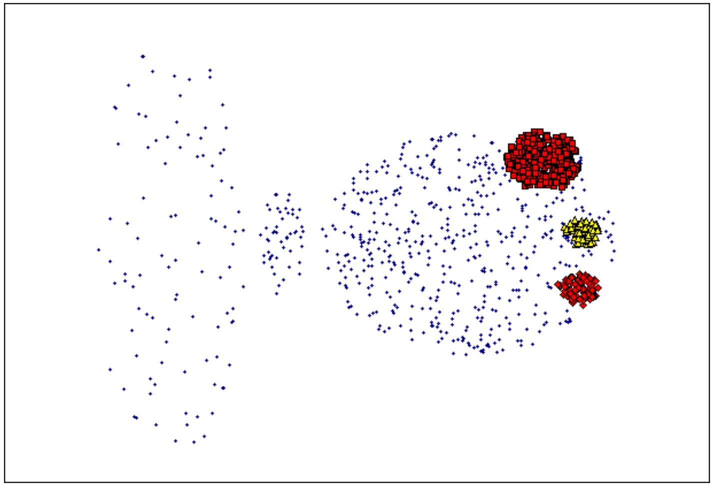
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

- **Cannot handle Varying densities**
- **sensitive to parameters**

DBSCAN: Sensitive to Parameters

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

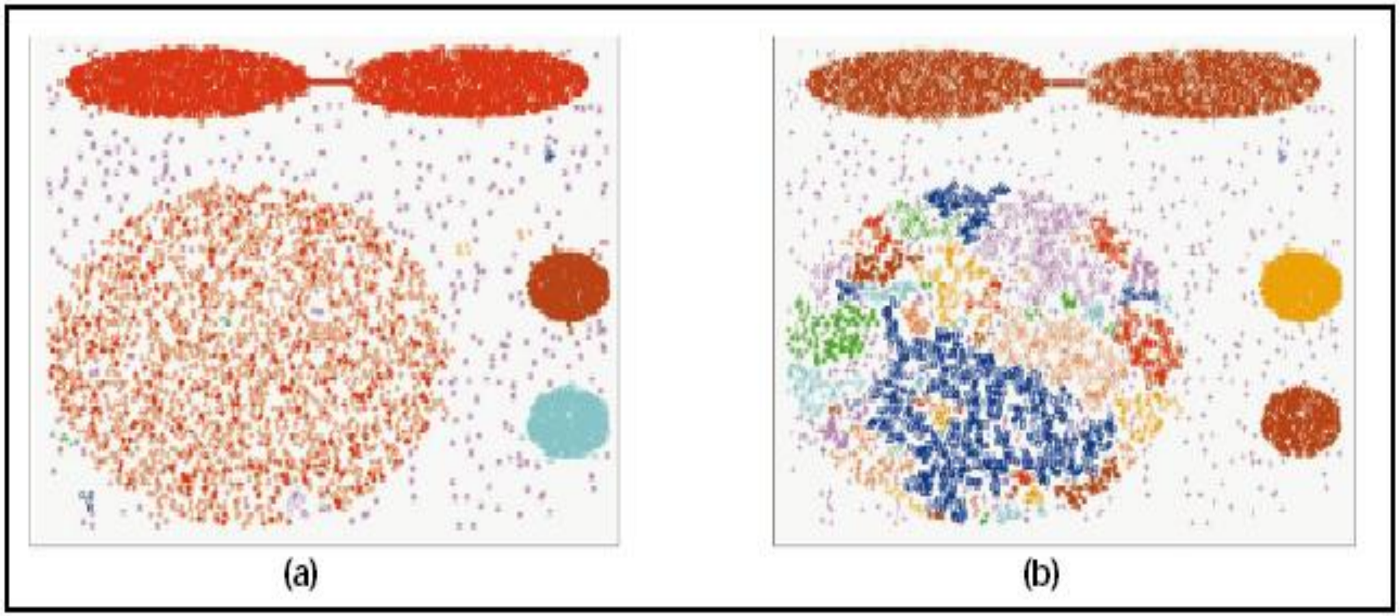
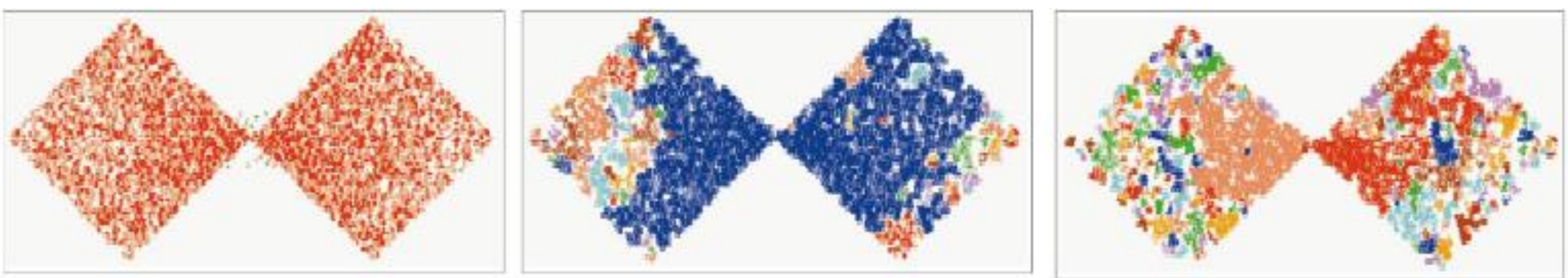


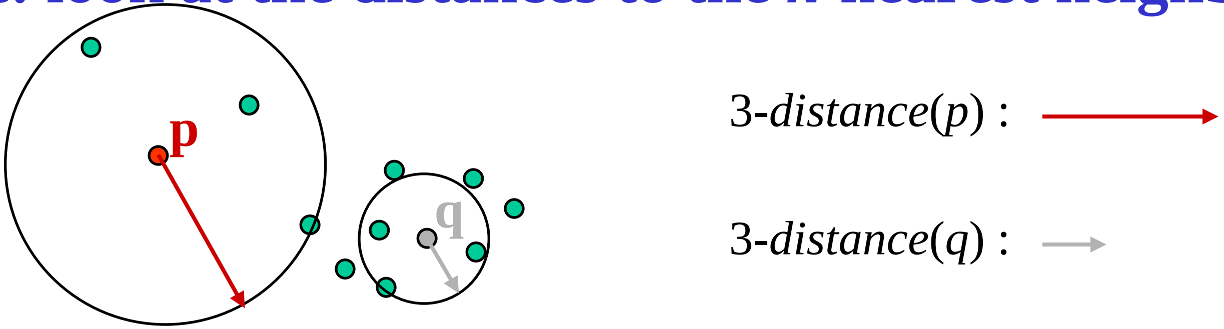
Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



