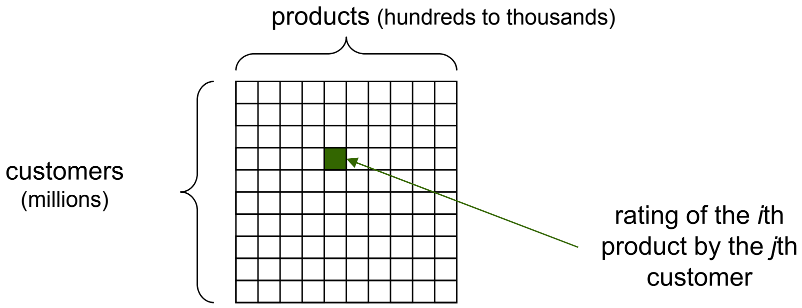
Chapter Overview

1. Introduciton and Challenges
2. Feature Selection
3. Feature Reduction and Metric Learning
4. Clustering in High-Dimensional Data

Challenges for Clustering High-Dimensional Data

- Customer Recommendation / Target Marketing
  - Data
    - Customer ratings for given products
    - Data matrix:

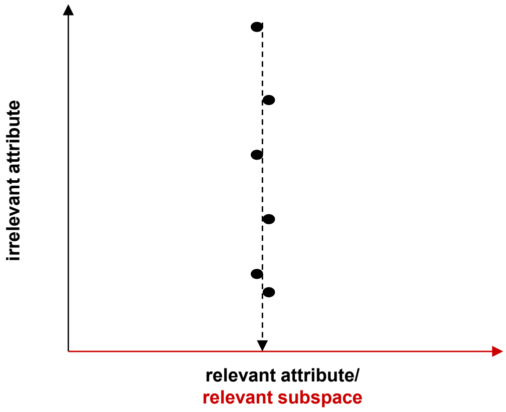

products (hundreds to thousands)
customers (millions)
rating of the *i*th product by the *j*th customer

  - Task: Cluster customers to find groups of persons that share similar preferences or disfavor (e.g. to do personalized target marketing)
    - *Challenge*:

      customers may be grouped differently according to different preferences/disfavors, i.e. different subsets of products

Challenges for Clustering High-Dimensional Data

- *Relevant and Irrelevant attributes*
  - A subset of the features may be relevant for clustering
  - Groups of similar ("dense") points may be identified when considering these features only

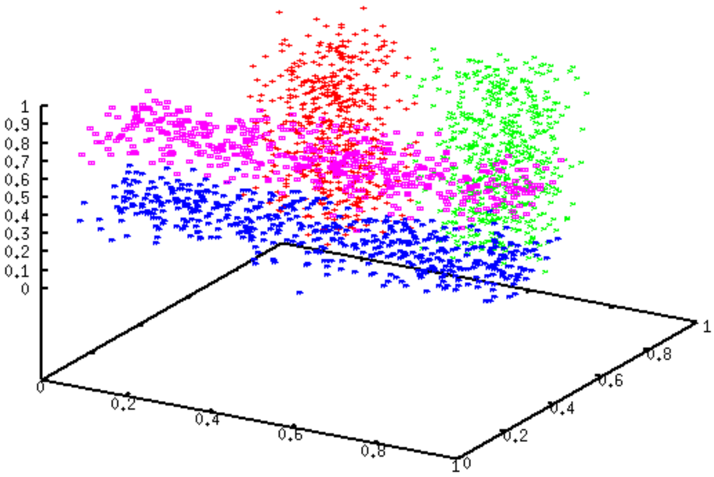


  - Different subsets of attributes may be relevant for different clusters

Challenges for Clustering High-Dimensional Data

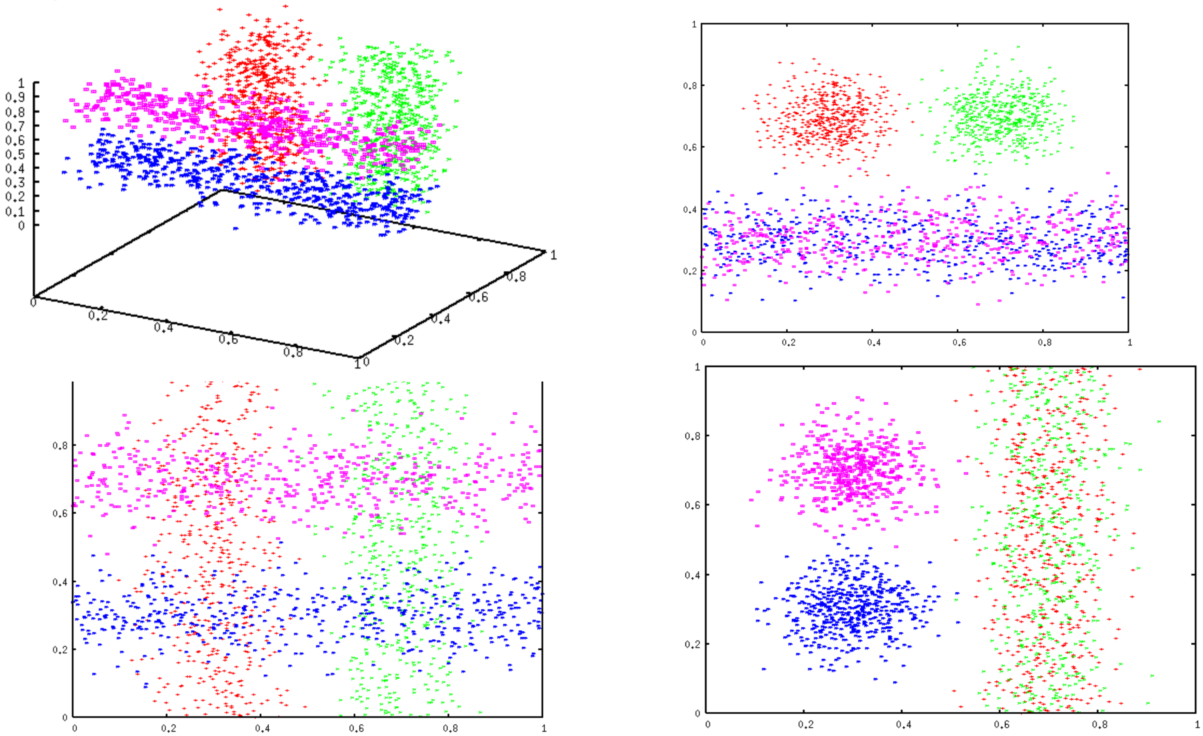
Effect on clustering:

- Usually the distance functions used give equal weight to all dimensions
- However, not all dimensions are of equal importance
- Adding irrelevant dimensions ruins any clustering based on a distance function that equally weights all dimensions



Challenges for Clustering High-Dimensional Data

again: different attributes are relevant for different clusters

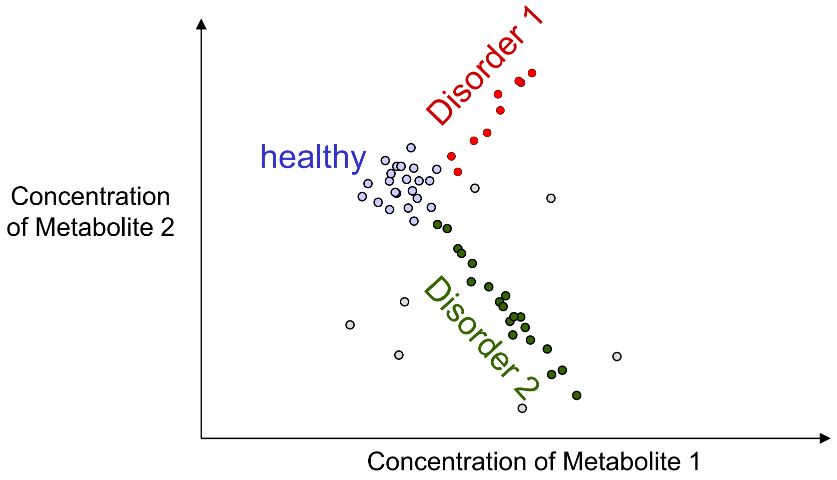


Challenges for Clustering High-Dimensional Data

Task: Cluster test persons to find groups of individuals with similar correlation among the concentrations of metabolites indicating homogeneous metabolic behavior (e.g. disorder)

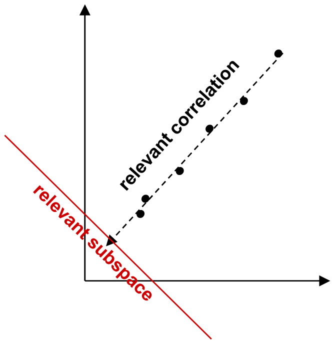
- *Challenge*:

  different metabolic disorders appear through different correlations of (subsets of) metabolites



Challenges for Clustering High-Dimensional Data

- *Correlation among attributes*
  - A subset of features may be correlated
  - Groups of similar ("dense") points may be identified when considering this correlation of features only

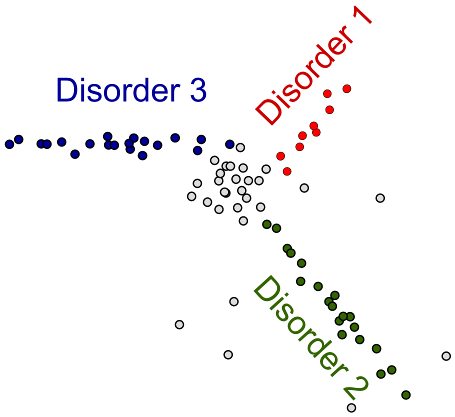


  - Different correlations of attributes may be relevant for different clusters

Challenges for Clustering High-Dimensional Data

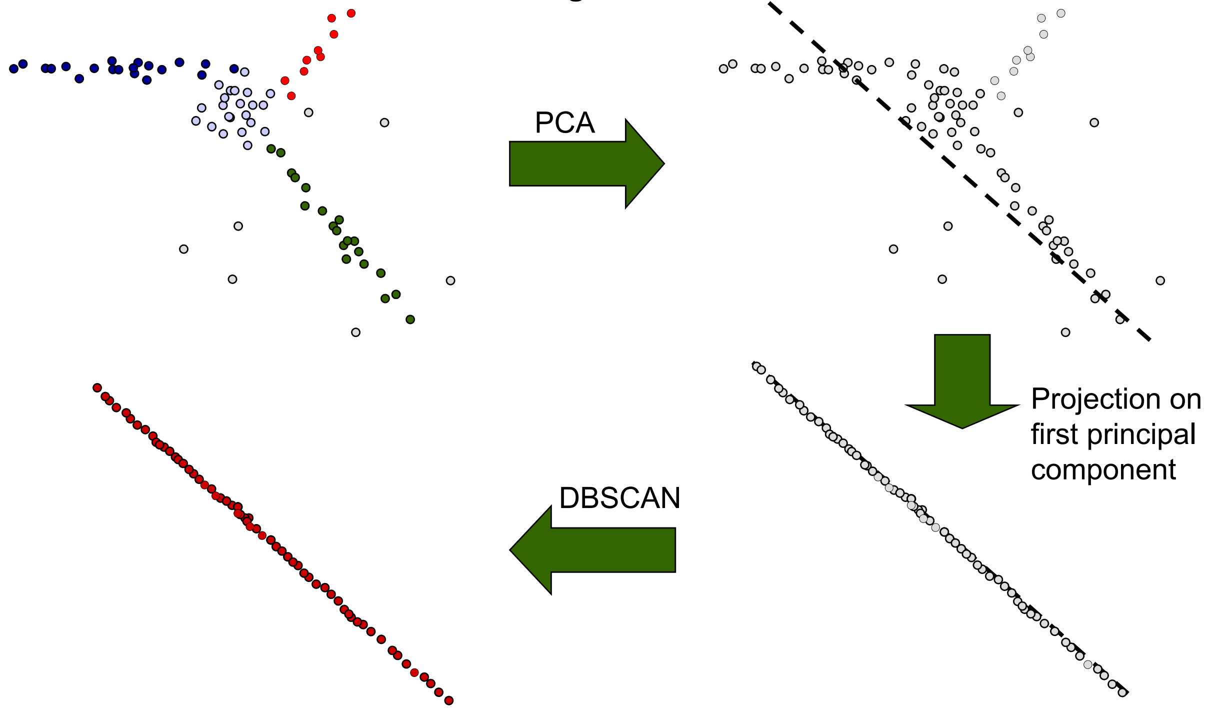
Why not feature selection?

