Going Big in Data Dimensionality:

Challenges and Solutions for Mining High Dimensional Data

Peer Kröger
Lehrstuhl für Datenbanksysteme
Ludwig-Maximilians-Universität München

Going Big in Data Dimensionality?

- The three V's of big data
  - Volume
  - Velocity
  - Variety

Let's talk about variety before talking about velocity and volume

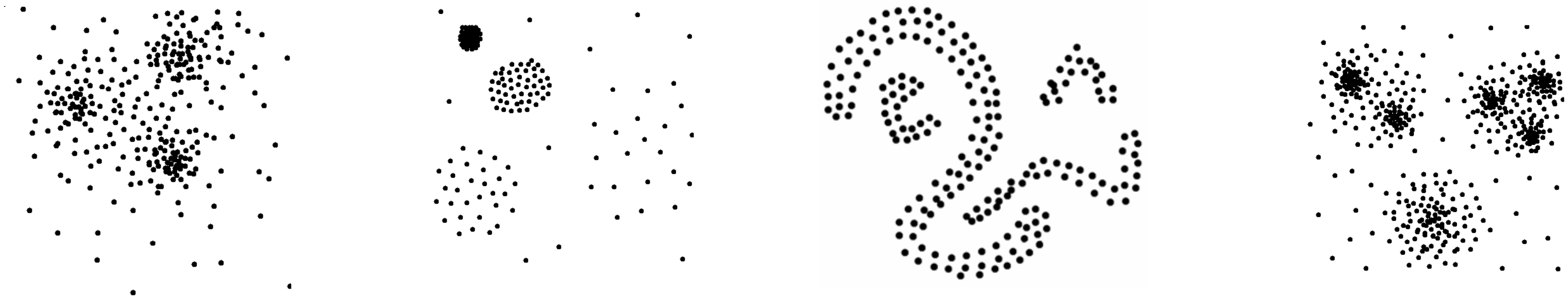
- An aspect of variety (and volume) is high dimensionality
  - An archaelogical finding can have 10+ attributes/dimensions
  - Recommendation data can feature 100+ dimensions per object
  - Micro array data may contain 1000+ dimensions per object
  - TF vectors may contain 10,000+ dimensions per object
  - …
- Note: we are talking about structured data!!!

Outline

- Why Bother?
- Solutions
- Perspectives – Open Issues

Why Bother?

- Clustering
  - Automatically partition the data into clusters of similar objects
  - Diverse applications ranging from business applications (e.g. customer segmentation) to science (e.g. molecular biology)
- Similarity?
  - Objects are points in some feature spaces (let us assume an Euclidean space for convenience)
  - Similarity can be defined as the vicinity of two feature vectors in the feature space, usually applying a distance function like some Lp norm, the cosine distances, etc.



Why Bother?

- But:
  - In the early days of data mining:
    - The relevance of features was carefully analyzed before recording them because data acquisition was costly
    - So far so good: only relevant features were recorded
  - Nowadays:
    - Data acquisition is cheap and easy (mobile devices, sensors, modern machines, etc. – ***everyone*** measures ***everything***, ***everywhere***)
    - Consequence: sure big data, but what bothers?
      - The relevance of a given feature to the analysis task is not necessarily clear
      - Data is high dimensional containing a possibly large number of irrelevant attributes
    - OK, but why does this bother?

Why Bother?

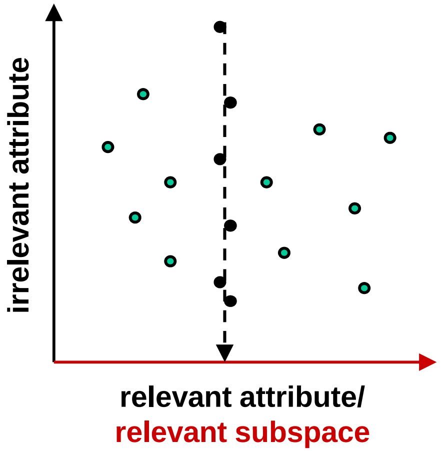
- High-dimensional data problems
  - General: "the curse of dimensionality"

$$\forall \varepsilon > 0 : \lim_{d\to\infty} P[dist(\frac{\mathrm{Dmax}_d - \mathrm{Dmin}_d}{\mathrm{Dmin}_d}, 0) \leq \varepsilon] = 1$$

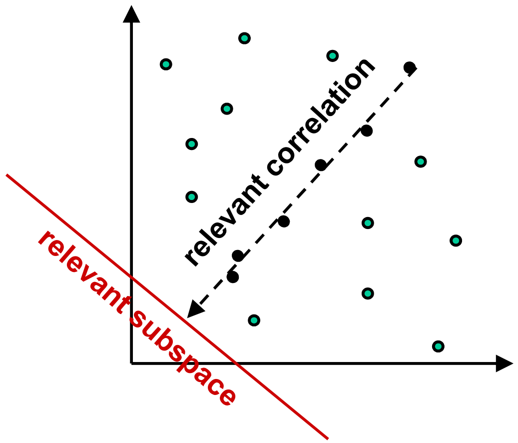
Dmax = Distance to the farthest neighbor
Dmin = Distance to the nearest neighbor
d = dimensionality of the data space

    - Relative contrast of distances decrease
    - No cluster to find any more, only noise
  - Special/Additional: Clusters in subspaces

