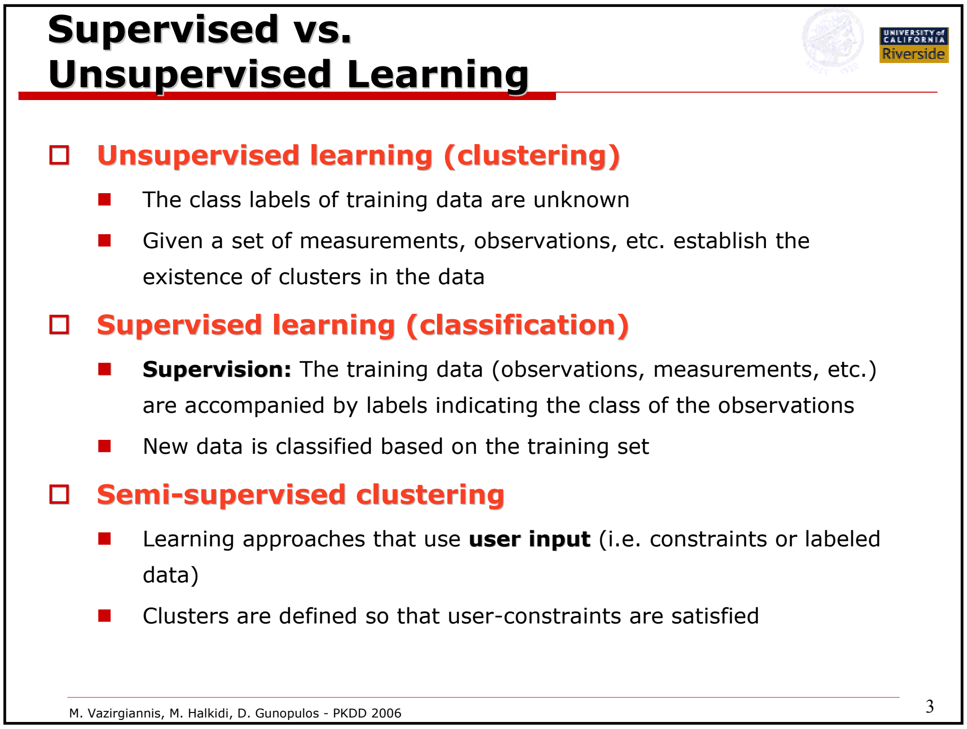
Novel Aspects in Unsupervised Learning: Semi-Supervised and Distributed Algorithms

Maria Halkidi **Michalis Vazirgiannis**
Dept of Informatics
AUEB
Email: `{mhalk, mvazirg}@aueb.gr`

Dimitrios Gunopulos
Dept of CS & Engineering
UCR
Email: `dg@cs.ucr.edu`

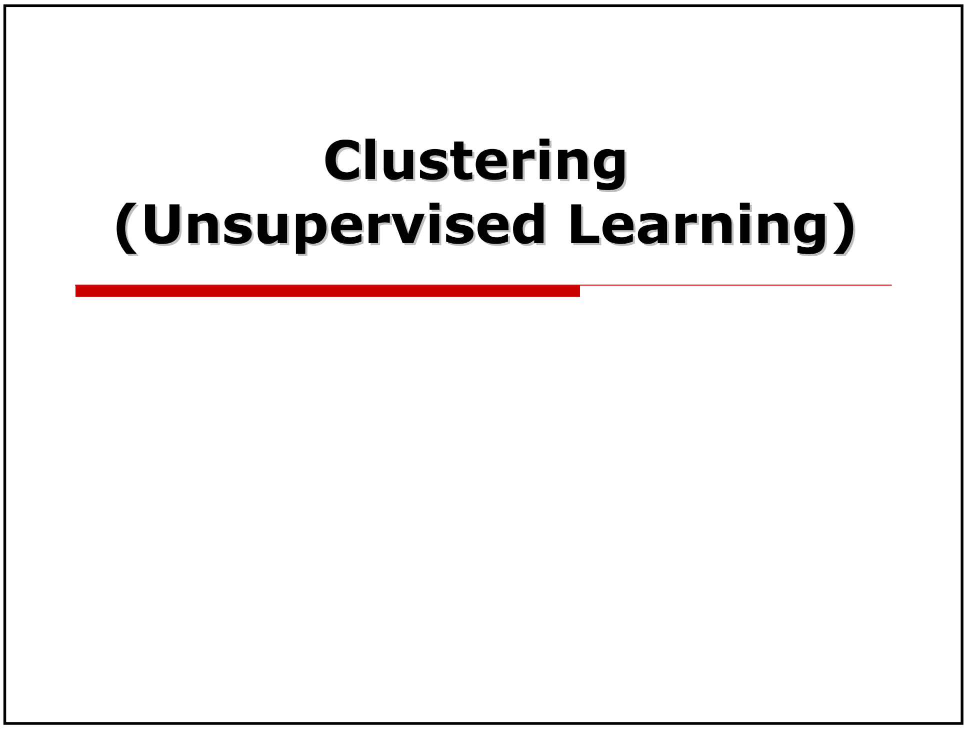
Outline

- Introduction
- Unsupervised learning
- Semi-supervised learning
  - Semi-supervised learning & cluster quality assessment
- Dimensionality reduction techniques
- Distributed clustering approaches

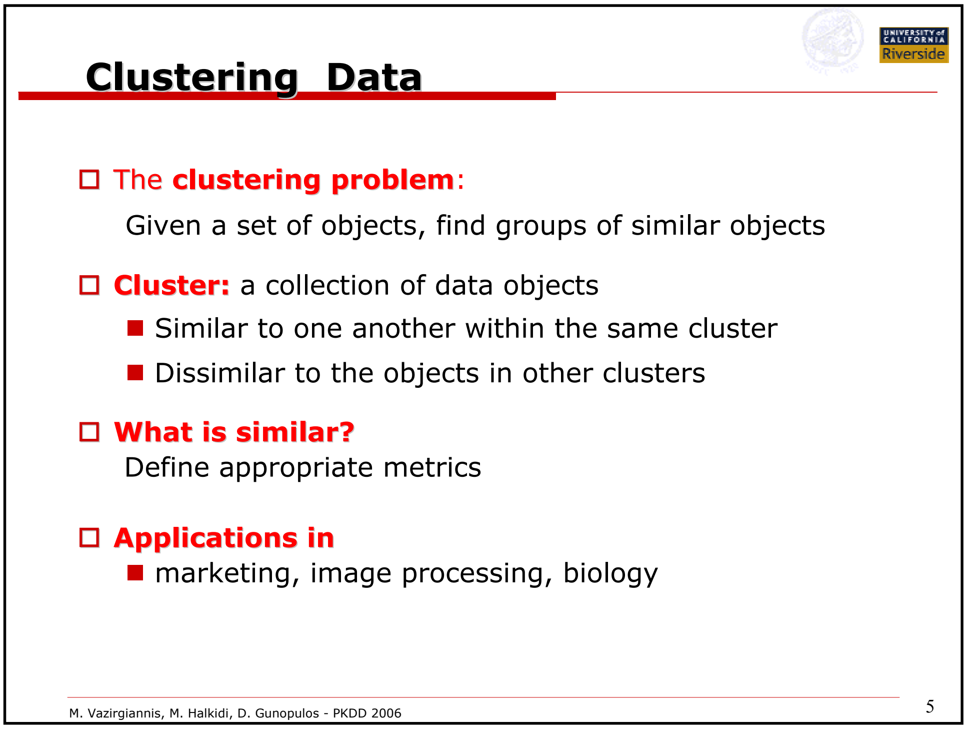


Supervised vs. Unsupervised Learning

- **Unsupervised learning (clustering)**
  - The class labels of training data are unknown
  - Given a set of measurements, observations, etc. establish the existence of clusters in the data
- **Supervised learning (classification)**
  - **Supervision:** The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set
- **Semi-supervised clustering**
  - Learning approaches that use **user input** (i.e. constraints or labeled data)
  - Clusters are defined so that user-constraints are satisfied



Clustering (Unsupervised Learning)



Clustering Data

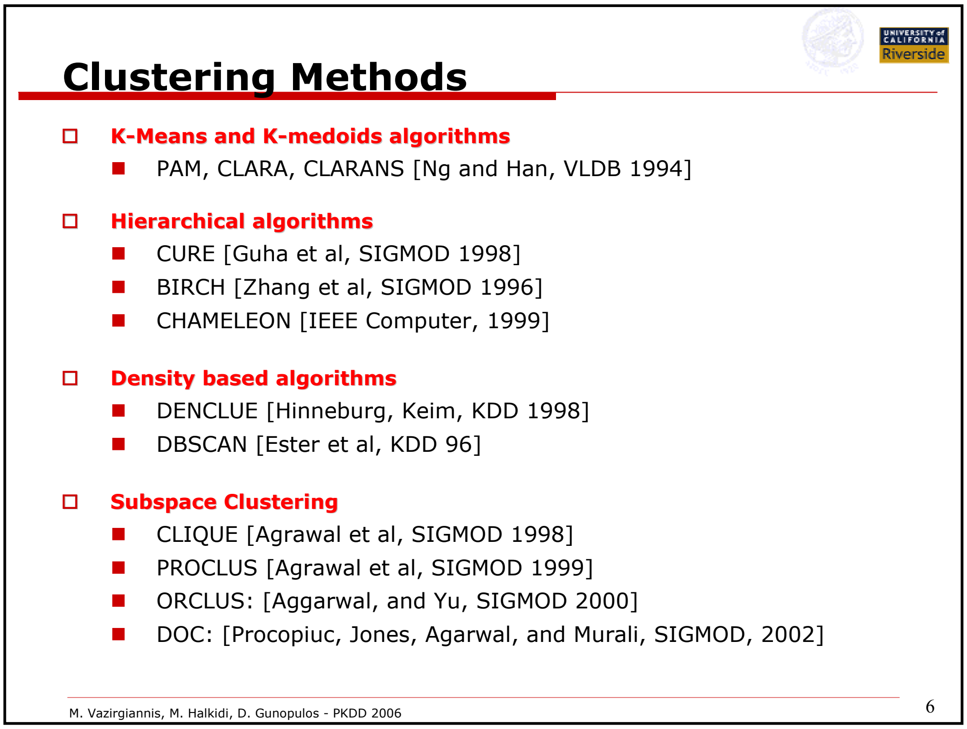
- □ The **clustering problem**:

  Given a set of objects, find groups of similar objects

- □ **Cluster:** a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- □ **What is similar?**

  Define appropriate metrics

- □ **Applications in**
  - marketing, image processing, biology



Clustering Methods

- □ **K-Means and K-medoids algorithms**
  - PAM, CLARA, CLARANS [Ng and Han, VLDB 1994]
- □ **Hierarchical algorithms**
  - CURE [Guha et al, SIGMOD 1998]
  - BIRCH [Zhang et al, SIGMOD 1996]
  - CHAMELEON [IEEE Computer, 1999]
- □ **Density based algorithms**
  - DENCLUE [Hinneburg, Keim, KDD 1998]
  - DBSCAN [Ester et al, KDD 96]
- □ **Subspace Clustering**
  - CLIQUE [Agrawal et al, SIGMOD 1998]
  - PROCLUS [Agrawal et al, SIGMOD 1999]
  - ORCLUS: [Aggarwal, and Yu, SIGMOD 2000]
  - DOC: [Procopiuc, Jones, Agarwal, and Murali, SIGMOD, 2002]

Partitional Algorithms: Basic Concept

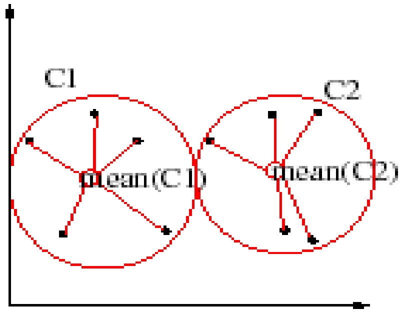
- **Partitional method:**
  - Partition the data set into a set of **k** disjoint clusters.
- **Problem Definition:**
  - Given an integer **k**, find a partitioning of **k** clusters that optimizes the chosen partitioning criterion

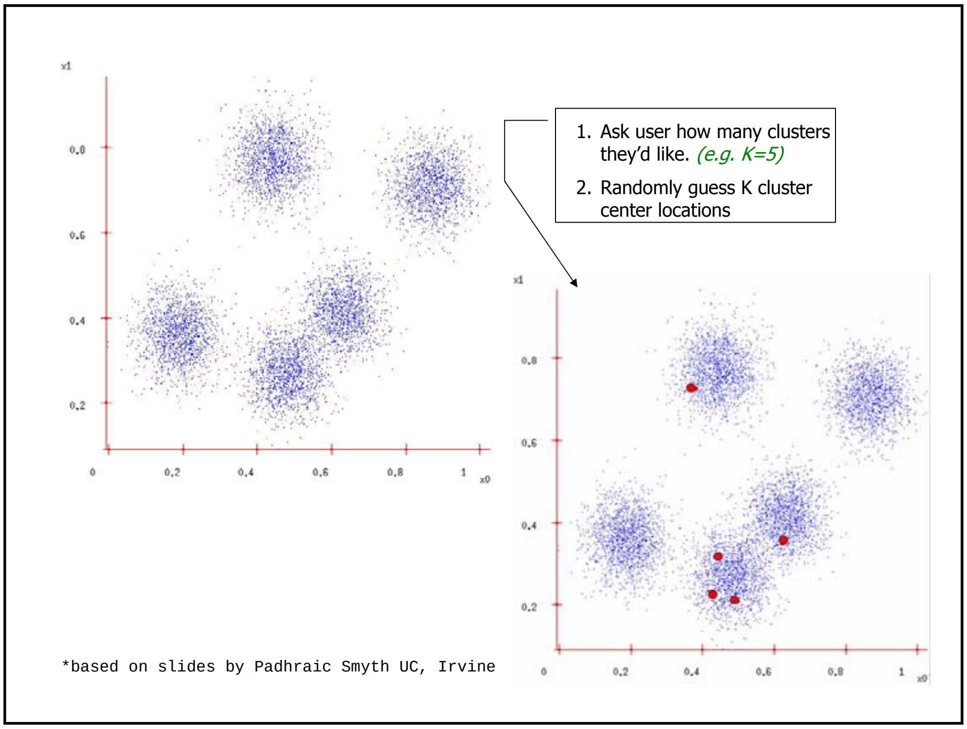
K-Means and K-Medoids algorithms

□Minimizes the sum of square distances of points to cluster representative

$$E_K = \sum_k \left\| x_k - m_{c(x_k)} \right\|^2$$

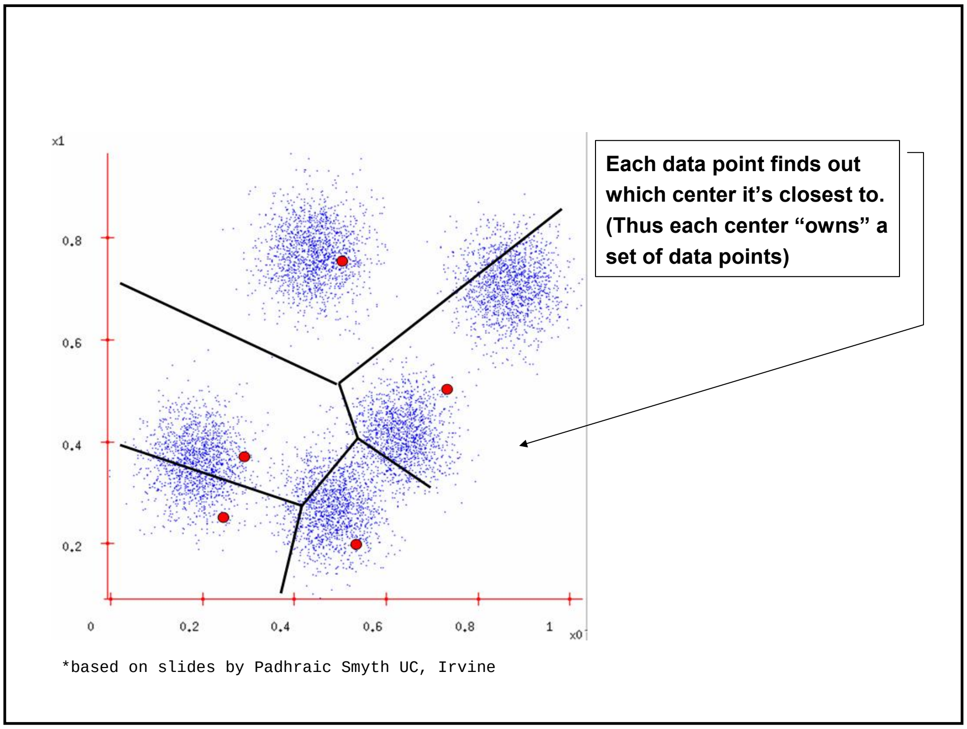
□Efficient iterative algorithms (*O(n)*)





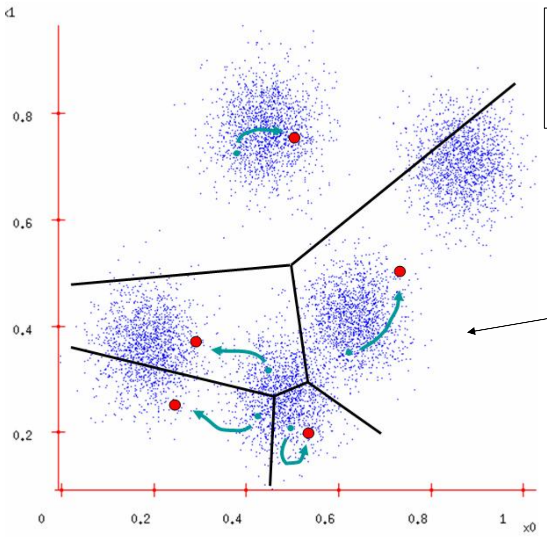
1. Ask user how many clusters they'd like. *(e.g. K=5)*
2. Randomly guess K cluster center locations

*based on slides by Padhraic Smyth UC, Irvine



Each data point finds out which center it's closest to. (Thus each center "owns" a set of data points)

*based on slides by Padhraic Smyth UC, Irvine



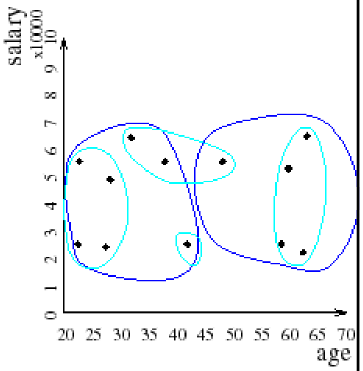
- Redefine each center finding out the set of the points it owns

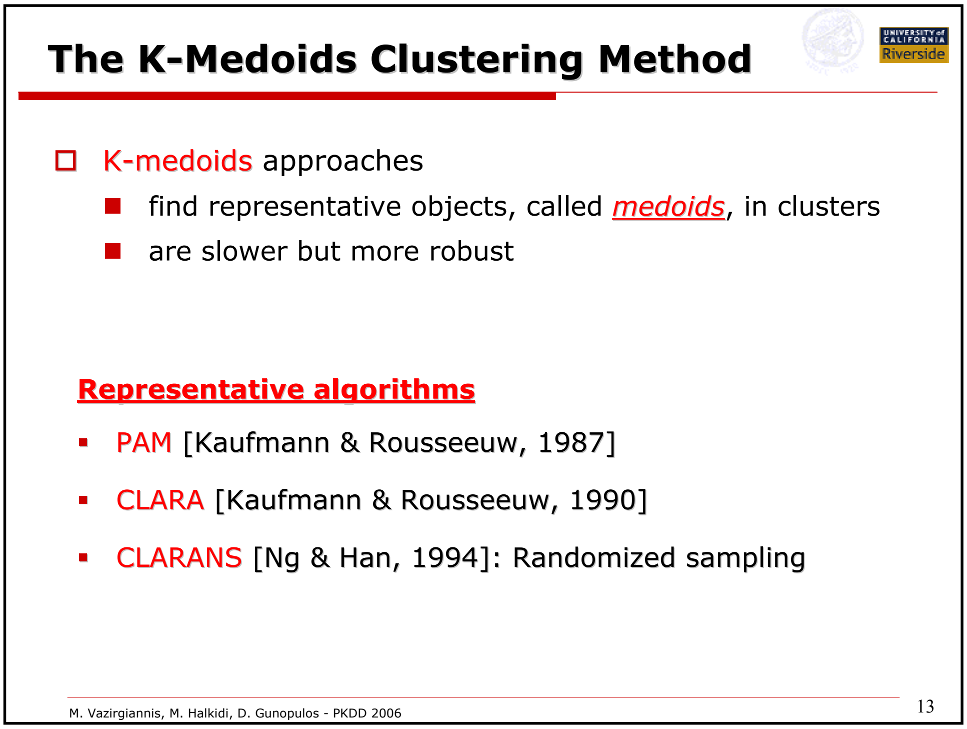
*based on slides by Padhraic Smyth UC, Irvine

Problems with K-Means type algorithms

- **Advantages**
  - Relatively efficient: **O(*tkn*)**
  - where *n* is the number of objects, ***k*** is the number of clusters, and ***t*** is the number of iterations.

    Normally, ***k, t << n*.**
  - Often terminates at a local optimum.
- **Problems**
  - Clusters are approximately spherical
  - Unable to handle noisy data and outliers
  - High dimensionality may be a problem
  - The value of *k* is an input parameter



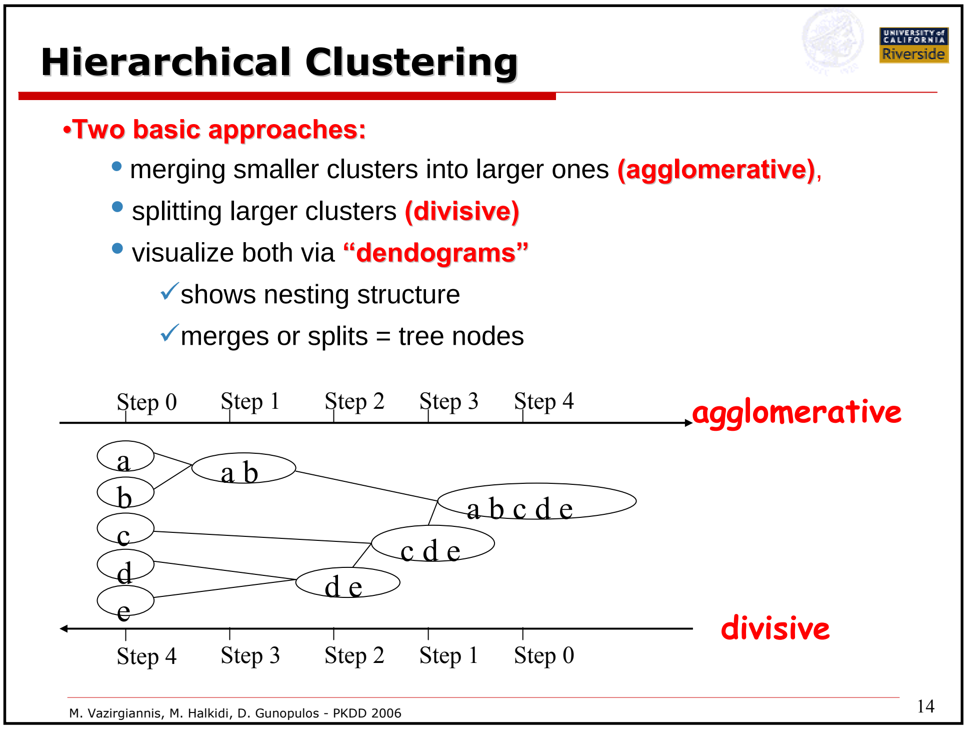


The K-Medoids Clustering Method

- □ K-medoids approaches
  - find representative objects, called *medoids*, in clusters
  - are slower but more robust

Representative algorithms

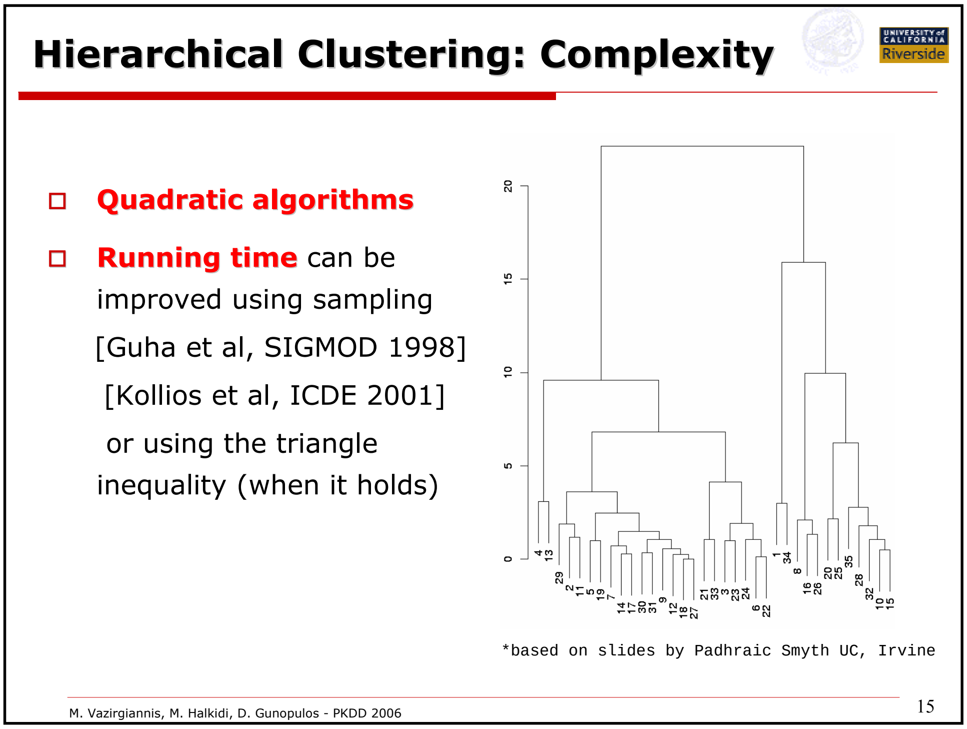
- PAM [Kaufmann & Rousseeuw, 1987]
- CLARA [Kaufmann & Rousseeuw, 1990]
- CLARANS [Ng & Han, 1994]: Randomized sampling



Hierarchical Clustering

•**Two basic approaches:**

- merging smaller clusters into larger ones **(agglomerative)**,
- splitting larger clusters **(divisive)**
- visualize both via **"dendograms"**
  - ✓shows nesting structure
  - ✓merges or splits = tree nodes

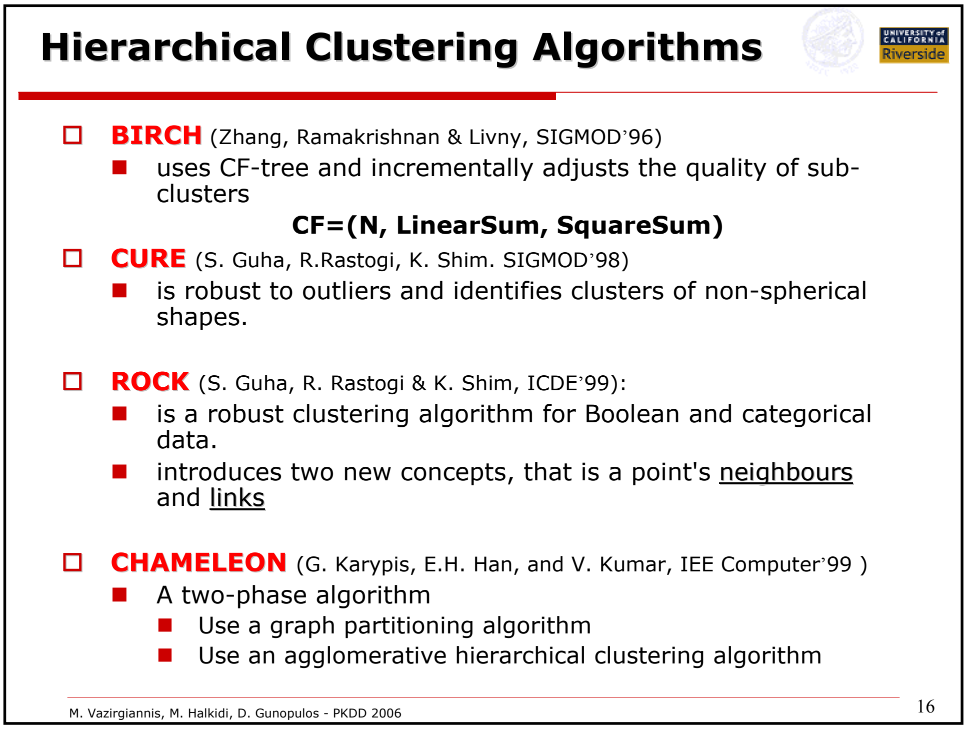


Hierarchical Clustering: Complexity

- **Quadratic algorithms**
- **Running time** can be improved using sampling [Guha et al, SIGMOD 1998] [Kollios et al, ICDE 2001] or using the triangle inequality (when it holds)

0
5
10
15
20
4 13 29 2 11 5 19 7 14 17 30 31 9 12 18 27 21 33 3 23 24 6 22 1 34 8 16 26 20 25 35 28 32 10 15

*based on slides by Padhraic Smyth UC, Irvine



Hierarchical Clustering Algorithms

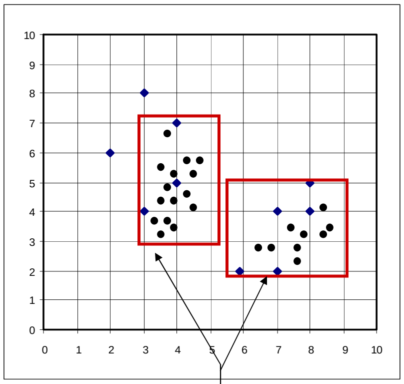
- **BIRCH** (Zhang, Ramakrishnan & Livny, SIGMOD'96)
  - uses CF-tree and incrementally adjusts the quality of sub-clusters

    CF=(N, LinearSum, SquareSum)
- **CURE** (S. Guha, R.Rastogi, K. Shim. SIGMOD'98)
  - is robust to outliers and identifies clusters of non-spherical shapes.
- **ROCK** (S. Guha, R. Rastogi & K. Shim, ICDE'99):
  - is a robust clustering algorithm for Boolean and categorical data.
  - introduces two new concepts, that is a point's neighbours and links
- **CHAMELEON** (G. Karypis, E.H. Han, and V. Kumar, IEE Computer'99)
  - A two-phase algorithm
    - Use a graph partitioning algorithm
    - Use an agglomerative hierarchical clustering algorithm