- (Unsupervised) feature selection is global (e.g. PCA)
- We face a local feature relevance/correlation: some features (or combinations of them) may be relevant for one cluster, but may be irrelevant for a second one



Challenges for Clustering High-Dimensional Data

Use feature selection before clustering



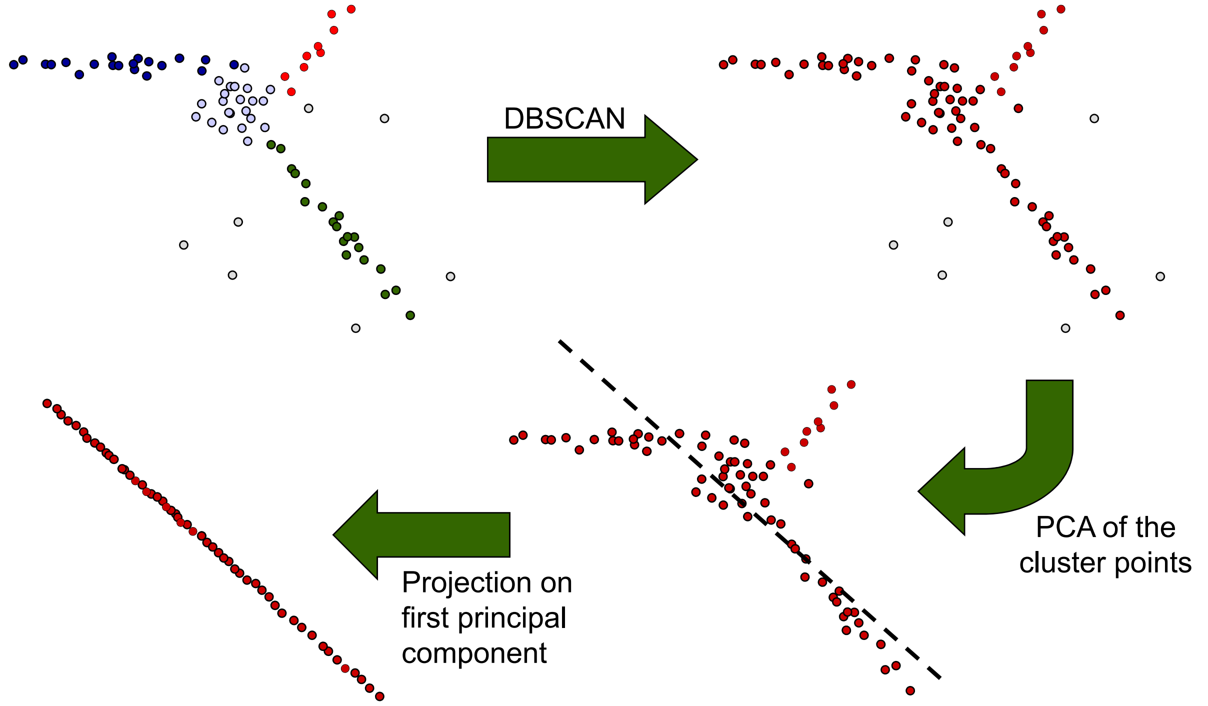
PCA

Projection on first principal component

DBSCAN

Challenges for Clustering High-Dimensional Data

Cluster first and then apply PCA



DBSCAN

PCA of the cluster points

Projection on first principal component

Challenges for Clustering High-Dimensional Data

Problem Summary

- Curse of dimensionality/Feature relevance and correlation
  - Usually, no clusters in the full dimensional space
  - Often, clusters are hidden in subspaces of the data, i.e. only a subset of features is relevant for the clustering
  - E.g. a gene plays a certain role in a subset of experimental conditions
- Local feature relevance/correlation
  - For each cluster, a different subset of features or a different correlation of features may be relevant
  - E.g. different genes are responsible for different phenotypes
- Overlapping clusters
  - Clusters may overlap, i.e. an object may be clustered differently in varying subspaces
  - E.g. a gene plays different functional roles depending on the environment

Challenges for Clustering High-Dimensional Data

- General problem setting of clustering high dimensional data

Search for clusters in
(in general arbitrarily oriented) subspaces
of the original feature space

- Challenges:
  - Find the correct subspace of each cluster
    - Search space:
      - all possible arbitrarily oriented subspaces of a feature space
      - infinite
  - Find the correct cluster in each relevant subspace
    - Search space:
      - “Best” partitioning of points (see: minimal cut of the similarity graph)
      - NP-complete [SCH75]

Challenges for Clustering High-Dimensional Data

- Even worse: ***Circular Dependency***
  - Both challenges 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
  - Thus, we need heuristics to solve
    - the clustering problem
    - the subspace search problem

    simultaneously

Overview over discussed methods

- Buttom-Up approach: Subspace Clustering
  - Clique *[AGGR98]*
  - Subclue *[KKK04]*
- Top-Down Approaches: Correlation and Projected Clustering
  - ProCLUS [APW+99] and ORCLUS[AY00]
  - 4C *[BKKZ04]*
  - CASH
- Pattern based clustering
  - P-Clustering

Bottom-up Algorithms

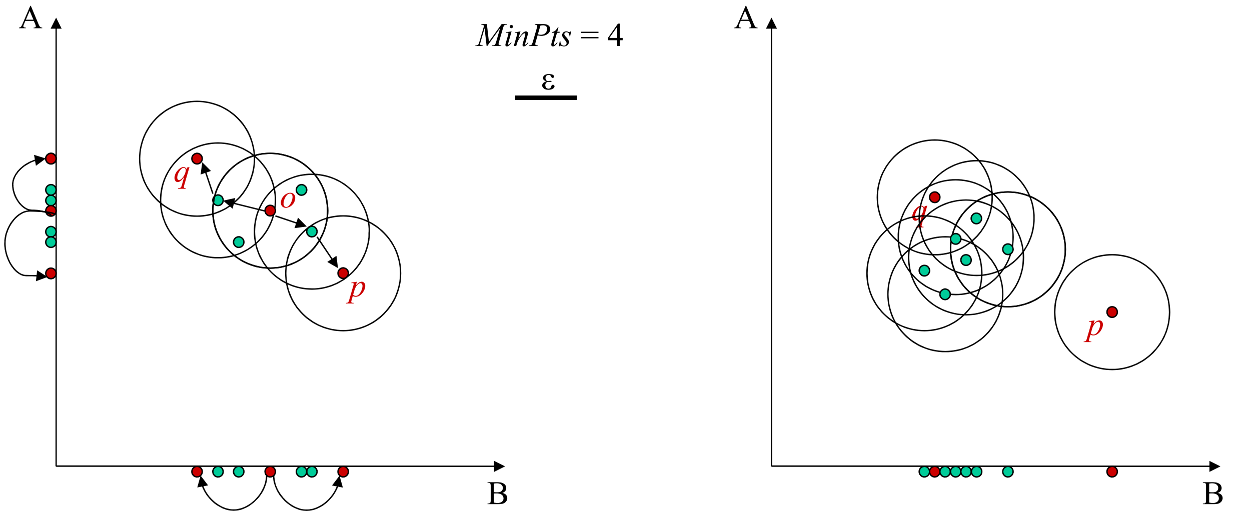
- Rational:
  - Start with 1-dimensional subspaces and merge them to compute higher dimensional ones
  - Most approaches transfer the problem of subspace search into frequent item set mining
    - The cluster criterion must implement the downward closure property
      - If the criterion holds for any *k*-dimensional subspace *S*, then it also holds for any (*k*–1)-dimensional projection of *S*
      - Use the reverse implication for pruning:

        If the criterion does not hold for a (*k*–1)-dimensional projection of *S*, then the criterion also does not hold for *S*
    - Apply any frequent itemset mining algorithm (e.g. APRIORI)
  - Some approaches use other search heuristics like best-first-search, greedy-search, etc.
    - Better average and worst-case performance
    - No guaranty on the completeness of results

Bottom-up Algorithms

Downward-closure property

> if *C* is a dense set of points in subspace *S*,
> then *C* is also a dense set of points in any subspace $T \subset S$



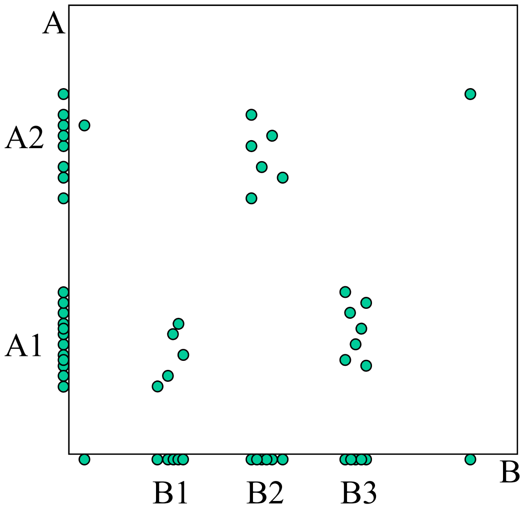
p and *q* density-connected in {A,B}, {A} and {B}

p and *q* not density-connected in {B} and {A,B}

Bottom-up Algorithms

Downward-closure property

the reverse implication does not hold necessarily



CLIQUE [AGGR98]

CLIQUE serves two purposes:

1. Identify subspaces containing clusters
2. Identify the clusters

Approach

- *Cluster*: „dense regions“ in the feature space
- Partition the feature space into ξ equal sized parts in each dimension (regions = grid cells)
- Density threshold τ:

  If region r contain more than τ objects => r is dense
- Clusters are maximal sets of neighboring regions

CLIQUE

Identify subspaces containing clusters

Task: Find dense regions

- Greedy approach (Bottom-Up), comparable to the APRIORI Algorithm for finding frequent Item Set (Downward Closure):
  - Start with the empty set
  - Add one dimension in each step
- Downward Closure for dense grid cells

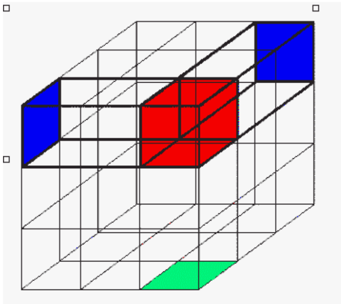
  If region r is dense in a k-dimensional space then each projection of the region into a k-1 dimensional subspace has to be dense as well.

  Inversion:

  If any (k-1) dimensional projection of r is not dense, then r cannot be dense in the k-dimensional feature space.

CLIQUE

Example



🟦 2-dim. dense region

🟥 3-dim. candidate region

🟩 2-dim. region which has to be checked

- If all ξ k-1 dimensional regions are dense
  => check candidate on data set
- heuristics reduction of uninteresting subspace
  => prevents the exponential growth of interesting subspaces

CLIQUE

Identify Clusters

Task: Find maximal sets of connected dense regions

Given: all dense regions in a *k*-dimensional subspace

- „depth-first“-search on the following graph (search space):

    nodes: dense regions

    edges: common hyperplanes (neighboring regions)

- runtime complexity:

Under the condition that dense region can be held in main memory (e.g. in a hash tree)

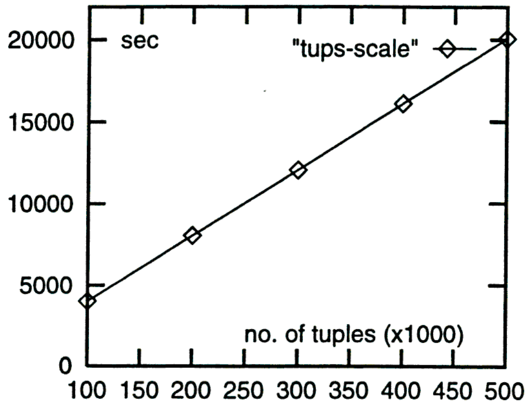
For each region we have to check 2k neighbors

$\Rightarrow$ number of tree accesses: O $(2 \cdot k \cdot n)$

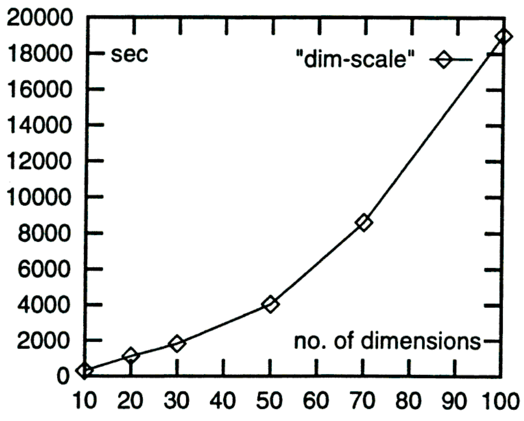
CLIQUE

Experimental Evaluation

runtime vs. number of objects *n*



runtime vs. dimensionality *d*



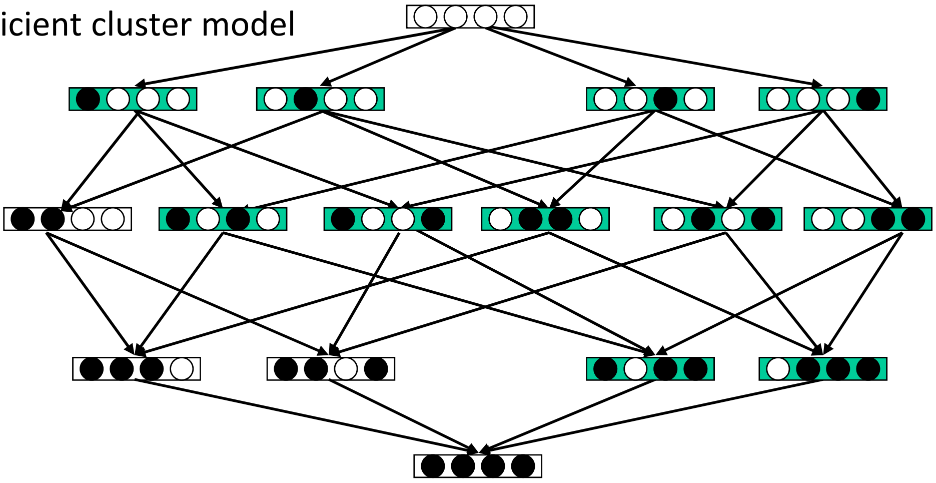
runtime complexity of CLIQUE

linear in *n*, super linear in *d*

CLIQUE

Discussion

- Input: ξ and τ specifying the density threshold
- Output: *all* clusters in *all* subspaces, clusters may overlap
- Uses a fixed density threshold for all subspaces (in order to ensure the downward closure property)
- Simple but efficient cluster model



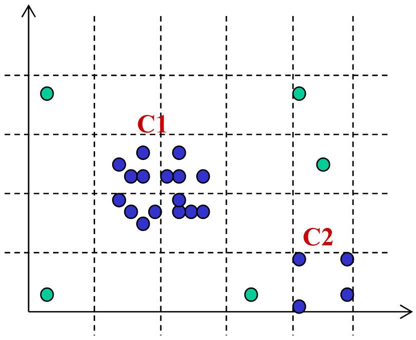
SUBCLU [KKK04]

Motivation:

Drawbacks of a grid-based regions:

- Positioning of the grid influences the clustering
- Only rectangular regions ξ
- Selection of ξ and τ is very sensitiv
  Example:

  Cluster for τ = 4
  (is C2 a cluster?)
  for τ > 4: no cluster
  (C1 is lost)