(a) (b) (c)

Determining the Parameters ε and *MinPts*

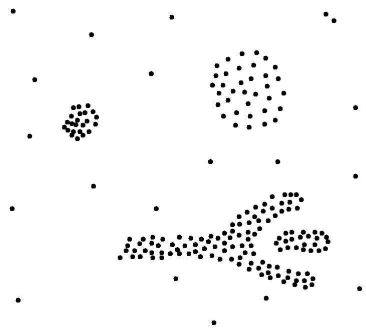
- Cluster: Point density higher than specified by ε and *MinPts*
- Idea: use the point density of the least dense cluster in the data set as parameters – but how to determine this?
- Heuristic: look at the distances to the *k*-nearest neighbors

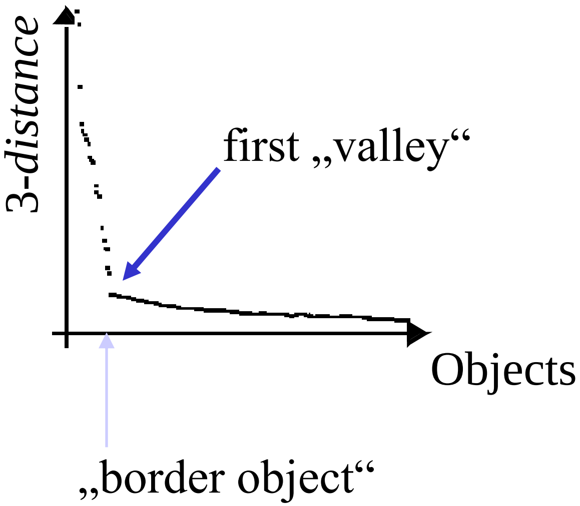


- Function *k-distance*(*p*): distance from *p* to the its *k*-nearest neighbor
- *k-distance plot*: *k*-distances of all objects, sorted in decreasing order

Determining the Parameters ε and *MinPts*

- **Example *k*-distance plot**

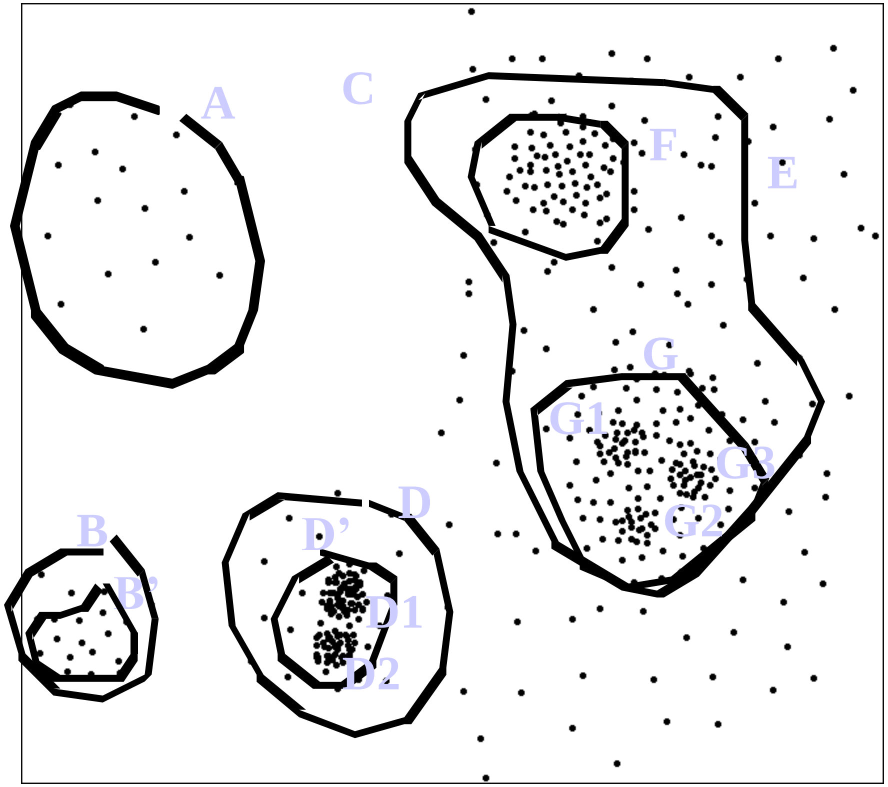


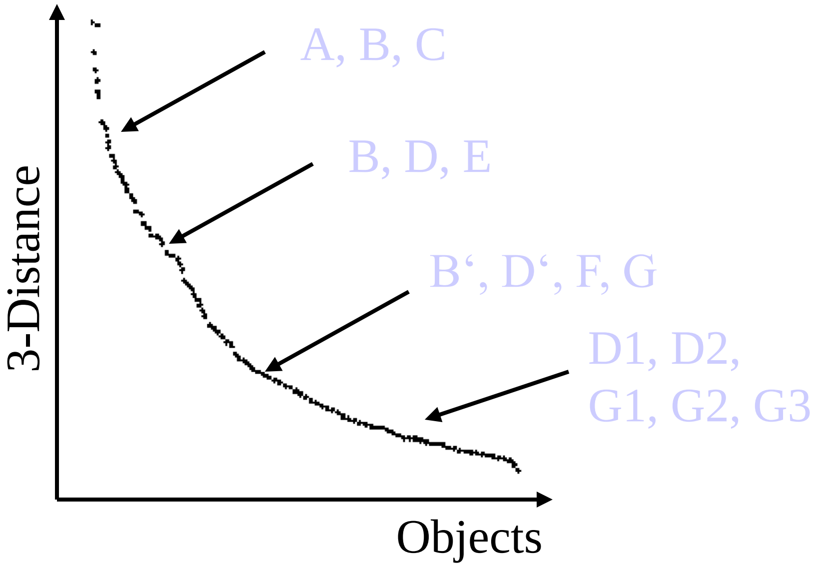


- **Heuristic method:**
  - **Fix a value for *MinPts* (default: $2 \times d - 1$)**
  - **User selects "border object" *o* from the *MinPts-distance* plot; ε is set to *MinPts-distance*(o)**