Density-based Algorithms

- □ **Clusters** are **regions of space which have a high density** of points

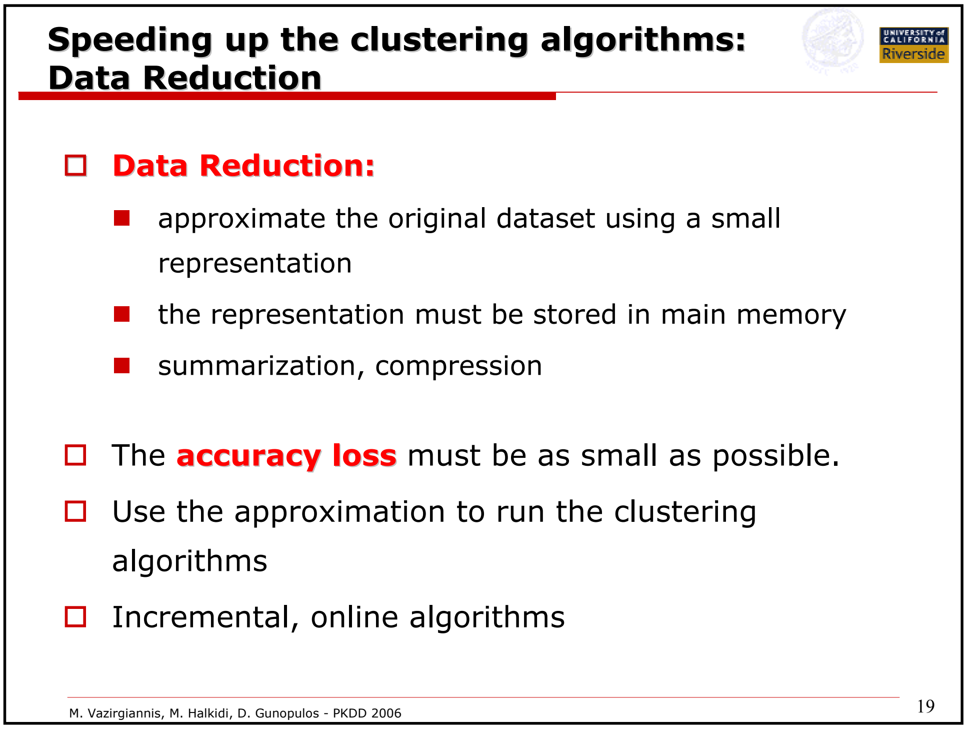
- □ Clusters can have **arbitrary shapes**



Regions of high density

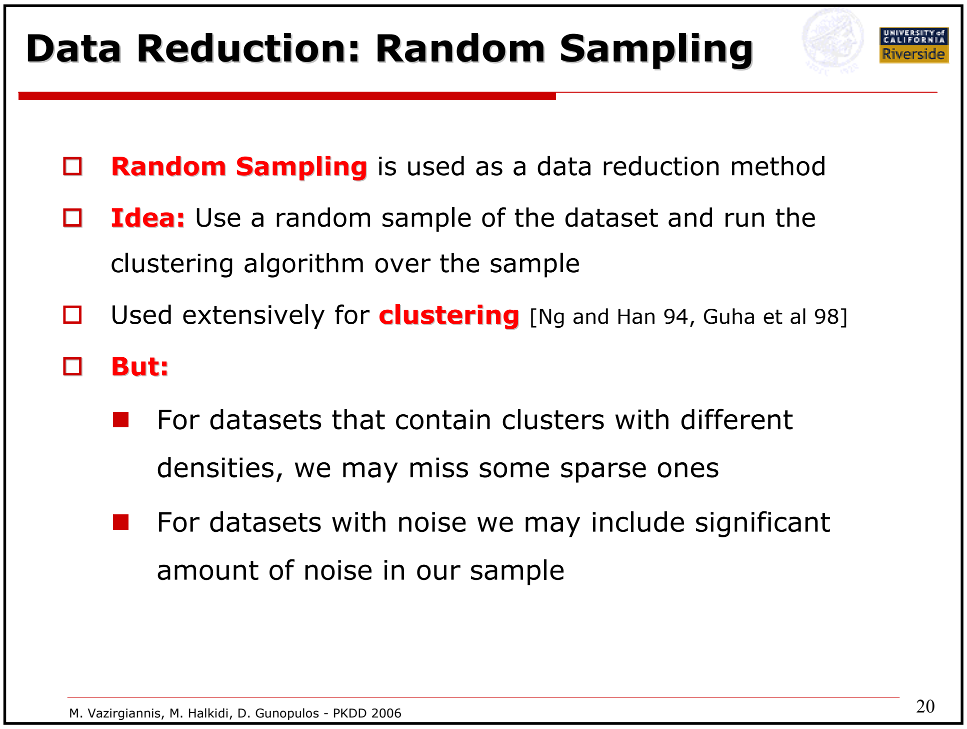
Density-based Clustering Algorithms

- **Clustering** based on density (local cluster criterion), such as density-connected points
- **Major features:**
  - ✓ Discover clusters of arbitrary shape
  - ✓ Handle noise
  - ✓ Need density parameters as termination condition
  - ✓ Work for low dimensional spaces
- **Representative algorithms:**
  - ✓ **DBSCAN:** Ester, et al. (KDD'96)
  - ✓ **DENCLUE:** Hinneburg & D. Keim (KDD'98)



Speeding up the clustering algorithms: Data Reduction

- □ **Data Reduction:**
  - ■ approximate the original dataset using a small representation
  - ■ the representation must be stored in main memory
  - ■ summarization, compression
- □ The **accuracy loss** must be as small as possible.
- □ Use the approximation to run the clustering algorithms
- □ Incremental, online algorithms

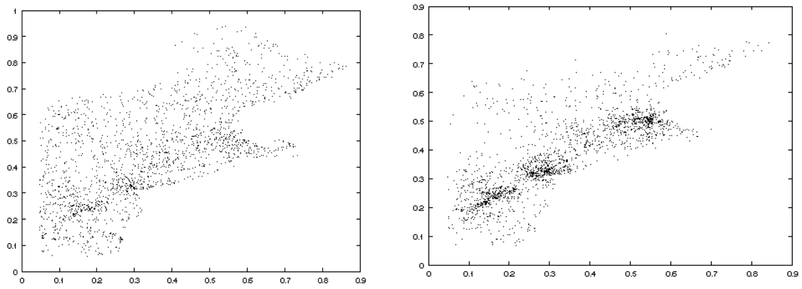


Data Reduction: Random Sampling

- □ **Random Sampling** is used as a data reduction method
- □ **Idea:** Use a random sample of the dataset and run the clustering algorithm over the sample
- □ Used extensively for **clustering** [Ng and Han 94, Guha et al 98]
- □ **But:**
  - ■ For datasets that contain clusters with different densities, we may miss some sparse ones
  - ■ For datasets with noise we may include significant amount of noise in our sample

Biased Sampling

- □ In **biased sampling**, the probability that a point is included in the sample depends on the local density
- □ We can oversample or undersample regions in our datasets depending on the DM task at hand



The Biased Sampling Technique

- □ **Basic idea:**
  - First compute an approximation of the density function of the dataset
  - Use the density function to define the probability for including a point to the sample

    [Palmer and Faloutsos, SIGMOD 2000]

    [Kollios et al, ICDE 2001]

Clustering High Dimensional Data

- Fundamental to all clustering techniques is the choice of distance measure between data points;

$$D(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{k=1}^{q} (x_{ik} - x_{jk})^2$$

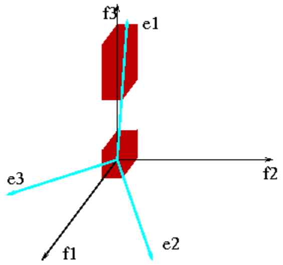
Squared Euclidean distance

- **Assumption:** All features are **equally important**;
- Such approaches fail in high dimensional spaces
- Feature selection (Dy and Brodley, 2000)

Dimensionality Reduction

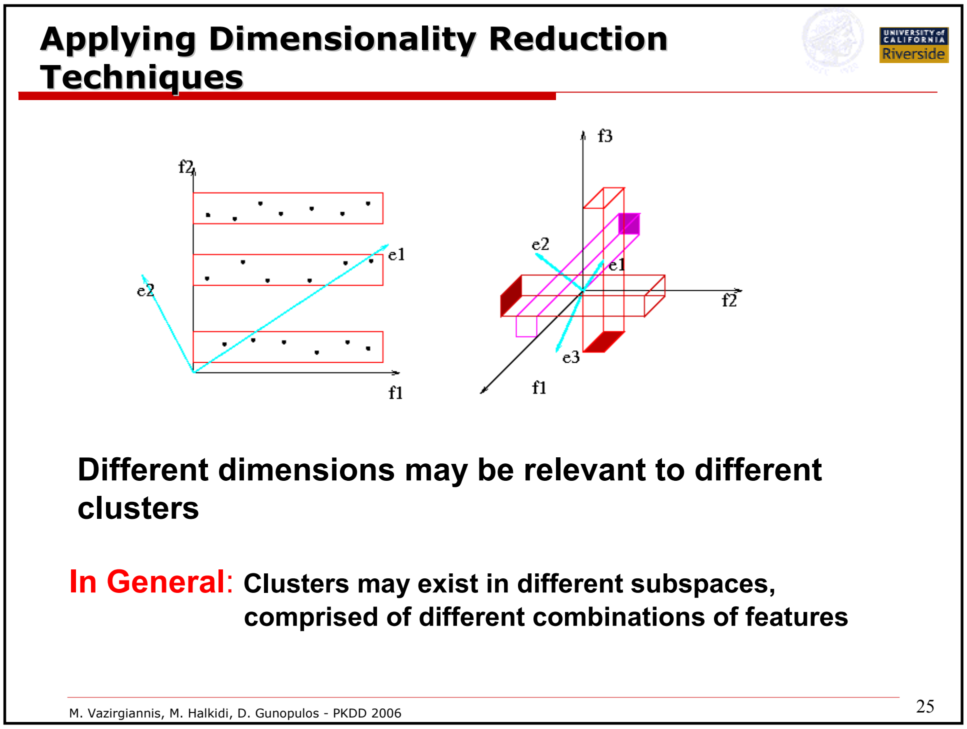
Applying Dimensionality Reduction Techniques

Dimensionality reduction techniques (such as **Singular Value Decomposition**) can provide a solution by reducing the dimensionality of the dataset:



Drawbacks:

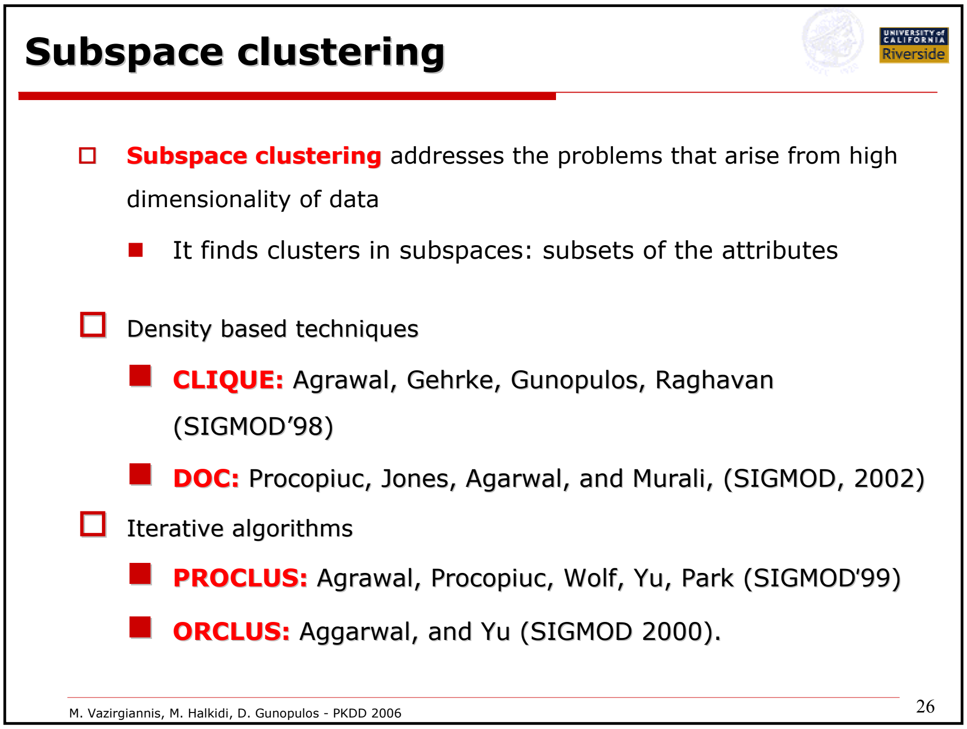
- The new dimensions may be difficult to interpret
- They don’t improve the clustering in all cases



Applying Dimensionality Reduction Techniques

Different dimensions may be relevant to different clusters

In General: **Clusters may exist in different subspaces, comprised of different combinations of features**

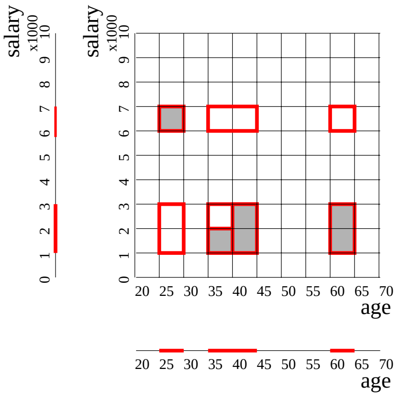


Subspace clustering

- **Subspace clustering** addresses the problems that arise from high dimensionality of data
  - It finds clusters in subspaces: subsets of the attributes
- Density based techniques
  - **CLIQUE:** Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98)
  - **DOC:** Procopiuc, Jones, Agarwal, and Murali, (SIGMOD, 2002)
- Iterative algorithms
  - **PROCLUS:** Agrawal, Procopiuc, Wolf, Yu, Park (SIGMOD'99)
  - **ORCLUS:** Aggarwal, and Yu (SIGMOD 2000).

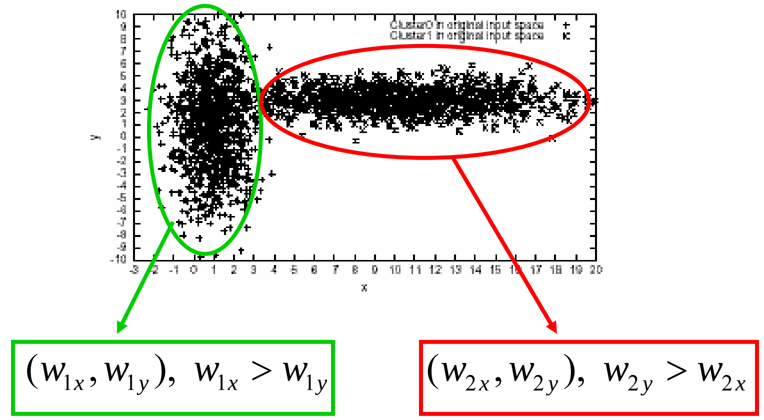
Subspace clustering

- □ **Density based clusters:** find dense areas in subspaces
- □ Identifying the right sets of attributes is hard
- □ Assuming a global threshold allows bottom-up algorithms
- □ Constrained monotone search in a lattice space



Locally Adaptive Clustering

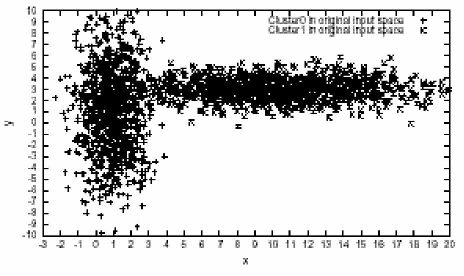
Each cluster is characterized by different attribute weights (Friedman and Meulman 2002, Domeniconi 2002)



$(w_{1x}, w_{1y}),\ w_{1x} > w_{1y}$

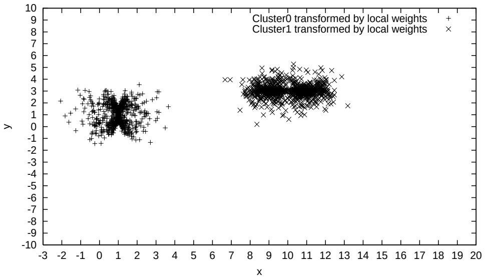
$(w_{2x}, w_{2y}),\ w_{2y} > w_{2x}$

Locally Adaptive Clustering : Example



before local transformations

after local transformations



LAC

[C. Domeniconi et al SDM04]

- **Computing the weights:**

X_{ji} : average squared distance along dimension i of points in S_j from $\boldsymbol{c}_j$

$$X_{ji} = \frac{1}{\left|S_j\right|}\sum_{x \in S_j}\left(c_{ji} - x_i\right)^2$$

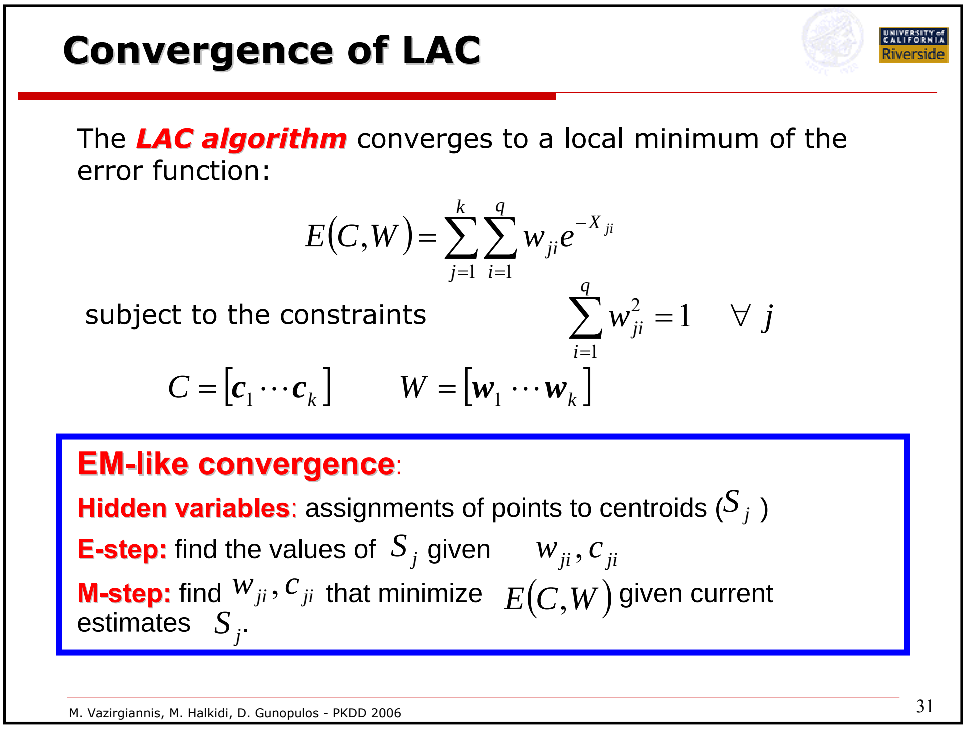
$$w_{ji} = \frac{e^{-X_{ji}}}{\sum_l e^{-X_{jl}}}$$

Exponential weighting scheme

Result :

$\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_k$

A weight vector for each cluster



Convergence of LAC

The ***LAC algorithm*** converges to a local minimum of the error function:

$$E(C,W)=\sum_{j=1}^{k}\sum_{i=1}^{q} w_{ji}e^{-X_{ji}}$$

subject to the constraints $\sum_{i=1}^{q} w_{ji}^2 = 1 \quad \forall\, j$

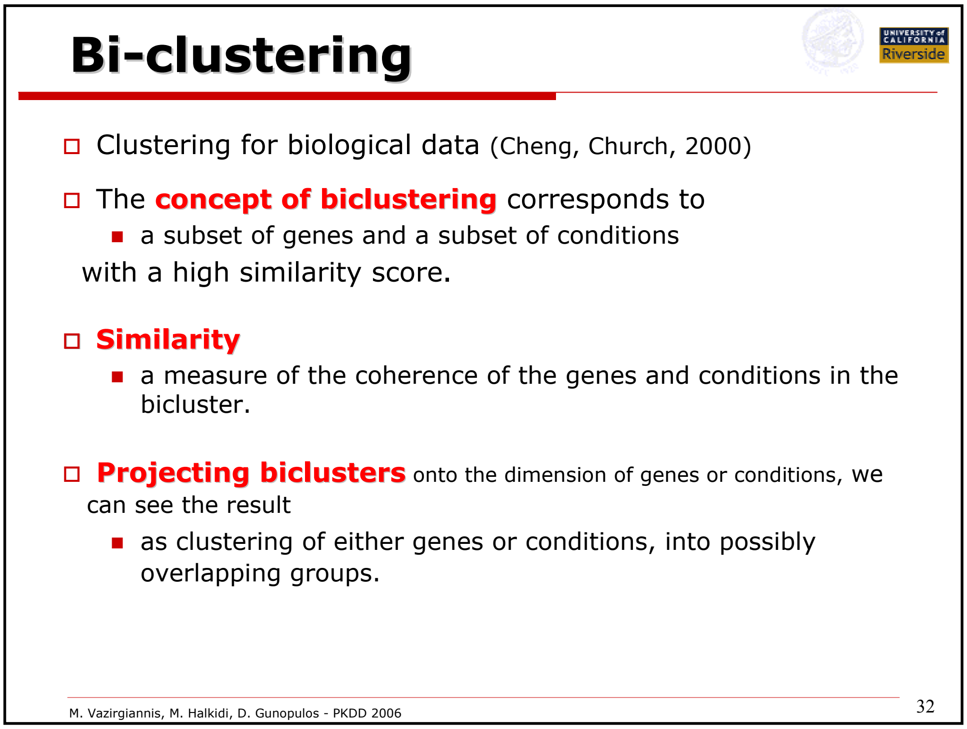
$$C=[\mathbf{c}_1 \cdots \mathbf{c}_k] \qquad W=[\mathbf{w}_1 \cdots \mathbf{w}_k]$$

EM-like convergence:

Hidden variables: assignments of points to centroids (S_j)

E-step: find the values of S_j given w_{ji}, c_{ji}

M-step: find w_{ji}, c_{ji} that minimize $E(C,W)$ given current estimates S_j.



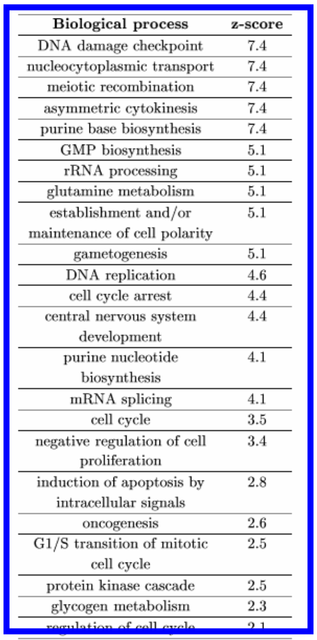
Bi-clustering

- Clustering for biological data (Cheng, Church, 2000)
- The **concept of biclustering** corresponds to
  - a subset of genes and a subset of conditions

  with a high similarity score.
- **Similarity**
  - a measure of the coherence of the genes and conditions in the bicluster.
- **Projecting biclusters** onto the dimension of genes or conditions, we can see the result
  - as clustering of either genes or conditions, into possibly overlapping groups.

Biological processes annotated in one cluster generated by the LAC algorithm

There exists a number of cell cycle genes. The terms for cell cycle regulation all score high. As with all cancers, BRCA1-BRCA2-related tumors involve the loss of control over cell growth and proliferation. Thus, the presence of strong cell-cycle components in the clustering is expected.



Biological process	z-score
DNA damage checkpoint	7.4
nucleocytoplasmic transport	7.4
meiotic recombination	7.4
asymmetric cytokinesis	7.4
purine base biosynthesis	7.4
GMP biosynthesis	5.1
rRNA processing	5.1
glutamine metabolism	5.1
establishment and/or maintenance of cell polarity	5.1
gametogenesis	5.1
DNA replication	4.6
cell cycle arrest	4.4
central nervous system development	4.4
purine nucleotide biosynthesis	4.1
mRNA splicing	4.1
cell cycle	3.5
negative regulation of cell proliferation	3.4
induction of apoptosis by intracellular signals	2.8
oncogenesis	2.6
G1/S transition of mitotic cell cycle	2.5
protein kinase cascade	2.5
glycogen metabolism	2.3
regulation of cell cycle	2.1

Spectral Clustering (I)

- □ **Algorithms that cluster points using eigenvectors** of matrices derived from the data
- □ Obtain data representation in the **low-dimensional space** that can be easily clustered
- □ Variety of methods that use the eigenvectors differently

[Ng, Jordan, Weiss. NIPS 2001]

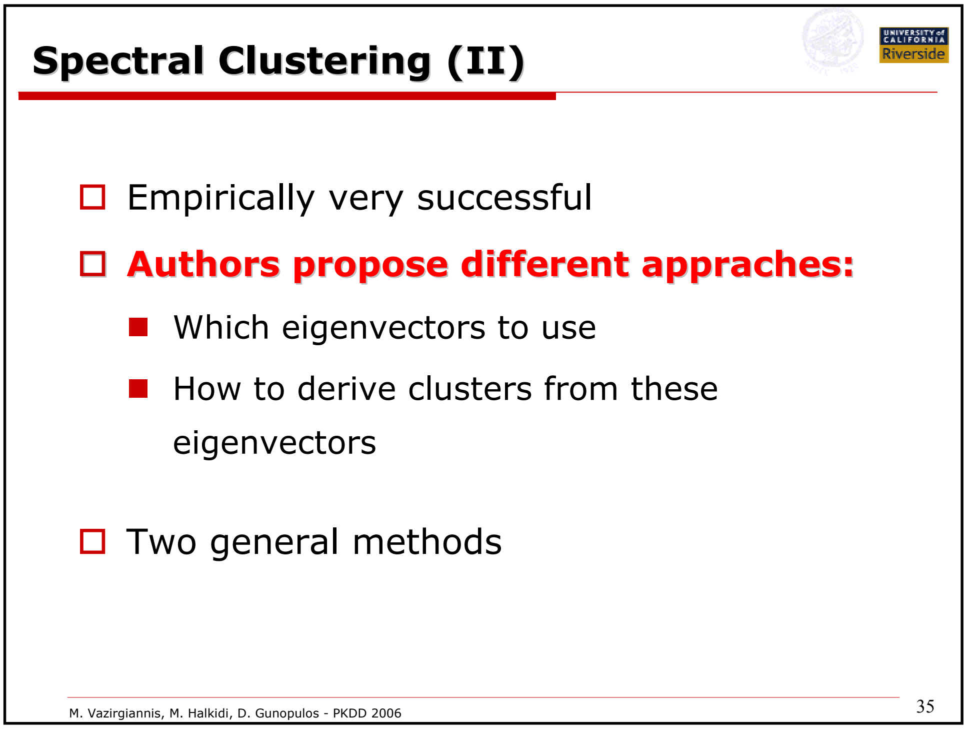
[Belkin, Niyogi, NIPS 2001]

[Dhillon, KDD 2001]

[Bach, Jordan NIPS 2003]

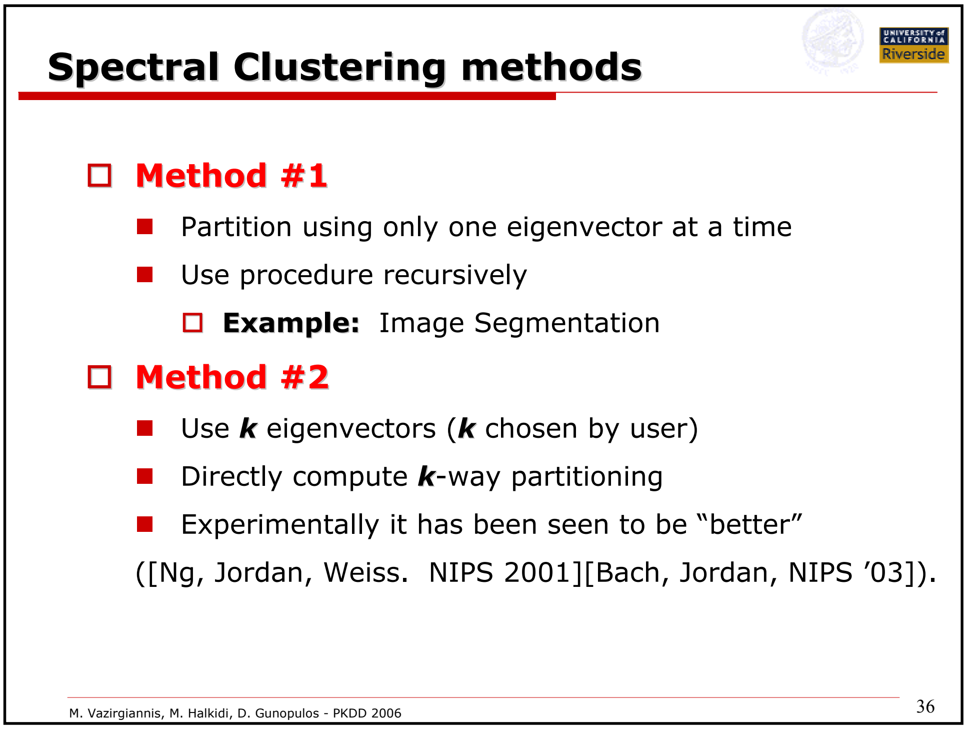
[Kamvar, Klein, Manning. IJCAI 2003]

[Jin, Ding, Kang, NIPS 2005]



Spectral Clustering (II)

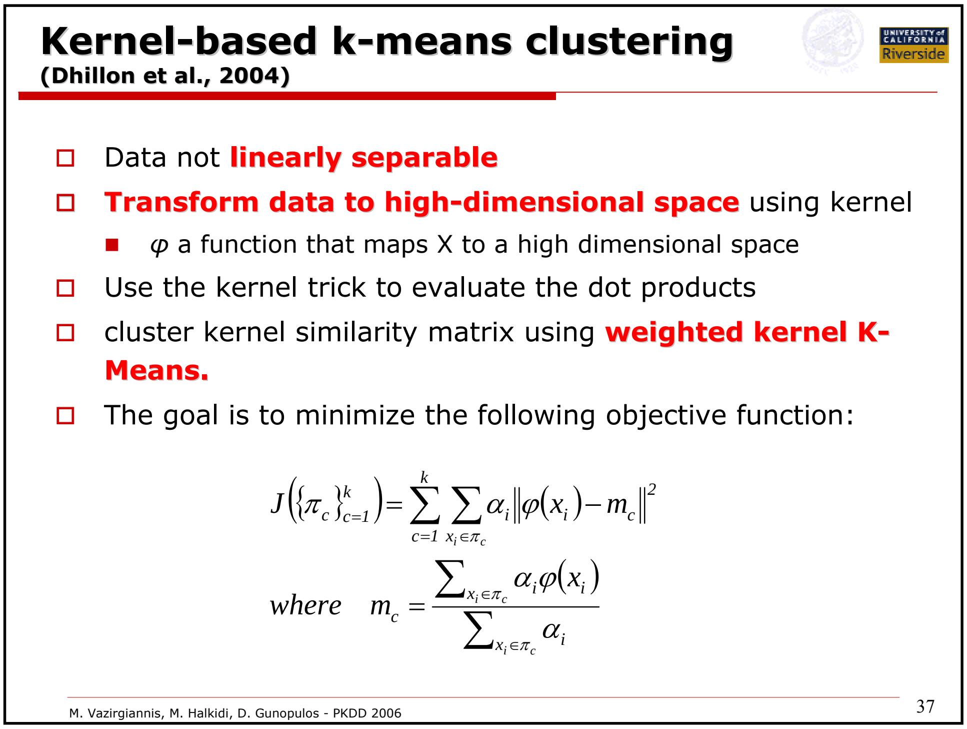
- □ Empirically very successful
- □ **Authors propose different appraches:**
  - Which eigenvectors to use
  - How to derive clusters from these eigenvectors
- □ Two general methods



Spectral Clustering methods

- □ **Method #1**
  - Partition using only one eigenvector at a time
  - Use procedure recursively
    - □ **Example:** Image Segmentation
- □ **Method #2**
  - Use ***k*** eigenvectors (***k*** chosen by user)
  - Directly compute ***k***-way partitioning
  - Experimentally it has been seen to be “better”

([Ng, Jordan, Weiss. NIPS 2001][Bach, Jordan, NIPS ’03]).