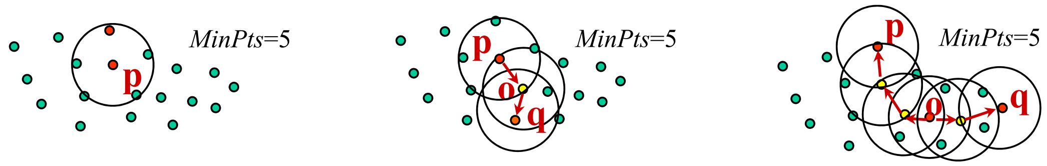


⇒ define regions based on the neighborhood of data points
⇒ use densit-based clustering

SUBCLU

Cluster model:

- Density-based cluster model of DBSCAN
- Clusters are maximal sets of density-connected points
- Density connectivity is defined based on core points
- Core points have at least *MinPts* points in their ε-neighborhood



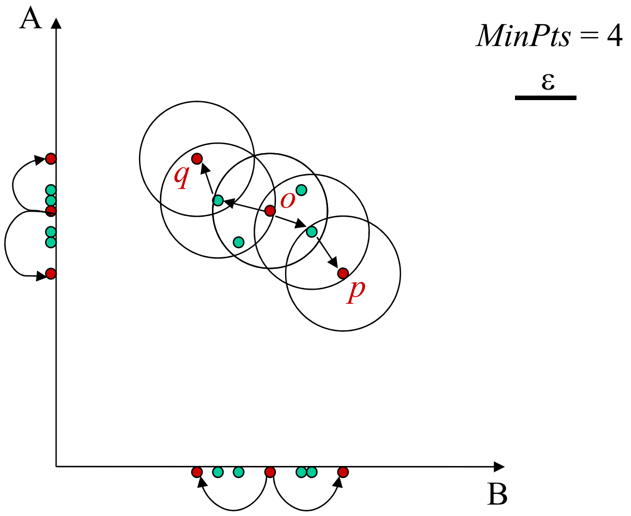
- Detects clusters of arbitrary size and shape (in the corresponding subspaces)
- Downward-closure property holds for sets of density-connected points

SUBCLU

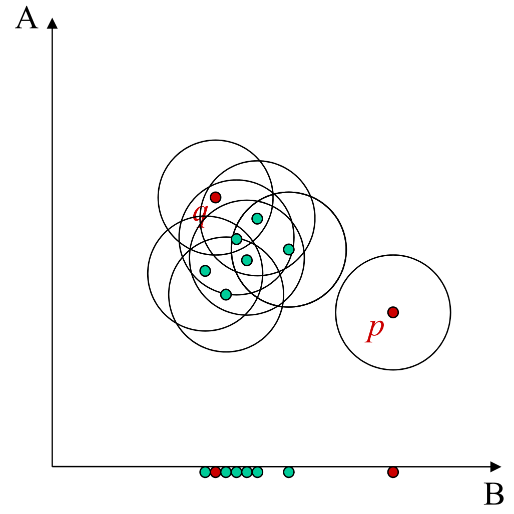
Downward closure of density connected sets

If C is a densitiy connected set in subspace S then C is a density connected set in any subspace $T \subset S$.

(Does not hold for clusters because of the maximality of clusters.)



p and *q* density connected in {A,B}, {A} and {B}



p and *q* not density connected in {B} and {A,B}

SUBCLU

Algorithm

Init:

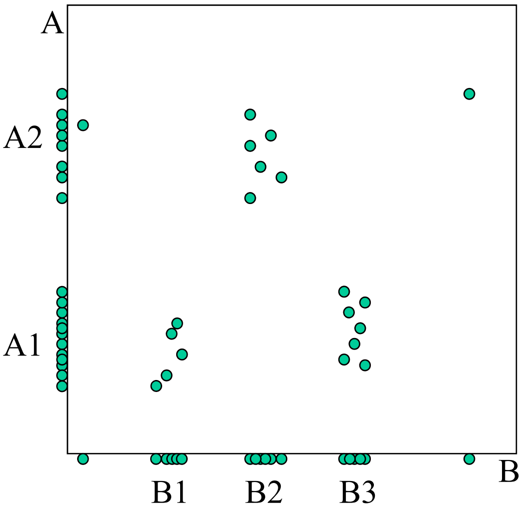
- For each 1 dimensional Subspace S generate all clusters

Loop: Terminate if no k dimensional subspace contains any cluster

- Build: build (k+1) dimensional candidate spaces:
  - Combine subspaces with overlapping dimensions but for one.
  - Prune candidates having a k-dimensional subspace without any cluster
- On each candidate perform DBSCAN to extend the underlying clusters: (noise in any subspace must remain noise for extended spaces)
  - If any cluster is found add candidate subspace to the k+1 subspaces and collect the clusters
  - Else prune the candidate

Remark: Algorithmic pattern is rather close to APRIORI for frequent item set mining.

SUBCLU



Function DBSCAN(*D*, S, ε, *MinPts*) computes all density-connected clusters w.r.t. ε and *MinPts* in data set D and subspace *S*

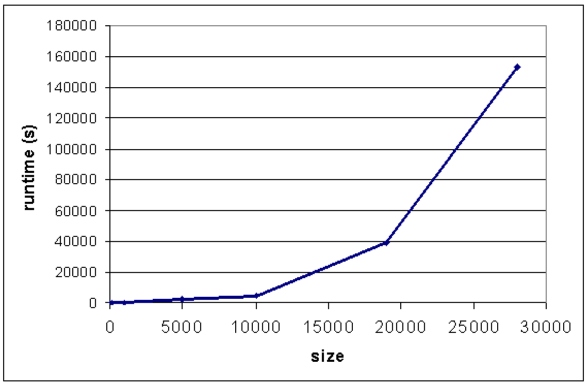
S1 = {{A}, {B}}
C{A} = {A1, A2}
C{B} = {B1, B2, B3}
C1 = {C{A}, C{B}}

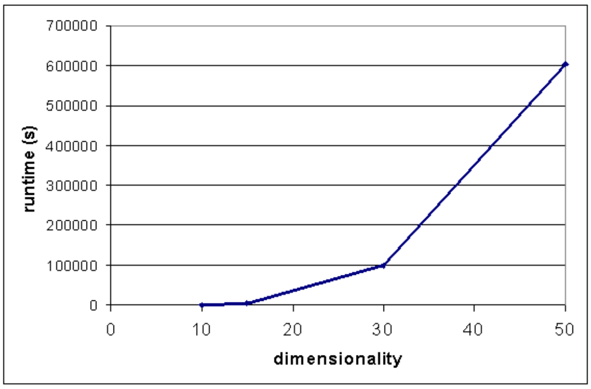
tuning:

- Call DBSCAN(*c*, *U*, ε, *MinPts*) for subspace *U* ⊂ *Cand* having the smallest amount of data objects in clusters (example: *U* = {B})
- Reduces the amout of range queries for each call of DBSCAN minimiert

SUBCLU

Experimental Evaluation





- Scales super linear with the dimension and the number of objects
- Finds more clusters than CLIQUE

Bottom-up Algorithms

The key limitation: ***global density thresholds***

- Usually, the cluster criterion relies on density
- In order to ensure the downward closure property, the density threshold must be fixed
- Consequence: the points in a 20-dimensional subspace cluster must be as dense as in a 2-dimensional cluster
- This is a rather optimistic assumption since the data space grows exponentially with increasing dimensionality
- Consequences:
  - A strict threshold will most likely produce only lower dimensional clusters
  - A loose threshold will most likely produce higher dimensional clusters but also a huge amount of (potentially meaningless) low dimensional clusters

Bottom-up Algorithms

- Algorithm
  - All subspaces that contain any density-connected set are computed using the bottom-up approach
  - Density-connected clusters are computed using a specialized DBSCAN run in the resulting subspace to generate the subspace clusters

- Discussion
  - Input: ε and *MinPts* specifying the density threshold
  - Output: all clusters in all subspaces, clusters may overlap
  - Uses a fixed density threshold for all subspaces
  - Advanced but costly cluster model

Top-down Algorithms

Rational:

- Cluster-based approach:
  - Learn the subspace of a cluster in the ***entire*** *d*-dimensional feature space
  - Start with full-dimensional clusters
  - Iteratively refine the cluster memberships of points and the subspaces of the cluster

- Instance-based approach:
  - Learn for each point its subspace preference in the ***entire*** *d*-dimensional feature space
  - The subspace preference specifies the subspace in which each point "clusters best"
  - Merge points having similar subspace preferences to generate the clusters

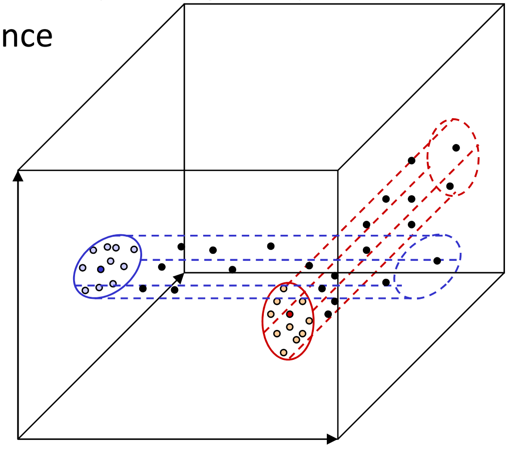
Top-down Algorithms

The key problem:
How should we learn the subspace preference of a cluster or a point?

- Most approaches rely on the so-called "locality assumption"
  - The subspace is usually learned from the local neighborhood of cluster representatives/cluster members in the entire feature space:
    - Cluster-based approach: the ***local neighborhood*** of each cluster representative is evaluated in the *d*-dimensional space to learn the "correct" subspace of the cluster
    - Instance-based approach: the ***local neighborhood*** of each point is evaluated in the *d*-dimensional space to learn the "correct" subspace preference of each point
- ***The locality assumption***: the subspace preference can be learned from the ***local neighborhood*** in the *d*-dimensional space
  - Other approaches learn the subspace preference of a cluster or a point from ***randomly sampled points***

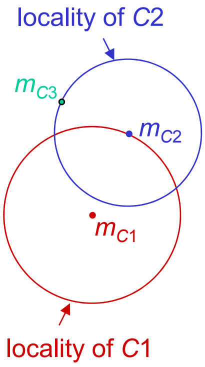
Top-down Algorithms

- PROCLUS [APW+99]
  - *K*-medoid cluster model
    - Cluster is represented by its medoid
    - To each cluster a subspace (of relevant attributes) is assigned
    - Each point is assigned to the nearest medoid (where the distance to each medoid is based on the corresponding subspaces of the medoids)
    - Points that have a large distance to its nearest medoid are classified as noise



Top-down Algorithms

- 3-Phase Algorithm
  - Initialization of cluster medoids
    - A superset *M* of *b·k* medoids is computed from a sample of *a·k* data points such that these medoids are well separated
    - *k* randomly chosen medoids from *M* are the initial cluster representatives
    - Input parameters *a* and *b* are introduced for performance reasons
  - Iterative phase works similar to any *k*-medoid clustering
    - Approximate subspaces for each cluster *C*



      - » The locality of *C* includes all points that have a distance to the medoid of *C* less than the distance between the medoid of *C* and the medoid of the neighboring cluster
      - » Compute standard deviation of distances from the medoid of *C* to the points in the locality of *C* along each dimension
      - » Add the dimensions with the smallest standard deviation to the relevant dimensions of cluster *C* such that
        - in summary *k·l* dimensions are assigned to all clusters
        - each cluster has at least 2 dimensions assigned

Top-down Algorithms

- Reassign points to clusters
  - » Compute for each point the distance to each medoid taking only the relevant dimensions into account
  - » Assign points to a medoid minimizing these distances
- Termination (criterion not really clearly specified in [APW+99])
  - » Terminate if the clustering quality does not increase after a given number of current medoids have been exchanged with medoids from *M*

    (it is not clear, if there is another hidden parameter in that criterion)

- Refinement
  - Reassign subspaces to medoids as above (but use only the points assigned to each cluster rather than the locality of each cluster)
  - Reassign points to medoids; points that are not in the locality of their corresponding medoids are classified as noise

Correlation Clustering Algorithms

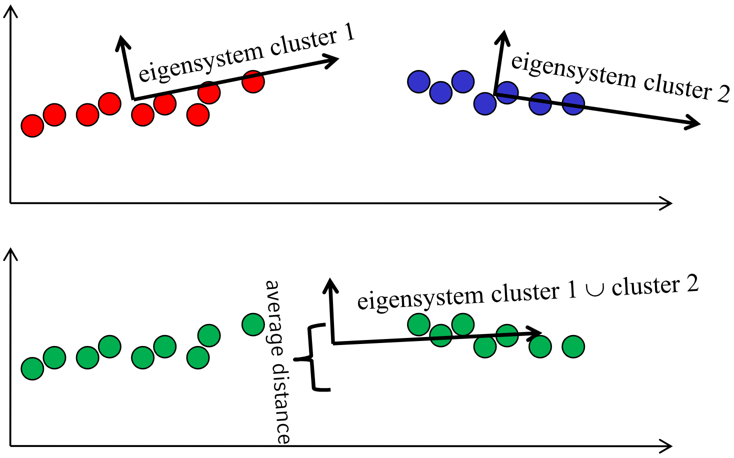
ORCLUS [AY00]:

first approach to *generalized projected clustering*

- similar ideas to PROCLUS [APW+99]
- *k*-means like approach
- start with $k_c > k$ seeds
- assign cluster members according to distance function based on the eigensystem of the current cluster (starting with axes of data space, i.e. Euclidean distance)
- reduce k_c in each iteration by merging best-fitting cluster pairs

Correlation Clustering Algorithms

- best fitting pair of clusters: least average distance in the projected space spanned by weak eigenvectors of the merged clusters



- assess average distance in all merged pairs of clusters and finally merge the best fitting pair