Determining the Parameters ε and *MinPts*

- Problematic example





Density Based Clustering: Discussion

- Advantages
  - Clusters can have arbitrary shape and size
  - Number of clusters is determined automatically
  - Can separate clusters from surrounding noise
  - Can be supported by spatial index structures
- Disadvantages
  - Input parameters may be difficult to determine
  - In some situations very sensitive to input parameter setting

OPTICS: Ordering Points To Identify the Clustering Structure

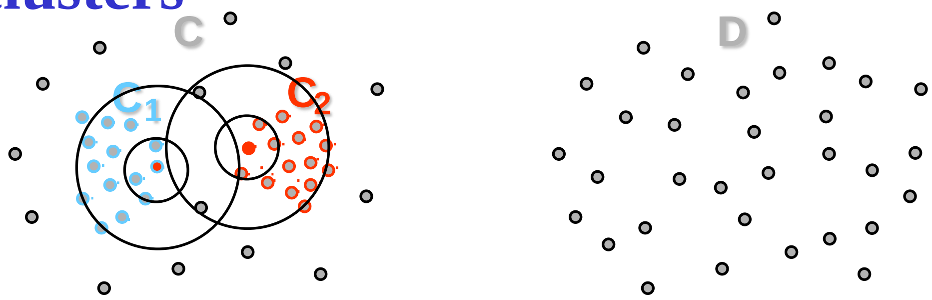
- **DBSCAN**
  - **Input parameter – hard to determine.**
  - **Algorithm very sensitive to input parameters.**
- **OPTICS – Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)**
  - **Based on DBSCAN.**
  - **Does not produce clusters explicitly.**
  - **Rather generate an ordering of data objects representing density-based clustering structure.**

OPTICS con't

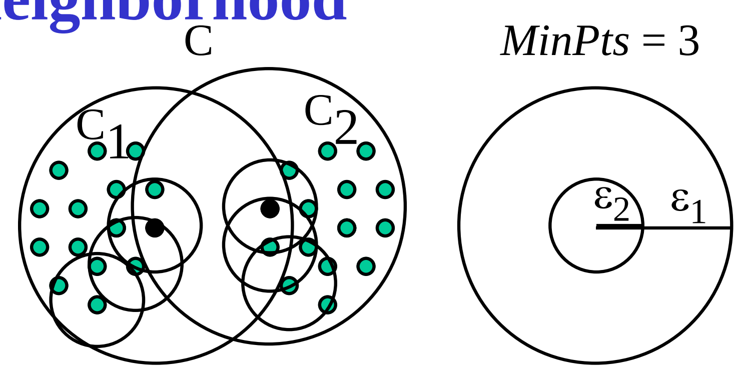
- Produces a special order of the database wrt its density-based clustering structure
- This cluster-ordering contains info equiv to the density-based clusterings corresponding to a broad range of parameter settings
- Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
- Can be represented graphically or using visualization techniques

Density-Based Hierarchical Clustering

- *Observation*: Dense clusters are completely contained by less dense clusters



- *Idea*: Process objects in the "right" order and keep track of point density in their neighborhood



Core- and Reachability Distance

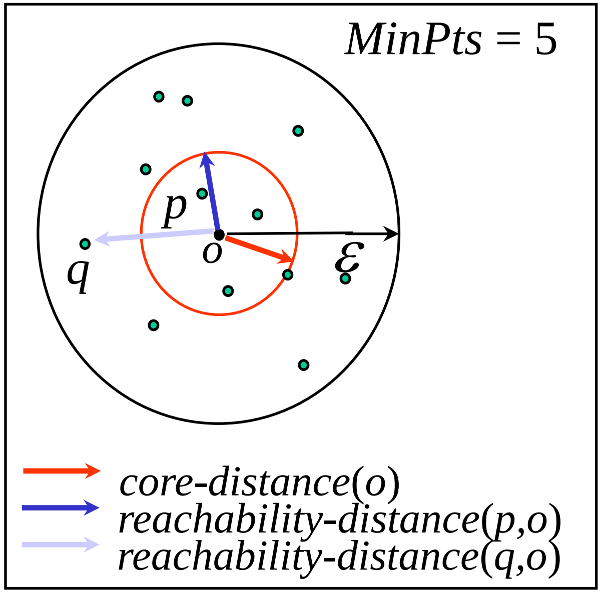
- **Parameters: "generating" distance ε, fixed value *MinPts***

- *core-distance*$_{\varepsilon,MinPts}(o)$

  "smallest distance such that *o* is a core object"
  (if that distance is $\leq \varepsilon$; "?" otherwise)

- *reachability-distance*$_{\varepsilon,MinPts}(p, o)$

  "smallest distance such that *p* is *directly* density-reachable from *o*"
  (if that distance is $\leq \varepsilon$; "?" otherwise)

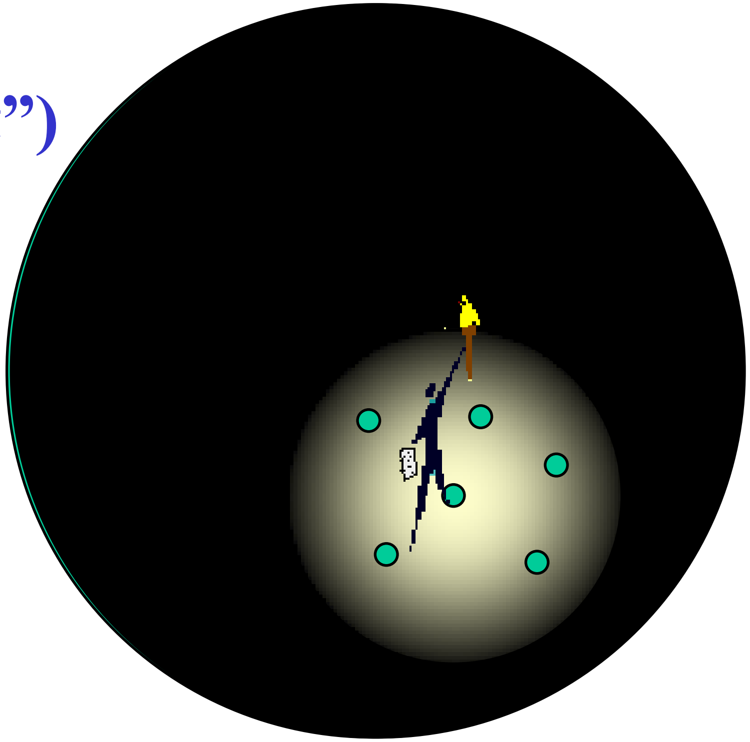


OPTICS: Extension of DBSCAN

- Order points by shortest *reachability distance* to guarantee that clusters w.r.t. higher density are finished first. (for a constant MinPts, higher density requires lower ε)