Local feature relevance



Local feature correlation



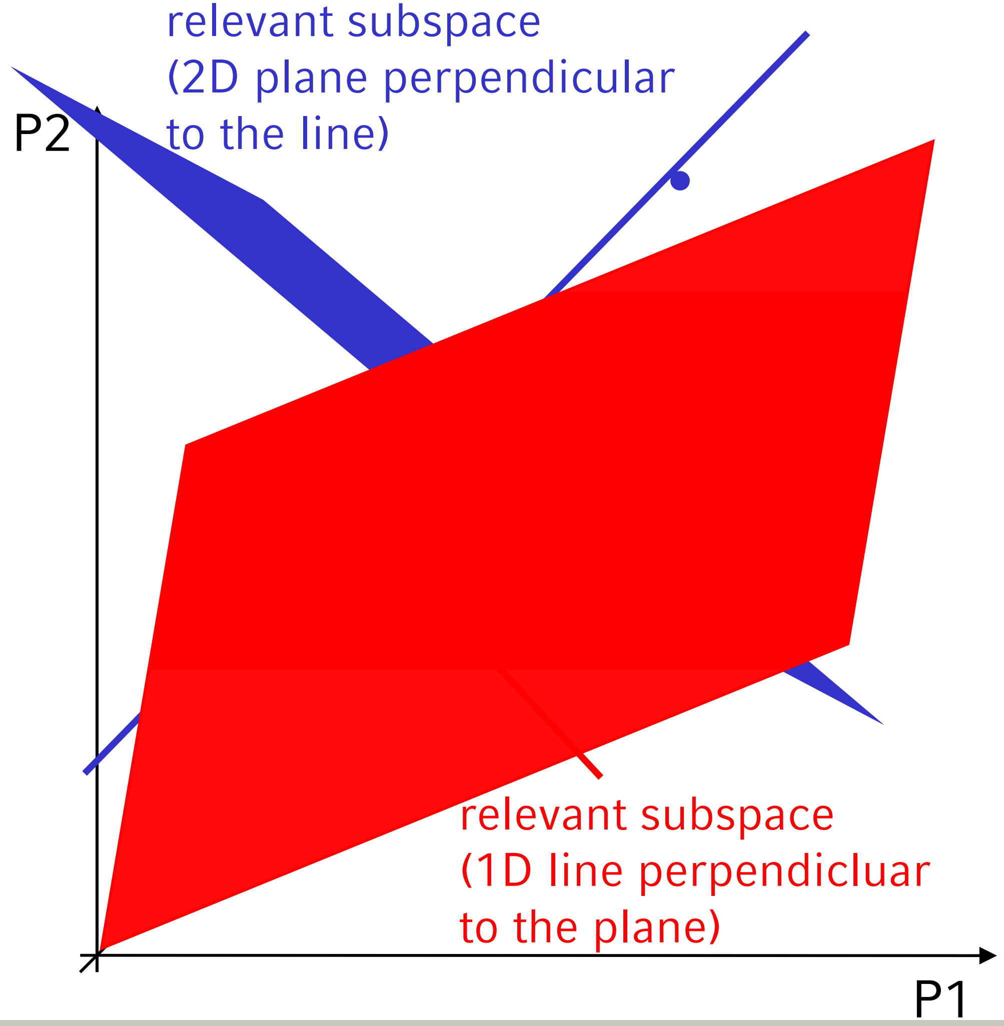
Why Bother?

- Example: local feature correlation
  - customers (C1 – C10) rate products (P1 – P3)

	P1	P2	P3
C1	3	2	1
C2	2	3	4
C3	0	1	2
C4	3	4	5
C5	5	5	5
C6	1	3	10
C7	4	6	8
C8	0	2	3
C9	7	9	1
C10	3	5	5

C1–C5: $P1 - 2 \cdot P2 + P3 = 0$

C6–C10: $P1 - P2 + 2 = 0$



DATABASE SYSTEMS GROUP

Why Bother?

LMU

- H

ighbor
ghbor
e

Consequences:

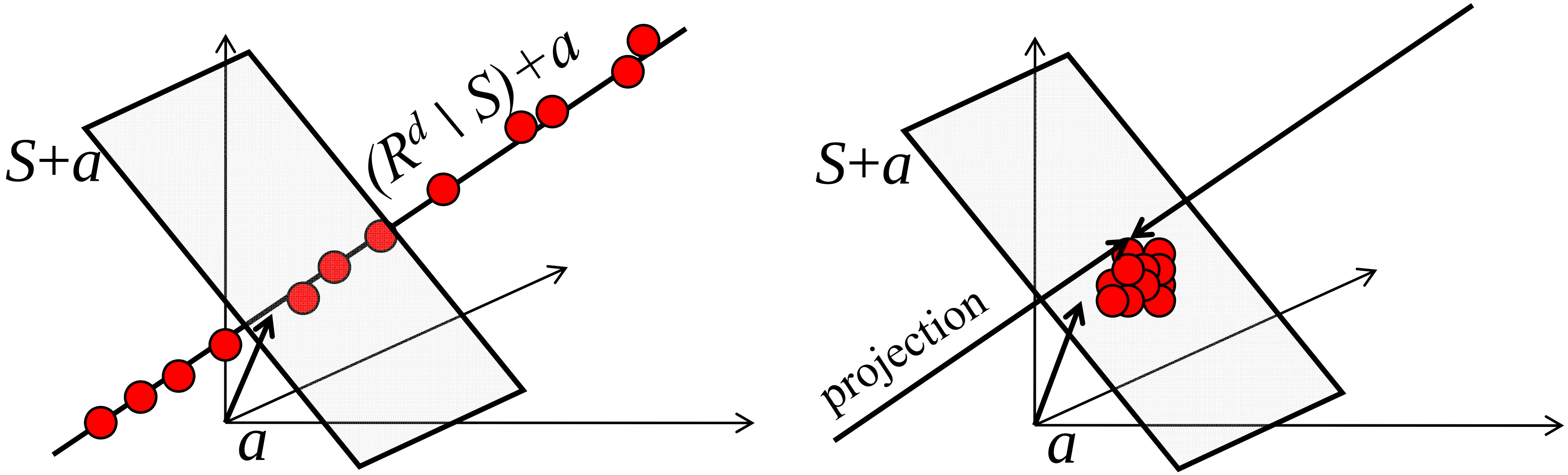
We should care about accuracy before caring about scalability

If we take a traditional clustering method and optimize it for efficiency, we will most likely not get the desired results

Why Bother?