Top-down Algorithms

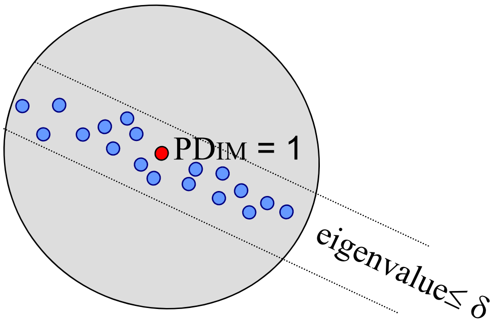
- Discussion
  - Input:
    - Number of clusters *k*
    - Average dimensionality of clusters *l*
    - Factor *a* to determine the size of the sample in the initialization step
    - Factor *b* to determine the size of the candidate set for the medoids
  - Output: partitioning of points into *k* disjoint clusters and noise, each cluster has a set of relevant attributes specifying its subspace
  - Relies on cluster-based locality assumption: subspace of each cluster is learned from local neighborhood of its medoid
  - Biased to find *l*-dimensional subspace clusters
  - Simple but efficient cluster model

4C [BKKZ04]

Idea: Integrate PCA into into density-based clustering.
4C = Compution Correlation Connected Clusters

Approach:

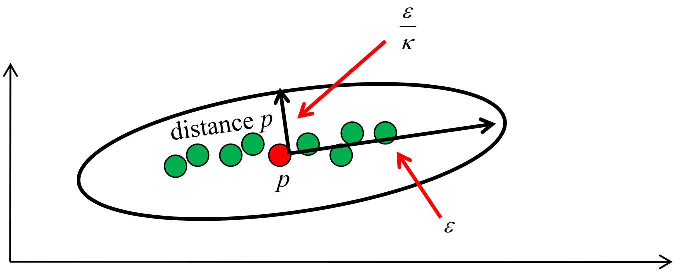
- Determine the core point property in the complete space
- Perform PCA on the local neighborhood to local determine subspace correlations



PCA factorizes M_P into $M_P = V E V^T$
V: eigenvectors
E: eigenvalues

Correlation Clustering Algorithms

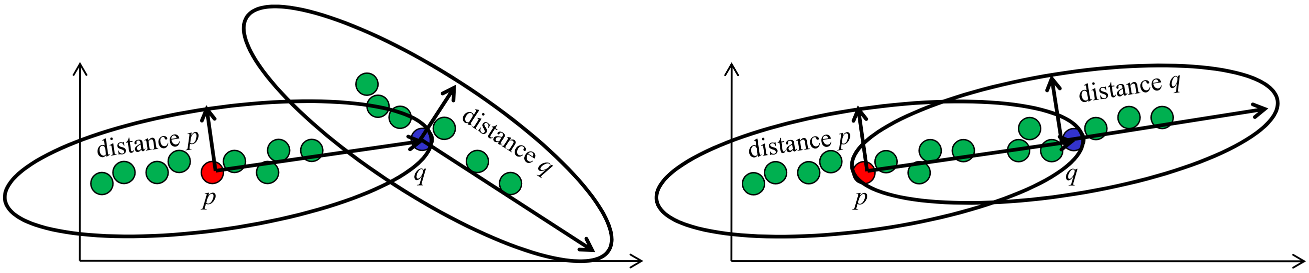
- effect on distance measure:



- distance of *p* and *q* w.r.t. *p*: $\sqrt{(p-q)\cdot V_p \cdot E'_p \cdot V_p^{\mathrm{T}} \cdot (p-q)^{\mathrm{T}}}$
- distance of *p* and *q* w.r.t. *q*: $\sqrt{(q-p)\cdot V_q \cdot E'_q \cdot V_q^{\mathrm{T}} \cdot (q-p)^{\mathrm{T}}}$

Correlation Clustering Algorithms

- symmetry of distance measure by choosing the maximum:



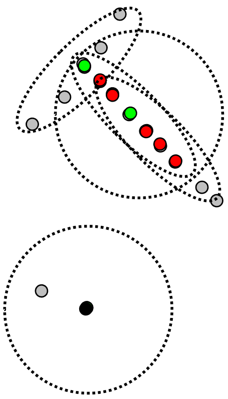
- *p* and *q* are correlation-neighbors if

$$\max\left\{\begin{array}{l}\sqrt{(p-q)\cdot V_p \cdot E'_p \cdot V_p^{\mathrm{T}} \cdot (p-q)^{\mathrm{T}}},\\ \sqrt{(q-p)\cdot V_q \cdot E'_q \cdot V_q^{\mathrm{T}} \cdot (q-p)^{\mathrm{T}}}\end{array}\right\} \le \varepsilon$$

4C

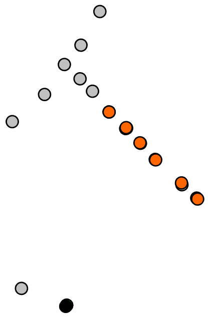
```
Algorithm 4C ($DB$, $\varepsilon$, $\mu$, $\lambda$, $\delta$)
// assumption: each point in $DB$ is marked
// as unclassified
for each unclassified $O \in DB$ do
    compute $N_\varepsilon(O)$;
    if $|N_\varepsilon(O)| \geq \mu$ then
        if CorDim( $N_\varepsilon(O)$ ) $\leq \lambda$ then
            if $|N_\varepsilon^{M'_O}(O)| \geq \mu$ then
                expand a new cluster;
    In all other cases: mark $O$ as noise;
```

μ = 3



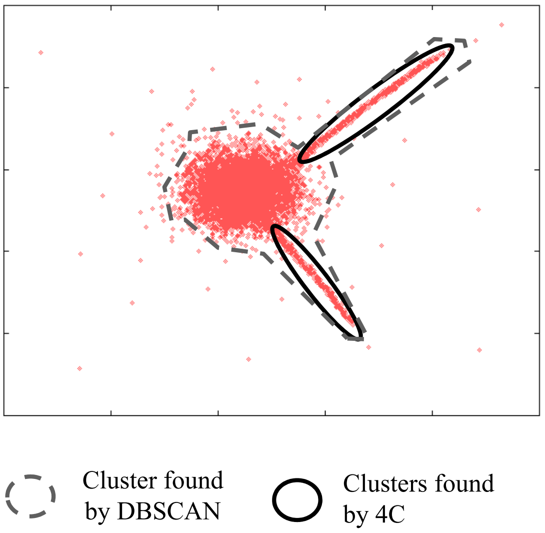
4C

```
//expand cluster
generate new clusterID;
Insert all $X$ with DirCorReach(O,X) into queue $\Phi$;
while $\Phi \neq \emptyset$ do
    $Q$ = first point in $\Phi$;
    compute $N_\varepsilon^{M'_Q}(Q)$ ;
    for each $X$ with DirCorReach(Q,X) do
        if $X$ is unclassified or noise then
            assign current clusterID to $X$;
        if $X$ is unclassified then
            insert $X$ into $\Phi$;
    remove $Q$ from $\Phi$;
```



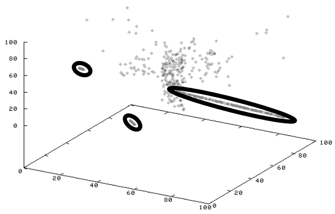
4C

4C vs. DBSCAN

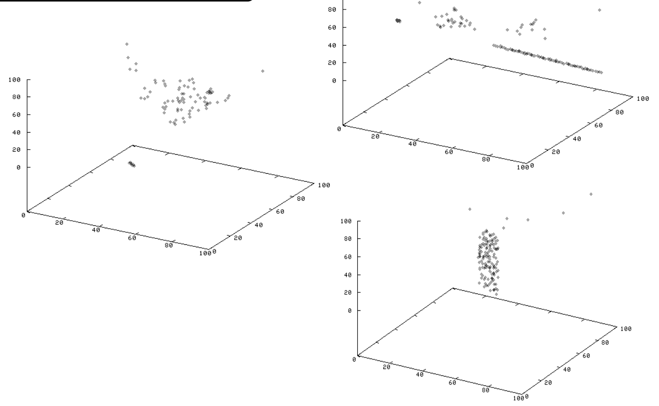


4C vs. ORCLUS

4C



ORCLUS



Correlation Clustering Algorithms

properties:

- finds arbitrary number of clusters
- requires specification of density-thresholds
  - – μ (minimum number of points): rather intuitive
  - – ε (radius of neighborhood): hard to guess
- biased to maximal dimensionality λ of correlation clusters (user specified)
- instance-based locality assumption: correlation distance measure specifying the subspace is learned from local neighborhood of each point in the *d*-dimensional space

enhancements also based on PCA:

- COPAC [ABK+07c] and
- ERiC [ABK+07b]

Use Hough Transformation to determine Clusters

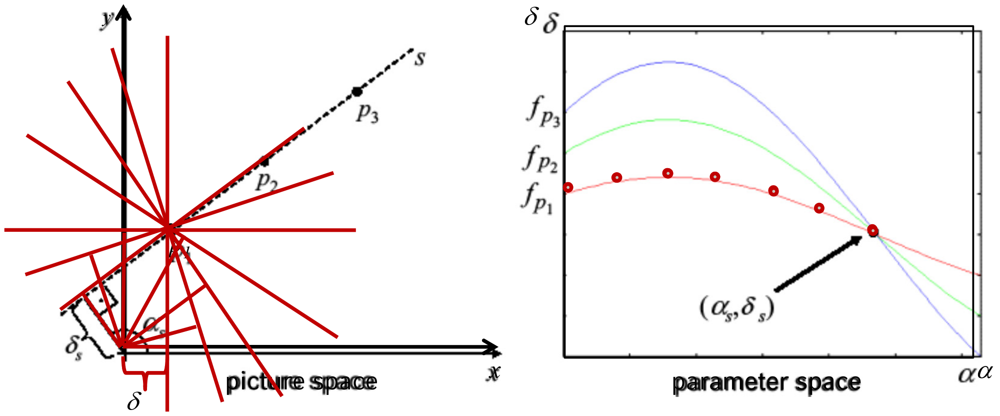
- Hough-Transformation

  Known from image analysis (finds geometric primitives lines, circles..) in 2D pixel images
- Extension to arbitrary dimensions
- Transfers clustering into a new space ("parameter space" of the Hough transform)
- reduces the search space from not countable infinity to *O(n!)*
- Common search heuristic is full enumeration

=> *For efficient clustering a better heuristic is necessary!!*

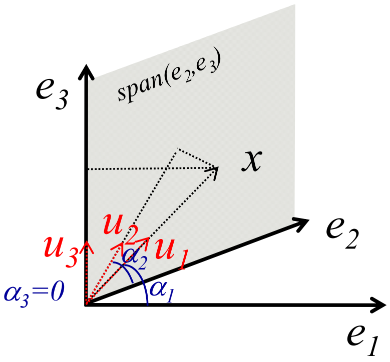
Hough-Transformation

- Given: $D \subseteq IR^d$
- target: linear subspaces, containing many points x $x \in D$
- Idea: Maps points from the data space (picture space) to functions in the parameters space



d-dimensional Polar Coordinates

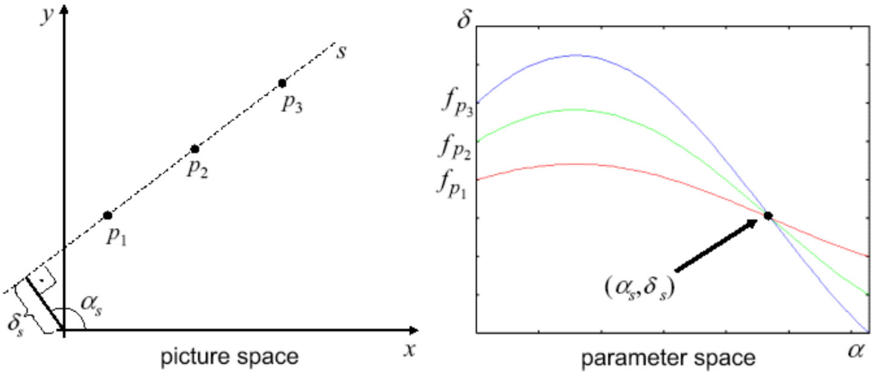
- e_i, $1 \leq i \leq d$: Orthonormal basis
- $x = (x_1, \ldots, x_d)^T$: d-dimensional Vector on the hyper sphere around the origin with radius r
- u_i: unity vector in the direction of the projection of x to the subspace $span(e_i, \ldots, e_d)$
- $\alpha_1, \ldots, \alpha_{d-1}$: α_i angle between u_i and e_i



$$x_i = r \cdot \left(\prod_{j=1}^{i-1} \sin(\alpha_j) \right) \cdot \cos(\alpha_i)$$

Correlation Clustering Algorithms

- points in data space are mapped to functions in the parameter space

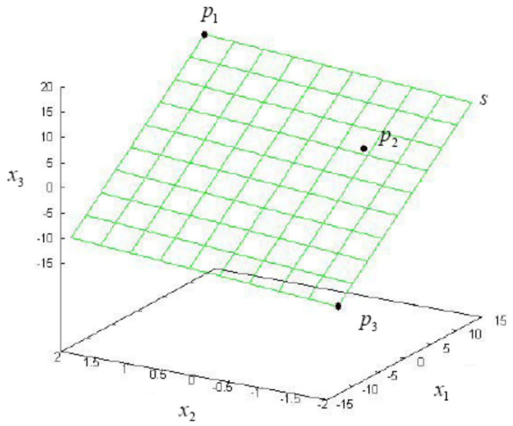


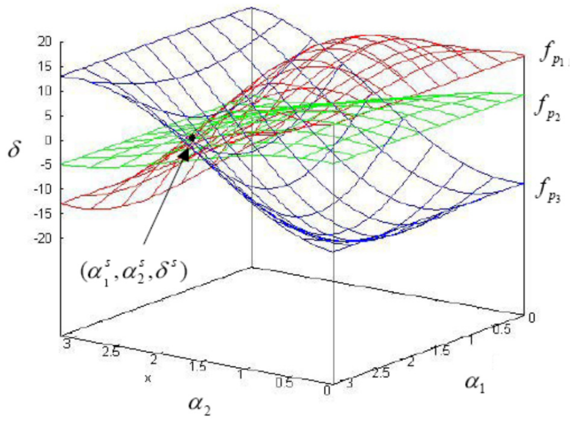
$$f_p(\alpha_1, \ldots, \alpha_{d-1}) = \langle p, n \rangle = \sum_{i=1}^{d} p_i \cdot \left(\prod_{j=1}^{i-1} \sin(\alpha_j) \right) \cdot \cos(\alpha_i)$$