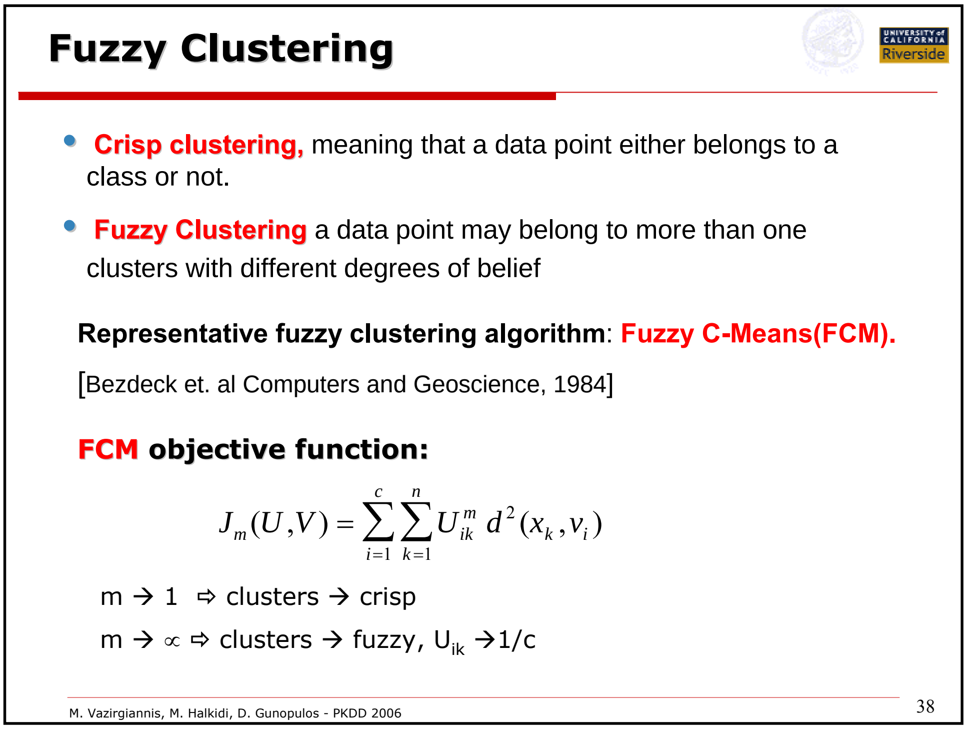
Kernel-based k-means clustering

(Dhillon et al., 2004)

- □ Data not **linearly separable**
- □ **Transform data to high-dimensional space** using kernel
  - ■ φ a function that maps X to a high dimensional space
- □ Use the kernel trick to evaluate the dot products
- □ cluster kernel similarity matrix using **weighted kernel K-Means.**
- □ The goal is to minimize the following objective function:

$$J\left(\{\pi_c\}_{c=1}^{k}\right)=\sum_{c=1}^{k}\sum_{x_i\in\pi_c}\alpha_i\left\|\varphi(x_i)-m_c\right\|^2$$

$$where \quad m_c=\frac{\sum_{x_i\in\pi_c}\alpha_i\varphi(x_i)}{\sum_{x_i\in\pi_c}\alpha_i}$$



Fuzzy Clustering

- **Crisp clustering,** meaning that a data point either belongs to a class or not.
- **Fuzzy Clustering** a data point may belong to more than one clusters with different degrees of belief

Representative fuzzy clustering algorithm: **Fuzzy C-Means(FCM).**

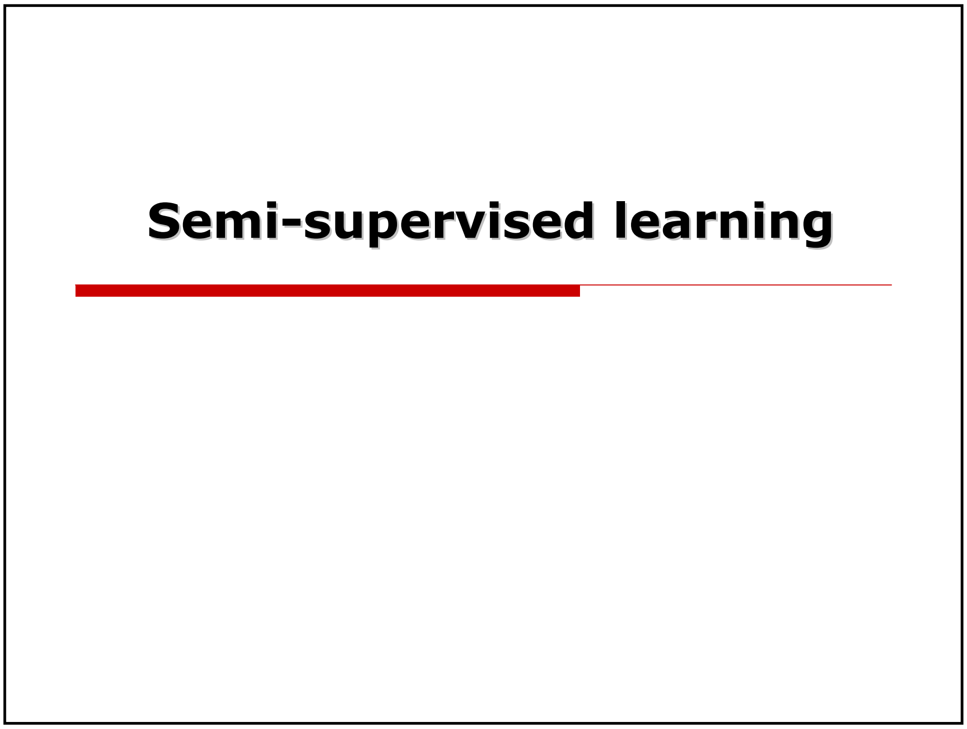
[Bezdeck et. al Computers and Geoscience, 1984]

FCM objective function:

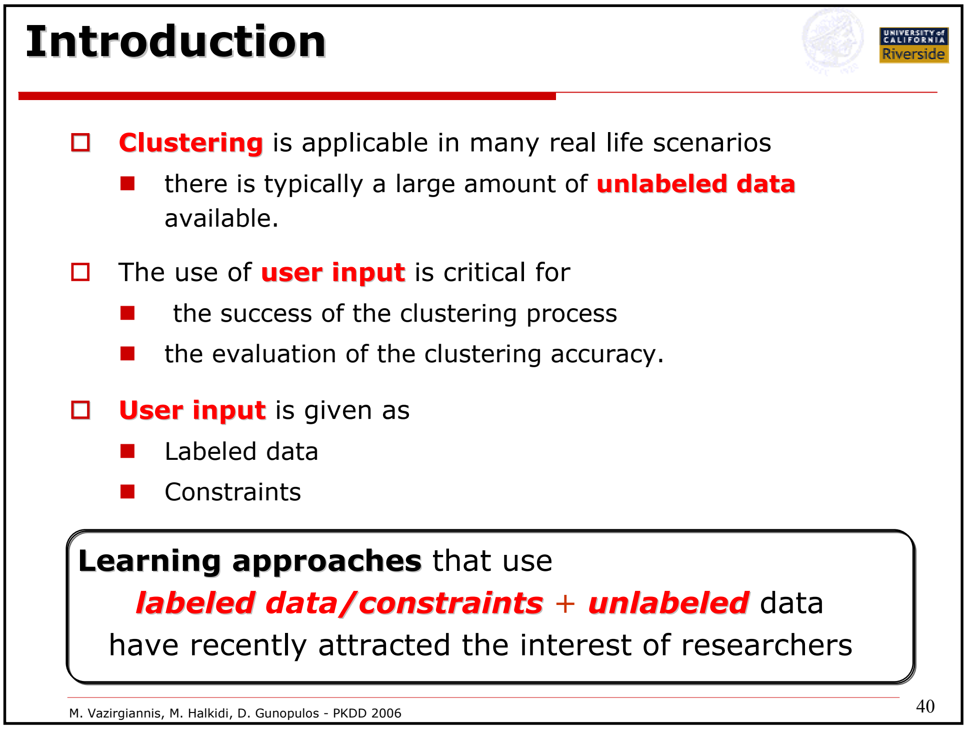
$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}U_{ik}^{m}\,d^2(x_k,v_i)$$

m → 1 ⇨ clusters → crisp

m → ∝ ⇨ clusters → fuzzy, U_{ik} →1/c



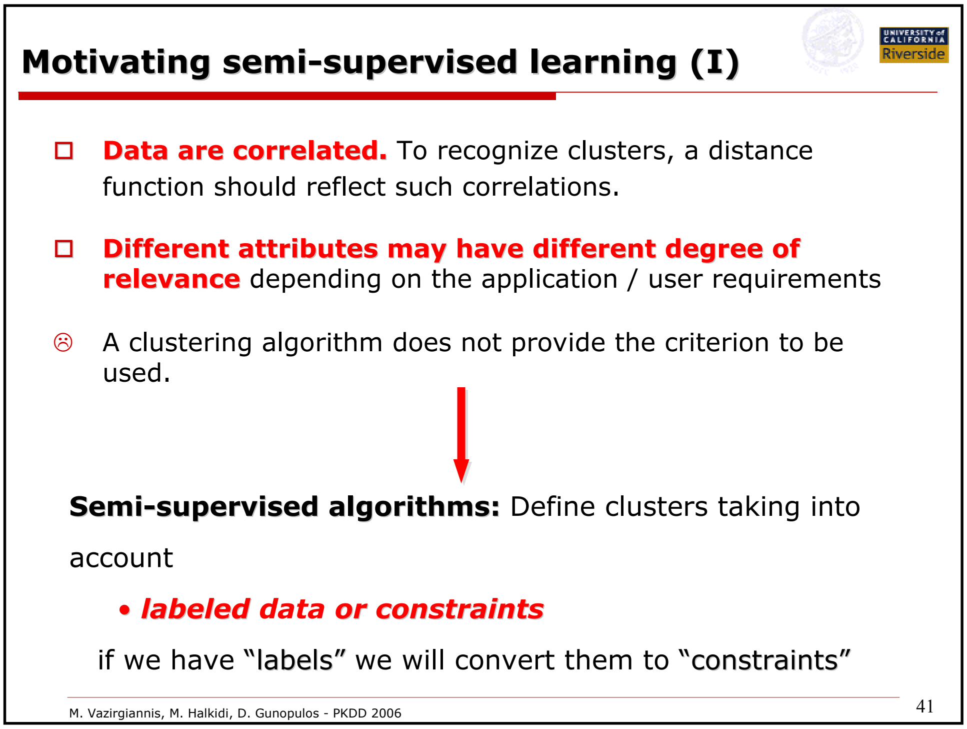
Semi-supervised learning



Introduction

- **Clustering** is applicable in many real life scenarios
  - there is typically a large amount of **unlabeled data** available.
- The use of **user input** is critical for
  - the success of the clustering process
  - the evaluation of the clustering accuracy.
- **User input** is given as
  - Labeled data
  - Constraints

Learning approaches that use
labeled data/constraints + ***unlabeled*** data
have recently attracted the interest of researchers



Motivating semi-supervised learning (I)

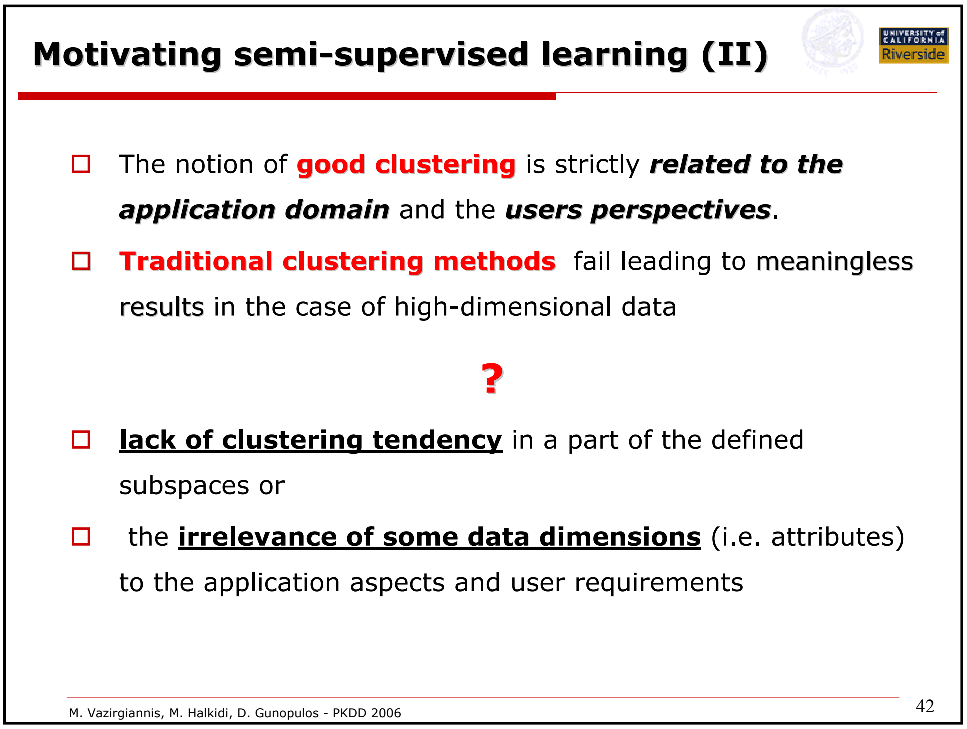
- □ **Data are correlated.** To recognize clusters, a distance function should reflect such correlations.
- □ **Different attributes may have different degree of relevance** depending on the application / user requirements
- ☹ A clustering algorithm does not provide the criterion to be used.

↓

Semi-supervised algorithms: Define clusters taking into account

- ***labeled data or constraints***

if we have "labels" we will convert them to "constraints"

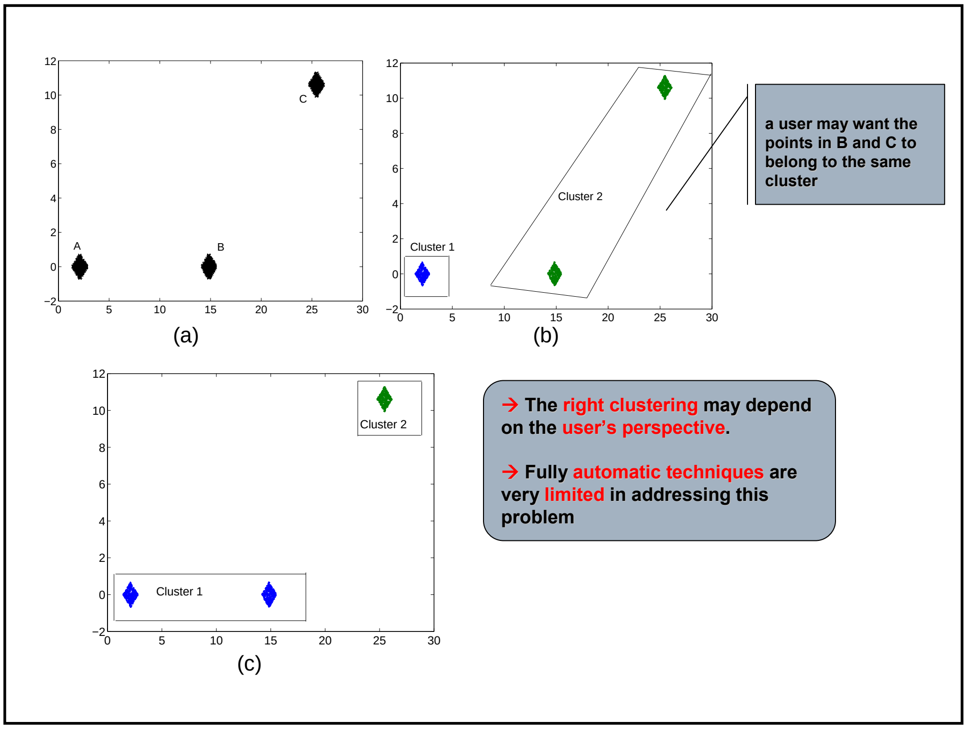


Motivating semi-supervised learning (II)

- □ The notion of **good clustering** is strictly ***related to the application domain*** and the ***users perspectives***.
- □ **Traditional clustering methods** fail leading to meaningless results in the case of high-dimensional data

?

- □ **lack of clustering tendency** in a part of the defined subspaces or
- □ the **irrelevance of some data dimensions** (i.e. attributes) to the application aspects and user requirements



(a)

(b)

(c)

a user may want the points in B and C to belong to the same cluster

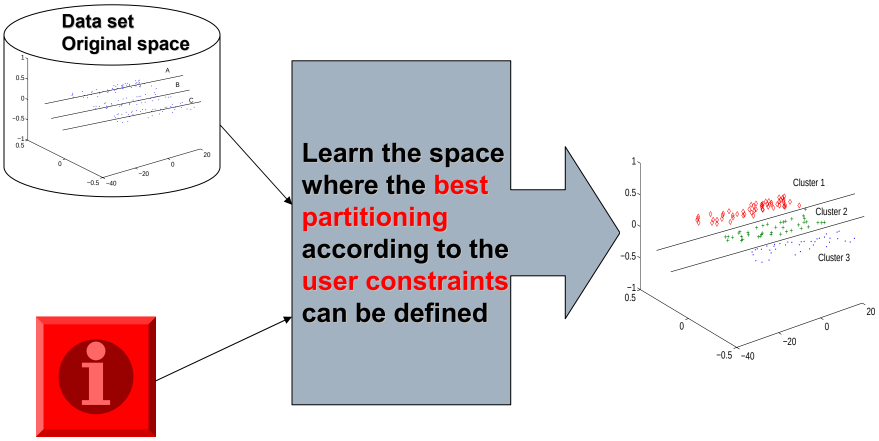
→ The **right clustering** may depend on the **user's perspective**.

→ Fully **automatic techniques** are very **limited** in addressing this problem

Clustering under constraints

- Use **constraints** to
  - learn a distortion/distance function
    - Points surrounding a pair of **must-link/cannot-link** points should be close to/far from each other
  - guide the algorithm to a useful solution
    - Two points should be in the same/different clusters

Semi-supervised learning framework



Learn the space where the **best partitioning** according to the **user constraints** can be defined

Semi-supervised learning Framework

Defining the constraints

- A set of points X = {x_1, …, x_n} on which sets of ***must-link(S)*** and ***cannot-link constraints(D)*** have been defined.
- **Must-link constraints**
  - **S:** {(x_i, x_j) in X }: x_i and x_j **should belong** to the same cluster
- **Cannot-link constraints**
  - **D:** {(x_i, x_j) in X} : x_i and x_j **cannot belong** to the same cluster
- **Conditional constraints**
  - δ-constraint and ε-constraint

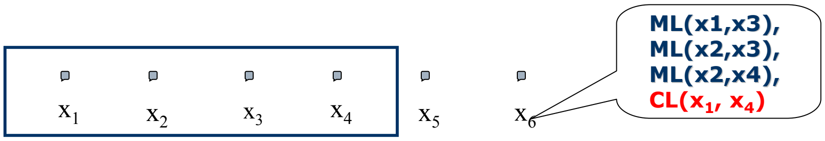
Clustering with constraints: Feasibility issues

- □ **Constraints** provide information that should be satisfied.
- □ Options for **constraint-based clustering**
  - **Satisfy all constraints**
    - □ **Not always possible:** A with B, B with C, C not with A.
    - □ Any combination of constraints involving cannot-link constraints is generally computationally intractable (Davidson & Ravi, ISMB 2000),
  - **Satisfy as many constraints as possible**

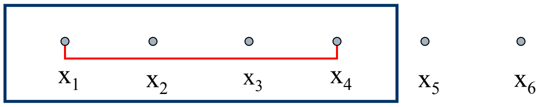
Feasibility under *Must-link(ML)* and *Cannot-link(CL) constraints*

x_1
x_2
x_3
x_4
x_5
x_6

Form the clusters implied by the ML={CC_1 … CC_r} constraints → Transitive closure of the ML constraints

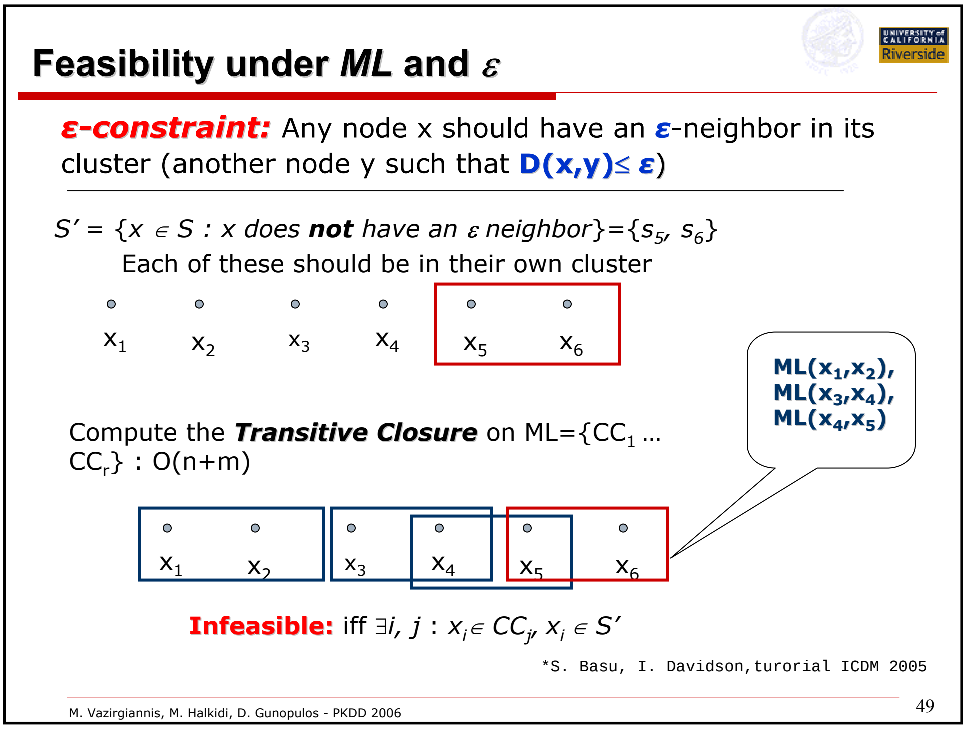


Construct Edges {E} between Nodes based on CL



Infeasible: iff $\exists h, k : e_h(x_i, x_j) : x_i, x_j \in CC_k$

*S. Basu, I. Davidson, tutorial ICDM 2005



Feasibility under *ML* and ε

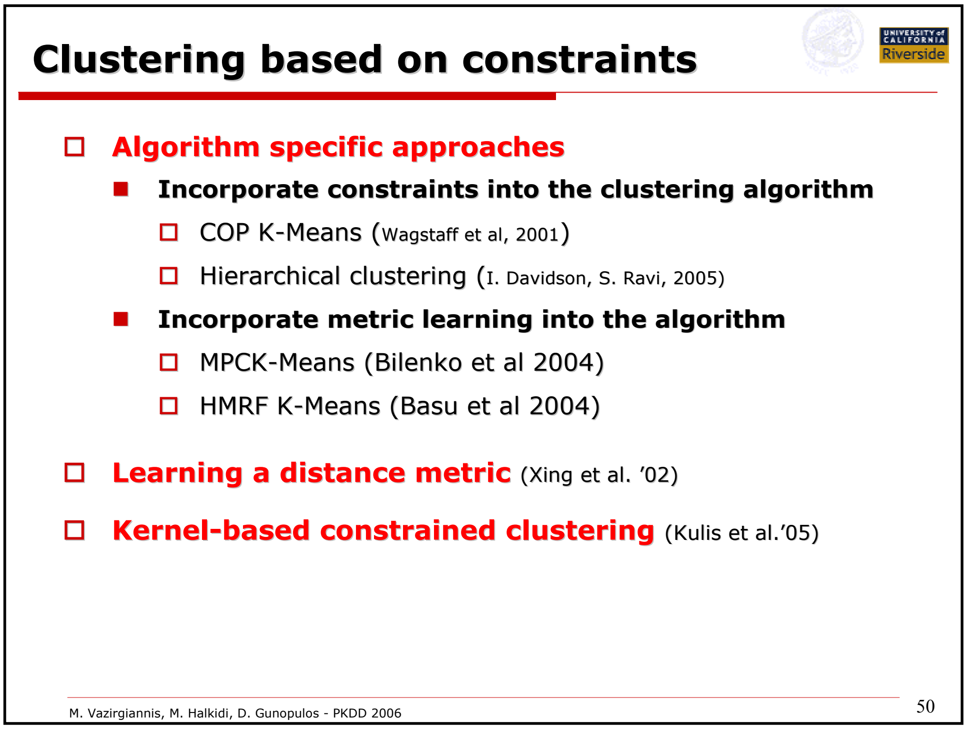
ε-constraint: Any node x should have an **ε**-neighbor in its cluster (another node y such that **D(x,y)≤ ε**)

*S′ = {x ∈ S : x does **not** have an ε neighbor}={s$_5$, s$_6$}*
Each of these should be in their own cluster

Compute the ***Transitive Closure*** on ML={CC$_1$ … CC$_r$} : O(n+m)

Infeasible: iff $\exists i, j : x_i \in CC_j, x_i \in S'$

*S. Basu, I. Davidson,turorial ICDM 2005



Clustering based on constraints

- **Algorithm specific approaches**
  - **Incorporate constraints into the clustering algorithm**
    - COP K-Means (Wagstaff et al, 2001)
    - Hierarchical clustering (I. Davidson, S. Ravi, 2005)
  - **Incorporate metric learning into the algorithm**
    - MPCK-Means (Bilenko et al 2004)
    - HMRF K-Means (Basu et al 2004)
- **Learning a distance metric** (Xing et al. ’02)
- **Kernel-based constrained clustering** (Kulis et al.’05)

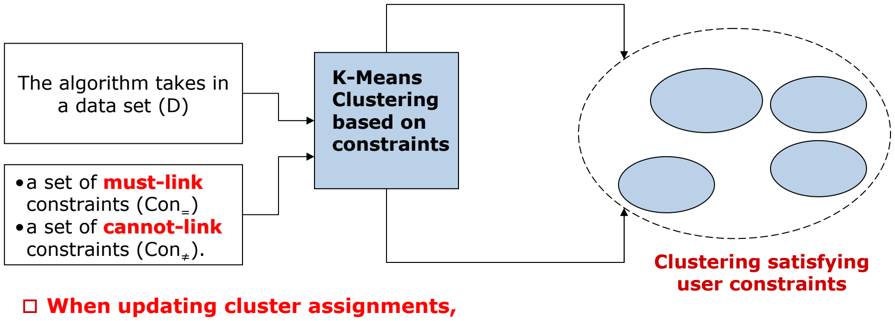
COP K-Means (I)

[Wagstaff et al, 2001]

- □ Semi-supervised variants of K-Means
- □ **Constraints:** Initial background knowledge
- □ **Must-link** & **Cannot-link** constraints are used in the clustering process
  - Generate a partition that satisfies all the given constraints

K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *ICML*, pages 577–584, 2001.

COP K-Means (II)



The algorithm takes in a data set (D)

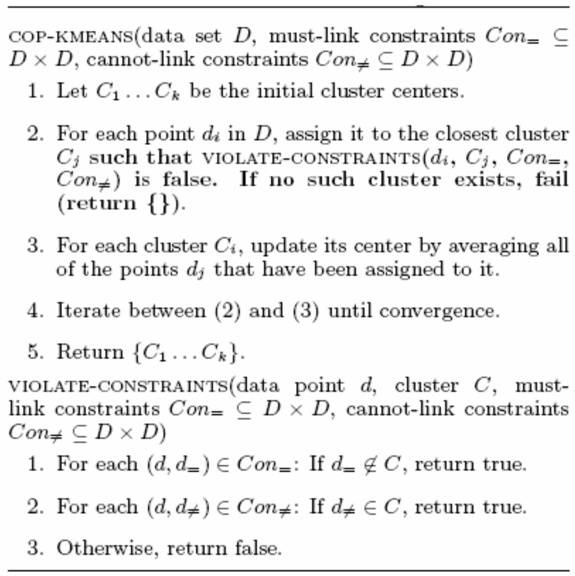
- •a set of **must-link** constraints ($Con_=$)
- •a set of **cannot-link** constraints ($Con_{\neq}$).

K-Means Clustering based on constraints

Clustering satisfying user constraints

- □ **When updating cluster assignments,**
  - we ensure that none of the specified constraints are violated.
- □ **Assign each point d_i to its closest cluster C_j.** This will succeed unless a constraint would be violated.
  - If there is another point $d_=$ that must be assigned to the same cluster as d, but that is already in some other cluster, or
  - there is another point $d_{\neq}$ that cannot be grouped with *d* but is already in *C*, then *d* cannot be placed in *C*.
- □ **Constraints are never broken**; if a legal cluster cannot be found for d, the empty partition (f_g) is returned.