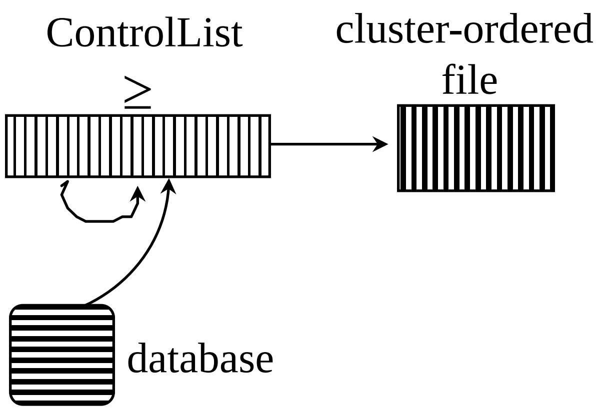
The Algorithm OPTICS

- Basic data structure: controlList
  - Memorize shortest reachability distances seen so far
    ("distance of a jump to that point")
- Visit each point
  - Make always a shortest jump
- Output:
  - order of points
  - core-distance of points
  - reachability-distance of points



The Algorithm OPTICS

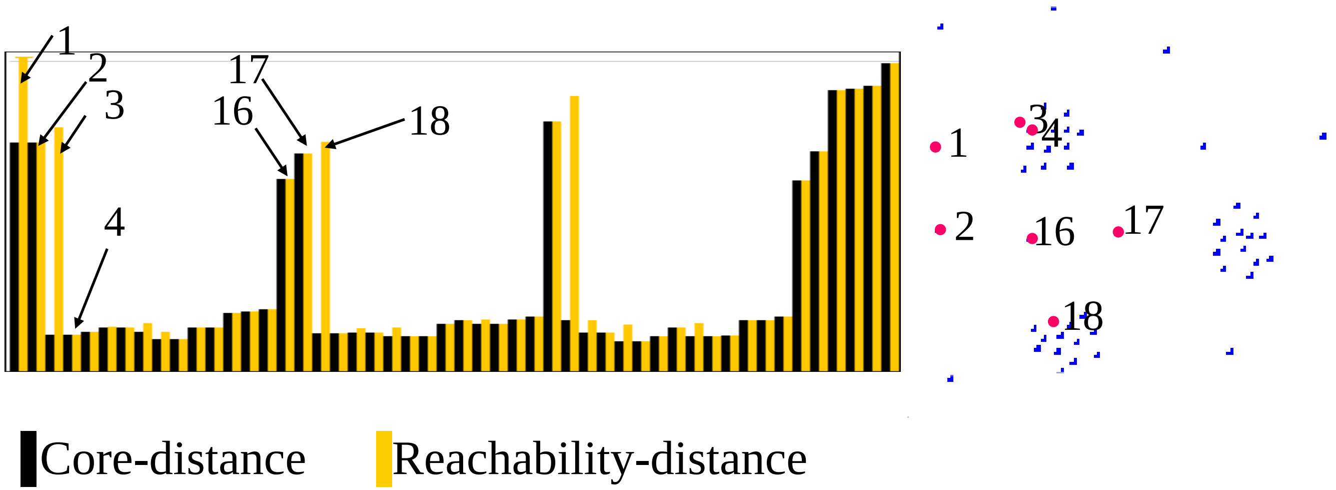
- *ControlList* ordered by reachability-distance.



foreach $o \in$ Database
 // initially, o.processed = false for all objects o
 if o.processed = false;
  insert (o, "?") 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: Properties

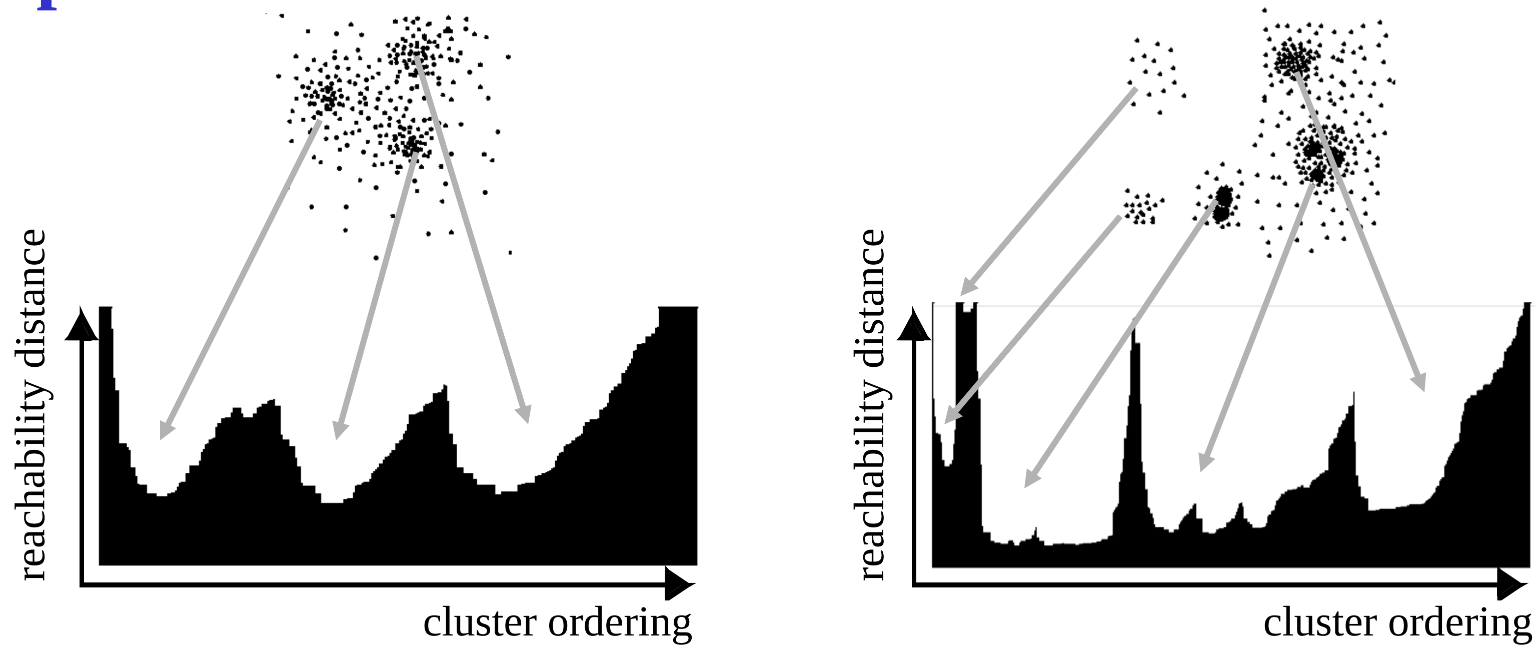
- "Flat" density-based clusters wrt. $\varepsilon^* \leq \varepsilon$ and *MinPts* afterwards:
  - Starts with an object *o* where *c-dist*(*o*) $\leq \varepsilon^*$ and *r-dist*(*o*) $> \varepsilon^*$
  - Continues while *r-dist* $\leq \varepsilon^*$