- functions in the parameter space define all lines possibly crossing the point in the data space

Properties of the transformation

- Point in the data space = sinusoidal curve in parameter space
- Point in parameter space = hyper-plane in data space
- Points on a common hyper-plane in data space = sinusoidal curves through a common point in parameter space
- Intersections of sinusoidal curves in parameter space = hyper-plane through the corresponding points in data space



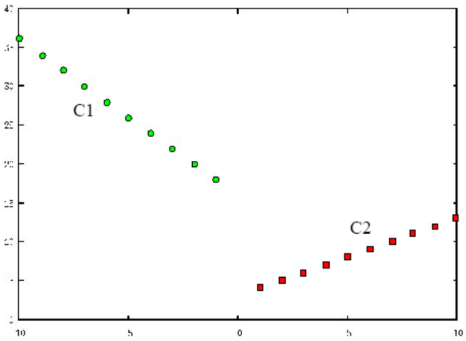


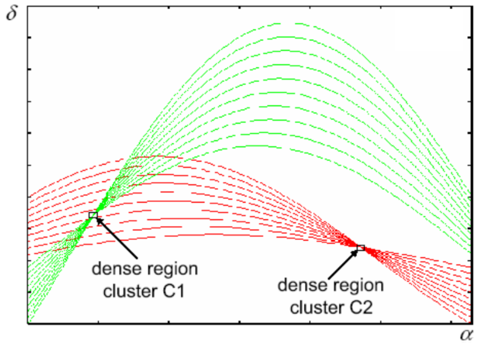
Correlation Clustering based on the Hough-Transformation

- Dense region in parameter space ⇔ lineare regions in the data space

  (hyper planes wherer $\lambda \leq d\text{-}1$)
- Exact solutions: Determine all Intersections
  - Computation too expensive
  - Too exact to find linear clusters
- approximative solution: gridbased clustering in parameter spaces

  $\rightarrow$ determine grid cells intersecting at least m sinusoids
  - Search space is finite but in $O(r^d)$
  - Cluster quality depends on the resolutio r (Auflösung des Grids)





Correlation Clustering Algorithms

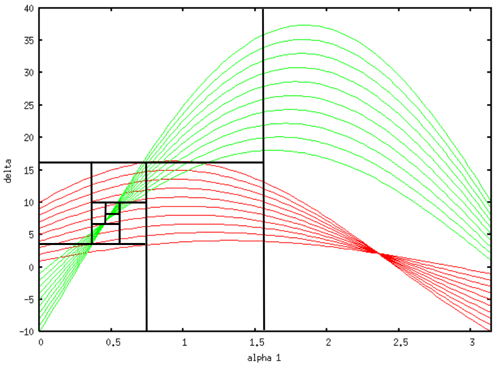
Idea: find dense regions in parameter space

- construct a grid by recursively splitting the parameter space (best-first-search)
- identify dense grid cells as intersected by many parametrization functions
- dense grid represents (*d-1*)-dimensional linear structure
- transform corresponding data objects in corresponding (*d-1*)-dimensional space and repeat the search recursively

Algorithmus CASH: effiziente Suchheuristik

CASH: Clustering in Arbitrary Subspaces based on the Hough-Transform []

- Parameter space is recursively partitioned per axis in a predefined order $[\alpha_1, \dots, \alpha_{d-1}, \delta]$
- Select the hyper rectangle representing the most points to continue (Best-First Search)



Algorithm CASH: efficient search heuristics

- Hyper rectangle representing less than m points can be pruned from the search space → early determination of the search path
- Hyper rectangles intersecting at least m sinusoids after s recursive partitionings represent correlation clusters (where $\lambda \leq d-1$)
  - Cluster points (i.e. sinusoids) are removed from any other hyper rectangle
  - To detect correlation clusters in subspaces with $\lambda \leq d-2$:
    recursive processing of the cluster after transformation into the corresponding $d-1$-dimensional subspace

Algorithmus CASH: Characateristics

- Detects an arbitrary amount of cluster
- Required input:
  - search depth (number of splits ⇔ maximal size of a cluster cell/accuracy)
  - minimal density of a cell (⇔ minimal number of point in a cluster)
- Density of a cell is not based on the “locality assumption”

  => method for global correlation clustering
- In average the search heuristic scales with $\sim d^3$
- BUT: worst case runtime degenerates to exhaustive search (exponential growth in d)

Correlation Clustering Algorithms

properties:

- finds arbitrary number of clusters
- requires specification of depth of search (number of splits per axis)
- requires minimum density threshold for a grid cell
- Note: this minimum density does not relate to the locality assumption: CASH is a global approach to correlation clustering
- search heuristic: linear in number of points, but ~ d^4
- But: complete enumeration in worst case (exponential in d)

Summary and Perspectives

- PCA: mature technique, allows construction of a broad range of similarity measures for local correlation of attributes
- drawback: all approaches suffer from locality assumption
- successfully employing PCA in correlation clustering in “really” high-dimensional data requires more effort henceforth
- new approach based on Hough-transform:
  - does not rely on locality assumption
  - but worst case again complete enumeration

Summary and Perspectives

- some preliminary approaches base on concept of self-similarity (intrinsic dimensionality, fractal dimension): [BC00,PTTF02,GHPT05]
- interesting idea, provides quite a different basis to grasp correlations in addition to PCA
- drawback: self-similarity assumes locality of patterns even by definition

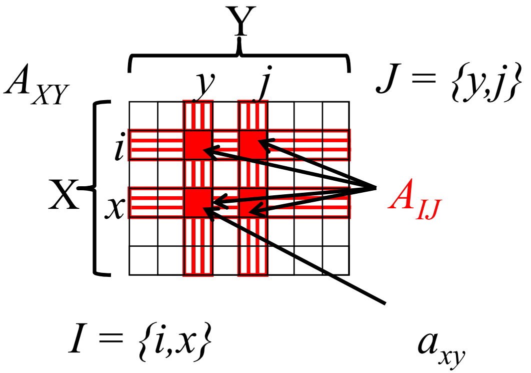
Outline: Pattern-based Clustering

- Challenges and Approaches, Basic Models for
  - Constant Biclusters
  - Biclusters with Constant Values in Rows or Columns
  - Pattern-based Clustering: Biclusters with Coherent Values
  - Biclusters with Coherent Evolutions
- Algorithms for
  - Constant Biclusters
  - Pattern-based Clustering: Biclusters with Coherent Values
- Summary

Challenges and Approaches, Basic Models

Pattern-based clustering relies on patterns in the data matrix.

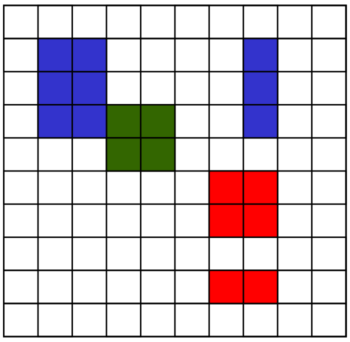
- Simultaneous clustering of rows and columns of the data matrix (hence *bi*clustering).
  - Data matrix ***A*** = (X,Y) with set of rows X and set of columns Y
  - a_{xy} is the element in row *x* and column *y*.
  - submatrix A_{IJ} = (I,J) with subset of rows I $\subseteq$ X and subset of columns J $\subseteq$ Y contains those elements a_{ij} with *i* $\in$ I und *j* $\in$ J



Challenges and Approaches, Basic Models

General aim of biclustering approaches:

Find a set of submatrices $\{(I_1,J_1),(I_2,J_2),...,(I_k,J_k)\}$ of the matrix ***A***=(X,Y) (with $I_i \subseteq$ X and $J_i \subseteq$ Y for *i* = 1,...,k) where each submatrix (= bicluster) meets a given homogeneity criterion.



Challenges and Approaches, Basic Models

- Some values often used by bicluster models:
  - mean of row *i*:

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}$$

  - mean of column *j*:

$$a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$$

  - mean of all elements:

$$a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij} = \frac{1}{|J|} \sum_{j \in J} a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{iJ}$$

Challenges and Approaches, Basic Models

Different types of biclusters (cf. [MO04]):

- constant biclusters
- biclusters with
  - constant values on columns
  - constant values on rows
- biclusters with coherent values (aka. pattern-based clustering)
- biclusters with coherent evolutions

Challenges and Approaches, Basic Models

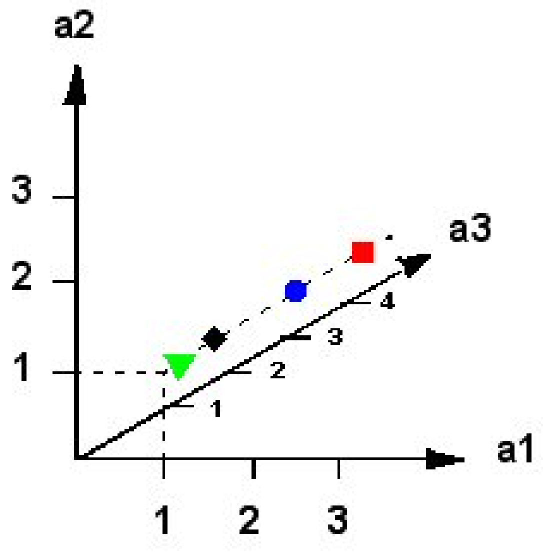
Constant biclusters

- all points share identical value in selected attributes.
- The constant value μ is a typical value for the cluster.
- Cluster model: $a_{ij} = \mu$
- Obviously a special case of an axis-parallel subspace cluster.

Challenges and Approaches, Basic Models

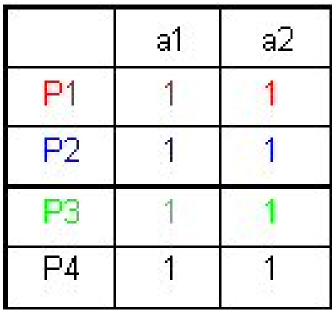
- example – embedding 3-dimensional space:

	a1	a2	a3
P1	1	1	3.5
P2	1	1	2.3
P3	1	1	0.2
P4	1	1	0.7

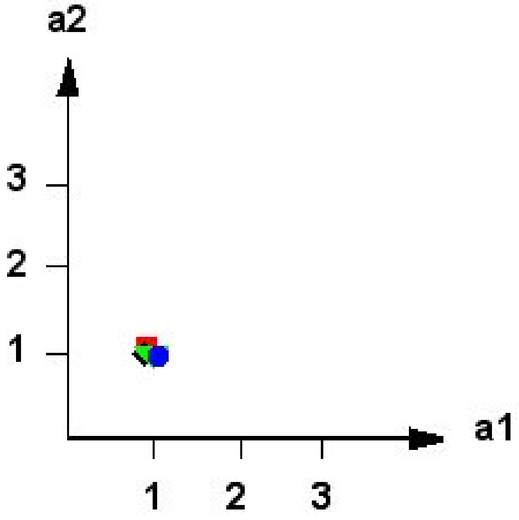


Challenges and Approaches, Basic Models

- example – 2-dimensional subspace:



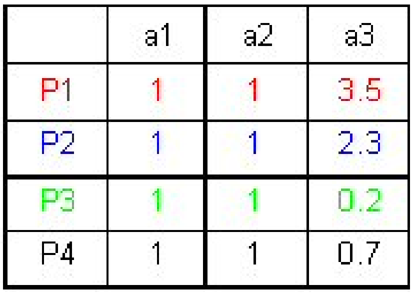
	a1	a2
P1	1	1
P2	1	1
P3	1	1
P4	1	1



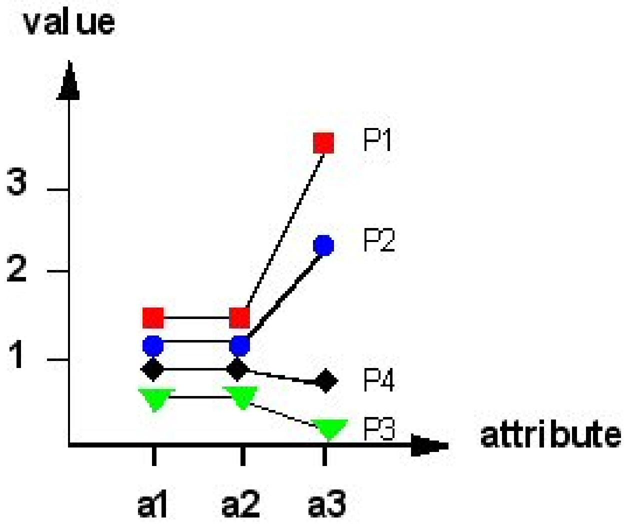
- points located on the bisecting line of participating attributes

Challenges and Approaches, Basic Models

- example – transposed view of attributes:



	a1	a2	a3
P1	1	1	3.5
P2	1	1	2.3
P3	1	1	0.2
P4	1	1	0.7



- pattern: identical constant lines

Challenges and Approaches, Basic Models

- real-world constant biclusters will not be perfect
- cluster model relaxes to: $a_{ij} \approx \mu$
- Optimization on matrix *A* = (X,Y) may lead to |X|·|Y| singularity-biclusters each containing one entry.
- Challenge: Avoid this kind of overfitting.