COP K-Means Algorithm
[Wagstaff et al]



```
COP-KMEANS(data set $D$, must-link constraints $Con_= \subseteq D \times D$, cannot-link constraints $Con_{\neq} \subseteq D \times D$)
  1. Let $C_1 \ldots C_k$ be the initial cluster centers.
  2. For each point $d_i$ in $D$, assign it to the closest cluster $C_j$ such that VIOLATE-CONSTRAINTS($d_i$, $C_j$, $Con_=$, $Con_{\neq}$) is false. If no such cluster exists, fail (return {}).
  3. For each cluster $C_i$, update its center by averaging all of the points $d_j$ that have been assigned to it.
  4. Iterate between (2) and (3) until convergence.
  5. Return $\{C_1 \ldots C_k\}$.

VIOLATE-CONSTRAINTS(data point $d$, cluster $C$, must-link constraints $Con_= \subseteq D \times D$, cannot-link constraints $Con_{\neq} \subseteq D \times D$)
  1. For each $(d, d_=) \in Con_=$: If $d_= \notin C$, return true.
  2. For each $(d, d_{\neq}) \in Con_{\neq}$: If $d_{\neq} \in C$, return true.
  3. Otherwise, return false.
```

Hierarchical Clustering based on constraints
[I. Davidson, S. Ravi, 2005]

Instance: A set S of nodes, the (symmetric) distance **d(x,y)≥0** for each pair of nodes x and y and a collection C of constraints

- **Question:** Can we create a dendrogram for S so that all the constraints in C are satisfied?

Davidson I. and Ravi, S. S. "Hierarchical Clustering with Constraints: Theory and Practice", *In PKDD 2005*

Constraints and Irreducible Clusterings

- A **feasible clustering $C=\{C_1, C_2, \ldots, C_k\}$** of a set S is irreducible if no pair of clusters in C can be merged to obtain a feasible clustering with k-1 clusters.

If mergers are not done correctly, the dendrogram may stop prematurely

- **$X=\{x_1, x_2, \ldots, x_k\}$, $Y=\{y_1, y_2, \ldots, y_k\}$, $Z=\{z_1, z_2, \ldots, z_k\}$, $W=\{w_1, w_2, \ldots, w_k\}$**
- **CL-constraints**
  - $\forall\{x_i, x_j\}$, $i \neq j$
  - $\forall\{w_i, w_j\}$, $i \neq j$
  - $\forall\{y_i, z_j\}$, $i \leq j$, $j \leq k$

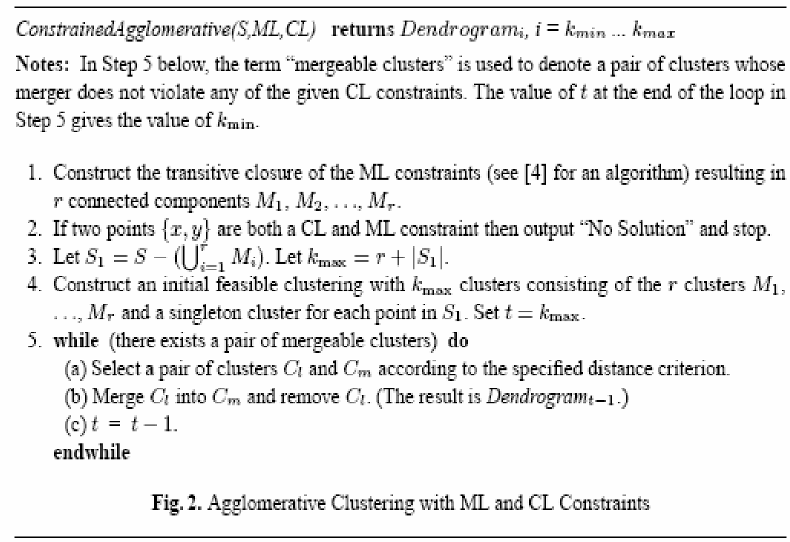
- Feasible clustering with 2k clusters: $\{x_1, y_1\}, \{x_2, y_2\}, \ldots, \{x_k, y_k\}, \{z_1, w_1\}, \{z_2, w_2\}, \ldots, \{z_k, w_k\}$

But then get stuck

- **Alternative is:**

$\{x_1, w_1, y_1, y_2, \ldots, y_k\}, \{x_2, w_2, z_1, z_2, \ldots, z_k\}, \{x_3, w_3\}, \ldots, \{x_k, w_k\}$

Using constraints for hierarchical clustering



ConstrainedAgglomerative(S,ML,CL) **returns** *Dendrogram$_i$*, $i = k_{min} \ldots k_{max}$

Notes: In Step 5 below, the term "mergeable clusters" is used to denote a pair of clusters whose merger does not violate any of the given CL constraints. The value of t at the end of the loop in Step 5 gives the value of $k_{\min}$.

```
1. Construct the transitive closure of the ML constraints (see [4] for an algorithm) resulting in
   $r$ connected components $M_1, M_2, \ldots, M_r$.
2. If two points $\{x, y\}$ are both a CL and ML constraint then output "No Solution" and stop.
3. Let $S_1 = S - (\bigcup_{i=1}^{r} M_i)$. Let $k_{\max} = r + |S_1|$.
4. Construct an initial feasible clustering with $k_{\max}$ clusters consisting of the $r$ clusters $M_1$,
   $\ldots, M_r$ and a singleton cluster for each point in $S_1$. Set $t = k_{\max}$.
5. while (there exists a pair of mergeable clusters) do
     (a) Select a pair of clusters $C_l$ and $C_m$ according to the specified distance criterion.
     (b) Merge $C_l$ into $C_m$ and remove $C_l$. (The result is Dendrogram$_{t-1}$.)
     (c) $t = t - 1$.
   endwhile
```

Fig. 2. Agglomerative Clustering with ML and CL Constraints

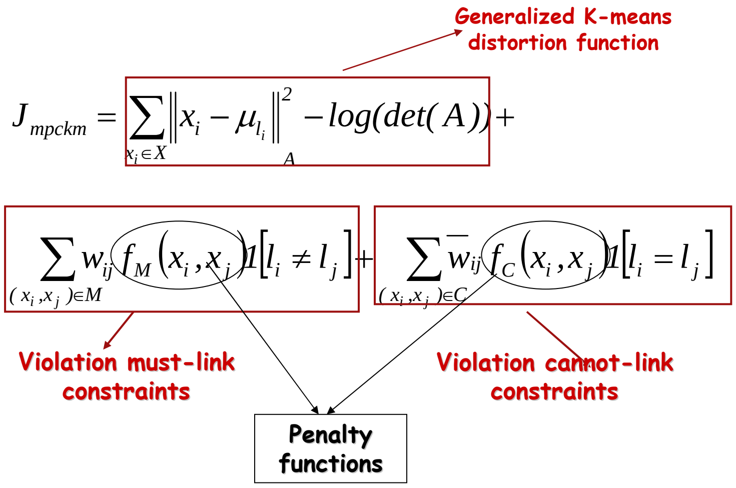
MPCK-Means

[Bilenko et al 2004]

- **Incorporate metric learning directly into the clustering algorithm**
  - Unlabeled data influence the metric learning process
- **Objective function**
  - Sum of total square distances between the points and cluster centroids
  - Cost of violating the pair-wise constraints

M. Bilenko, S. Basu, R. Mooney. "Integrating Constraints and Metric Learning in Semi-supervised clustering. In Proceedings of the 21st ICML Conference, July 2004.

Unifying constraints and Metric learning



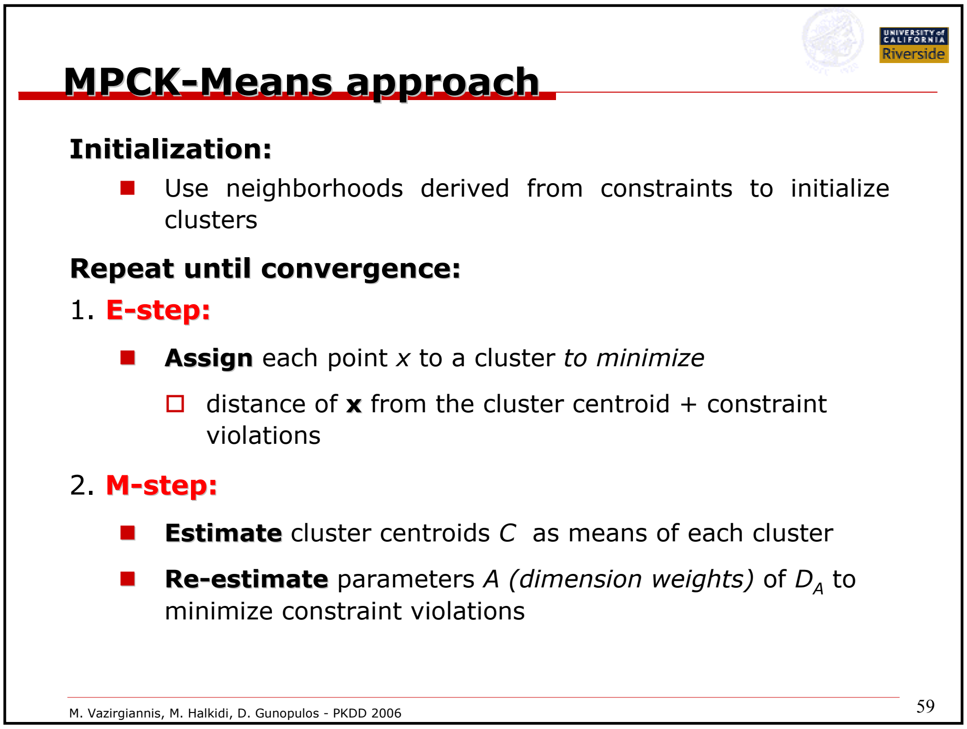
Generalized K-means distortion function

$$J_{mpckm} = \boxed{\sum_{x_i \in X} \left\| x_i - \mu_{l_i} \right\|_A^2 - log(det(A))} + \boxed{\sum_{(x_i,x_j)\in M} w_{ij} f_M(x_i,x_j) 1[l_i \neq l_j]} + \boxed{\sum_{(x_i,x_j)\in C} \bar{w}_{ij} f_C(x_i,x_j) 1[l_i = l_j]}$$

Violation must-link constraints

Violation cannot-link constraints

Penalty functions



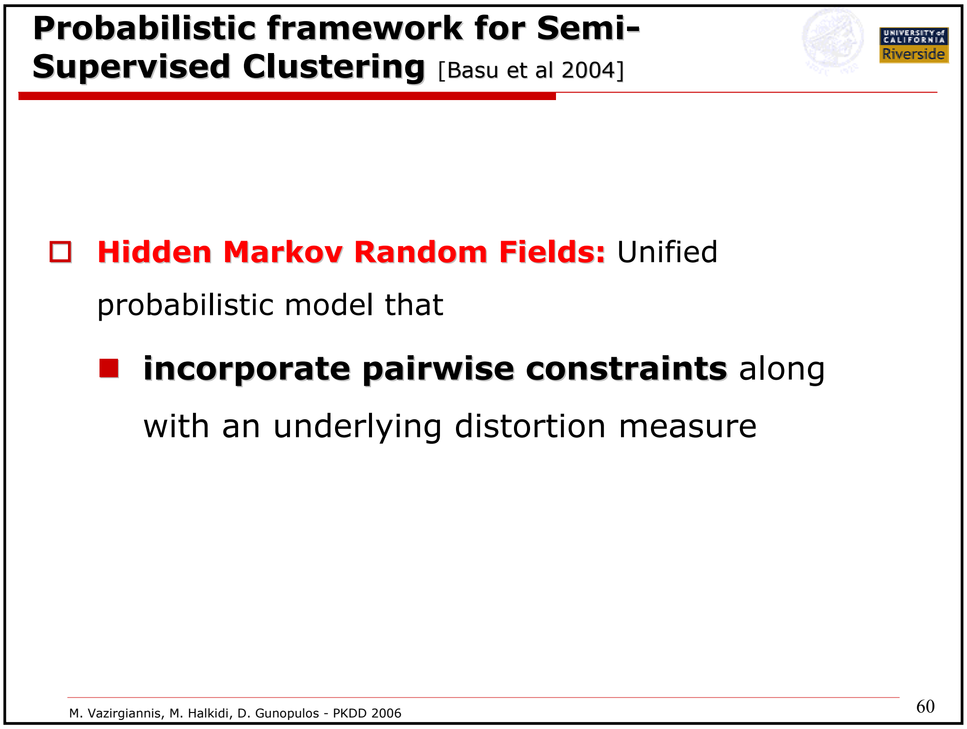
MPCK-Means approach

Initialization:

- Use neighborhoods derived from constraints to initialize clusters

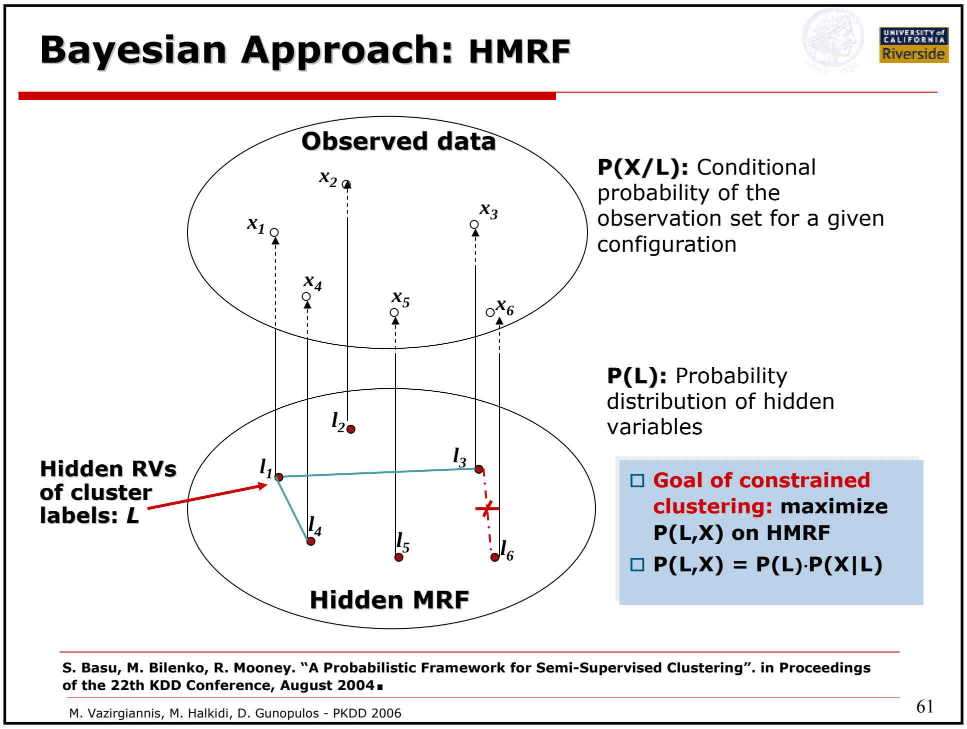
Repeat until convergence:

1. **E-step:**
   - **Assign** each point *x* to a cluster *to minimize*
     - distance of **x** from the cluster centroid + constraint violations
2. **M-step:**
   - **Estimate** cluster centroids *C* as means of each cluster
   - **Re-estimate** parameters *A (dimension weights)* of D_A to minimize constraint violations



Probabilistic framework for Semi-Supervised Clustering [Basu et al 2004]

- **Hidden Markov Random Fields:** Unified probabilistic model that
  - **incorporate pairwise constraints** along with an underlying distortion measure



Bayesian Approach: HMRF

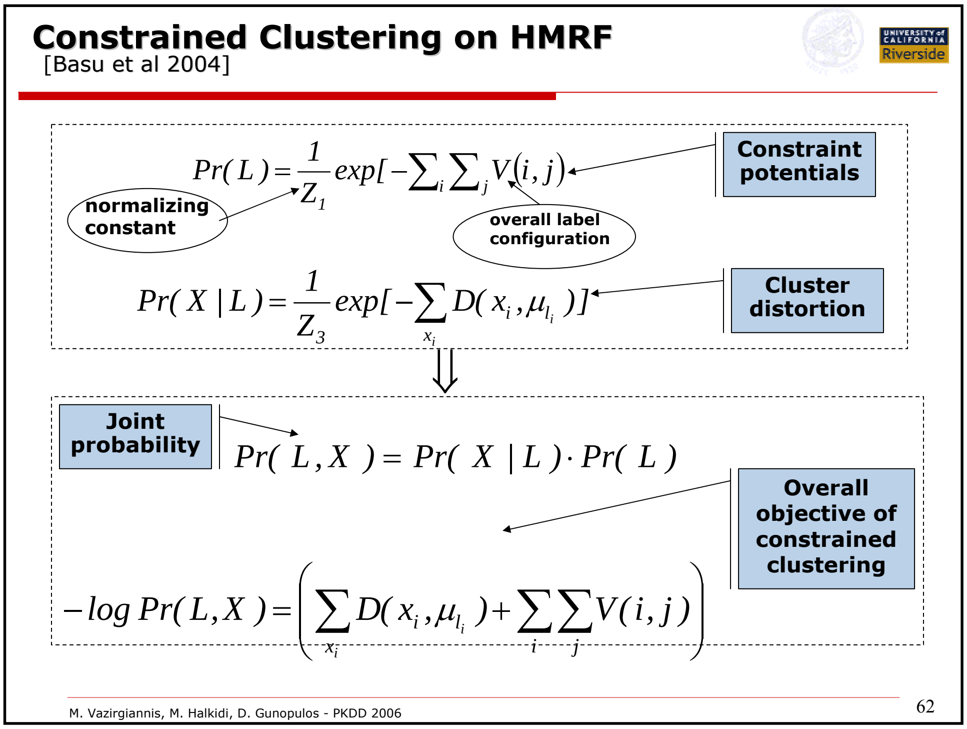
P(X/L): Conditional probability of the observation set for a given configuration

P(L): Probability distribution of hidden variables

Hidden RVs of cluster labels: *L*

- □ **Goal of constrained clustering: maximize P(L,X) on HMRF**
- □ **P(L,X) = P(L)·P(X|L)**

S. Basu, M. Bilenko, R. Mooney. "A Probabilistic Framework for Semi-Supervised Clustering". in Proceedings of the 22th KDD Conference, August 2004



Constrained Clustering on HMRF

[Basu et al 2004]

$$Pr(L) = \frac{1}{Z_1} exp[-\sum_i \sum_j V(i,j)]$$

normalizing constant

overall label configuration

Constraint potentials

$$Pr(X|L) = \frac{1}{Z_3} exp[-\sum_{x_i} D(x_i, \mu_{l_i})]$$

Cluster distortion

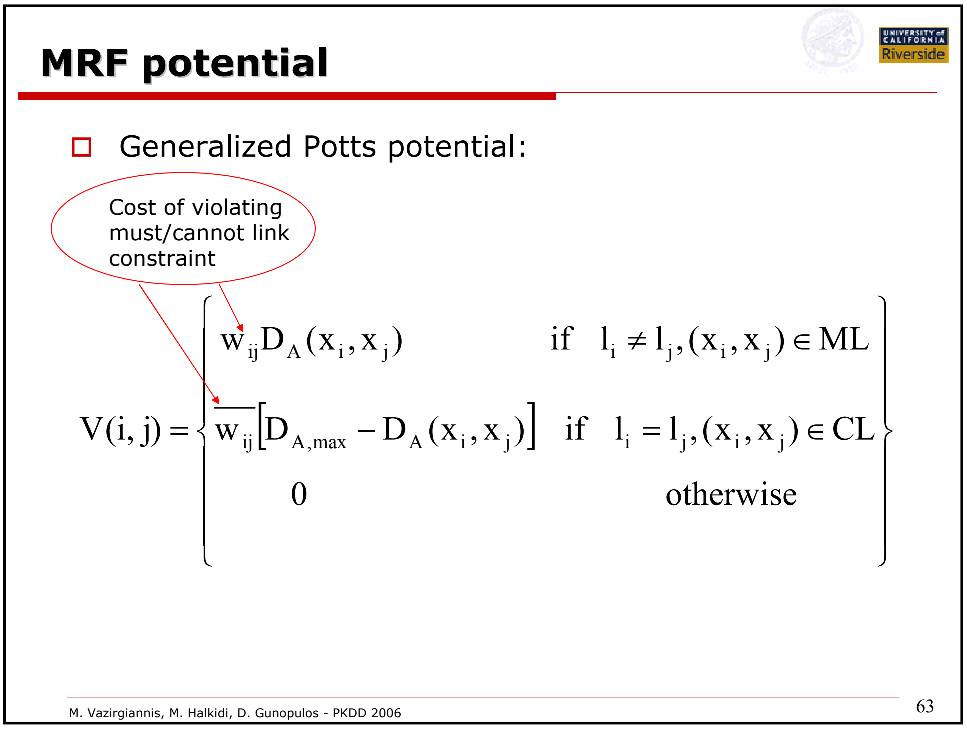
$$\Downarrow$$

Joint probability

$$Pr(L,X) = Pr(X|L) \cdot Pr(L)$$

Overall objective of constrained clustering

$$-log\, Pr(L,X) = \left(\sum_{x_i} D(x_i, \mu_{l_i}) + \sum_i \sum_j V(i,j) \right)$$

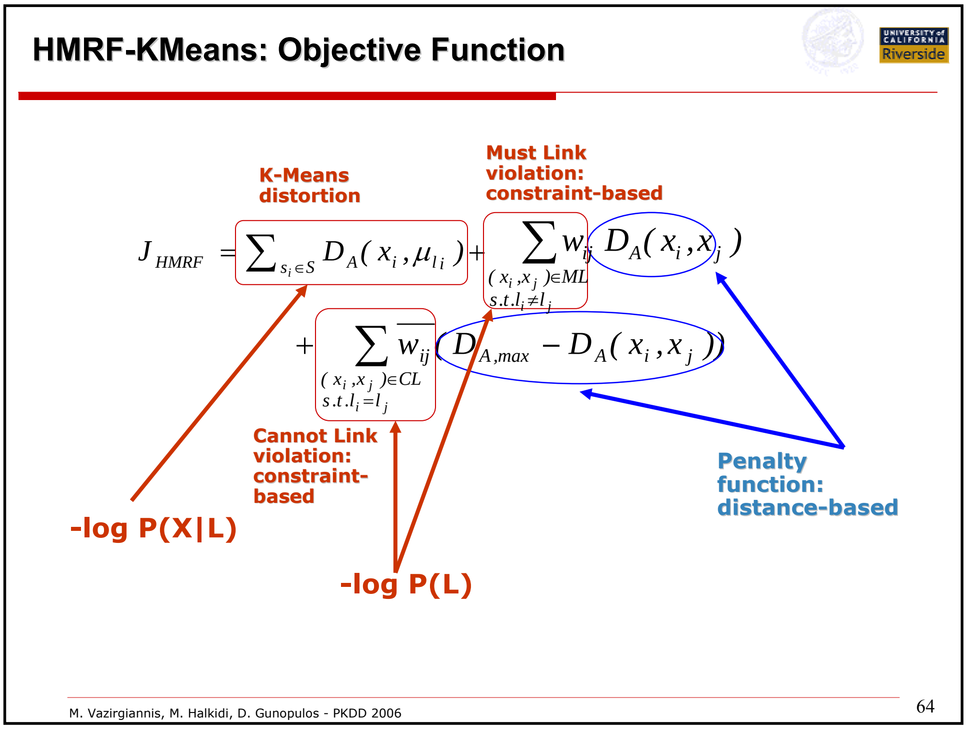


MRF potential

- Generalized Potts potential:

Cost of violating must/cannot link constraint

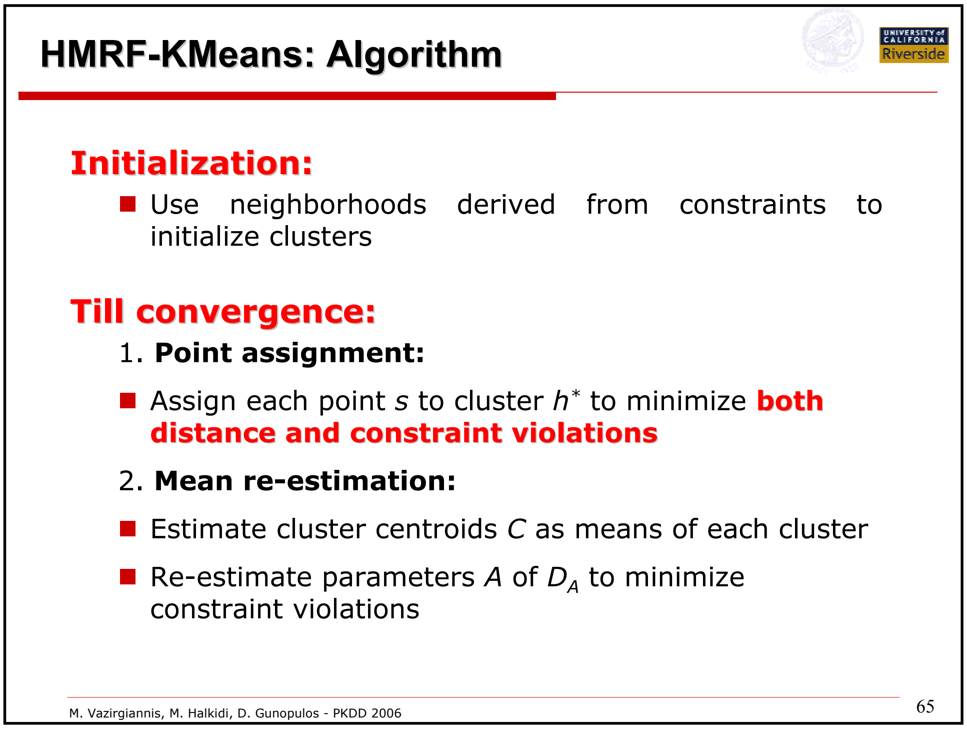
$$V(i,j)=\left\{\begin{array}{ll} w_{ij}D_A(x_i,x_j) & \text{if } \; l_i \neq l_j, (x_i,x_j)\in ML \\ \overline{w_{ij}}\left[D_{A,\max}-D_A(x_i,x_j)\right] & \text{if } \; l_i = l_j, (x_i,x_j)\in CL \\ 0 & \text{otherwise} \end{array}\right\}$$



HMRF-KMeans: Objective Function

$$J_{HMRF} = \boxed{\sum_{s_i\in S} D_A(x_i,\mu_{l_i})} + \boxed{\sum_{\substack{(x_i,x_j)\in ML\\ s.t. l_i\neq l_j}} w_{ij}} D_A(x_i,x_j) + \boxed{\sum_{\substack{(x_i,x_j)\in CL\\ s.t. l_i=l_j}} \overline{w_{ij}}} (D_{A,\max} - D_A(x_i,x_j))$$

K-Means distortion
Must Link violation: constraint-based
Cannot Link violation: constraint-based
Penalty function: distance-based
-log P(X|L)
-log P(L)



HMRF-KMeans: Algorithm

Initialization:

- Use neighborhoods derived from constraints to initialize clusters

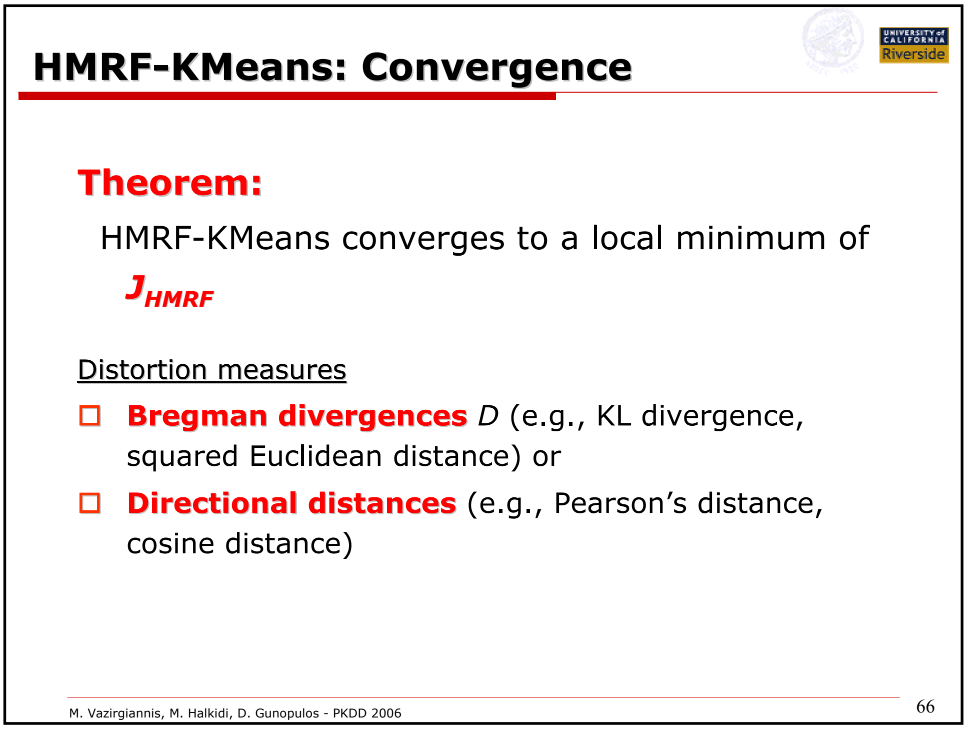
Till convergence:

1. **Point assignment:**

- Assign each point s to cluster h^* to minimize **both distance and constraint violations**

2. **Mean re-estimation:**

- Estimate cluster centroids C as means of each cluster
- Re-estimate parameters A of D_A to minimize constraint violations



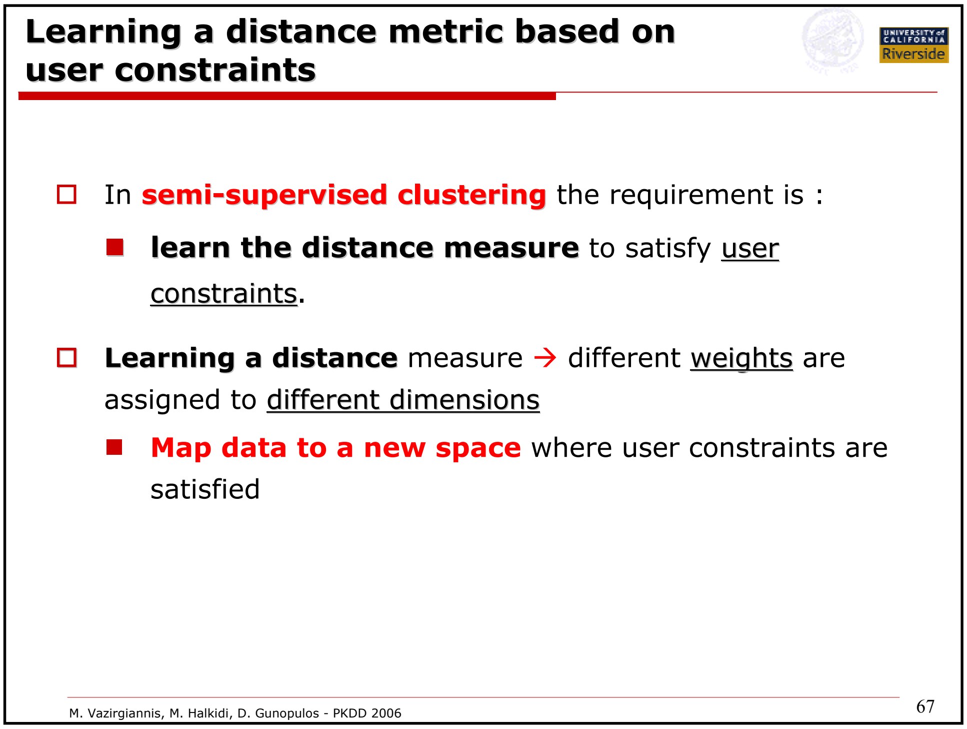
HMRF-KMeans: Convergence

Theorem:

HMRF-KMeans converges to a local minimum of J_{HMRF}

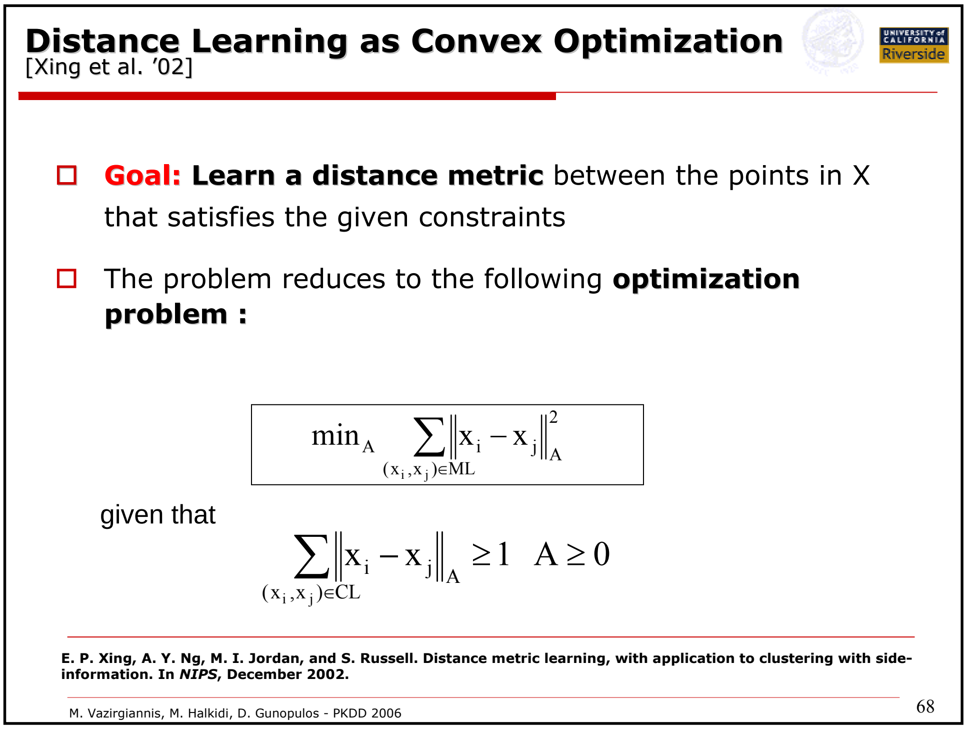
Distortion measures

- □ **Bregman divergences** D (e.g., KL divergence, squared Euclidean distance) or
- □ **Directional distances** (e.g., Pearson’s distance, cosine distance)



Learning a distance metric based on user constraints

- ☐ In **semi-supervised clustering** the requirement is :
  - **learn the distance measure** to satisfy user constraints.
- ☐ **Learning a distance** measure → different weights are assigned to different dimensions
  - **Map data to a new space** where user constraints are satisfied



Distance Learning as Convex Optimization

[Xing et al. '02]

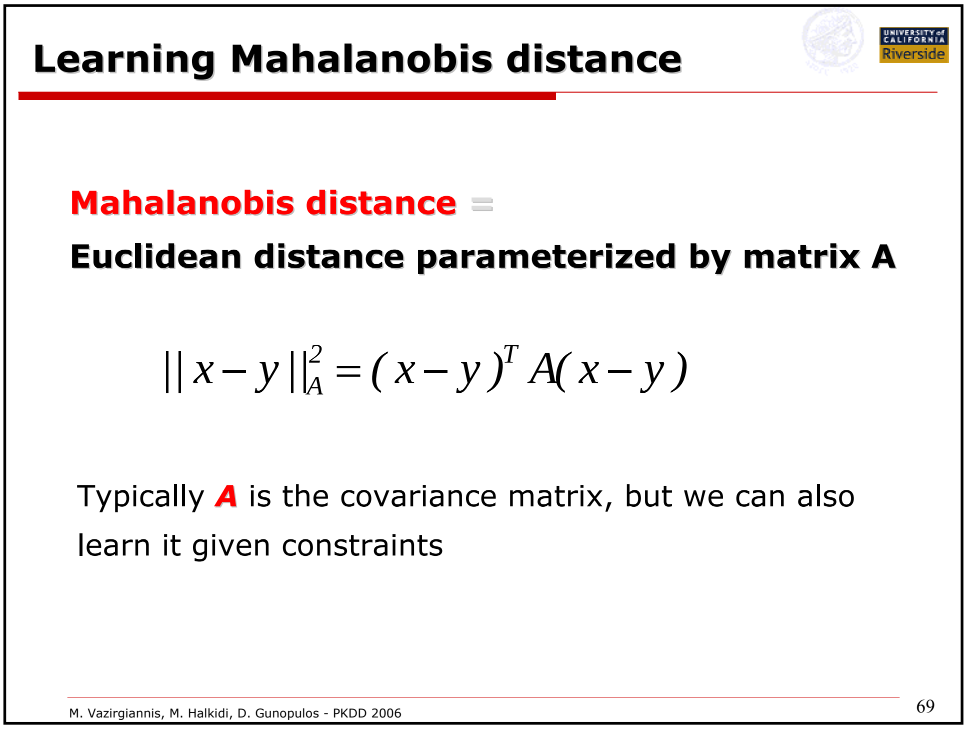
- ☐ **Goal: Learn a distance metric** between the points in X that satisfies the given constraints
- ☐ The problem reduces to the following **optimization problem :**

$$\boxed{\min_A \sum_{(x_i,x_j)\in ML} \left\| x_i - x_j \right\|_A^2}$$

given that

$$\sum_{(x_i,x_j)\in CL} \left\| x_i - x_j \right\|_A \geq 1 \quad A \geq 0$$

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *NIPS*, December 2002.



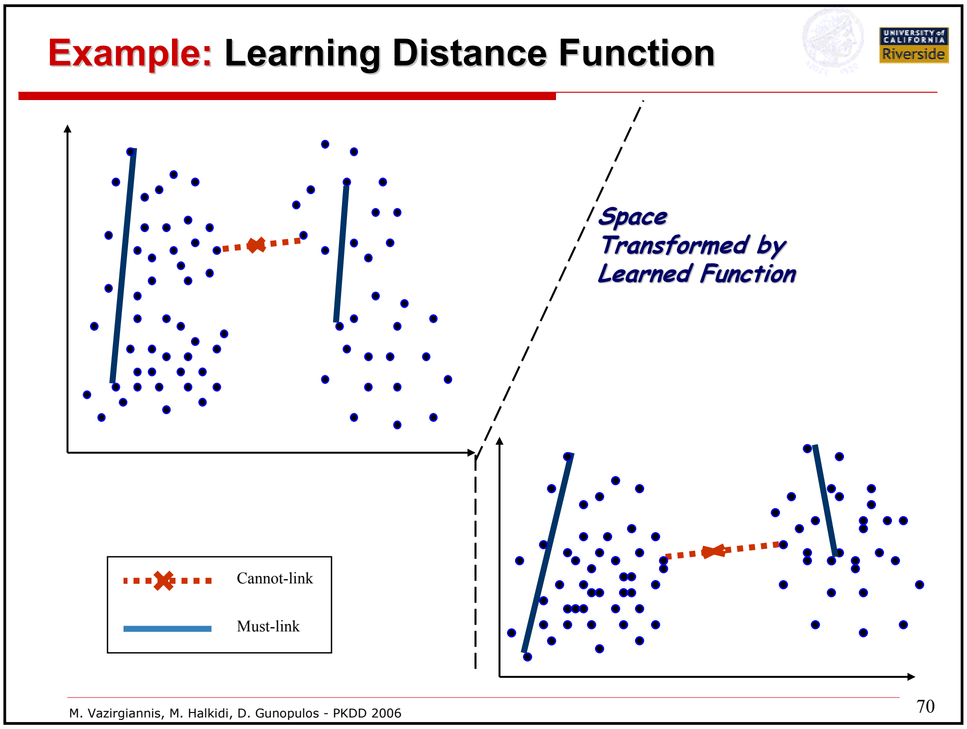
Learning Mahalanobis distance

Mahalanobis distance =

Euclidean distance parameterized by matrix A

$$||x-y||_A^2=(x-y)^T A(x-y)$$

Typically ***A*** is the covariance matrix, but we can also learn it given constraints



Example: Learning Distance Function

The Diagonal *A* Case

- □ Considering the case of learning ***a diagonal*** A
- □ we can solve the original ***optimization problem*** using Newton-Raphson to efficiently optimize the following

$$g(A) = \sum_{(x_i,x_j)\in ML} \left\| x_i - x_j \right\|_A^2 - \log\left(\sum_{(x_i,x_j)\in CL} \left\| x_i - x_j \right\|_A \right)$$

Use **Newton Raphson Technique**:

x’ = x – g(x)/g’(x)

g(A’)=A-g(A).J^{-1}(A)

Full *A* Case: Alternative Formulation

- □ **Equivalent optimization problem**

$$\max_A g(A) = \sum_{(s_i,s_j)\in CL} \| x_i \text{ - } x_j \|_A$$

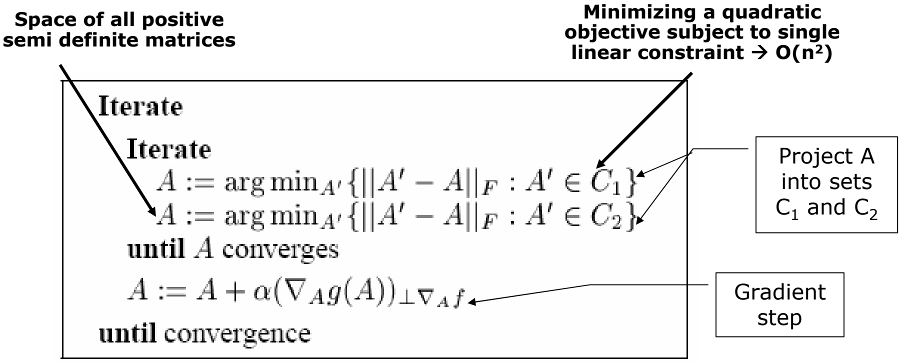
$$\text{s.t.} \quad f(A) = \sum_{(s_i,s_j)\in ML} \| x_i - x_j \|_A^2 \leq 1 \quad : \quad C_1$$

$$A \geq 0 \quad : \quad C_2$$

Optimization Algorithm - Full *A* Case

□ Solve optimization problem using combination of

- **gradient ascent:** to optimize the objective
- **iterated projection algorithm:** to satisfy the constraints



Space of all positive semi definite matrices

Minimizing a quadratic objective subject to single linear constraint → O(n²)

```
Iterate
    Iterate
        $A := \arg\min_{A'} \{||A' - A||_F : A' \in C_1\}$
        $A := \arg\min_{A'} \{||A' - A||_F : A' \in C_2\}$
    until $A$ converges
    $A := A + \alpha(\nabla_A g(A))_{\perp \nabla_A f}$
until convergence
```

Project A into sets C_1 and C_2

Gradient step

Kernel based Semi-supervised clustering

[Kulis et al.'05]

A non-linear transformation, φ

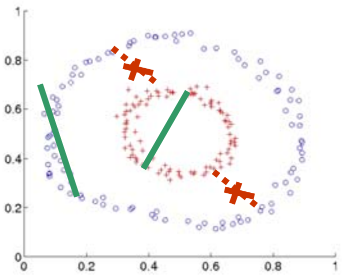
- maps data to a high dimensional space
- the data are expected to be more separable
- a kernel function **k (x, y)** computes **φ(x)·φ(y)**

The user gives constraints

The appropriate kernel is created based on constraints

$$J\left(\{\pi\}_{c=1}^{k}\right) = \sum_{c=1}^{k} \sum_{x_i \in \pi_c} \|\phi(x_i) - m_c\|^2 - \sum_{\substack{x_i, x_j \in ML \\ l_i = l_j}} w_{ij} + \sum_{\substack{x_i, x_j \in CL \\ l_i = l_j}} w_{ij}$$

Reward for constraint satisfaction

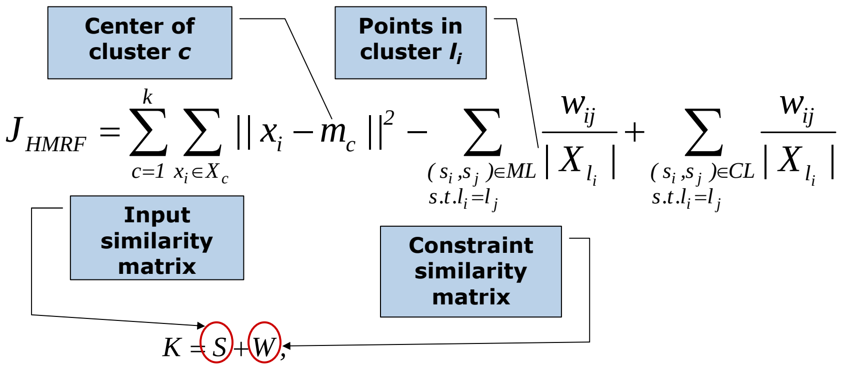


Semi-Supervised Kernel-KMeans

[Kulis et al.'05]

- **Algorithm:**
  - Constructs the appropriate kernel matrix from data and constraints
  - Runs weighted kernel K-Means
- **Input of the algorithm:** Kernel matrix
  - Kernel function on vector data or
  - Graph affinity matrix
- **Benefits:**
  - HMRF-KMeans and Spectral Clustering are special cases
  - Fast algorithm for constrained graph-based clustering
  - Kernels allow constrained clustering with non-linear cluster boundaries

Kernel for HMRF-KMeans with squared Euclidean distance



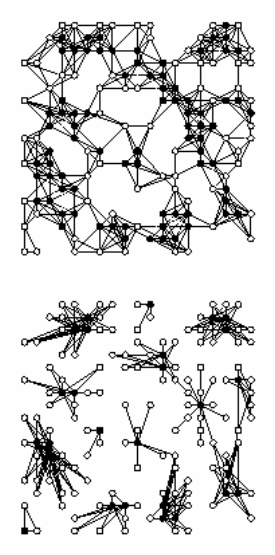
$$J_{HMRF} = \sum_{c=1}^{k} \sum_{x_i \in X_c} || x_i - m_c ||^2 - \sum_{\substack{(s_i,s_j)\in ML \\ s.t. l_i = l_j}} \frac{w_{ij}}{| X_{l_i} |} + \sum_{\substack{(s_i,s_j)\in CL \\ s.t. l_i = l_j}} \frac{w_{ij}}{| X_{l_i} |}$$

$$K = S + W,$$

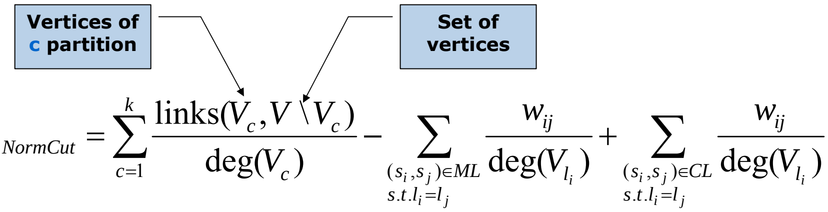
$$\text{where} \begin{cases} S_{ij} = x_i \cdot x_j \text{ , input similarity matrix,} \\ W_{ij} = \begin{matrix} + w_{ij} \text{ if } (x_i, x_j) \in ML \\ - w_{ij} \text{ if } (x_i, x_j) \in CL \end{matrix} \end{cases}$$

Graph-based constrained clustering

- **Constrained graph clustering:**
  - minimize cut in input graph while maximally respecting a given set of constraints



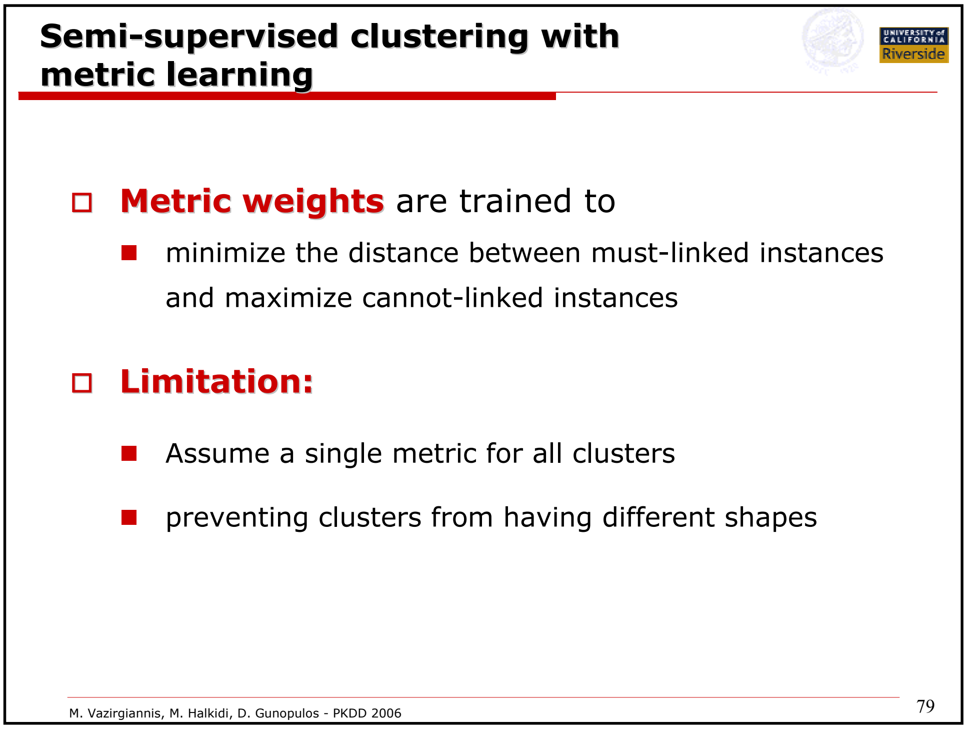
Kernel for Constrained Normalized-Cut Objective



$$J_{NormCut} = \sum_{c=1}^{k} \frac{\text{links}(V_c, V \setminus V_c)}{\deg(V_c)} - \sum_{\substack{(s_i, s_j) \in ML \\ s.t. l_i = l_j}} \frac{w_{ij}}{\deg(V_{l_i})} + \sum_{\substack{(s_i, s_j) \in CL \\ s.t. l_i = l_j}} \frac{w_{ij}}{\deg(V_{l_i})}$$

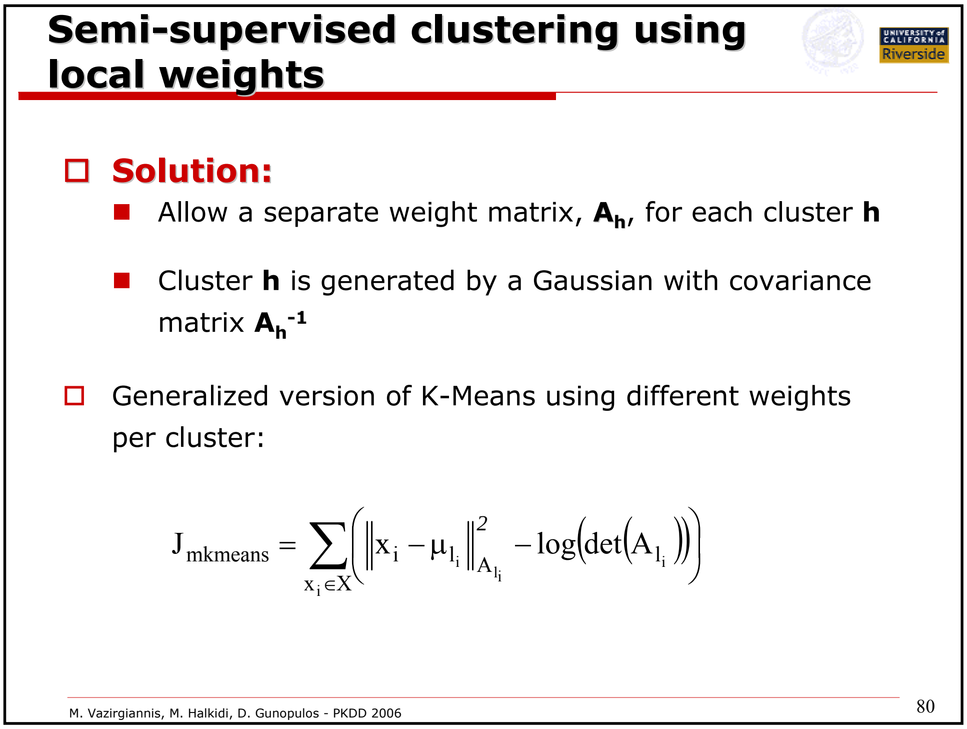
$$\mathrm{K} = \mathrm{D}^{-1}\mathrm{AD} + \mathrm{D}^{-1}\mathrm{WD},$$

$$\text{where} \begin{cases} \mathrm{A}_{\mathrm{ij}} = \text{graph affinity (i, j)}, \\ \mathrm{D} = \text{diagonal degree matrix} \\ \mathrm{W}_{\mathrm{ij}} = \begin{matrix} +\mathrm{w}_{\mathrm{ij}} \text{ if } (\mathrm{x}_{\mathrm{i}}, \mathrm{x}_{\mathrm{j}}) \in \mathrm{ML} \\ -\mathrm{w}_{\mathrm{ij}} \text{ if } (\mathrm{x}_{\mathrm{i}}, \mathrm{x}_{\mathrm{j}}) \in \mathrm{CL} \end{matrix} \end{cases}$$



Semi-supervised clustering with metric learning

- **Metric weights** are trained to
  - minimize the distance between must-linked instances and maximize cannot-linked instances
- **Limitation:**
  - Assume a single metric for all clusters
  - preventing clusters from having different shapes

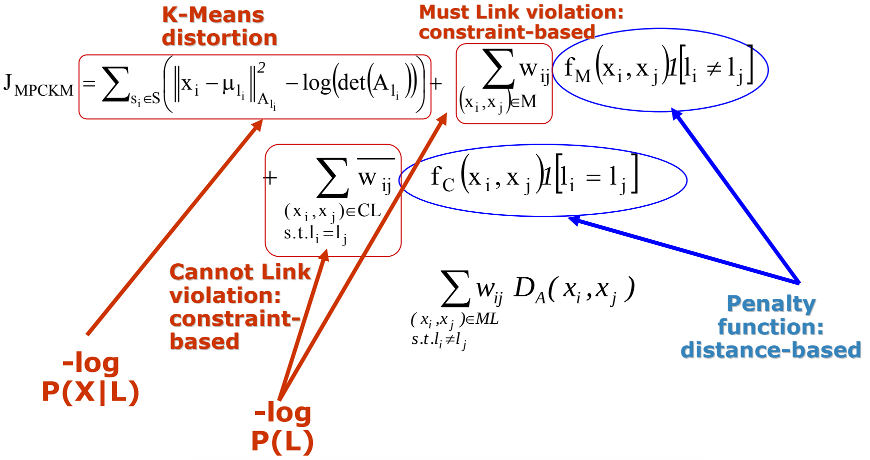


Semi-supervised clustering using local weights

- **Solution:**
  - Allow a separate weight matrix, $\mathbf{A_h}$, for each cluster **h**
  - Cluster **h** is generated by a Gaussian with covariance matrix $\mathbf{A_h^{-1}}$
- Generalized version of K-Means using different weights per cluster:

$$J_{\mathrm{mkmeans}} = \sum_{x_i \in X} \left(\left\| x_i - \mu_{l_i} \right\|^2_{A_{l_i}} - \log\left(\det\left(A_{l_i}\right)\right) \right)$$

Integrating Constraints and Metric Learning



K-Means distortion

Must Link violation: constraint-based

$$J_{MPCKM} = \sum_{s_i \in S}\left(\left\|x_i - \mu_{l_i}\right\|^2_{A_{l_i}} - \log\left(\det\left(A_{l_i}\right)\right)\right) + \sum_{(x_i,x_j)\in M} w_{ij}\, f_M(x_i,x_j)\mathbb{1}[l_i \neq l_j]$$

$$+ \sum_{\substack{(x_i,x_j)\in CL \\ s.t. l_i = l_j}} \overline{w}_{ij}\, f_C(x_i,x_j)\mathbb{1}[l_i = l_j]$$

$$\sum_{\substack{(x_i,x_j)\in ML \\ s.t. l_i \neq l_j}} w_{ij}\, D_A(x_i,x_j)$$

Cannot Link violation: constraint-based

Penalty function: distance-based

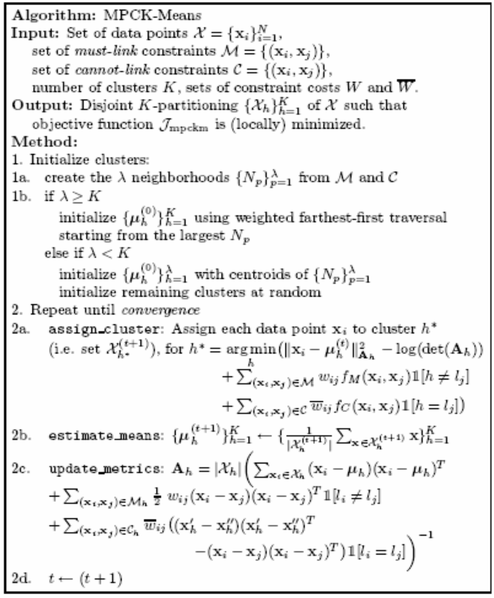
-log P(X|L)

-log P(L)

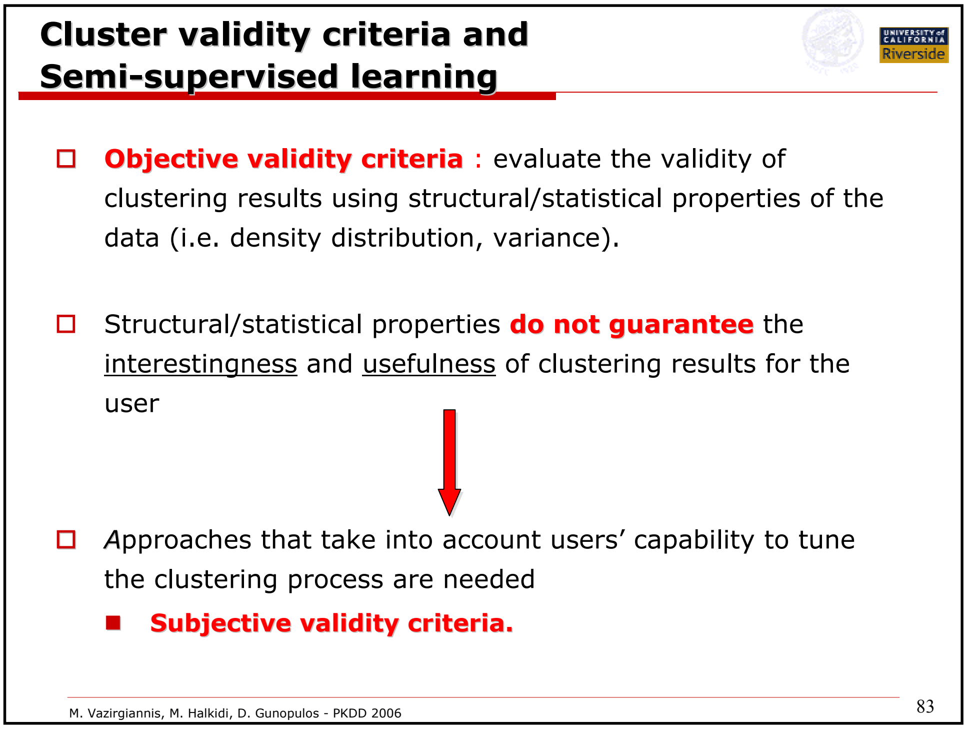
$$f_C(\mathbf{x}_i,\mathbf{x}_j) = \|\mathbf{x}'_{l_i} - \mathbf{x}''_{l_i}\|^2_{\mathbf{A}_{l_i}} - \|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{A}_{l_i}}$$

$$f_M(\mathbf{x}_i,\mathbf{x}_j) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{A}_{l_i}} + \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2_{\mathbf{A}_{l_j}}$$