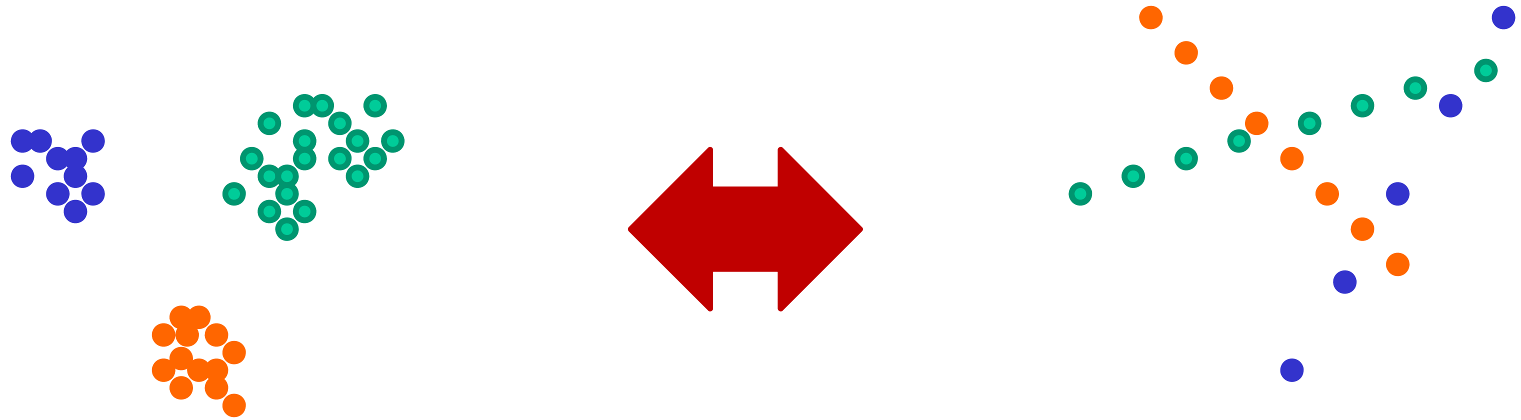
- General Solution: "correlation clustering"
  - Search for clusters in arbitrarily oriented subspaces
  - Affine subspace $S+a$, $S \subset R^d$, affinity $a \in R^d$, is interesting if a set of points clusters (are dense) within this subspace
  - The points of the cluster may exhibit high variance in the perpendicular subspace $(R^d \setminus S)+a$

    → ***points form a hyper-plane along this subspace***



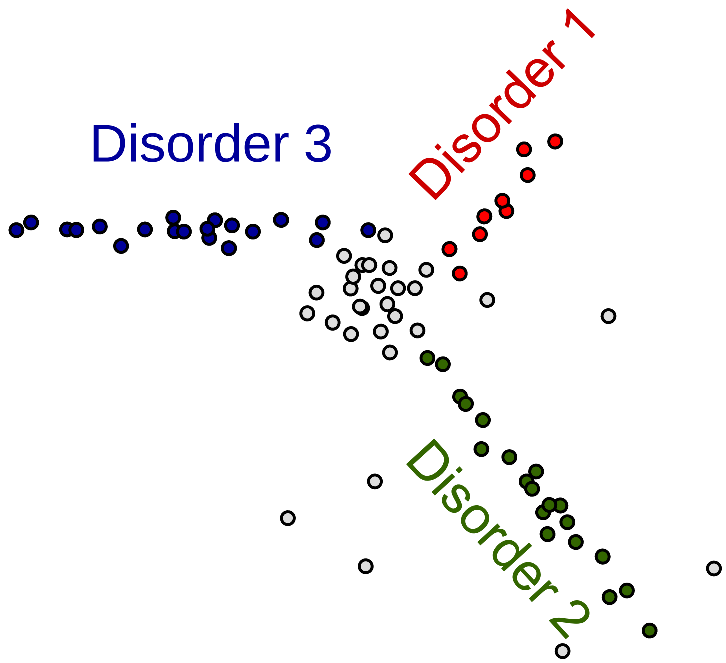
Why Bother?

- Back to the definition of clustering
  - Clustering: partition the data into clusters of ***similar*** objects
  - Traditionally, similarity means ***similar values in all attributes*** (this obviously does not account for high dimensional data)
  - Now, similarity is defined as a ***common correlation in a given (sub)set of features*** (actually, that is not too far apart)