- Performance: approx. runtime(DBSCAN(ε, *MinPts*))
  - O(*n* * runtime(ε-neighborhood-query))
    - without spatial index support (worst case): O(n^2)
    - e.g. tree-based spatial index support: O($n * \log(n)$)

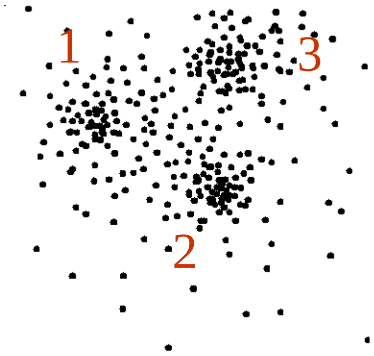
OPTICS: The Reachability Plot

- represents the density-based clustering structure
- easy to analyze
- independent of the dimension of the data

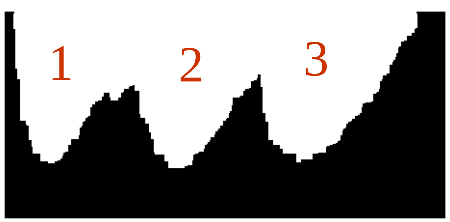


OPTICS: Parameter Sensitivity

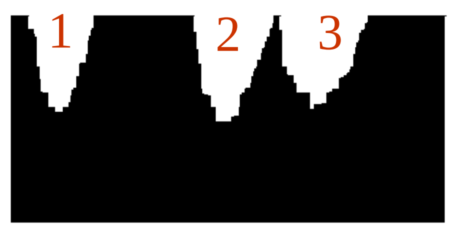
- Relatively insensitive to parameter settings
- Good result if parameters are just "large enough"