MPCK-Means with local weights

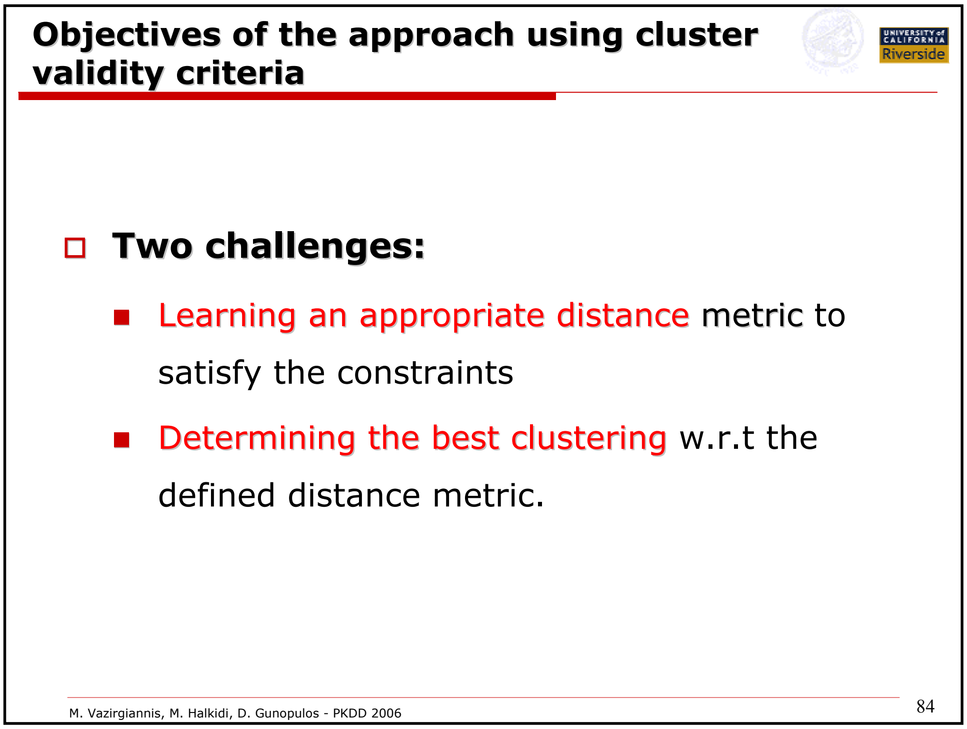


```
Algorithm: MPCK-Means
Input: Set of data points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$,
    set of must-link constraints $\mathcal{M} = \{(\mathbf{x}_i, \mathbf{x}_j)\}$,
    set of cannot-link constraints $\mathcal{C} = \{(\mathbf{x}_i, \mathbf{x}_j)\}$,
    number of clusters $K$, sets of constraint costs $W$ and $\overline{W}$.
Output: Disjoint $K$-partitioning $\{\mathcal{X}_h\}_{h=1}^K$ of $\mathcal{X}$ such that
    objective function $\mathcal{J}_{\text{mpckm}}$ is (locally) minimized.
Method:
1. Initialize clusters:
1a.  create the $\lambda$ neighborhoods $\{N_p\}_{p=1}^{\lambda}$ from $\mathcal{M}$ and $\mathcal{C}$
1b.  if $\lambda \geq K$
         initialize $\{\mu_h^{(0)}\}_{h=1}^K$ using weighted farthest-first traversal
         starting from the largest $N_p$
     else if $\lambda < K$
         initialize $\{\mu_h^{(0)}\}_{h=1}^{\lambda}$ with centroids of $\{N_p\}_{p=1}^{\lambda}$
         initialize remaining clusters at random
2. Repeat until convergence
2a.  assign_cluster: Assign each data point $\mathbf{x}_i$ to cluster $h^*$
     (i.e. set $\mathcal{X}_{h^*}^{(t+1)}$), for $h^* = \arg\min_h (\|\mathbf{x}_i - \mu_h^{(t)}\|^2_{\mathbf{A}_h} - \log(\det(\mathbf{A}_h))$
                 $+\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} w_{ij} f_M(\mathbf{x}_i,\mathbf{x}_j)\mathbb{1}[h \neq l_j]$
                 $+\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} \overline{w}_{ij} f_C(\mathbf{x}_i,\mathbf{x}_j)\mathbb{1}[h = l_j])$
2b.  estimate_means: $\{\mu_h^{(t+1)}\}_{h=1}^K \leftarrow \{\frac{1}{|\mathcal{X}_h^{(t+1)}|}\sum_{\mathbf{x}\in\mathcal{X}_h^{(t+1)}} \mathbf{x}\}_{h=1}^K$
2c.  update_metrics: $\mathbf{A}_h = |\mathcal{X}_h| \Big(\sum_{\mathbf{x}_i\in\mathcal{X}_h} (\mathbf{x}_i - \mu_h)(\mathbf{x}_i - \mu_h)^T$
     $+\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}_h} \frac{1}{2} w_{ij}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \mathbb{1}[l_i \neq l_j]$
     $+\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}_h} \overline{w}_{ij}((\mathbf{x}'_h - \mathbf{x}''_h)(\mathbf{x}'_h - \mathbf{x}''_h)^T$
                 $-(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T)\mathbb{1}[l_i = l_j]\Big)^{-1}$
2d.  $t \leftarrow (t+1)$
```



Cluster validity criteria and Semi-supervised learning

- □ **Objective validity criteria** : evaluate the validity of clustering results using structural/statistical properties of the data (i.e. density distribution, variance).
- □ Structural/statistical properties **do not guarantee** the interestingness and usefulness of clustering results for the user
- □ *A*pproaches that take into account users' capability to tune the clustering process are needed
  - ■ **Subjective validity criteria.**

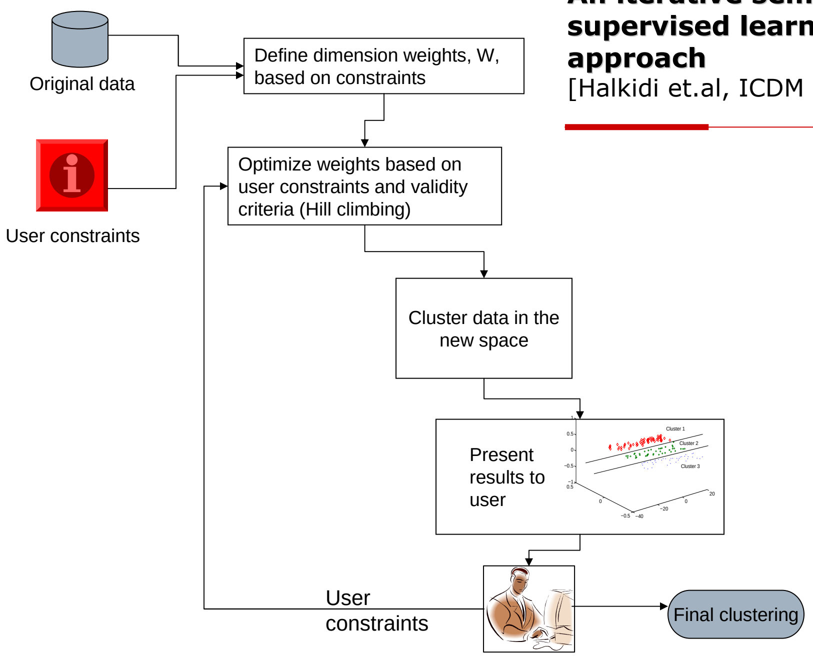


Objectives of the approach using cluster validity criteria

- □ **Two challenges:**
  - ■ Learning an appropriate distance metric to satisfy the constraints
  - ■ Determining the best clustering w.r.t the defined distance metric.

An iterative semi-supervised learning approach

[Halkidi et.al, ICDM 2005]


Original data
User constraints
Define dimension weights, W, based on constraints
Optimize weights based on user constraints and validity criteria (Hill climbing)
Cluster data in the new space
Present results to user
Cluster 1
Cluster 2
Cluster 3
User constraints
Final clustering

Initializing dimension weights based on user constraints

- Learn the distance measure to satisfy user constraints (must-link and cannot-link).
- Different weights are assigned to different dimensions
- Learn ***a diagonal*** matrix A using Newton-Raphson to efficiently optimize the following equation [Xing et al, 2002]

$$g(A) = \sum_{(x_i,x_j)\in S} \left\| x_i - x_j \right\|_A^2 - \log\left(\sum_{(x_i,x_j)\in D} \left\| x_i - x_j \right\|_A \right)$$

Best weighting of data dimensions

- □ **W:** set of different weightings defined for a set of d data dimensions.
- □ **$W_j \in W$** best weighting for a given dataset
  - ■ **if** the clustering of data in the ***d*−dimensional** space defined by

$$W_j = [w_{j1}, \ldots, w_{jd}] \ (w_{ji} > 0)$$

optimizes the quality measure:

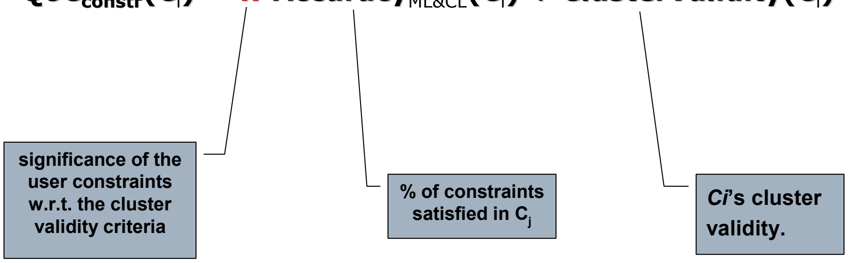
$$QoC_{constr}(C_j) = optim_{i=1,\ldots,m}\{QoC_{constr}(C_i)\}$$

given that C_j is the clustering for the W_j weighting vector.

Defining dimension weights

- □ **Clustering quality criterion (measure) :** evaluates a clustering, C_i, of a dataset in terms of
  - ■ its **accuracy w.r.t. the user constraints** (*ML* & *CL*)
  - ■ its **validity based on well-defined cluster validity criteria**.

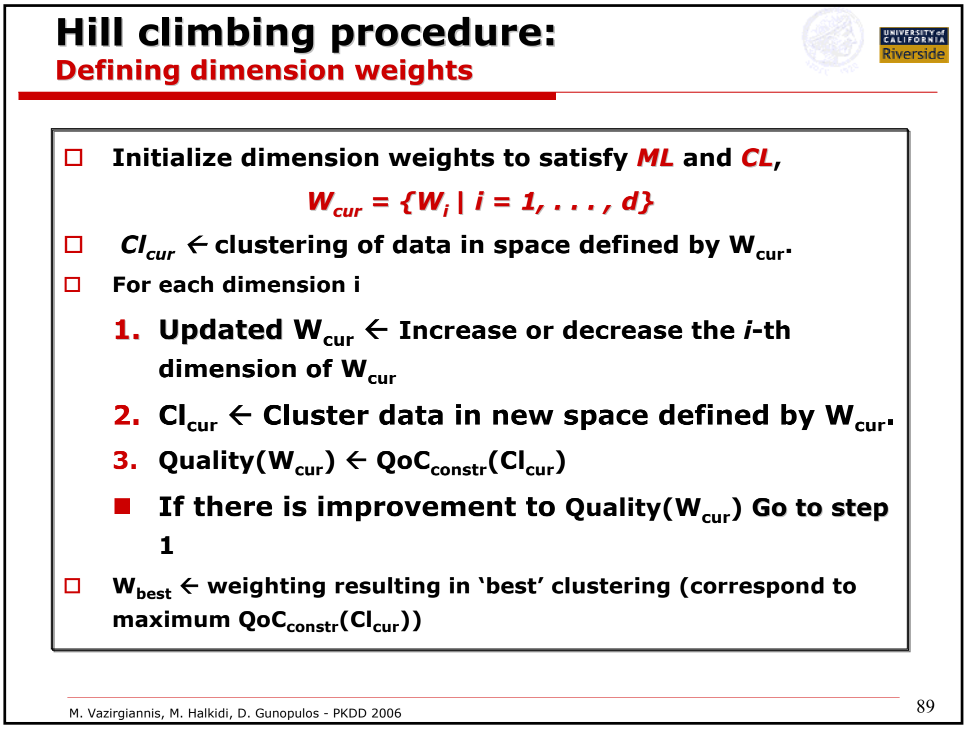
$$QoC_{constr}(C_i) = w \cdot Accuracy_{ML\&CL}(C_i) + ClusterValidity(C_i)$$



significance of the user constraints w.r.t. the cluster validity criteria

% of constraints satisfied in C_j

Ci's cluster validity.

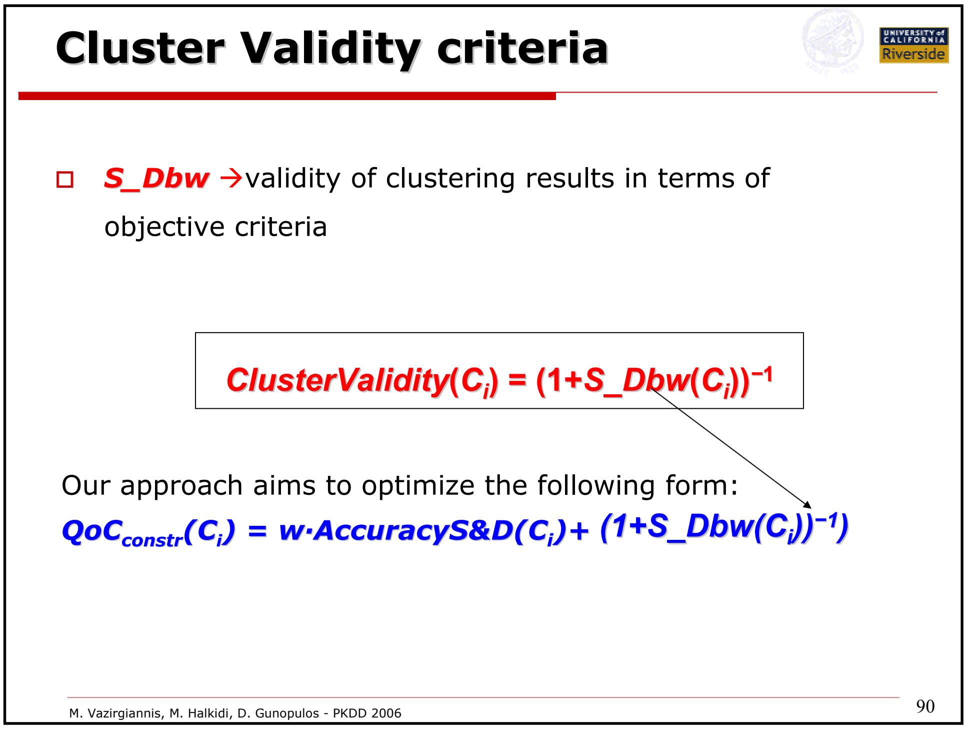


Hill climbing procedure:
Defining dimension weights

- □ Initialize dimension weights to satisfy ***ML*** and ***CL***,

  $W_{cur} = \{W_i \mid i = 1, \dots, d\}$

- □ Cl_{cur} ← clustering of data in space defined by W_{cur}.
- □ For each dimension i
  1. Updated W_{cur} ← Increase or decrease the *i*-th dimension of W_{cur}
  2. Cl_{cur} ← Cluster data in new space defined by W_{cur}.
  3. Quality(W_{cur}) ← $QoC_{constr}(Cl_{cur})$
  - ■ If there is improvement to Quality(W_{cur}) Go to step 1
- □ W_{best} ← weighting resulting in 'best' clustering (correspond to maximum $QoC_{constr}(Cl_{cur})$)



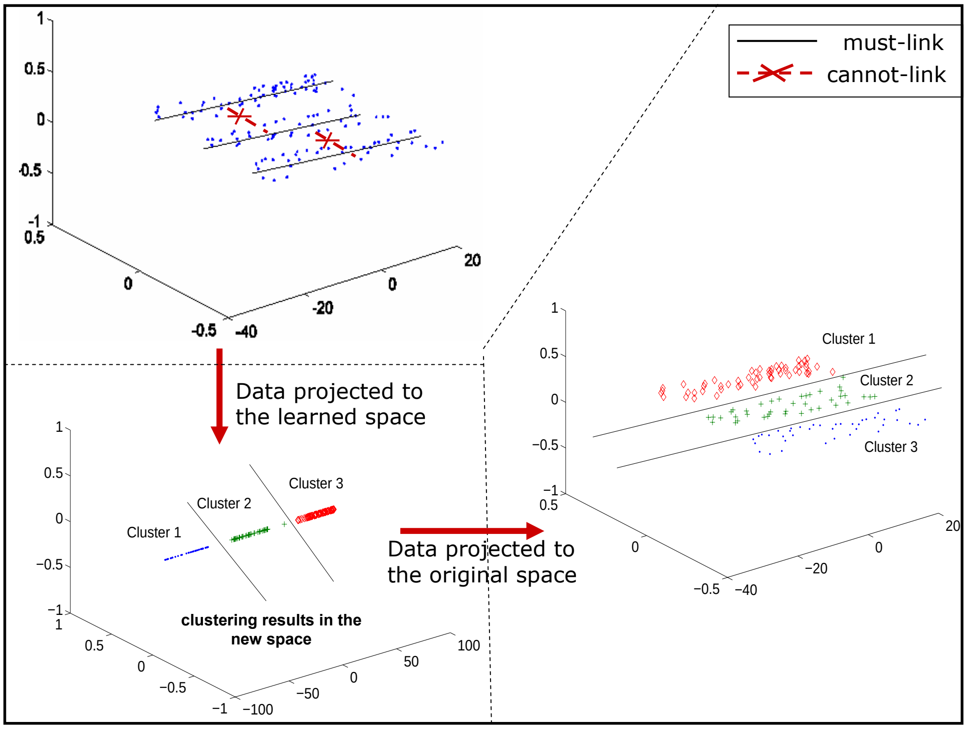
Cluster Validity criteria

- □ ***S_Dbw*** →validity of clustering results in terms of objective criteria

$$\boxed{ClusterValidity(C_i) = (1+S\_Dbw(C_i))^{-1}}$$

Our approach aims to optimize the following form:

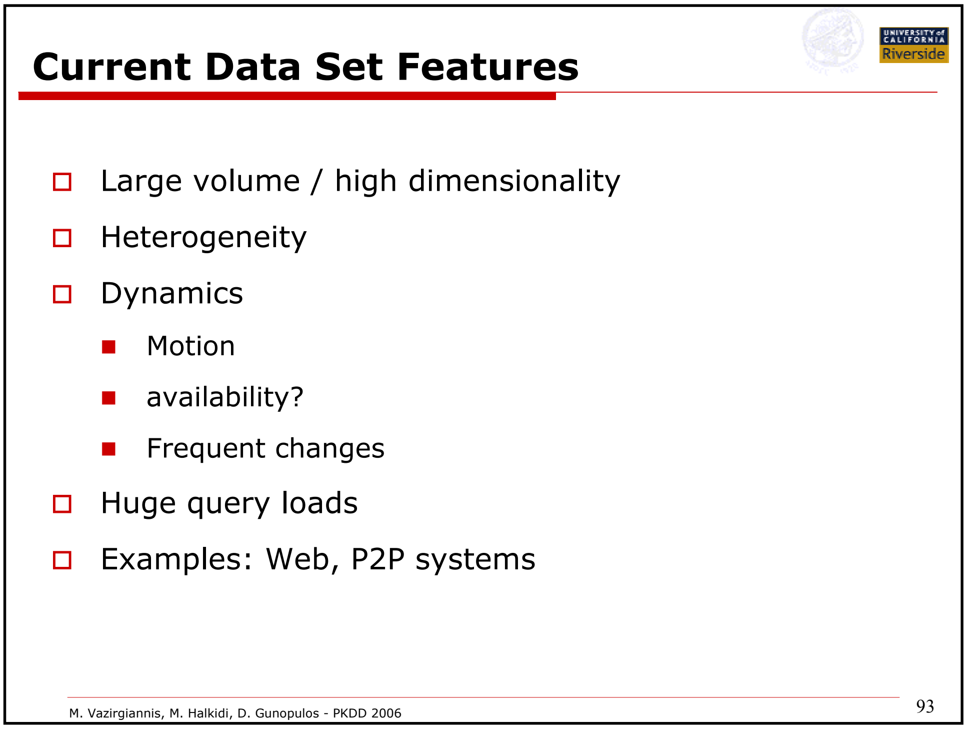
$$QoC_{constr}(C_i) = w \cdot AccuracyS\&D(C_i) + (1+S\_Dbw(C_i))^{-1}$$



Data projected to the learned space

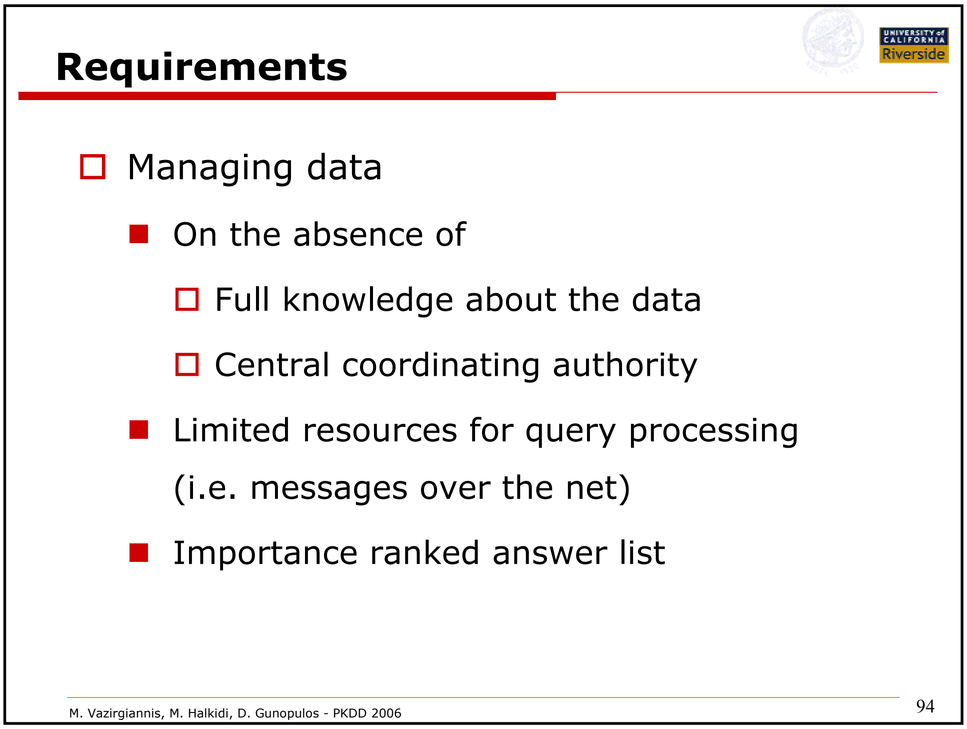
Data projected to the original space

Dimensionality Reduction for Clustering



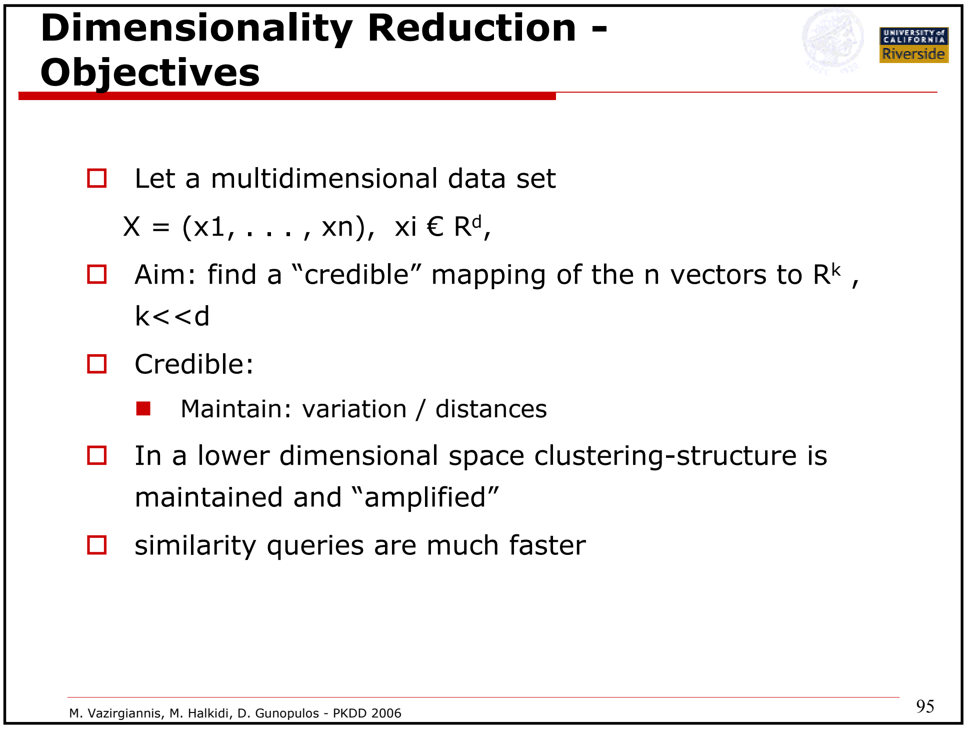
Current Data Set Features

- Large volume / high dimensionality
- Heterogeneity
- Dynamics
  - Motion
  - availability?
  - Frequent changes
- Huge query loads
- Examples: Web, P2P systems



Requirements

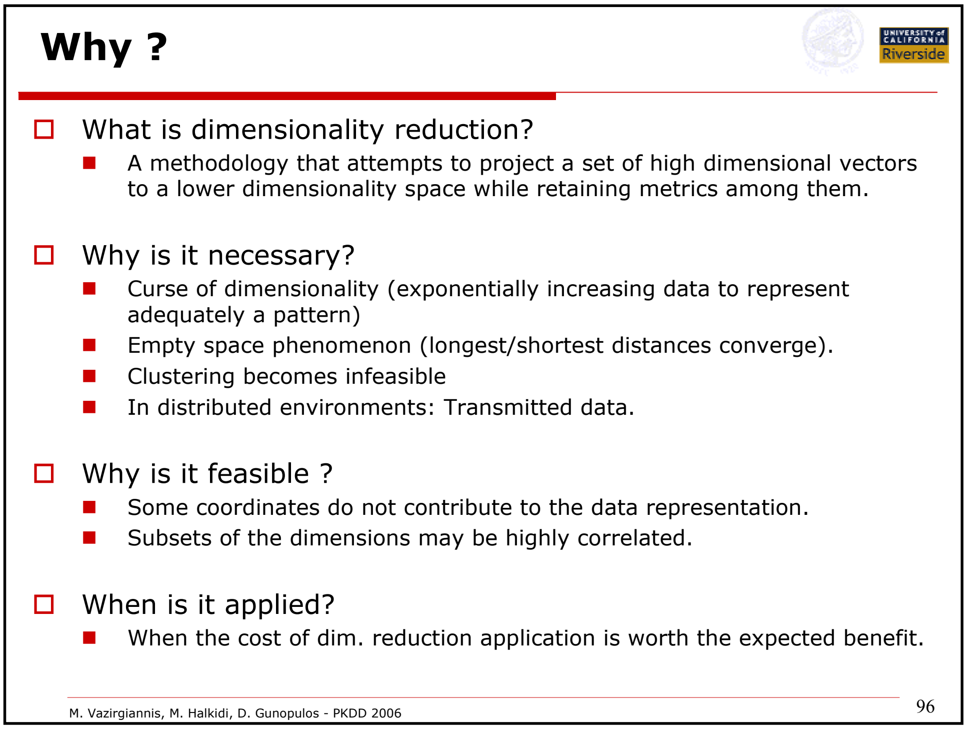
- Managing data
  - On the absence of
    - Full knowledge about the data
    - Central coordinating authority
  - Limited resources for query processing (i.e. messages over the net)
  - Importance ranked answer list



Dimensionality Reduction - Objectives

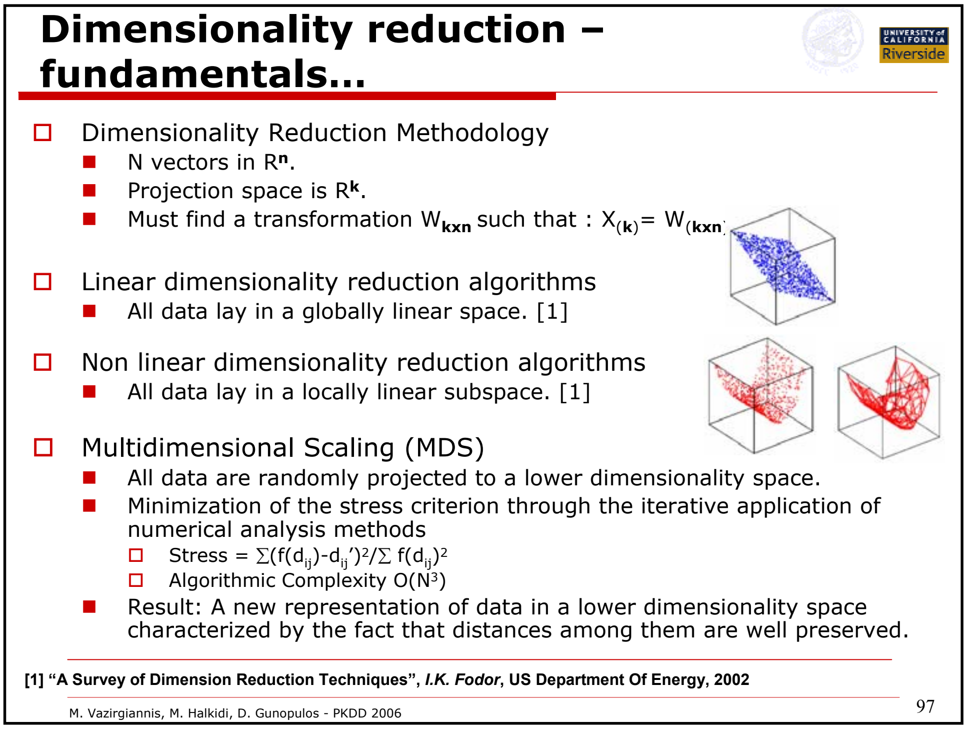
- □ Let a multidimensional data set

  X = (x1, . . . , xn), xi € R^d,
- □ Aim: find a “credible” mapping of the n vectors to R^k , k<<d
- □ Credible:
  - ■ Maintain: variation / distances
- □ In a lower dimensional space clustering-structure is maintained and “amplified”
- □ similarity queries are much faster



Why ?

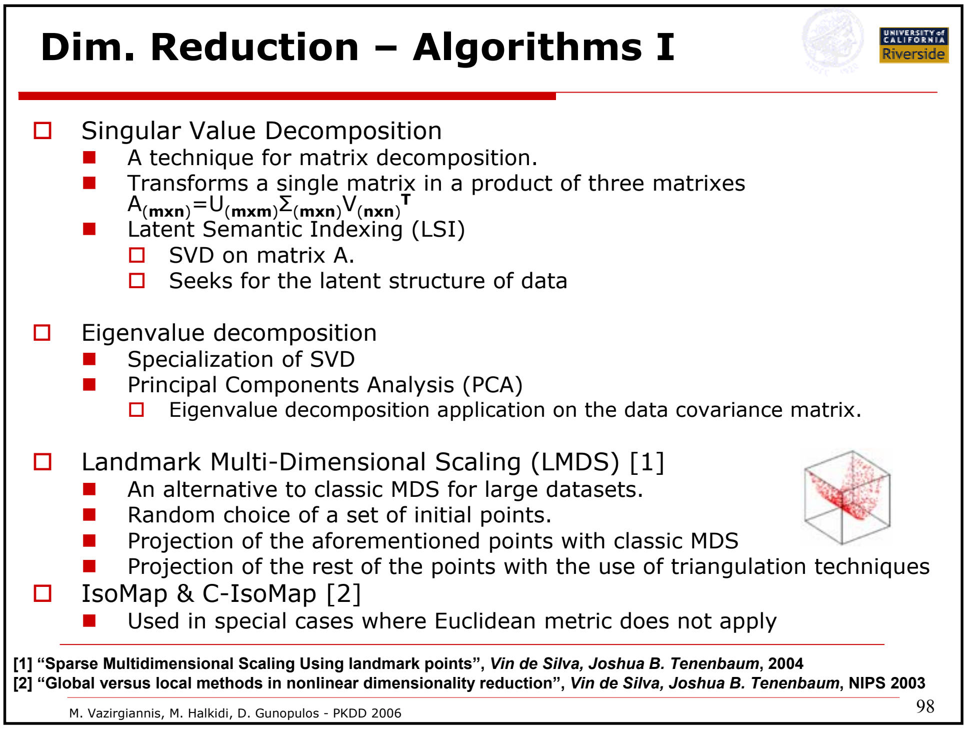
- □ What is dimensionality reduction?
  - ■ A methodology that attempts to project a set of high dimensional vectors to a lower dimensionality space while retaining metrics among them.
- □ Why is it necessary?
  - ■ Curse of dimensionality (exponentially increasing data to represent adequately a pattern)
  - ■ Empty space phenomenon (longest/shortest distances converge).
  - ■ Clustering becomes infeasible
  - ■ In distributed environments: Transmitted data.
- □ Why is it feasible ?
  - ■ Some coordinates do not contribute to the data representation.
  - ■ Subsets of the dimensions may be highly correlated.
- □ When is it applied?
  - ■ When the cost of dim. reduction application is worth the expected benefit.