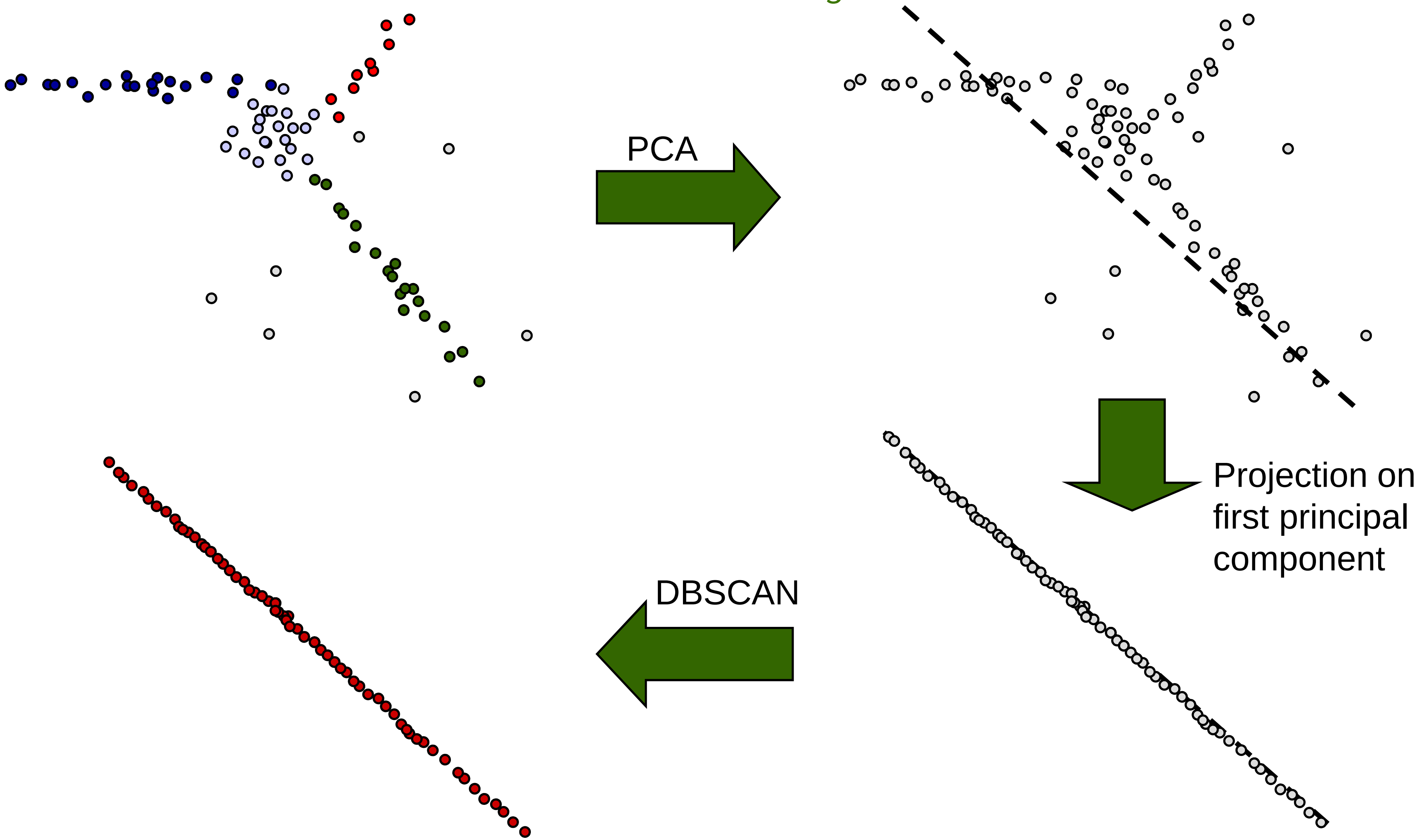
Why Bother?

- Why not feature selection?
  - (Unsupervised) feature selection (e.g. PCA, SVD, ...) is ***global***; it always returns only one (reduced) feature space
  - The ***local*** feature relevance/correlation problem states that we usually need multiple feature spaces (possibly one for each cluster)
  - Example: Simplified metabolic screening data (here: 2D, 43D in reality)



Why Bother?

- Use feature selection before clustering



Why Bother?

- Two tasks:
  1. We still need to search for clusters (depends on cluster model)
     - E.g. minimal cut of the similarity graph is NP-complete

  2. But now, we also need to search for arbitrarily oriented subspaces (search space probably infinite)
     - Naïve solution:
       - Given a cluster criterion and a database of *n* points
       - Compute for each subset of *k* points the subspace in which these points cluster and test the cluster criterion in this subspace
       - Search space:

$$\sum_{k=1}^{n} \binom{n}{k} = 2^n - 1 = O(2^n)$$

     - BTW:
       - How can we compute the subspace of the cluster? => see later
       - What is a cluster criterion? => see task 1

Why Bother?

- Even worse: ***Circular Dependency***
  - Both tasks depend on each other
    - In order to determine the correct subspace of a cluster, we need to know (at least some) cluster members
    - In order to determine the correct cluster memberships, we need to know the subspaces of all clusters
- How to solve the circular dependency problem?
  - Integrate subspace search into the clustering process
  - Due to the complexity of both tasks, we need **heuristics**
  - These heuristics should ***simultaneously*** solve
    - the clustering problem
    - the subspace search problem

DATABASE SYSTEMS GROUP

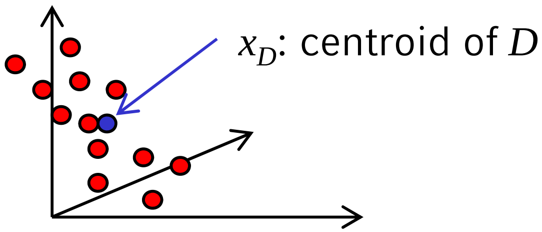
Outline

LMU

- Why Bother?
- Solutions
- Perspectives – Open Issues

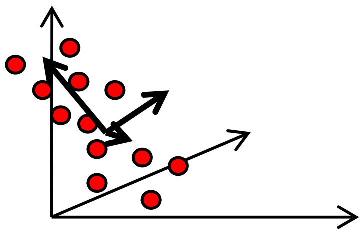
Solutions

- Finding clusters in arbitrarily oriented subspaces
  - Given a set D of points (e.g. a potential cluster); how can we determine the subspace in which these points cluster?
  - Principal Component Analysis (PCA) determines the directions of highest variance
    - Compute Covariance-matrix Σ_D für D



$$\boxed{\Sigma_D = \frac{1}{|D|} \sum_{x \in D} (x - x_D)(x - x_D)^{\mathrm{T}}}$$