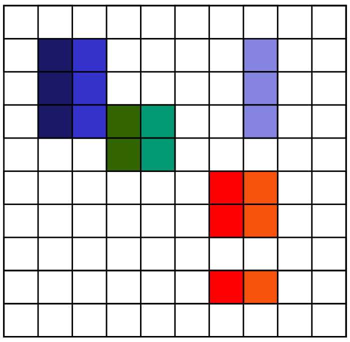
Challenges and Approaches, Basic Models

Biclusters with constant values on columns

- Cluster model for A_{IJ} = (I,J):

$$a_{ij} = \mu + c_j$$

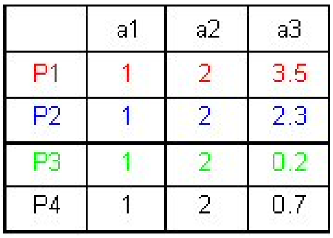
$$\forall i \in I, j \in J$$



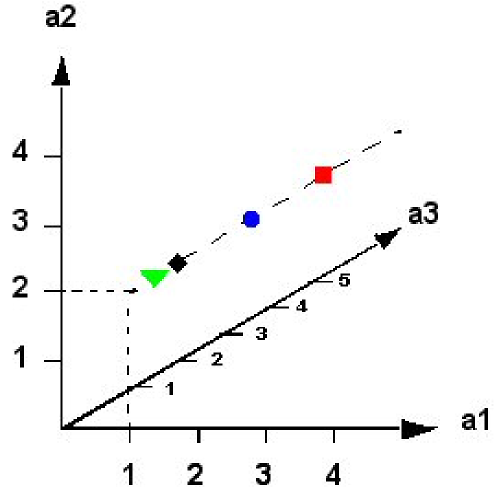
- adjustment value c_j for column *j* ∈ J
- results in axis-parallel subspace clusters

Challenges and Approaches, Basic Models

- example – 3-dimensional embedding space:

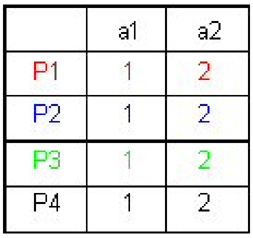


	a1	a2	a3
P1	1	2	3.5
P2	1	2	2.3
P3	1	2	0.2
P4	1	2	0.7

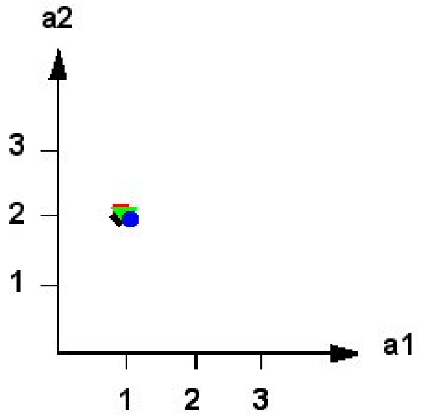


Challenges and Approaches, Basic Models

- example – 2-dimensional subspace:



	a1	a2
P1	1	2
P2	1	2
P3	1	2
P4	1	2

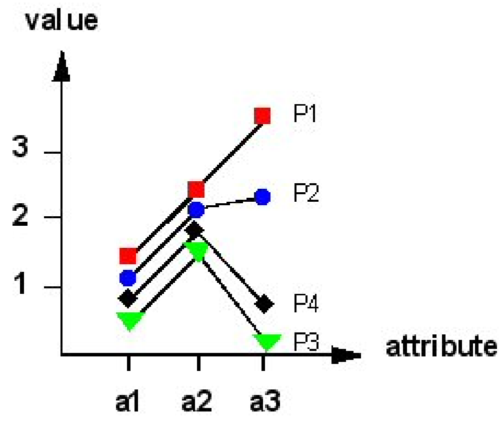


DATABASE SYSTEMS GROUP

Challenges and Approaches, Basic Models

- example – transposed view of attributes:

	a1	a2	a3
P1	1	2	3.5
P2	1	2	2.3
P3	1	2	0.2
P4	1	2	0.7



- pattern: identical lines

DATABASE SYSTEMS GROUP

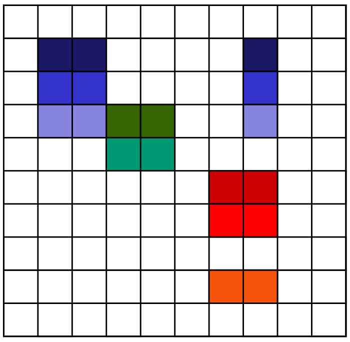
Challenges and Approaches, Basic Models

Biclusters with constant values on rows

- Cluster model for A_{IJ} = (I,J):

$$a_{ij} = \mu + r_i$$

$$\forall i \in I, j \in J$$

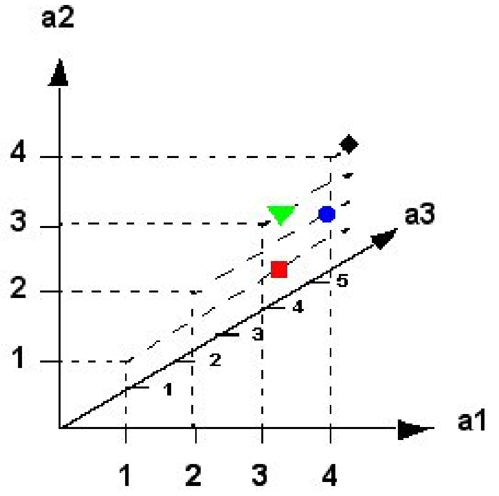


- adjustment value r_i for row $i \in$ I

Challenges and Approaches, Basic Models

- example – 3-dimensional embedding space:

	a1	a2	a3
P1	1	1	3.5
P2	2	2	2.3
P3	3	3	0.2
P4	4	4	0.7

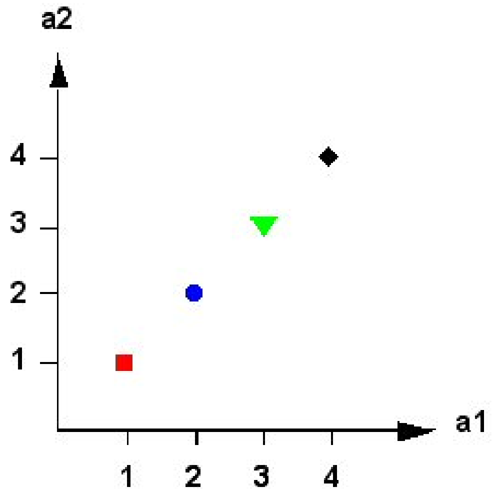


- in the embedding space, points build a sparse hyperplane parallel to irrelevant axes

Challenges and Approaches, Basic Models

- example – 2-dimensional subspace:

	a1	a2
P1	1	1
P2	2	2
P3	3	3
P4	4	4

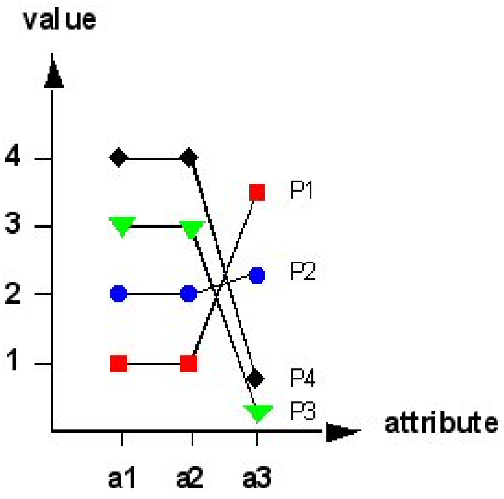


- points are accommodated on the bisecting line of participating attributes

Challenges and Approaches, Basic Models

- example – transposed view of attributes:

	a1	a2	a3
P1	1	1	3.5
P2	2	2	2.3
P3	3	3	0.2
P4	4	4	0.7



- pattern: parallel constant lines

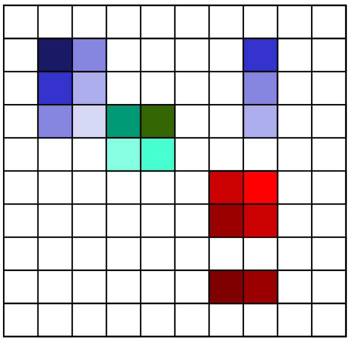
Challenges and Approaches, Basic Models

Biclusters with coherent values

- based on a particular form of covariance between rows and columns

$$a_{ij} = \mu + r_i + c_j$$

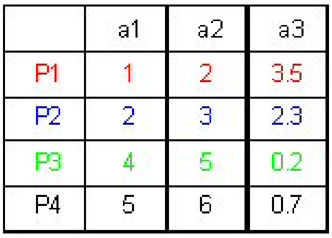
$$\forall i \in I, j \in J$$



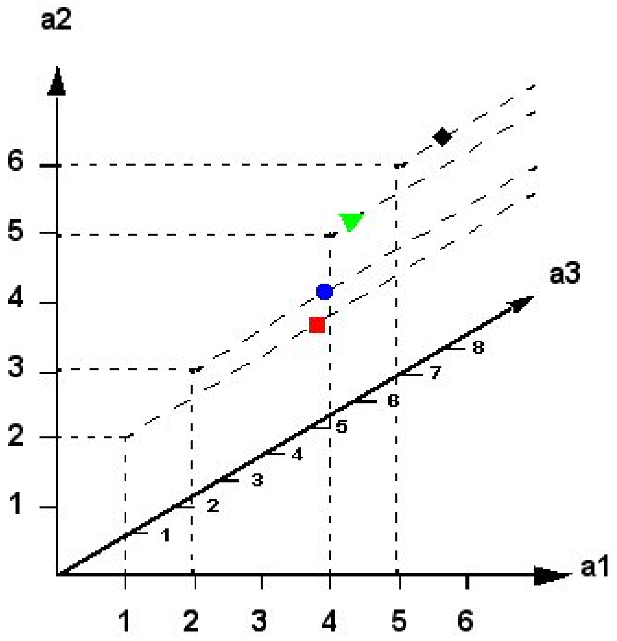
- special cases:
  - c_j = 0 for all j → constant values on rows
  - r_i = 0 for all i → constant values on columns

Challenges and Approaches, Basic Models

- embedding space: sparse hyperplane parallel to axes of irrelevant attributes

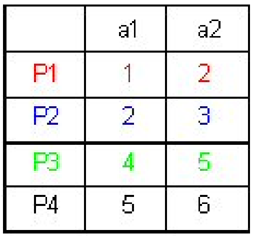


	a1	a2	a3
P1	1	2	3.5
P2	2	3	2.3
P3	4	5	0.2
P4	5	6	0.7

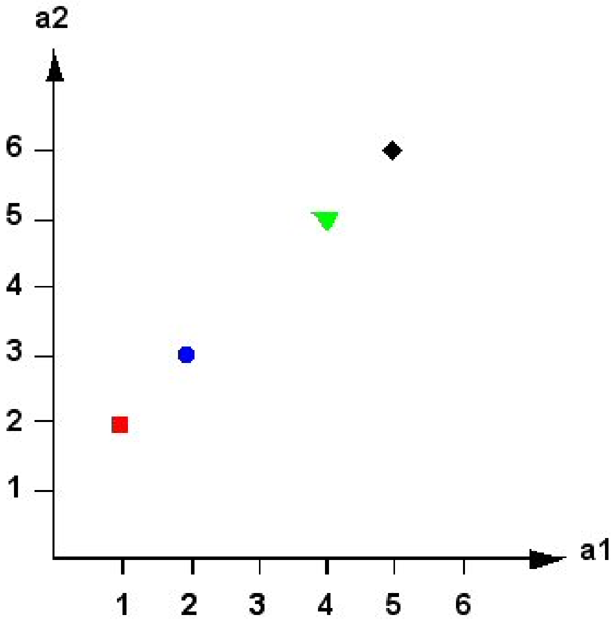


Challenges and Approaches, Basic Models

- subspace: increasing one-dimensional line



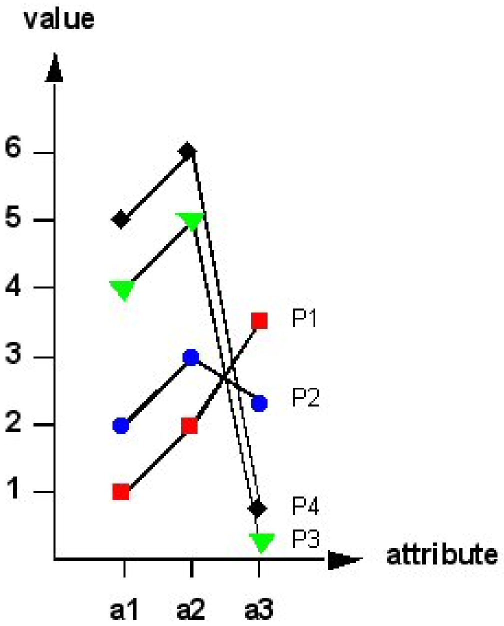
	a1	a2
P1	1	2
P2	2	3
P3	4	5
P4	5	6



Challenges and Approaches, Basic Models

- transposed view of attributes:

	a1	a2	a3
P1	1	2	3.5
P2	2	3	2.3
P3	4	5	0.2
P4	5	6	0.7

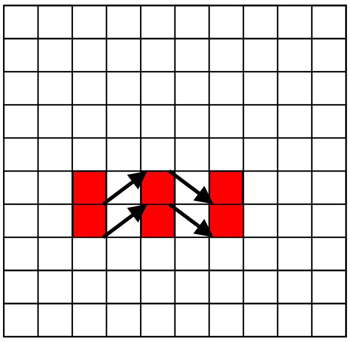


- pattern: parallel lines

Challenges and Approaches, Basic Models

Biclusters with coherent evolutions

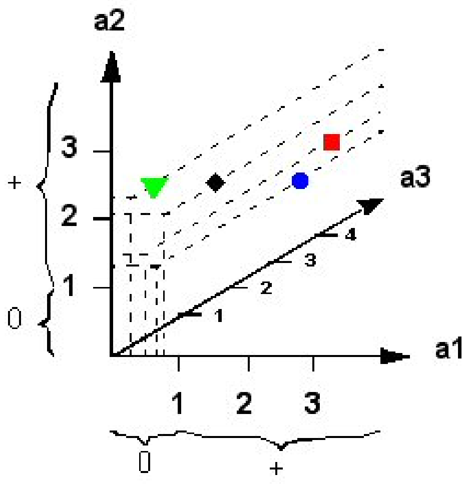
- for all rows, all pairs of attributes change simultaneously
  - discretized attribute space: coherent state-transitions
  - change in same direction irrespective of the quantity