Dimensionality reduction – fundamentals…

- □ Dimensionality Reduction Methodology
  - ■ N vectors in R^{n}.
  - ■ Projection space is R^{k}.
  - ■ Must find a transformation W_{kxn} such that : $X_{(k)}= W_{(kxn)}$
- □ Linear dimensionality reduction algorithms
  - ■ All data lay in a globally linear space. [1]
- □ Non linear dimensionality reduction algorithms
  - ■ All data lay in a locally linear subspace. [1]
- □ Multidimensional Scaling (MDS)
  - ■ All data are randomly projected to a lower dimensionality space.
  - ■ Minimization of the stress criterion through the iterative application of numerical analysis methods
    - □ Stress = $\Sigma(f(d_{ij})-d_{ij}')^2/\Sigma f(d_{ij})^2$
    - □ Algorithmic Complexity $O(N^3)$
  - ■ Result: A new representation of data in a lower dimensionality space characterized by the fact that distances among them are well preserved.

[1] **"A Survey of Dimension Reduction Techniques",** ***I.K. Fodor*****, US Department Of Energy, 2002**

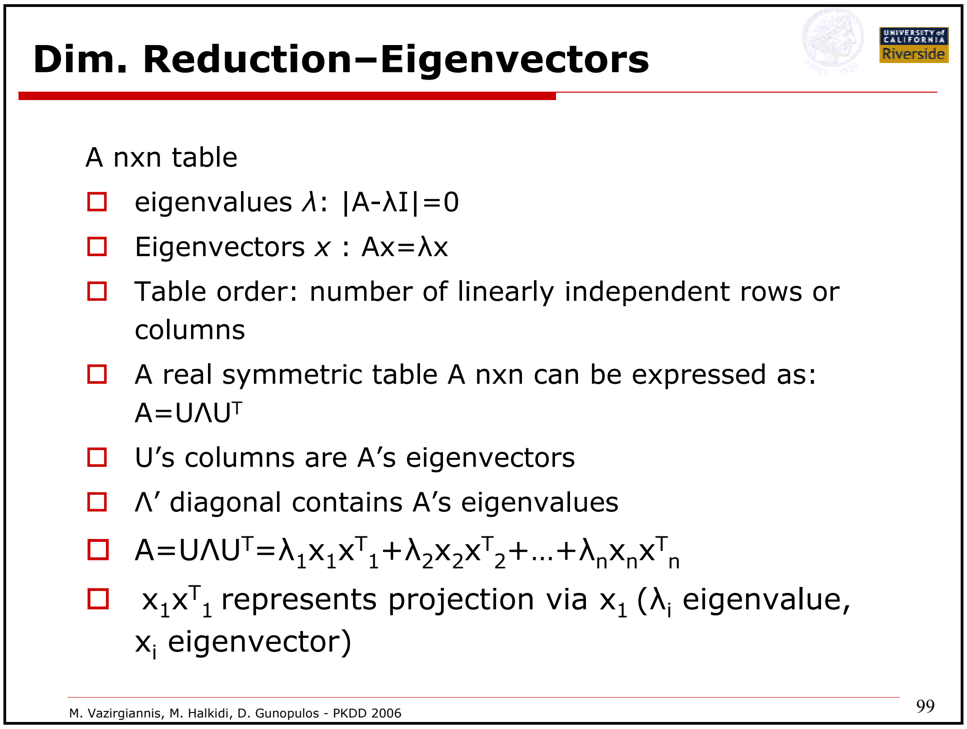


Dim. Reduction – Algorithms I

- □ Singular Value Decomposition
  - ■ A technique for matrix decomposition.
  - ■ Transforms a single matrix in a product of three matrixes $A_{(mxn)}=U_{(mxm)}\Sigma_{(mxn)}V_{(nxn)}^{T}$
  - ■ Latent Semantic Indexing (LSI)
    - □ SVD on matrix A.
    - □ Seeks for the latent structure of data
- □ Eigenvalue decomposition
  - ■ Specialization of SVD
  - ■ Principal Components Analysis (PCA)
    - □ Eigenvalue decomposition application on the data covariance matrix.
- □ Landmark Multi-Dimensional Scaling (LMDS) [1]
  - ■ An alternative to classic MDS for large datasets.
  - ■ Random choice of a set of initial points.
  - ■ Projection of the aforementioned points with classic MDS
  - ■ Projection of the rest of the points with the use of triangulation techniques
- □ IsoMap & C-IsoMap [2]
  - ■ Used in special cases where Euclidean metric does not apply

[1] **"Sparse Multidimensional Scaling Using landmark points",** ***Vin de Silva, Joshua B. Tenenbaum*****, 2004**

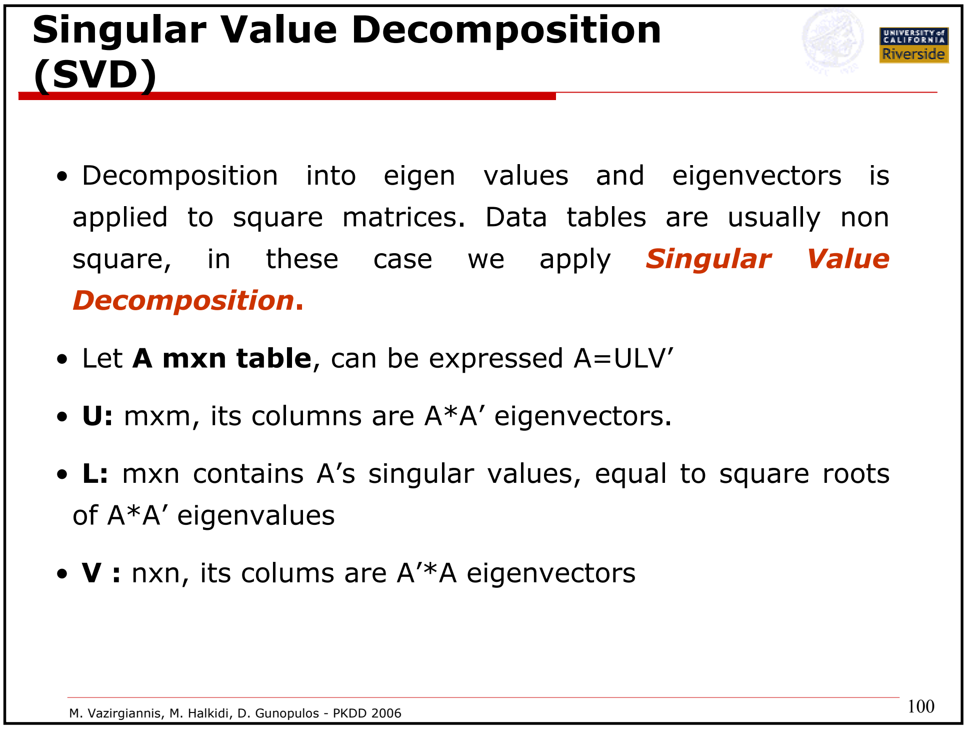
[2] **"Global versus local methods in nonlinear dimensionality reduction",** ***Vin de Silva, Joshua B. Tenenbaum*****, NIPS 2003**



Dim. Reduction–Eigenvectors

A nxn table

- □ eigenvalues λ: $|A-\lambda I|=0$
- □ Eigenvectors x : $Ax=\lambda x$
- □ Table order: number of linearly independent rows or columns
- □ A real symmetric table A nxn can be expressed as: $A=U\Lambda U^T$
- □ U's columns are A's eigenvectors
- □ Λ' diagonal contains A's eigenvalues
- □ $A=U\Lambda U^T=\lambda_1 x_1 x^T_1+\lambda_2 x_2 x^T_2+\ldots+\lambda_n x_n x^T_n$
- □ $x_1 x^T_1$ represents projection via x_1 (λ_i eigenvalue, x_i eigenvector)



Singular Value Decomposition (SVD)

- Decomposition into eigen values and eigenvectors is applied to square matrices. Data tables are usually non square, in these case we apply ***Singular Value Decomposition***.
- Let **A mxn table**, can be expressed A=ULV'
- **U:** mxm, its columns are A*A' eigenvectors.
- **L:** mxn contains A's singular values, equal to square roots of A*A' eigenvalues
- **V :** nxn, its colums are A'*A eigenvectors

Principal Components Analysis

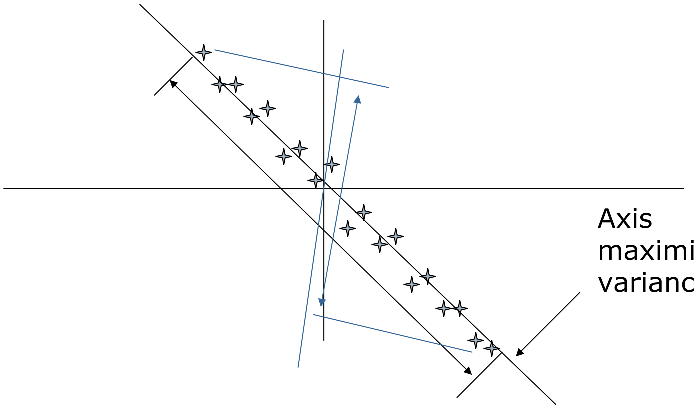
- □ The main concept behind *Principal Components Analysis* is dimensionality reduction, maintaining as much as possible data’s variance.
- □ variance: V(X)=σ^2=E[(X-μ)^2]
- □ Let *N* objects, with mean value, *m*, it is approximated as:

$$\frac{1}{N}\sum_{i=1}^{N}(x_i - m)^2,$$

- □ In a sample of *N* objects with unknown mean value:

$$\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2,$$

Dimensionality reduction based on variance maintenance



Axis maximizing variance

Principal Components Analysis

- □ Let n dimensional data, with dimensions: $x_1,\ldots,x_n$
- □ The objective is to projects the data to k dimensions via some linear decomposition:

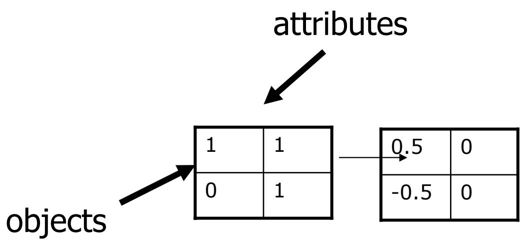
  $y_1=a_1*x_1+\ldots+a_n*x_n$

  ………

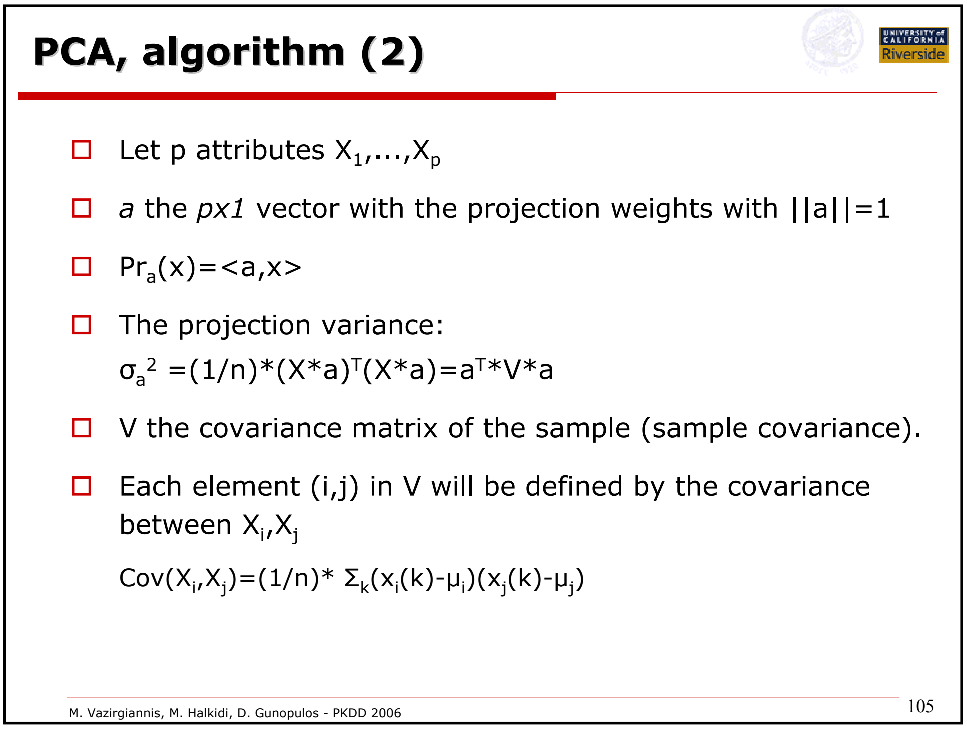
  $y_k=b_1*x_1+\ldots+b_n*x_n$
- □ the should maintain the variance of the original data

PCA, algorithm (1)

- □ X nxp data table, lines are the objects, columns the
- □ Initially data values are transformed such that μ=0



$$\begin{array}{|c|c|}\hline 1 & 1 \\ \hline 0 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline 0.5 & 0 \\ \hline -0.5 & 0 \\ \hline \end{array}$$

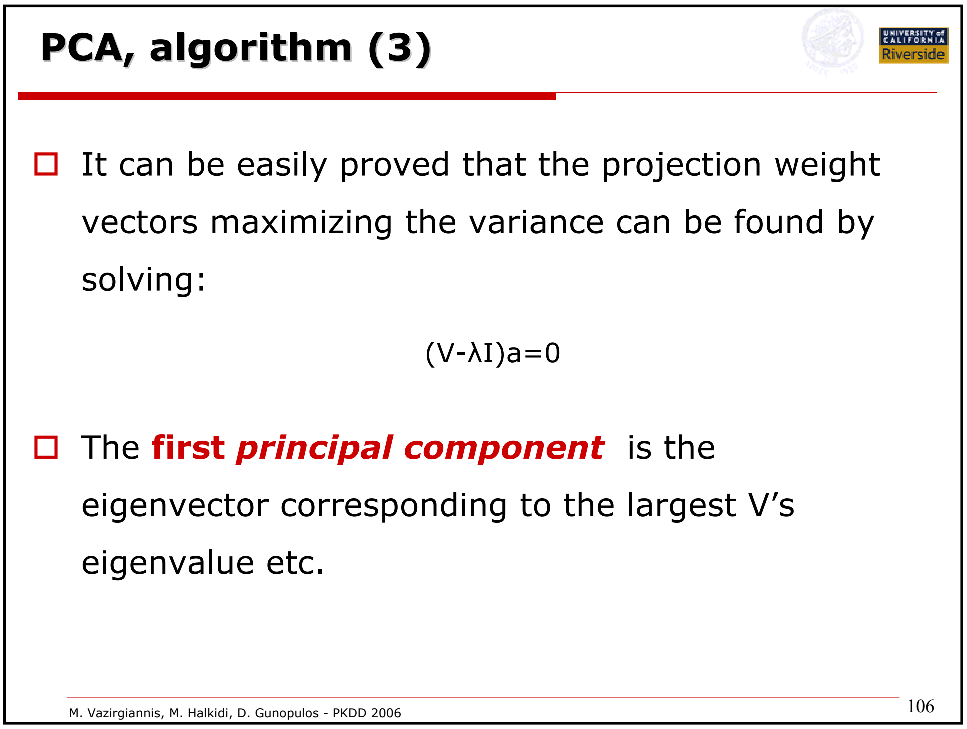


PCA, algorithm (2)

- Let p attributes $X_1,\dots,X_p$
- *a* the *px1* vector with the projection weights with ||a||=1
- $Pr_a(x)=<a,x>$
- The projection variance:

  $\sigma_a^2 = (1/n)*(X*a)^T(X*a)=a^T*V*a$
- V the covariance matrix of the sample (sample covariance).
- Each element (i,j) in V will be defined by the covariance between X_i,X_j

  $Cov(X_i,X_j)=(1/n)*\ \Sigma_k(x_i(k)-\mu_i)(x_j(k)-\mu_j)$



PCA, algorithm (3)

- It can be easily proved that the projection weight vectors maximizing the variance can be found by solving:

$$(V-\lambda I)a=0$$

- The ***first principal component*** is the eigenvector corresponding to the largest V's eigenvalue etc.

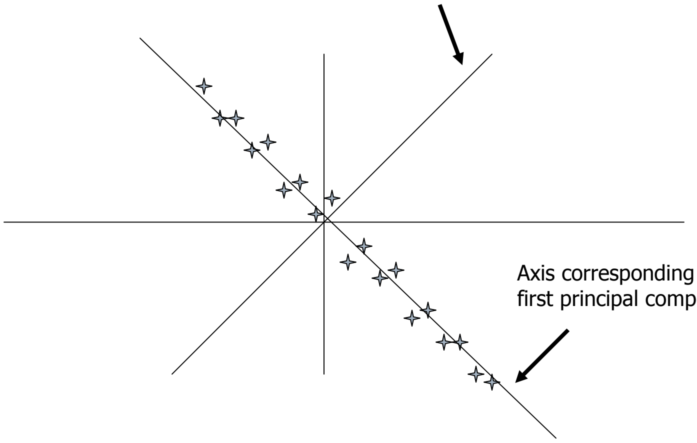
PCA, algorithm (4)

- Assuming the top k principal components, the deviation of the new variance to the original one is given by:

$$[\Sigma^{p}_{j=k+1} \lambda_j]/[\Sigma^{p}_{j=1} \lambda_j] \; [1]$$

- Termination criterion: when the deviation [1] is smaller than a threshold set.

PCA, example



Axis corresponding to the second principal component

Axis corresponding to the first principal component

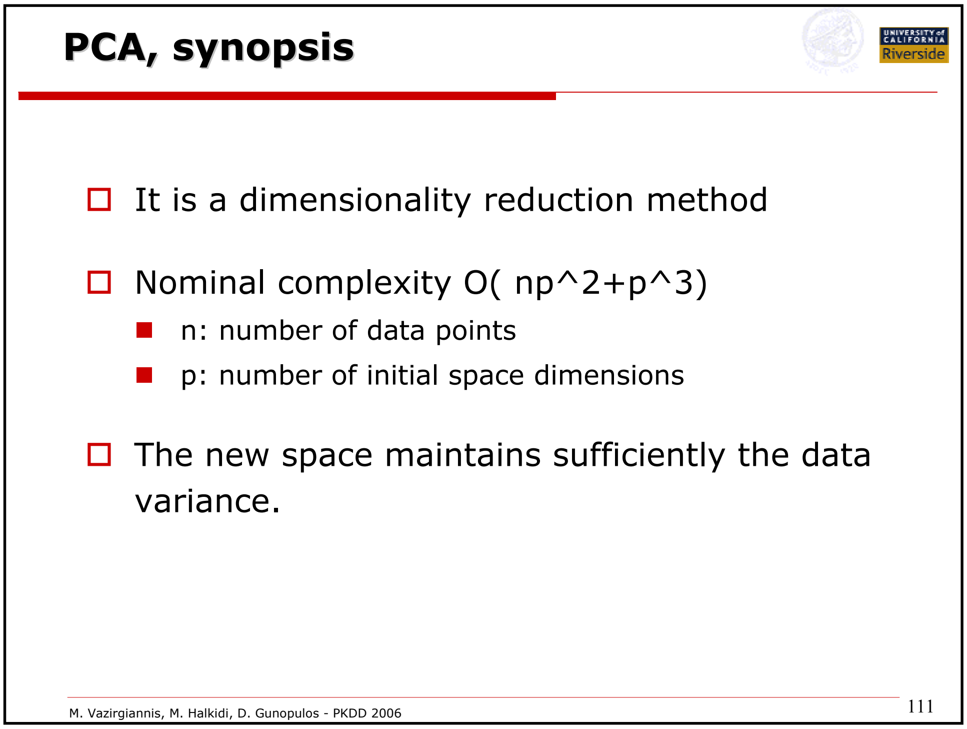
PCA Applications

- Preprocessing step preceding the application of data mining algorithms (such as clustering).
- Data Visualization.
- Noise reduction.

PCA, variations

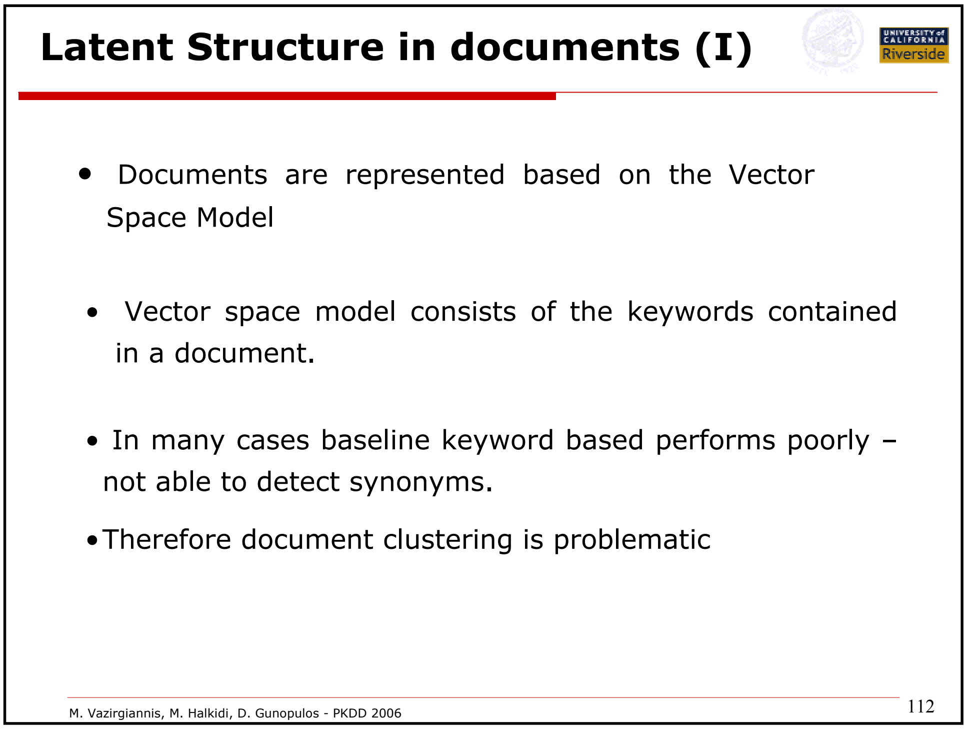
- There are variations on the definition of V generating the projection vectors.
- V may be defined as:

  $(1/n-1)* \Sigma_k(x_i(k)-\mu_i)(x_j(k)-\mu_j)$ (instead of 1/n).
- It can easily be proved that the two definitions result in exactly the same principal components.



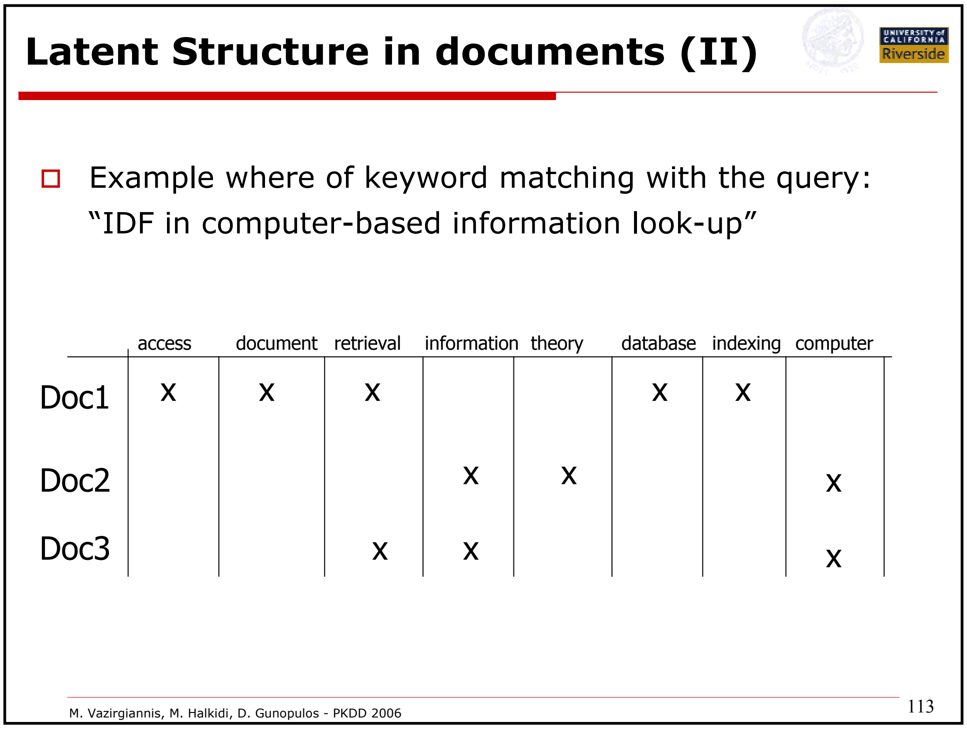
PCA, synopsis

- □ It is a dimensionality reduction method
- □ Nominal complexity O(np^2+p^3)
  - n: number of data points
  - p: number of initial space dimensions
- □ The new space maintains sufficiently the data variance.



Latent Structure in documents (I)

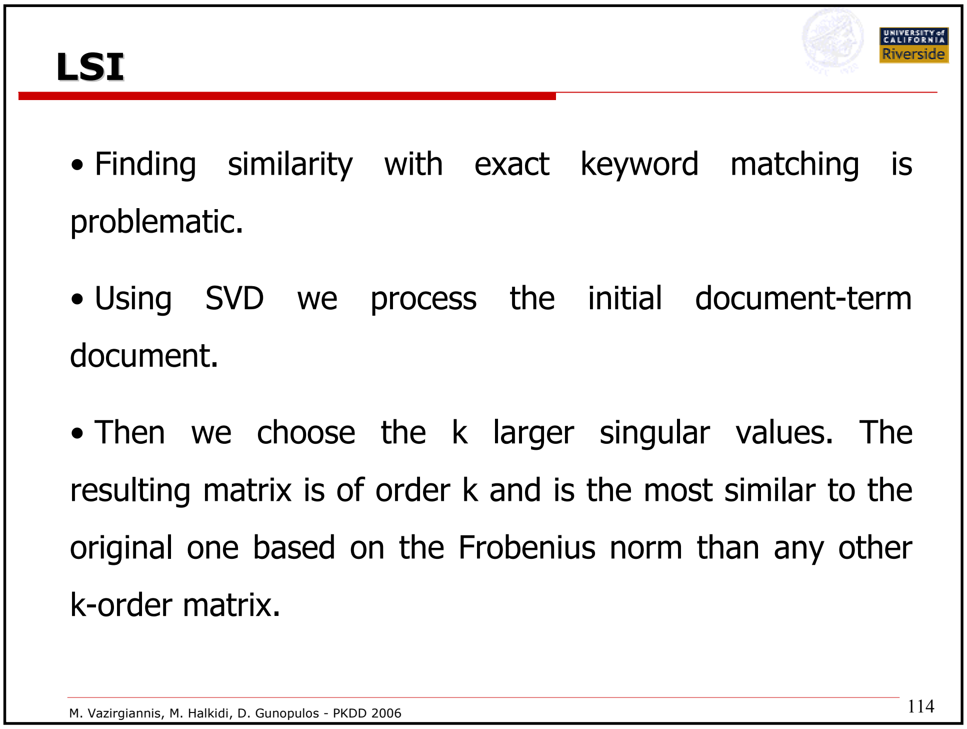
- Documents are represented based on the Vector Space Model
- Vector space model consists of the keywords contained in a document.
- In many cases baseline keyword based performs poorly – not able to detect synonyms.
- Therefore document clustering is problematic



Latent Structure in documents (II)

- Example where of keyword matching with the query: “IDF in computer-based information look-up”

	access	document	retrieval	information	theory	database	indexing	computer
Doc1	x	x	x			x	x	
Doc2				x	x			x
Doc3			x	x				x



LSI

- Finding similarity with exact keyword matching is problematic.
- Using SVD we process the initial document-term document.
- Then we choose the k larger singular values. The resulting matrix is of order k and is the most similar to the original one based on the Frobenius norm than any other k-order matrix.