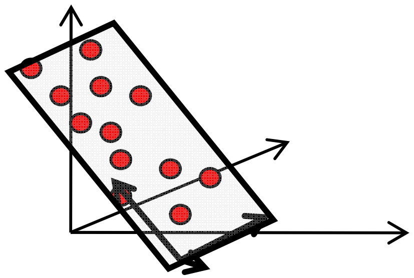
    - Obtain Eigenvalue-Matrix and Eigenvector-Matrix $\Sigma_D = V_D E_D V_D^{\mathrm{T}}$



- V_D: new basis, first Eigenvector = direction of the highest variance
- E_D: covariance-matrix of D in the new coordinate system V_D

Solutions

- If the points in D form a λ-dimensional hyper-plane then this hyper-plane is spanned by the first λ Eigenvectors
- The relevant subspace in which the points cluster is spanned by the remaining d-λ Eigenvectors $\hat{V}_D$



The sum of the smallest d-λ Eigenvalues $\sum_{i=\lambda+1}^{d} e_{D_{ii}}$

is minimal w.r.t. all possible transformations → points cluster optimal in this subspace

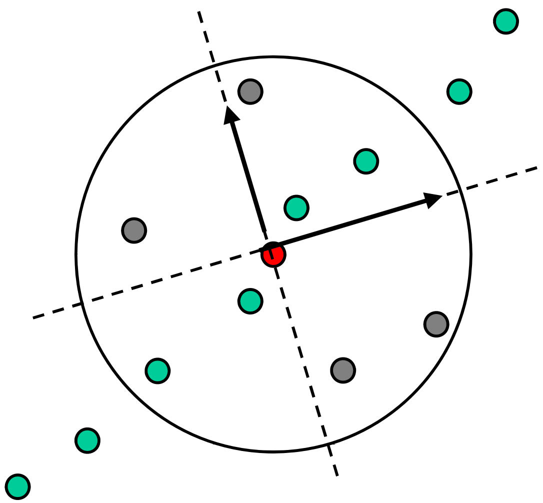
- Model for Correlation Cluster
  - The λ-dimensional hyper-plane accommodating the cluster $C \subset R^d$ is defined by a system of d-λ equations for d variables and an affinity (e.g. the centroid of the cluster x_C):

$$\hat{V}_C^{\mathrm{T}} x = \hat{V}_C^{\mathrm{T}} x_C$$

  - The equation system is approximately fulfilled by all $x \in C$

Solutions

- Correlation clustering methods based on PCA
  - Integrate (local) PCA into existing clustering algorithms
  - Learn a distance measure that reflects the subspace of points and/or parts of clusters (typically: specialized Mahalanobis distance)
  - Conquer the circular dependency of the two tasks by the so-called „Locality Assumption“
    - A local selection of points (e.g. the *k*-nearest neighbors of a potential cluster center) represents the hyper-plane of the corresponding cluster
    - The application of PCA on this local selection yields the subspace of the corresponding cluster
    - ***Curse of dimensionality???***



Solutions

- Many methods rely on a local application of PCA to sets of potential cluster members
  - Rely on locality assumption
  - Alternative: random sampling

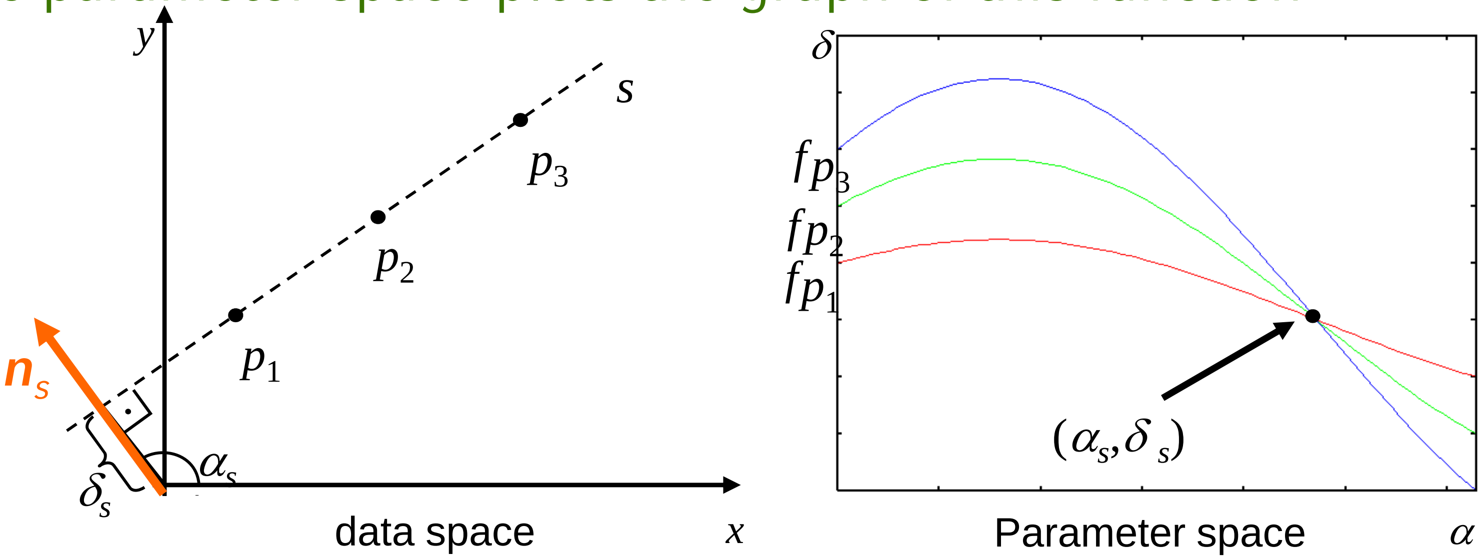
- How can we avoid the Locality Assumption/Random Sampling???
  - CASH (***C***lustering in ***A***rbitrary ***S***ubspaces based on the ***H***ough transform)