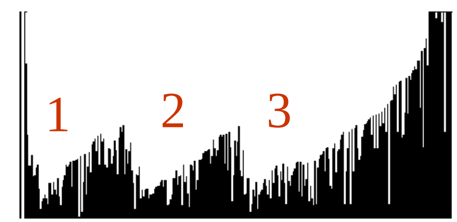
MinPts = 10, ε = 10



MinPts = 10, **ε = 5**

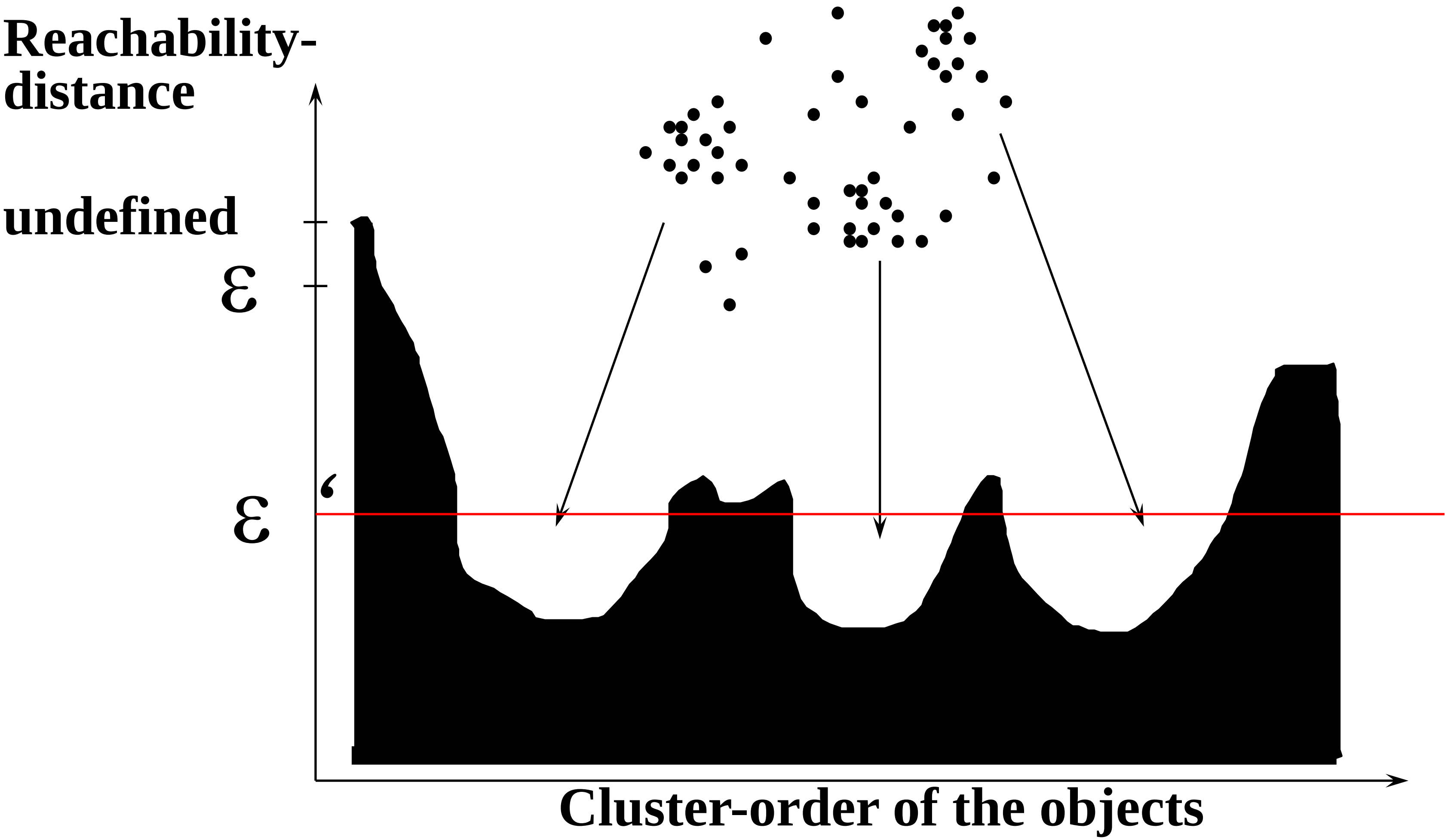


***MinPts* = 2**, ε = 10



An Example of OPTICS

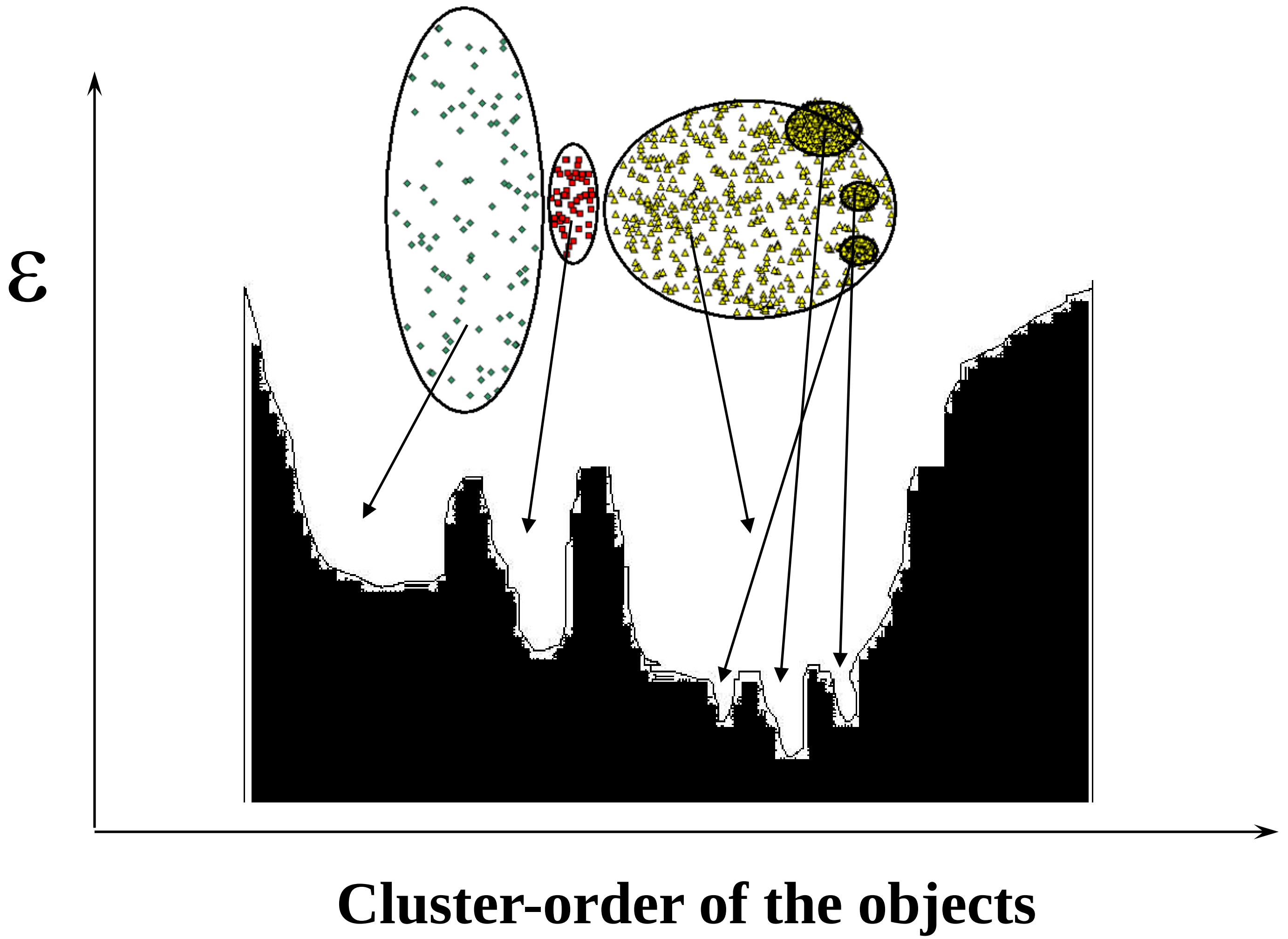
neighboring objects stay close to each other in a linear sequence.