Challenges and Approaches, Basic Models

- Approaches with coherent state-transitions: [TSS02,MK03]
- reduces the problem to grid-based axis-parallel approach:

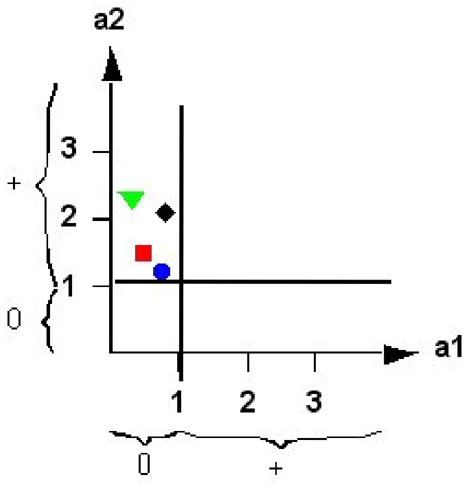
	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	2.3	0.2
P4	0.8	2.1	0.7

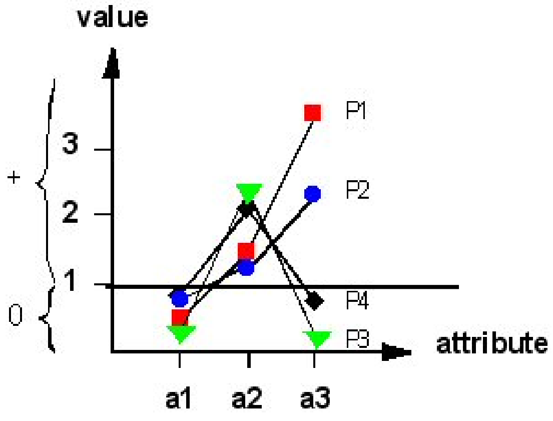


Challenges and Approaches, Basic Models

	a1	a2
P1	0	+
P2	0	+
P3	0	+
P4	0	+

	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	2.3	0.2
P4	0.8	2.1	0.7





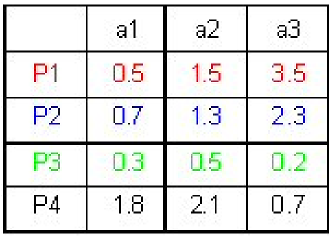
pattern: all lines cross border between states (in the same direction)

Challenges and Approaches, Basic Models

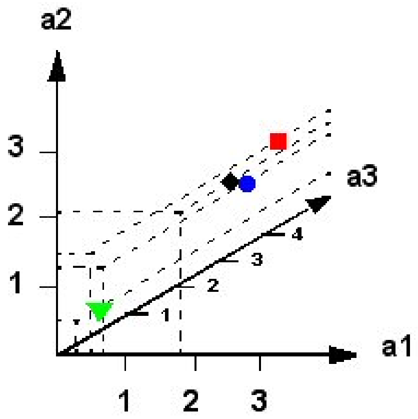
- change in same direction – general idea: find a subset of rows and columns, where a permutation of the set of columns exists such that the values in every row are increasing
- clusters do not form a subspace but rather half-spaces
- related approaches:
  - quantitative association rule mining [Web01,RRK04,GRRK05]
  - adaptation of formal concept analysis [GW99] to numeric data [Pfa07]

Challenges and Approaches, Basic Models

- example – 3-dimensional embedding space



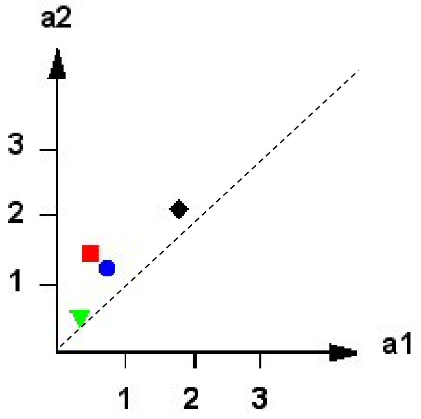
	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	0.5	0.2
P4	1.8	2.1	0.7



Challenges and Approaches, Basic Models

- example – 2-dimensional subspace

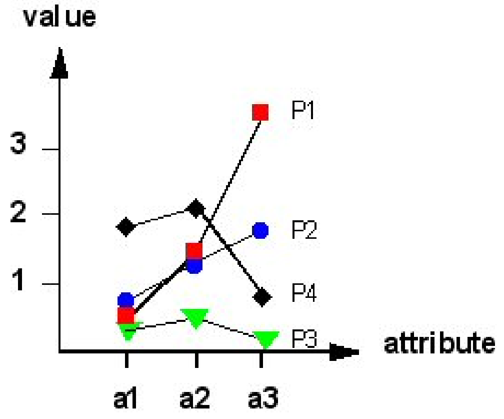
	a1	a2
P1	0.5	1.5
P2	0.7	1.3
P3	0.3	0.5
P4	1.8	2.1



Challenges and Approaches, Basic Models

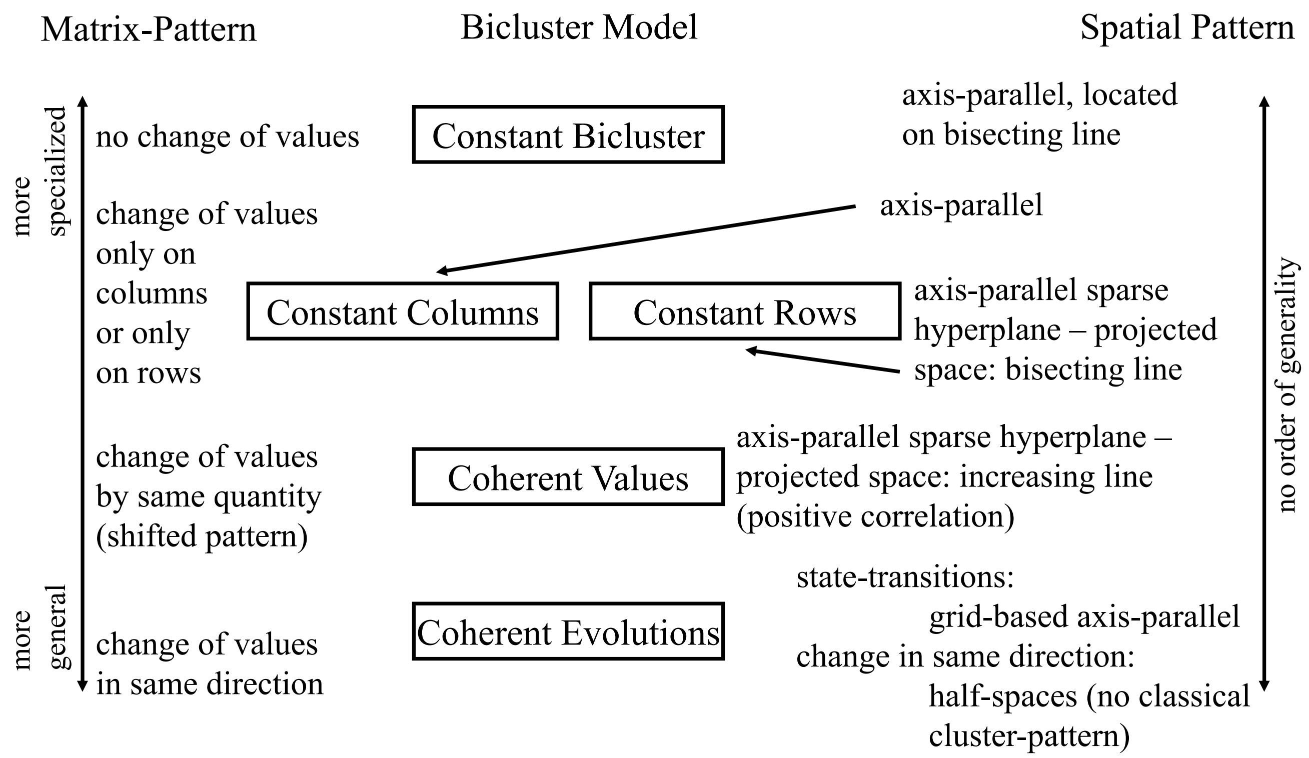
- example – transposed view of attributes

	a1	a2	a3
P1	0.5	1.5	3.5
P2	0.7	1.3	2.3
P3	0.3	0.5	0.2
P4	1.8	2.1	0.7



- pattern: all lines increasing

Challenges and Approaches, Basic Models



Matrix-Pattern

Bicluster Model

Spatial Pattern

more specialized

more general

no change of values

Constant Bicluster

axis-parallel, located on bisecting line

change of values only on columns or only on rows

Constant Columns

Constant Rows

axis-parallel

axis-parallel sparse hyperplane – projected space: bisecting line

change of values by same quantity (shifted pattern)

Coherent Values

axis-parallel sparse hyperplane – projected space: increasing line (positive correlation)

change of values in same direction

Coherent Evolutions

state-transitions:
grid-based axis-parallel
change in same direction:
half-spaces (no classical cluster-pattern)

no order of generality

Algorithms for Constant Biclusters

- classical problem statement by Hartigan [Har72]
- quality measure for a bicluster: variance of the submatrix $\boldsymbol{A}_{IJ}$:

$$VAR\left(A_{IJ}\right) = \sum_{i \in I, j \in J} \left(a_{ij} - a_{IJ}\right)^2$$

- avoids partitioning into |X|·|Y| singularity-biclusters (optimizing the sum of squares) by comparing the reduction with the reduction expected by chance
- recursive split of data matrix into two partitions
- each split chooses the maximal reduction in the overall sum of squares for all biclusters

Biclusters with Constant Values in Rows or Columns

- simple approach: normalization to transform the biclusters into constant biclusters and follow the first approach (e.g. [GLD00])
- some application-driven approaches with special assumptions in the bioinformatics community (e.g. [CST00,SMD03,STG+01])
- constant values on columns: general axis-parallel subspace/projected clustering
- constant values on rows: special case of general correlation clustering
- both cases special case of approaches to biclusters with coherent values

Algorithms for Biclusters with Coherent Values

classical approach: Cheng&Church [CC00]

- introduced the term biclustering to analysis of gene expression data
- quality of a bicluster: *mean squared residue* value *H*

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} \left(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}\right)^2$$

- submatrix (I,J) is considered a bicluster, if H(I,J) < δ

Algorithms for Biclusters with Coherent Values

- δ =0 → *perfect* bicluster:
  - each row and column exhibits absolutely consistent bias
  - bias of row *i* w.r.t. other rows: $a_{iJ} - a_{IJ}$

- the model for a perfect bicluster predicts value a_{ij} by a row-constant, a column-constant, and an overall cluster-constant:

$$a_{ij} = a_{iJ} + a_{Ij} - a_{IJ}$$

$$\Updownarrow \mu = a_{IJ}, r_i = a_{iJ} - a_{IJ}, c_j = a_{Ij} - a_{IJ}$$

$$a_{ij} = \mu + r_i + c_j$$

Algorithms for Biclusters with Coherent Values

- for a non-perfect bicluster, the prediction of the model deviates from the true value by a residue:

$$a_{ij} = \mathrm{res}(a_{ij}) + a_{iJ} + a_{Ij} - a_{IJ}$$

$$\Updownarrow$$

$$\mathrm{res}(a_{ij}) = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$$

- This residue is the optimization criterion:

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} \left(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}\right)^2$$

Algorithms for Biclusters with Coherent Values

- The optimization is also possible for the row-residue of row *i* or the column-residue of column *j*.

- Algorithm:
  1. find a δ -bicluster: greedy search by removing the row or column (or the set of rows/columns) with maximal mean squared residue until the remaining submatrix (I,J) satisfies H(I,J)< δ.
  2. find a maximal δ -bicluster by adding rows and columns to (I,J) unless this would increase *H*.
  3. replace the values of the found bicluster by random numbers and repeat the procedure until *k* δ -biclusters are found.

Algorithms for Biclusters with Coherent Values

Weak points in the approach of Cheng&Church:

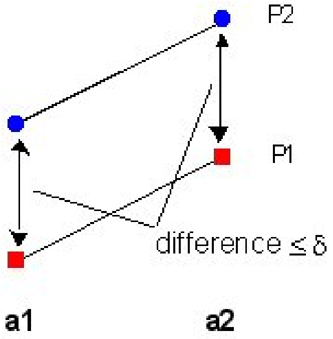
1. One cluster at a time is found, the cluster needs to be masked in order to find a second cluster.
2. This procedure bears an inefficient performance.
3. The masking may lead to less accurate results.
4. The masking inhibits simultaneous overlapping of rows and columns.
5. Missing values cannot be dealt with.
6. The user must specify the number of clusters beforehand.

Algorithms for Biclusters with Coherent Values

p-cluster model [WWYY02]

- p-cluster model: deterministic approach
- specializes δ -bicluster-property to a pairwise property of two objects in two attributes:

$$\left|\left(a_{i_1 j_1} - a_{i_1 j_2}\right) - \left(a_{i_2 j_1} - a_{i_2 j_2}\right)\right| \leq \delta$$

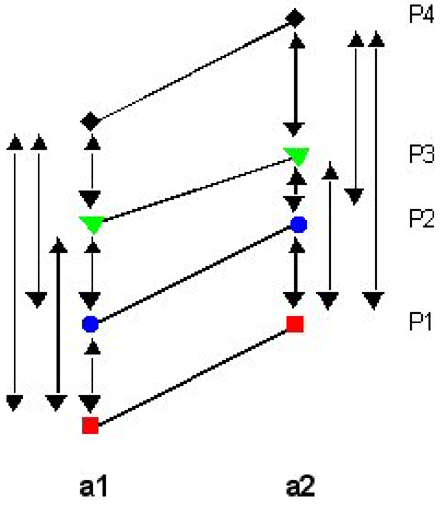


- submatrix (I,J) is a δ -p-cluster if this property is fulfilled for any 2x2 submatrix ({i_1, i_2}, {j_1, j_2}) where {i_1, i_2} $\in$ I and {j_1, j_2} $\in$J.