Solutions

- Basic idea of CASH
  - Transform each object into a so-called *parameter space* representing all possible subspaces accommodating this object (i.e. all hyper-planes through this object)
  - This parameter space is a *continuum* of all these subspaces
  - The subspaces are represented by a considerably small number of parameters
  - This transform is a generalization of the Hough Transform (which is designed to detect linear structures in 2D images) for arbitrary dimensions

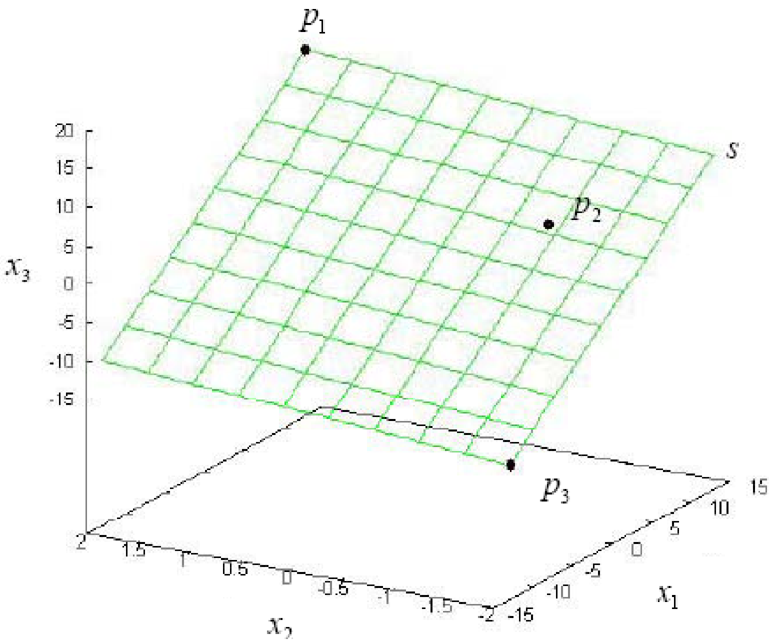
Solutions

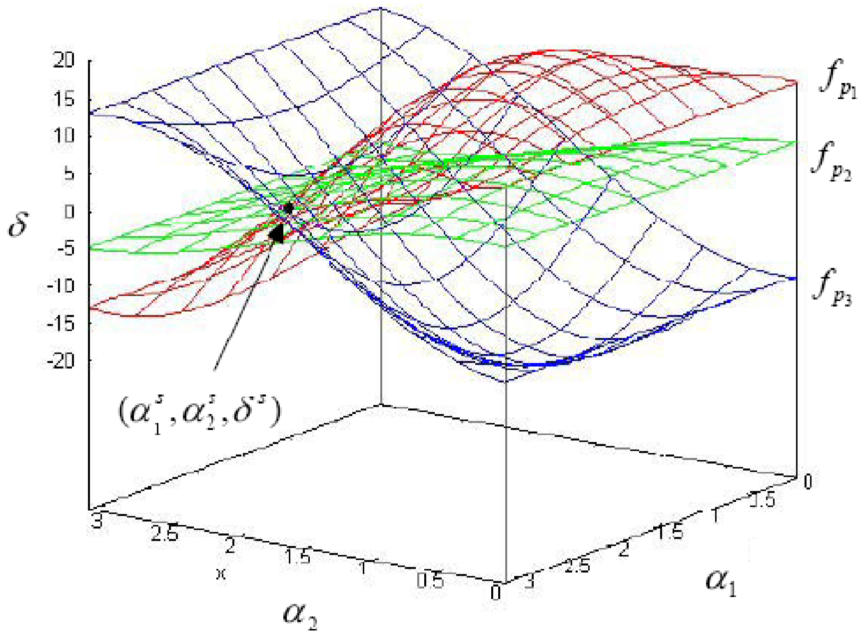
- Transform
  - For each *d*-dimensional point *p* there is an infinite number of (*d*-1)-dimensional hyper-planes through *p*
  - Each of these hyper-planes *s* is defined by ($p,\alpha_1,\ldots,\alpha_{d-1}$), where $\alpha_1,\ldots,\alpha_{d-1}$ is the normal vector $\boldsymbol{n}_s$ of the hyper-plane *s*
  - The function $f_p(\alpha_1,\ldots,\alpha_{d-1}) = \delta_s = \langle p,\boldsymbol{n}_s\rangle$ maps *p* and $\alpha_1,\ldots,\alpha_{d-1}$ onto the distance δ_s of the hyper-plane *s* to the origin
  - The parameter space plots the graph of this *function*



Solutions

- Properties of this transform
  - point in the data space = sinusoide curve in the parameter space
  - point in the parameter space = hyper-plane in the data space
  - points on a common hyper-plane in the data space (cluster) = sinusoide curves intersecting at ***one*** point in the parameter space
  - intersection of sinusoide curves in the parameter space = hyper-plane accommodating the corresponding points in data space