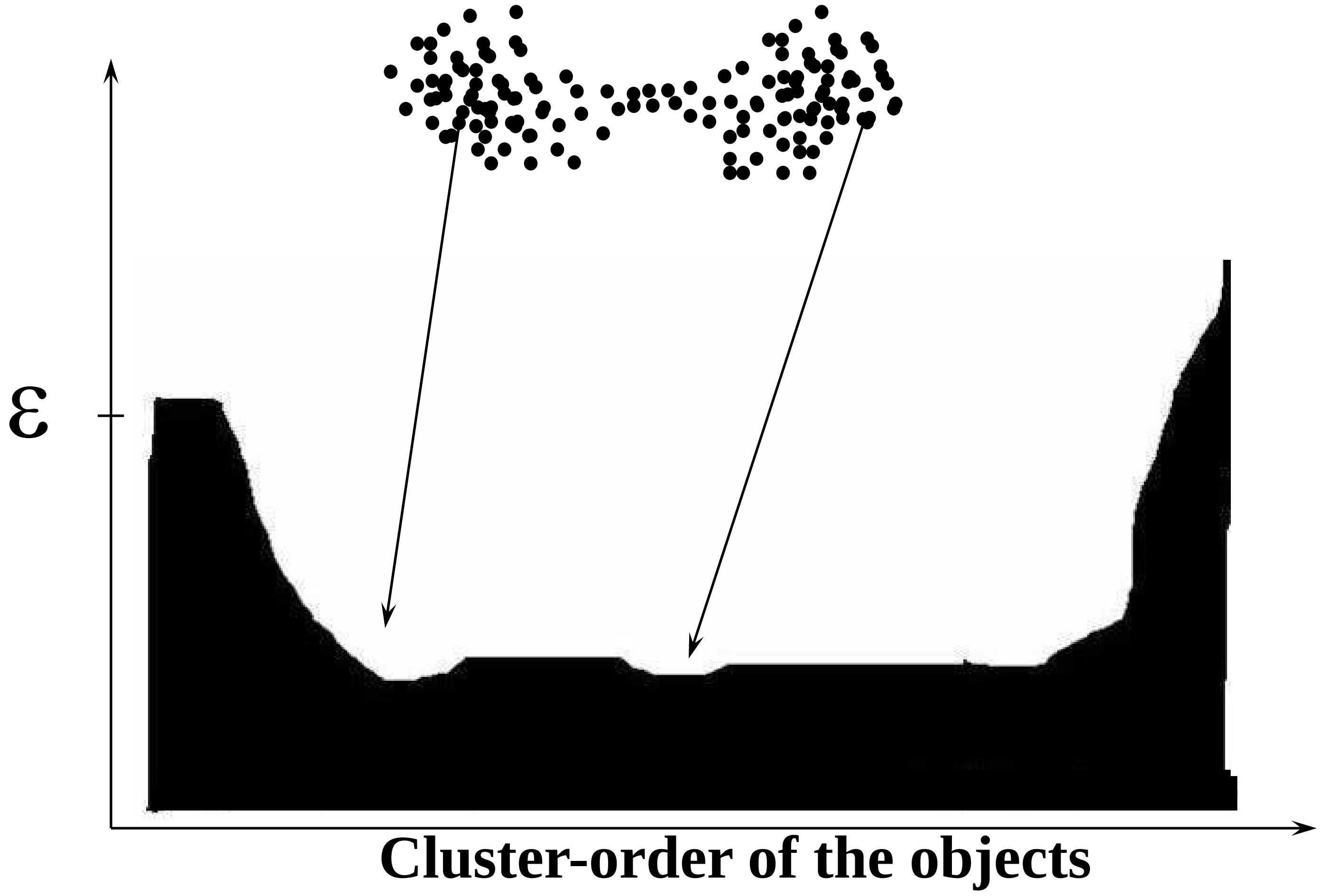
DBSCAN VS OPTICS

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

When OPTICS Works Well



When OPTICS Does NOT Work Well



DENCLUE: using density functions

- DENsity-based CLUstEring by Hinneburg & Keim (KDD'98)
- Major features
  - Solid mathematical foundation
  - Good for data sets with large amounts of noise
  - Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets
  - Significantly faster than existing algorithm (faster than DBSCAN by a factor of up to 45)
  - But needs a large number of parameters

Denclue: Technical Essence

- Model density by the notion of influence
- Each data object exert influence on its neighborhood.
- The influence decreases with distance
- Example:
  - Consider each object is a radio, the closer you are to the object, the louder the noise
- Key: Influence is represented by mathematical function

Denclue: Technical Essence

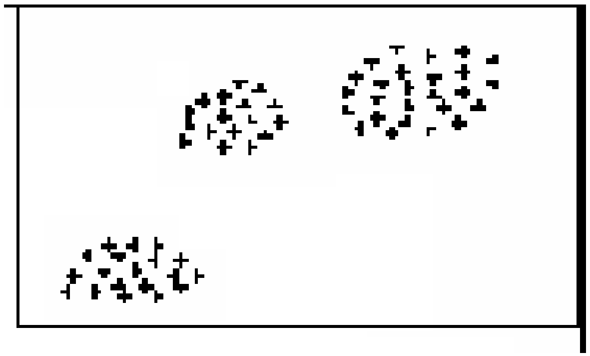
- **Influence functions: (influence of y on x, σ is a user given constant)**
  - **Square : $f^{y}_{square}(x)$ = 0, if dist(x,y) > σ,**
    1, otherwise
  - **Guassian:**

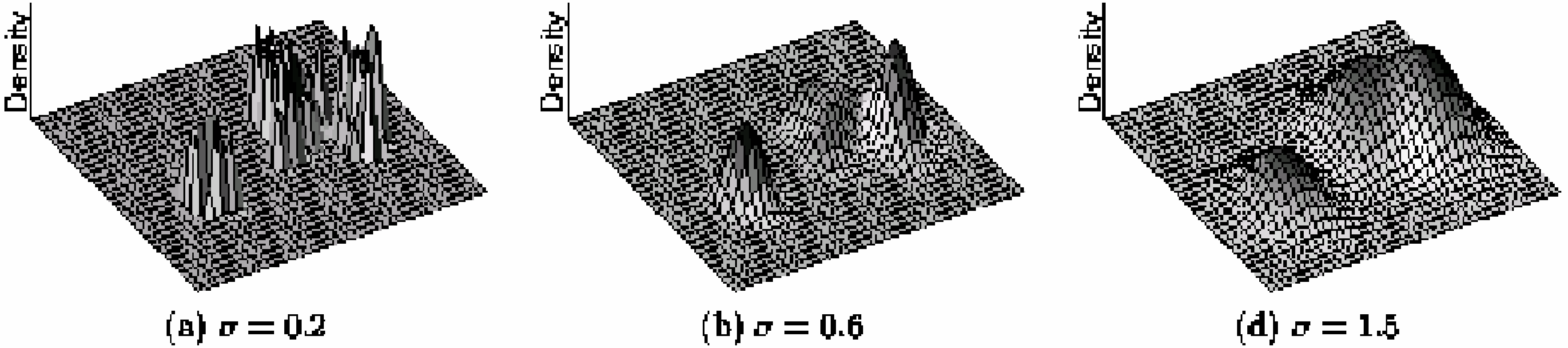
$$f^{y}_{Gaussian}(x) = e^{-\frac{d(x,y)^2}{2\sigma^2}}$$

Density Function

- Density Definition is defined as the sum of the influence functions of all data points.

$$f_{Gaussian}^{D}(x) = \sum_{i=1}^{N} e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$