Algorithms for Biclusters with Coherent Values

Algorithm:

1. create maximal set of attributes for each pair of objects forming a δ -p-cluster
2. create maximal set of objects for each pair of attributes forming a δ -p-cluster
3. pruning-step
4. search in the set of submatrices



Problem: complete enumeration approach

Addressed issues:

1. multiple clusters simultaneously
4. allows for overlapping rows and columns
6. allows for arbitrary number of clusters

Related approaches:

FLOC [YWWY02],MaPle [PZC+03]

Summary

- Biclustering models do not fit exactly into the spatial intuition behind subspace, projected, or correlation clustering.
- Models make sense in view of a data matrix.
- Strong point: the models generally do not rely on the locality assumption.
- Models differ substantially → fair comparison is a non-trivial task.
- Comparison of five methods: [PBZ+06]
- Rather specialized task – comparison in a broad context (subspace/projected/correlation clustering) is desirable.
- Biclustering performs generally well on microarray data – for a wealth of approaches see [MO04].

Summary and Perspectives

comparison: correlation clustering – biclustering:

- model for correlation clusters more general and meaningful
- models for biclusters rather specialized
- in general, biclustering approaches do not rely on locality assumption
- non-local approach and specialization of models may make biclustering successful in many applications
- correlation clustering is the more general approach but the approaches proposed so far are rather a first draft to tackle the complex problem

Literature

[ABD+08] E. Achtert, C. Böhm, J. David, P. Kröger, and A. Zimek.
Robust clustering in arbitrarily oriented subspaces.
In Proceedings of the 8th SIAM International Conference on Data Mining (SDM), Atlanta, GA, 2008

[ABK+06] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek.
Deriving quantitative models for correlation clusters.
In Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Philadelphia, PA, 2006.

[ABK+07a] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, I. Müller-Gorman, and A. Zimek.
Detection and visualization of subspace cluster hierarchies.
In Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA), Bangkok, Thailand, 2007.

[ABK+07b] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek.
On exploring complex relationships of correlation clusters.
In Proceedings of the 19th International Conference on Scientific and Statistical Database Management (SSDBM), Banff, Canada, 2007.

[ABK+07c] E. Achtert, C. Böhm, H.-P. Kriegel, P. Kröger, and A. Zimek.
Robust, complete, and efficient correlation clustering.
In Proceedings of the 7th SIAM International Conference on Data Mining (SDM), Minneapolis, MN, 2007.

Literature

[AGGR98] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan.
Automatic subspace clustering of high dimensional data for data mining applications.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Seattle, WA, 1998.

[AHK01] C. C. Aggarwal, A. Hinneburg, and D. Keim.
On the surprising behavior of distance metrics in high dimensional space.
In Proceedings of the 8th International Conference on Database Theory (ICDT), London, U.K., 2001.

[APW+99] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park.
Fast algorithms for projected clustering.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Philadelphia, PA, 1999.

[AS94] R. Agrawal and R. Srikant. **Fast algorithms for mining association rules**.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Minneapolis, MN, 1994.

[AY00] C. C. Aggarwal and P. S. Yu.
Finding generalized projected clusters in high dimensional space.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Dallas, TX, 2000.

Literature

[BBC04] N. Bansal, A. Blum, and S. Chawla.
Correlation clustering.
Machine Learning, 56:89–113, 2004.

[BC00] D. Barbara and P. Chen.
Using the fractal dimension to cluster datasets.
In Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Boston, MA, 2000.

[BDCKY02] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini.
Discovering local structure in gene expression data: The order-preserving submatrix problem.
In Proceedings of the 6th Annual International Conference on Computational Molecular Biology (RECOMB), Washington, D.C., 2002.

[Bel61] R. Bellman.
Adaptive Control Processes. A Guided Tour.
Princeton University Press, 1961.

[BFG99] K. P. Bennett, U. Fayyad, and D. Geiger.
Density-based indexing for approximate nearest-neighbor queries.
In Proceedings of the 5th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Diego, CA, 1999.

Literature

[BGRS99] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft.
When is "nearest neighbor" meaningful?
In Proceedings of the 7th International Conference on Database Theory (ICDT), Jerusalem, Israel, 1999.

[BKKK04] C. Böhm, K. Kailing, H.-P. Kriegel, and P. Kröger.
Density connected clustering with local subspace preferences.
In Proceedings of the 4th International Conference on Data Mining (ICDM), Brighton, U.K., 2004.

[BKKZ04] C. Böhm, K. Kailing, P. Kröger, and A. Zimek.
Computing clusters of correlation connected objects.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Paris, France, 2004.

[CC00] Y. Cheng and G. M. Church.
Biclustering of expression data.
In Proceedings of the 8th International Conference Intelligent Systems for Molecular Biology (ISMB), San Diego, CA, 2000.

[CDGS04] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra.
Minimum sum-squared residue co-clustering of gene expression data.
In Proceedings of the 4th SIAM International Conference on Data Mining (SDM), Orlando, FL, 2004.

DATABASE SYSTEMS GROUP

Literature

LMU

[CFZ99] C. H. Cheng, A. W.-C. Fu, and Y. Zhang.
Entropy-based subspace clustering for mining numerical data.
In Proceedings of the 5th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Diego, CA, pages 84–93, 1999.

[CST00] A. Califano, G. Stolovitzky, and Y. Tu.
Analysis of gene expression microarrays for phenotype classification.
In Proceedings of the 8th International Conference Intelligent Systems for Molecular Biology (ISMB), San Diego, CA, 2000.

[EKSX96] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu.
A density-based algorithm for discovering clusters in large spatial databases with noise.
In Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR, 1996.

[FM04] J. H. Friedman and J. J. Meulman.
Clustering objects on subsets of attributes.
Journal of the Royal Statistical Society: Series B (Statistical Methodology), 66(4):825–849, 2004.

[FWV07] D. Francois, V. Wertz, and M. Verleysen.
The concentration of fractional distances.
IEEE Transactions on Knowledge and Data Engineering, 19(7): 873-886, 2007.

DATABASE SYSTEMS GROUP

Literature

LMU

[GHPT05] A. Gionis, A. Hinneburg, S. Papadimitriou, and P. Tsaparas.
Dimension induced clustering.
In Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL, 2005.

[GLD00] G. Getz, E. Levine, and E. Domany.
Coupled two-way clustering analysis of gene microarray data.
Proceedings of the National Academy of Sciences of the United States of America, 97(22):12079–12084, 2000.

[GRRK05] E. Georgii, L. Richter, U. Rückert, and S. Kramer.
Analyzing microarray data using quantitative association rules.
Bioinformatics, 21(Suppl. 2):ii1–ii8, 2005.

[GW99] B. Ganter and R. Wille.
Formal Concept Analysis.
Mathematical Foundations. Springer, 1999.

[HAK00] A. Hinneburg, C. C. Aggarwal, and D. A. Keim.
What is the nearest neighbor in high dimensional spaces?
In Proceedings of the 26th International Conference on Very Large Data Bases (VLDB), Cairo, Egypt, 2000.

Literature

[Har72] J. A. Hartigan.
Direct clustering of a data matrix.
Journal of the American Statistical Association, 67(337):123–129, 1972.

[HKK+10] M. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek.
Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?
In Proceedings of the 22nd International Conference on Scientific and Statistical Data Management (SSDBM), Heidelberg, Germany, 2010.

[IBB04] J. Ihmels, S. Bergmann, and N. Barkai.
Defining transcription modules using large-scale gene expression data.
Bioinformatics, 20(13):1993–2003, 2004.

[Jol02] I. T. Jolliffe.
Principal Component Analysis.
Springer, 2nd edition, 2002.

[KKK04 K. Kailing, H.-P. Kriegel, and P. Kröger.
Density-connected subspace clustering for highdimensional data.
In Proceedings of the 4th SIAM International Conference on Data Mining (SDM), Orlando, FL, 2004.

Literature

[KKRW05] H.-P. Kriegel, P. Kröger, M. Renz, and S. Wurst.
A generic framework for efficient subspace clustering of high-dimensional data.
In Proceedings of the 5th International Conference on Data Mining (ICDM), Houston, TX, 2005.

[KKZ09] H.-P. Kriegel, P. Kröger, and A. Zimek.
Clustering High Dimensional Data: A Survey on Subspace Clustering, Pattern-based Clustering, and Correlation Clustering.
ACM Transactions on Knowledge Discovery from Data (TKDD), Volume 3, Issue 1 (March 2009), Article No. 1, pp. 1-58, 2009.

[LW03] J. Liu and W. Wang.
OP-Cluster: Clustering by tendency in high dimensional spaces.
In Proceedings of the 3th International Conference on Data Mining (ICDM), Melbourne, FL, 2003.

[MK03] T. M. Murali and S. Kasif.
Extracting conserved gene expression motifs from gene expression data.
In Proceedings of the 8th Pacific Symposium on Biocomputing (PSB), Maui, HI, 2003.

Literature

[MO04] S. C. Madeira and A. L. Oliveira.
Biclustering algorithms for biological data analysis: A survey.
IEEE Transactions on Computational Biology and Bioinformatics, 1(1):24–45, 2004.

[MSE06] G. Moise, J. Sander, and M. Ester.
P3C: A robust projected clustering algorithm.
In Proceedings of the 6th International Conference on Data Mining (ICDM), Hong Kong, China, 2006.

[NGC01] H.S. Nagesh, S. Goil, and A. Choudhary.
Adaptive grids for clustering massive data sets.
In Proceedings of the 1st SIAM International Conference on Data Mining (SDM), Chicago, IL, 2001.

[PBZ+06] A. Prelic, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Guissem, L. Hennig, L. Thiele, and E. Zitzler.
A systematic comparison and evaluation of biclustering methods for gene expression data.
Bioinformatics, 22(9):1122–1129, 2006.

[Pfa07] J. Pfaltz.
What constitutes a scientific database?
In Proceedings of the 19th International Conference on Scientific and Statistical Database Management (SSDBM), Banff, Canada, 2007.

Literature

[PHL04] L. Parsons, E. Haque, and H. Liu.
Subspace clustering for high dimensional data: A review.
SIGKDD Explorations, 6(1):90–105, 2004.

[PJAM02] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali.
A Monte Carlo algorithm for fast projective clustering.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Madison, WI, 2002.

[PTTF02] E. Parros Machado de Sousa, C. Traina, A. Traina, and C. Faloutsos.
How to use fractal dimension to find correlations between attributes.
In Proc. KDD-Workshop on Fractals and Self-similarity in Data Mining: Issues and Approaches, 2002.

[PZC+03] J. Pei, X. Zhang, M. Cho, H. Wang, and P. S. Yu.
MaPle: A fast algorithm for maximal pattern-based clustering.
In Proceedings of the 3th International Conference on Data Mining (ICDM), Melbourne, FL, 2003.

[RRK04] U. Rückert, L. Richter, and S. Kramer.
Quantitative association rules based on half-spaces: an optimization approach.
In Proceedings of the 4th International Conference on Data Mining (ICDM), Brighton, U.K., 2004.

Literature

[SCH75] J.L. Slagle, C.L. Chang, S.L. Heller.
A Clustering and Data-Reorganization Algorithm.
IEEE Transactions on Systems, Man and Cybernetics, 5: 121-128, 1975

[SLGL06] K. Sim, J. Li, V. Gopalkrishnan, and G. Liu.
Mining maximal quasi-bicliques to co-cluster stocks and financial ratios for value investment.
In Proceedings of the 6th International Conference on Data Mining (ICDM), Hong Kong, China, 2006.

[SMD03] Q. Sheng, Y. Moreau, and B. De Moor.
Biclustering microarray data by Gibbs sampling.
Bioinformatics, 19(Suppl. 2):ii196–ii205, 2003.

[STG+01] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller.
Rich probabilistic models for gene expression.
Bioinformatics, 17(Suppl. 1):S243–S252, 2001.

[SZ05] K. Sequeira and M. J. Zaki.
SCHISM: a new approach to interesting subspace mining.
International Journal of Business Intelligence and Data Mining, 1(2):137–160, 2005.

[TSS02] A. Tanay, R. Sharan, and R. Shamir.
Discovering statistically significant biclusters in gene expression data.
Bioinformatics, 18 (Suppl. 1):S136–S144, 2002.

Literature

[TXO05] A. K. H. Tung, X. Xu, and C. B. Ooi.
CURLER: Finding and visualizing nonlinear correlated clusters.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Baltimore, ML, 2005.

[Web01] G. I. Webb.
Discovering associations with numeric variables.
In Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), San Francisco, CA, pages 383–388, 2001.

[WLKL04] K.-G. Woo, J.-H. Lee, M.-H. Kim, and Y.-J. Lee.
FINDIT: a fast and intelligent subspace clustering algorithm using dimension voting.
Information and Software Technology, 46(4):255–271, 2004.

[WWYY02] H. Wang, W. Wang, J. Yang, and P. S. Yu.
Clustering by pattern similarity in large data sets.
In Proceedings of the ACM International Conference on Management of Data (SIGMOD), Madison, WI, 2002.

[YWWY02] J. Yang, W. Wang, H. Wang, and P. S. Yu.
δ-clusters: Capturing subspace correlation in a large data set.
In Proceedings of the 18th International Conference on Data Engineering (ICDE), San Jose, CA, 2002.