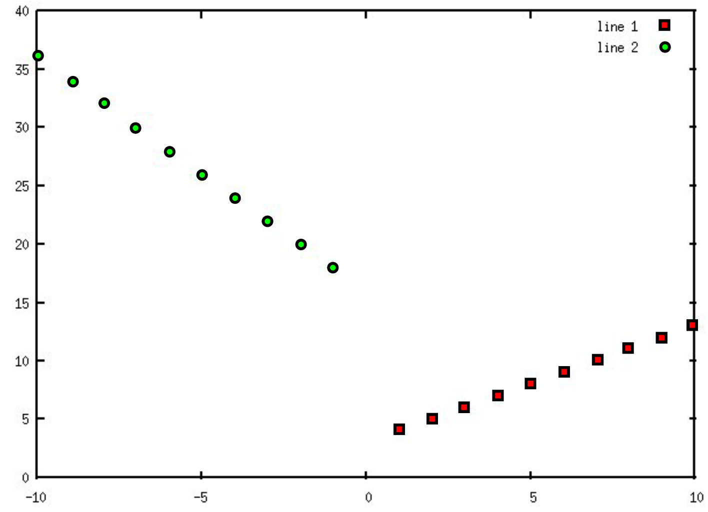
Solutions

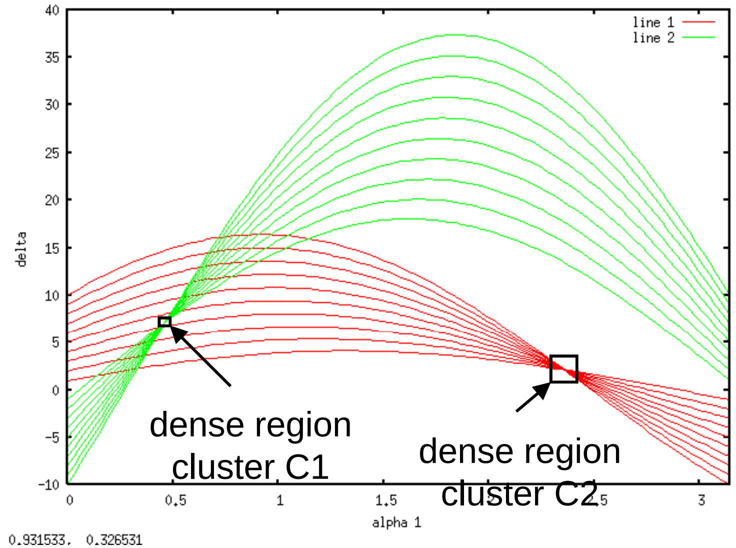
- Detecting clusters
  - determine all intersection points of at least *m* curves in the parameter space => (d-1)-dimensional cluster
  - Exact solution (check all pair-wise intersections) is too costly
  - Heuristics are employed
    - Grid-based bisecting search => Find cells with at least *m* curves

    ☺ determining the curves that are within a given cell is in $O(d^3)$

    ☹ Number of cells $O(\boldsymbol{r}^d)$, where r is the resolution of the grid

    ☹ high value for $\boldsymbol{r}$ necessary

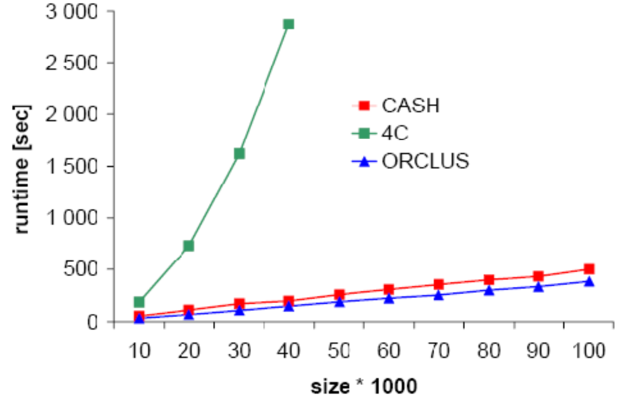


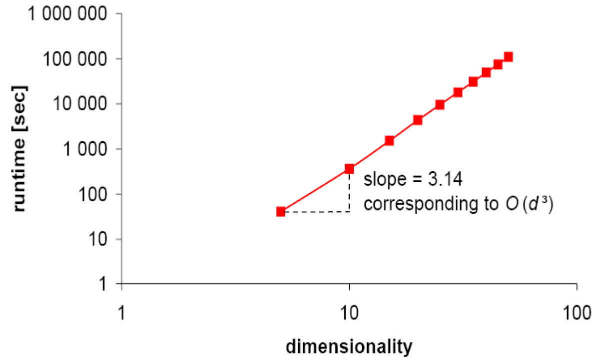


Solutions

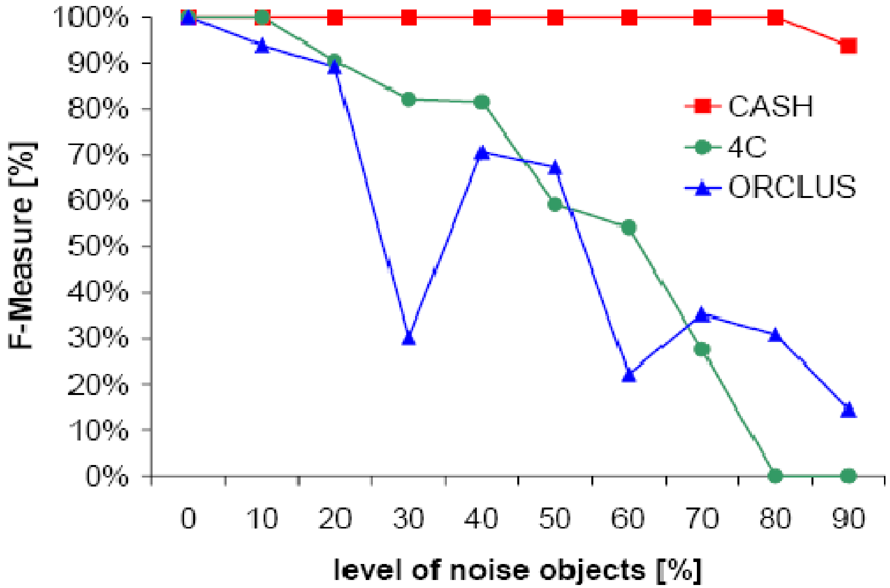
- Complexity (*c* = number of cluster found – not an input parameter!!!)
  - Bisecting search $O(s \cdot c)$
  - Determination of curves in a cell $O(n \cdot d^3)$
  - Over all $O(s \cdot c \cdot n \cdot d^3)$

    (algorithms for PCA are also in $O(d^3)$)