Gradient: The steepness of a slope

- Example

$$f_{Gaussian}(x,y) = e^{-\frac{d(x,y)^2}{2\sigma^2}}$$

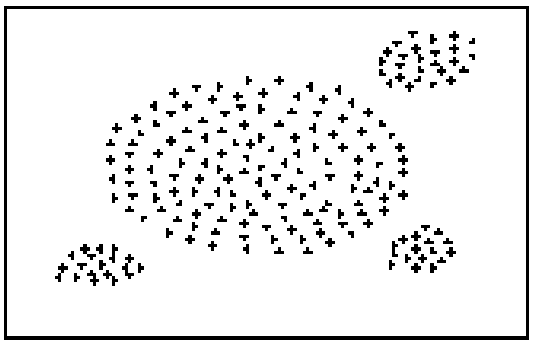
$$f_{Gaussian}^{D}(x) = \sum_{i=1}^{N} e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$

$$\nabla f_{Gaussian}^{D}(x,x_i) = \sum_{i=1}^{N} (x_i - x) \cdot e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$

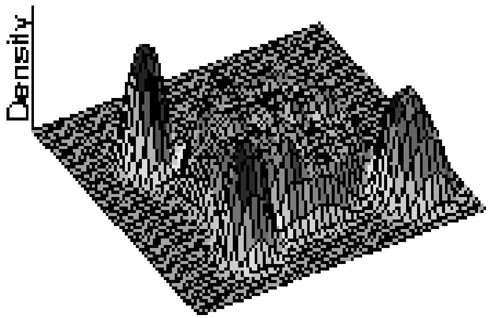
Denclue: Technical Essence

- Clusters can be determined mathematically by identifying density attractors.
- Density attractors are local maximum of the overall density function.

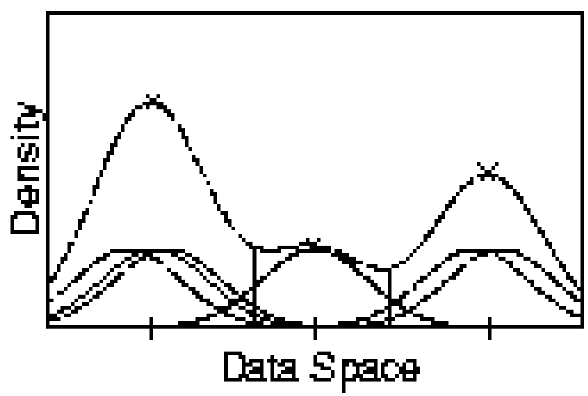
Density Attractor



(a) Data Set



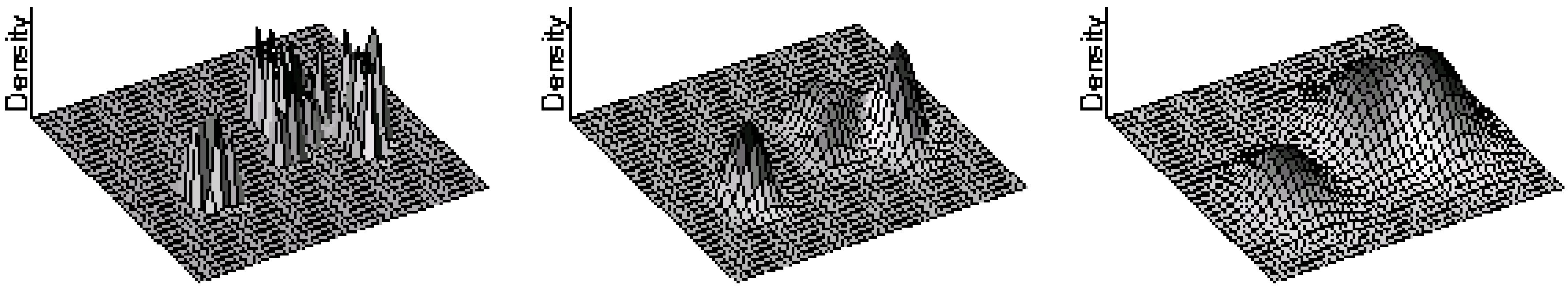
(c) Gaussian



Cluster Definition

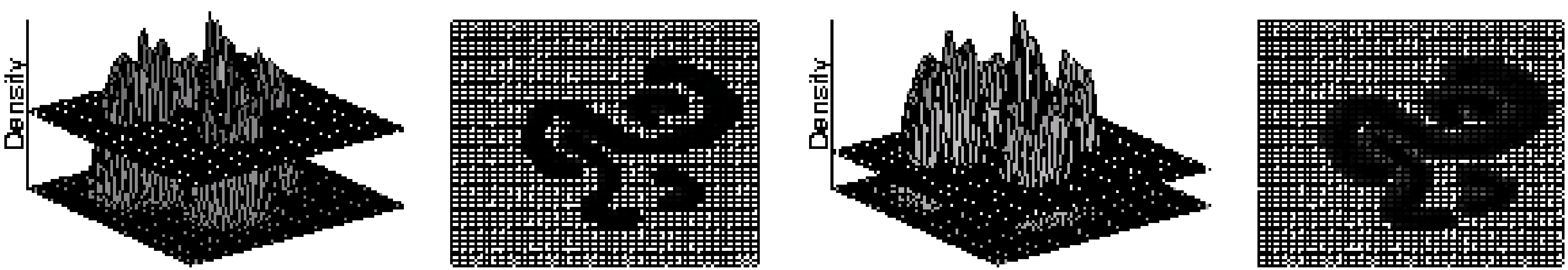
- Center-defined cluster
  - A subset of objects attracted by an attractor x
  - density(x) $\geq \xi$
- Arbitrary-shape cluster
  - A group of center-defined clusters which are connected by a path P
  - For each object x on P, density(x) $\geq \xi$.

Center-Defined and Arbitrary



(a) $\sigma = 0.2$ (b) $\sigma = 0.6$ (d) $\sigma = 1.5$

Figure 3: Example of Center-Defined Clusters for different σ



(a) $\xi = 2$ (b) $\xi = 2$ (c) $\xi = 1$ (d) $\xi = 1$

Figure 4: Example of Arbitray-Shape Clusters for different ξ

DENCLUE: How to find the clusters

- Divide the space into grids, with size 2σ
- Consider only grids that are highly populated
- For each object, calculate its density attractor using hill climbing technique
  - Tricks can be applied to avoid calculating density attractor of all points
- Density attractors form basis of all clusters

Features of DENCLUE

- Major features
  - Solid mathematical foundation
    - Compact definition for density and cluster
    - Flexible for both center-defined clusters and arbitrary-shape clusters
  - But needs a large number of parameters
    - σ: parameter to calculate density
    - ξ: density threshold
    - δ: parameter to calculate attractor