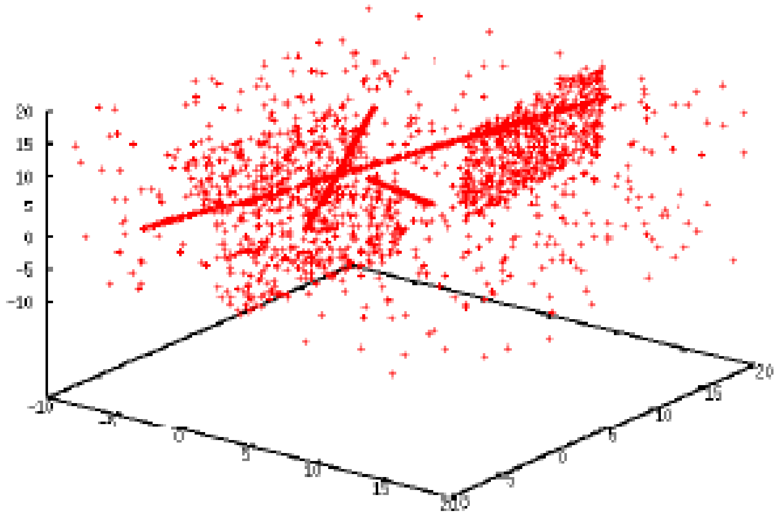




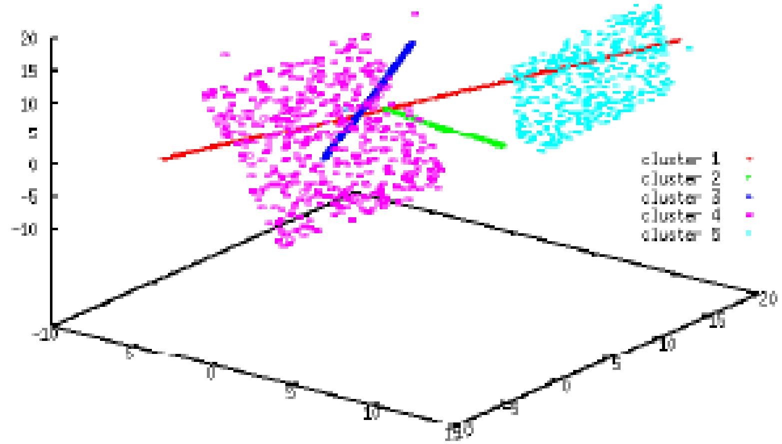
- Robustness against noise



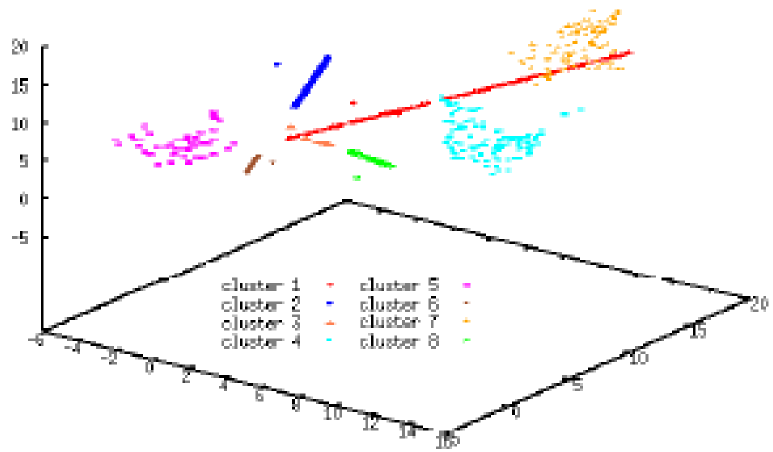
Solutions



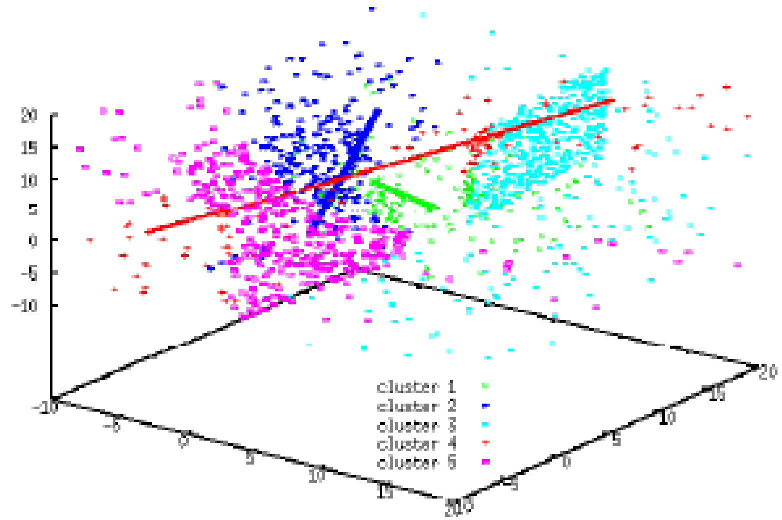
(a) Data set DS2.



(b) CASH – Cluster 1 - 5.



(c) 4C – Cluster 1 - 8.



(d) ORCLUS – Cluster 1 - 5.

Outline

- Why Bother?
- Solutions
- Perspectives – Open Issues

Perspectives – Open Issues

- What is next?
  - Still a lot to take care about at the accuracy end!!!
    - Examining the results (Are they significant? How to evaluate?).
    - Novel heuristics with new assumptions (limitations?).
    - Other patterns like outlier detection
    - …
- Big Data?
  - **Variety**:
    non-linear correlations, non-numeric data, relational data, …
  - **Velocity**:
    dynamic data, data streams, …
  - **Volume**:
    scalability (size/dimensionality), approximate solutions, …

Schluss