LSI

- The initial matrix is analyzed as: A=ULV′
- Choosing the k larger singular values from L we have: έχουμε

$$A_k=U_kL_kV_k' \text{ ,}$$

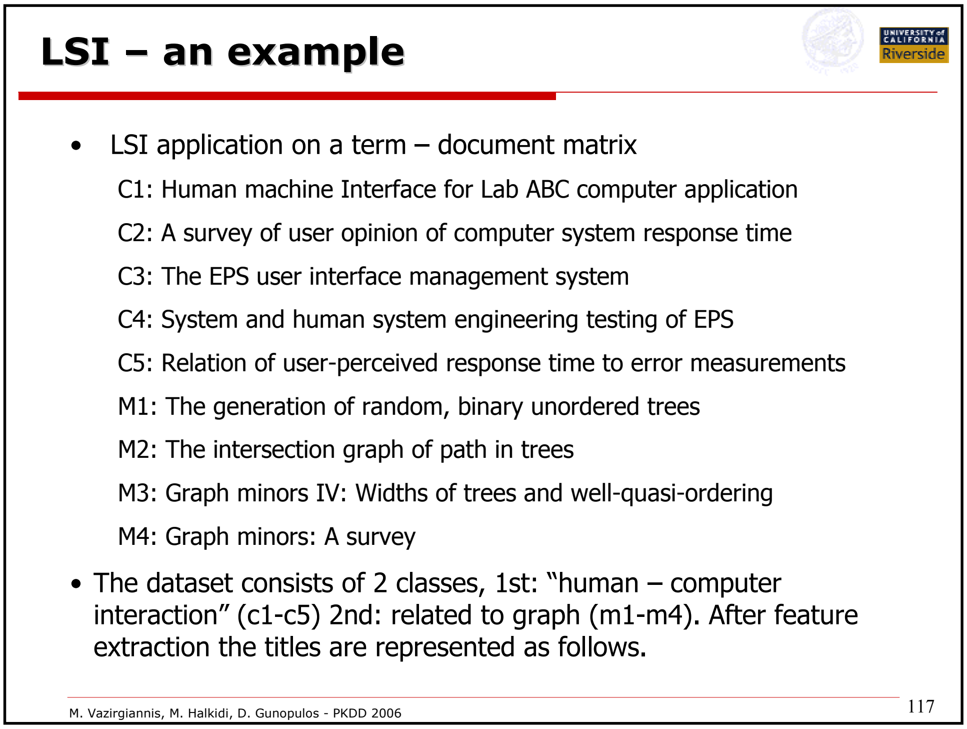
- L_k is square kxk containing the k larger eigenvalues of the diagonal in matrix L,
- $U_{k,}$ the mxk matrix containing the first k columns in U,
- $V_{k}'{}_{,}$ the kxn matrix containing the first k lines of V′

Typical values for κ~200-300 (empirically chosen based on experiments appearing in the bibliography)

LSI capabilities

- Term to term similarity:

  $A_kA\ _k'=U_kL_k^2U_k'$
- document-document similarity: $A_k'A\ _k=V_kL_k^2V_k'$
- term document similarity (as an element of the transformed – document matrix)
- Extended query capabilities transforming initial query q to q_n : $q_n=q'U_kL_k^{-1}$
- Thus q_n can be regarded a line in matrix V_k



LSI – an example

- LSI application on a term – document matrix

  C1: Human machine Interface for Lab ABC computer application

  C2: A survey of user opinion of computer system response time

  C3: The EPS user interface management system

  C4: System and human system engineering testing of EPS

  C5: Relation of user-perceived response time to error measurements

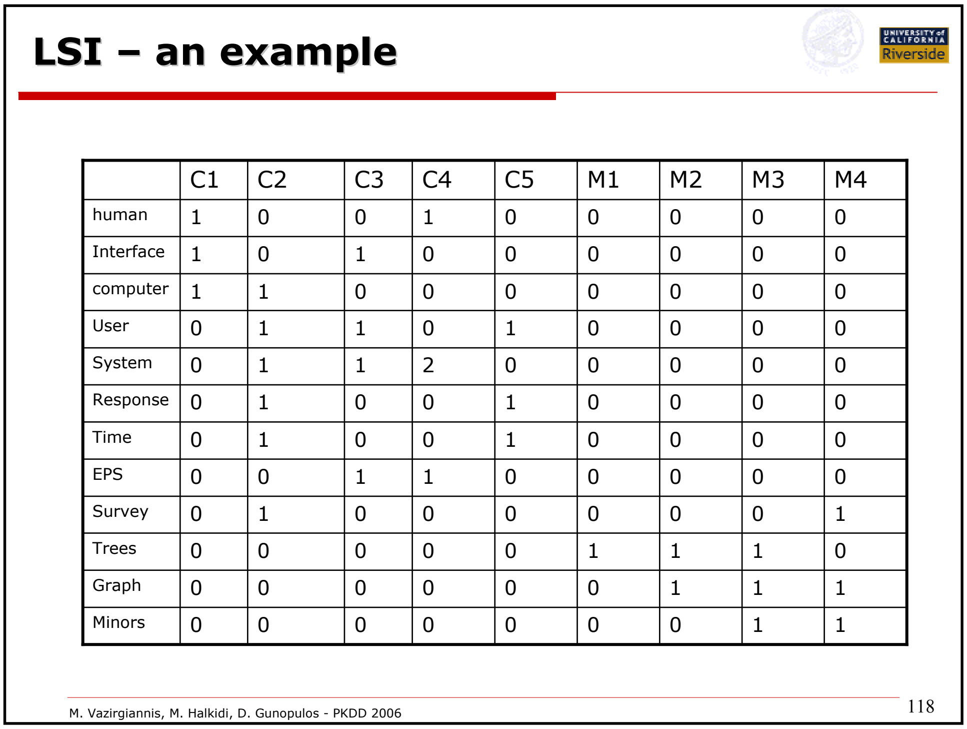
  M1: The generation of random, binary unordered trees

  M2: The intersection graph of path in trees

  M3: Graph minors IV: Widths of trees and well-quasi-ordering

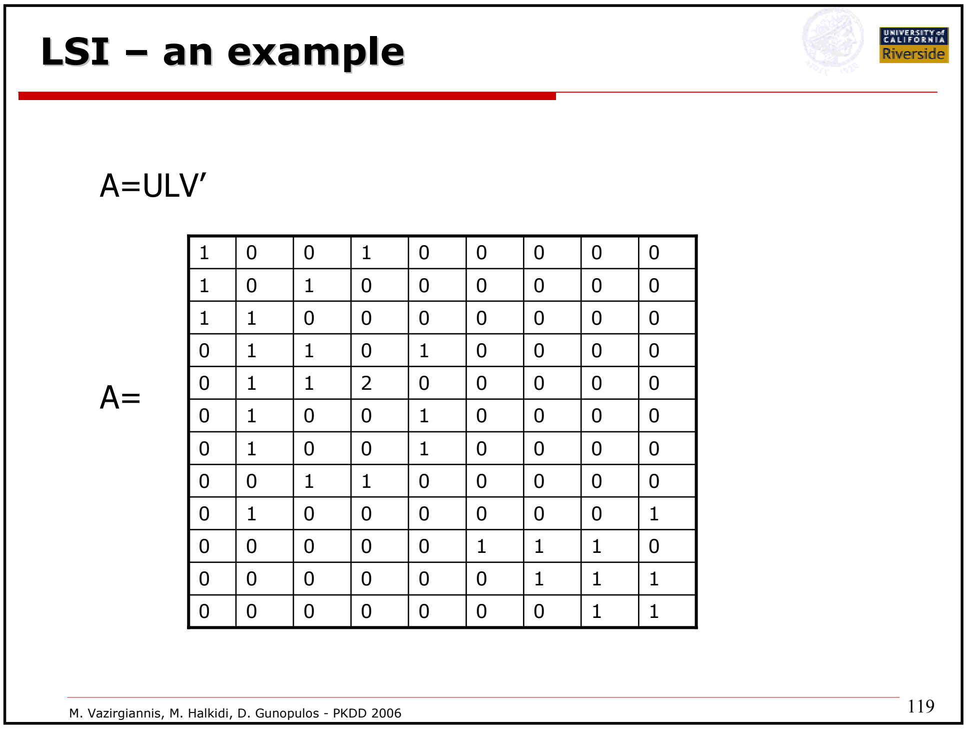
  M4: Graph minors: A survey

- The dataset consists of 2 classes, 1st: "human – computer interaction" (c1-c5) 2nd: related to graph (m1-m4). After feature extraction the titles are represented as follows.



LSI – an example

	C1	C2	C3	C4	C5	M1	M2	M3	M4
human	1	0	0	1	0	0	0	0	0
Interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	0	1	1	2	0	0	0	0	0
Response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	1
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

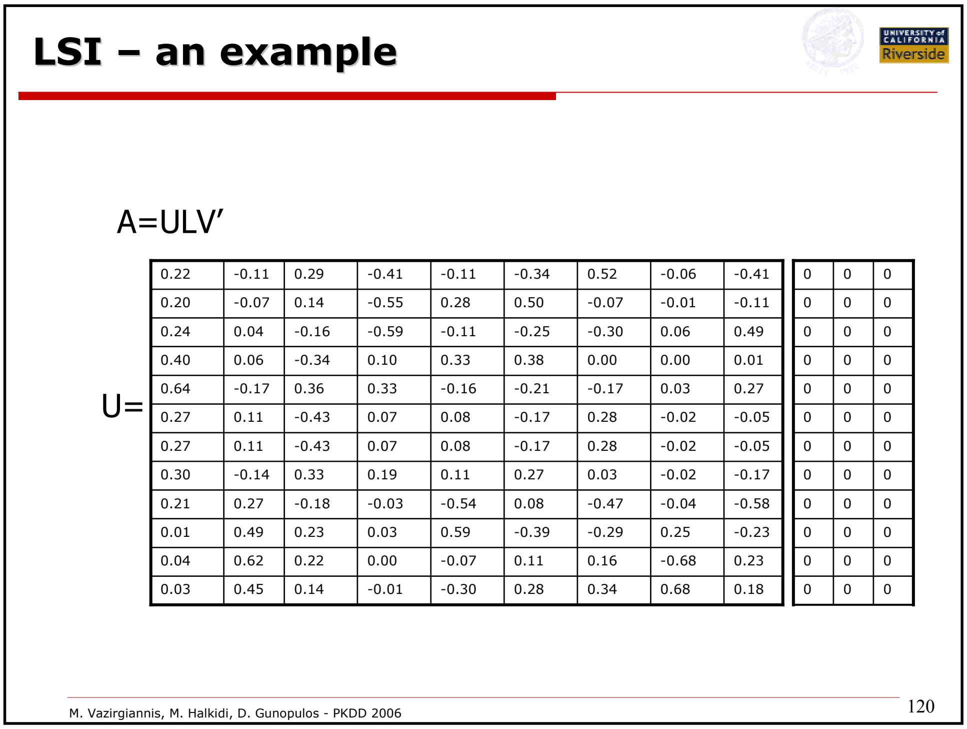


LSI – an example

A=ULV’

A=

1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0
0	1	1	2	0	0	0	0	0
0	1	0	0	1	0	0	0	0
0	1	0	0	1	0	0	0	0
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	0	0	1
0	0	0	0	0	1	1	1	0
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	1	1

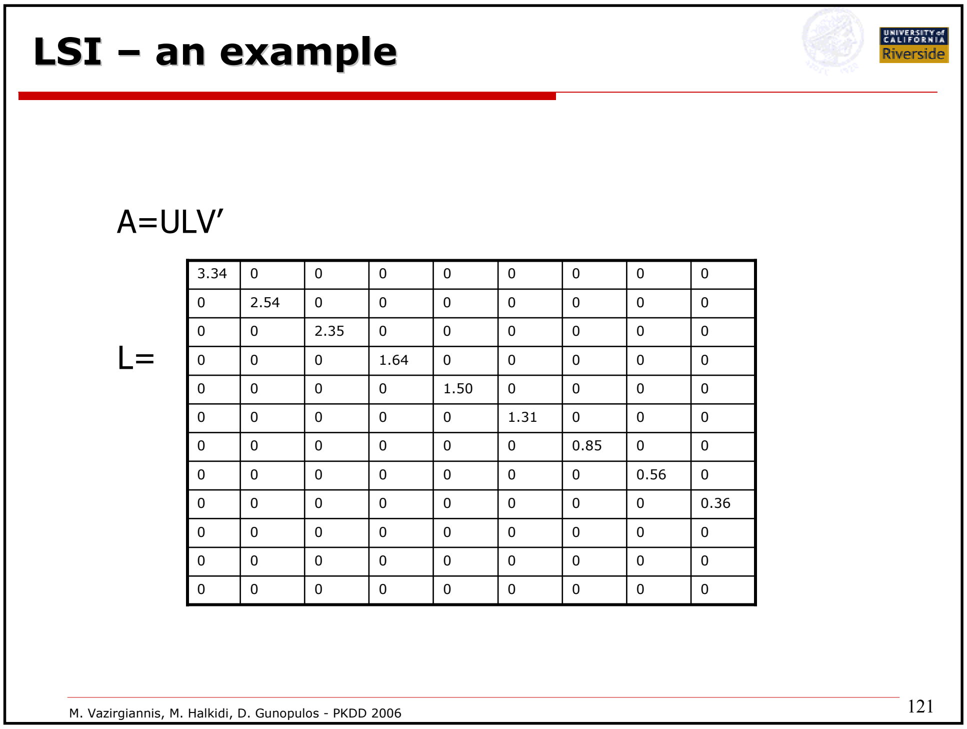


LSI – an example

A=ULV’

U=

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41	0	0	0
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11	0	0	0
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49	0	0	0
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01	0	0	0
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27	0	0	0
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05	0	0	0
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05	0	0	0
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17	0	0	0
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58	0	0	0
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23	0	0	0
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23	0	0	0
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18	0	0	0

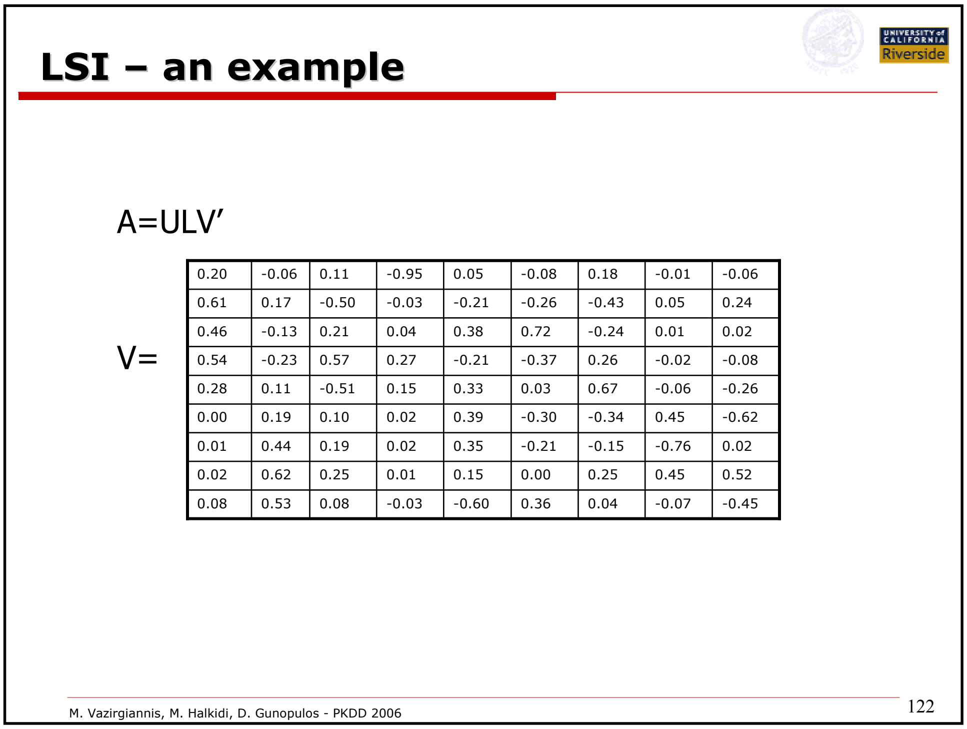


LSI – an example

A=ULV′

L=

3.34	0	0	0	0	0	0	0	0
0	2.54	0	0	0	0	0	0	0
0	0	2.35	0	0	0	0	0	0
0	0	0	1.64	0	0	0	0	0
0	0	0	0	1.50	0	0	0	0
0	0	0	0	0	1.31	0	0	0
0	0	0	0	0	0	0.85	0	0
0	0	0	0	0	0	0	0.56	0
0	0	0	0	0	0	0	0	0.36
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0



LSI – an example

A=ULV′

V=

0.20	-0.06	0.11	-0.95	0.05	-0.08	0.18	-0.01	-0.06
0.61	0.17	-0.50	-0.03	-0.21	-0.26	-0.43	0.05	0.24
0.46	-0.13	0.21	0.04	0.38	0.72	-0.24	0.01	0.02
0.54	-0.23	0.57	0.27	-0.21	-0.37	0.26	-0.02	-0.08
0.28	0.11	-0.51	0.15	0.33	0.03	0.67	-0.06	-0.26
0.00	0.19	0.10	0.02	0.39	-0.30	-0.34	0.45	-0.62
0.01	0.44	0.19	0.02	0.35	-0.21	-0.15	-0.76	0.02
0.02	0.62	0.25	0.01	0.15	0.00	0.25	0.45	0.52
0.08	0.53	0.08	-0.03	-0.60	0.36	0.04	-0.07	-0.45

LSI – an example

Choosing the 2 largest singular values we have

$U_k=$

0.22	-0.11
0.20	-0.07
0.24	0.04
0.40	0.06
0.64	-0.17
0.27	0.11
0.27	0.11
0.30	-0.14
0.21	0.27
0.01	0.49
0.04	0.62
0.03	0.45

$L_k=$

3.34	0
0	2.54

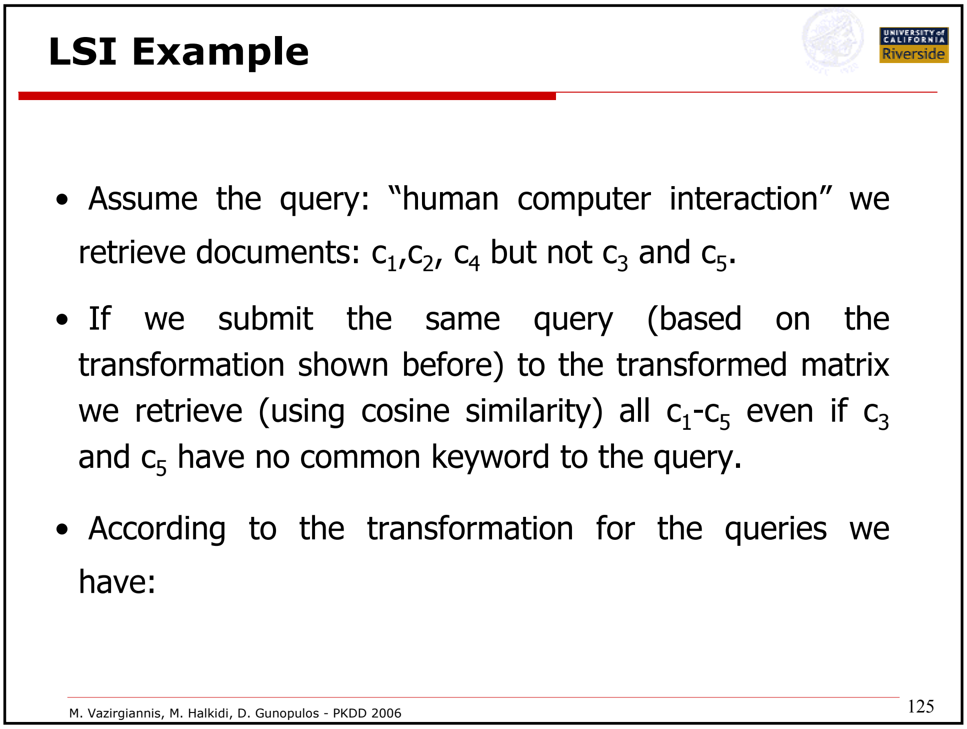
$V_k'=$

0.20	0.61	0.46	0.54	0.28	0.00	0.02	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53

LSI (2 singular values)

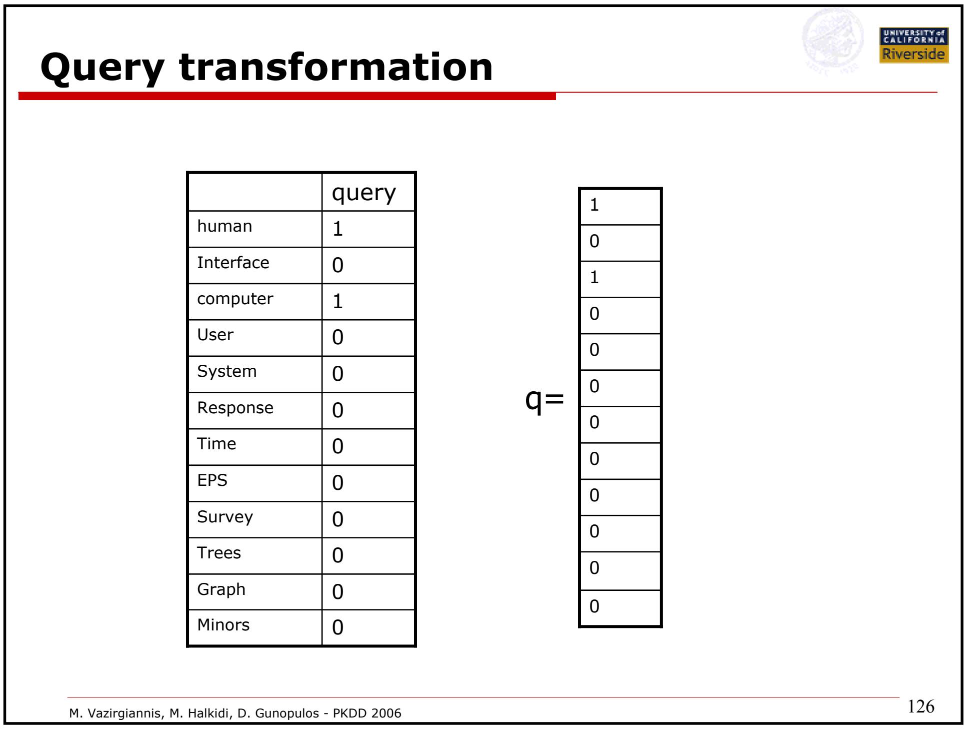
$A_k =$

	C1	C2	C3	C4	C5	M1	M2	M3	M4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
Interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
Computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
User	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
System	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
Response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
Time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
Survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
Trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
Graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
Minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62



LSI Example

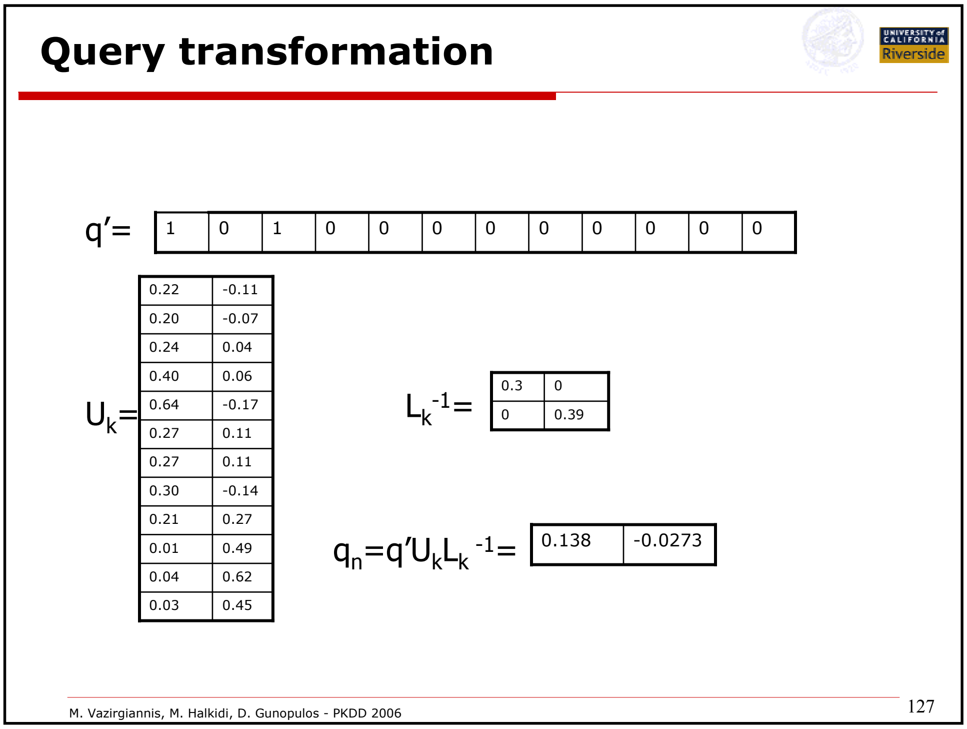
- Assume the query: “human computer interaction” we retrieve documents: c_1,c_2, c_4 but not c_3 and c_5.
- If we submit the same query (based on the transformation shown before) to the transformed matrix we retrieve (using cosine similarity) all c_1-c_5 even if c_3 and c_5 have no common keyword to the query.
- According to the transformation for the queries we have:



Query transformation

	query
human	1
Interface	0
computer	1
User	0
System	0
Response	0
Time	0
EPS	0
Survey	0
Trees	0
Graph	0
Minors	0

$$q=\begin{array}{|c|}\hline 1\\\hline 0\\\hline 1\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline 0\\\hline\end{array}$$



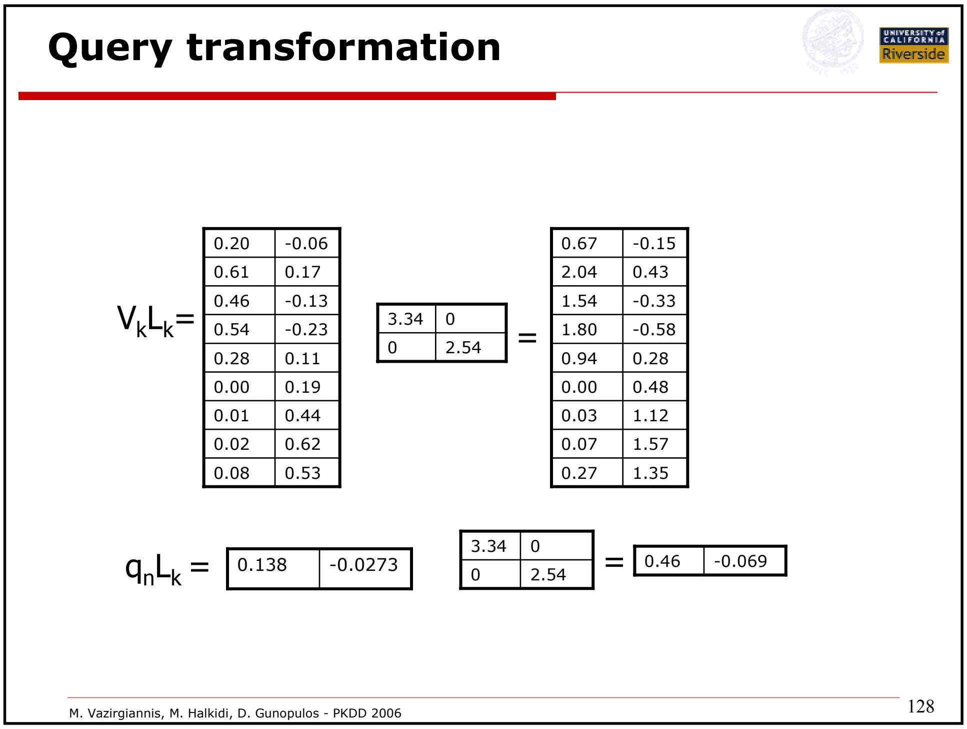
Query transformation

$$q' = \begin{bmatrix}1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$$

$$U_k = \begin{bmatrix}0.22 & -0.11\\ 0.20 & -0.07\\ 0.24 & 0.04\\ 0.40 & 0.06\\ 0.64 & -0.17\\ 0.27 & 0.11\\ 0.27 & 0.11\\ 0.30 & -0.14\\ 0.21 & 0.27\\ 0.01 & 0.49\\ 0.04 & 0.62\\ 0.03 & 0.45\end{bmatrix}$$

$$L_k^{-1} = \begin{bmatrix}0.3 & 0\\ 0 & 0.39\end{bmatrix}$$

$$q_n = q'U_kL_k^{-1} = \begin{bmatrix}0.138 & -0.0273\end{bmatrix}$$

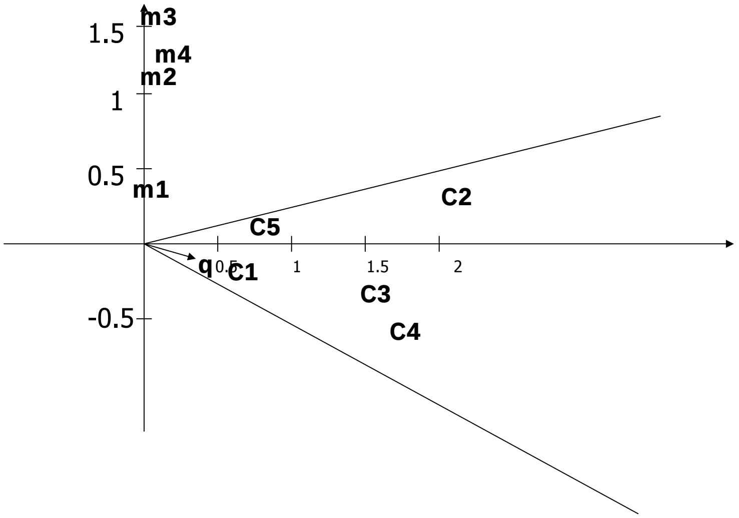


Query transformation

$$V_kL_k = \begin{bmatrix}0.20 & -0.06\\ 0.61 & 0.17\\ 0.46 & -0.13\\ 0.54 & -0.23\\ 0.28 & 0.11\\ 0.00 & 0.19\\ 0.01 & 0.44\\ 0.02 & 0.62\\ 0.08 & 0.53\end{bmatrix} \begin{bmatrix}3.34 & 0\\ 0 & 2.54\end{bmatrix} = \begin{bmatrix}0.67 & -0.15\\ 2.04 & 0.43\\ 1.54 & -0.33\\ 1.80 & -0.58\\ 0.94 & 0.28\\ 0.00 & 0.48\\ 0.03 & 1.12\\ 0.07 & 1.57\\ 0.27 & 1.35\end{bmatrix}$$

$$q_nL_k = \begin{bmatrix}0.138 & -0.0273\end{bmatrix} \begin{bmatrix}3.34 & 0\\ 0 & 2.54\end{bmatrix} = \begin{bmatrix}0.46 & -0.069\end{bmatrix}$$

Query transformation



Query transformation

- Comparison of the transformed query to the new document vectors based on cosine similarity, where the similarity is computed as: Cos(x,y)=<x,y>/||x||.||y||

Where $x=(x_1,\ldots,x_n)$, $y=(y_1,\ldots,y_n)$

$<x,y>=x_1*y_1+\ldots+x_n*y_n$

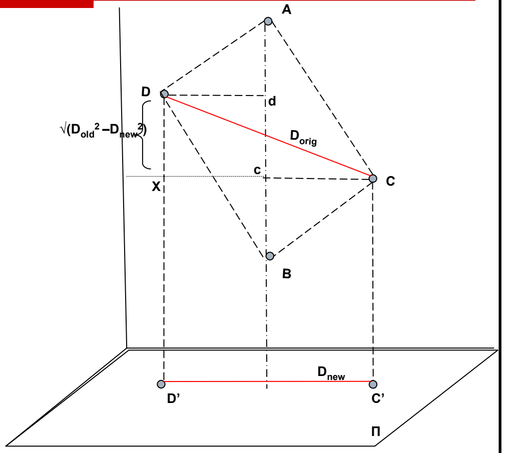
$||x||=\mathrm{sqrt}(<x,x>)$

Query transformation

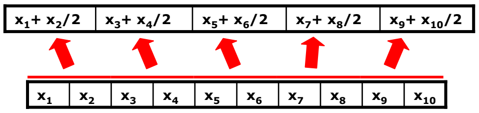
- The cosine similarity matrix of query vector to the documents is:

	query
C1	0.99
C2	0.94
C3	0.99
C4	0.99
C5	0.90
M1	-0.14
M2	-0.13
M3	-0.11
M4	0.05

Dim. Reduction – Algorithms II

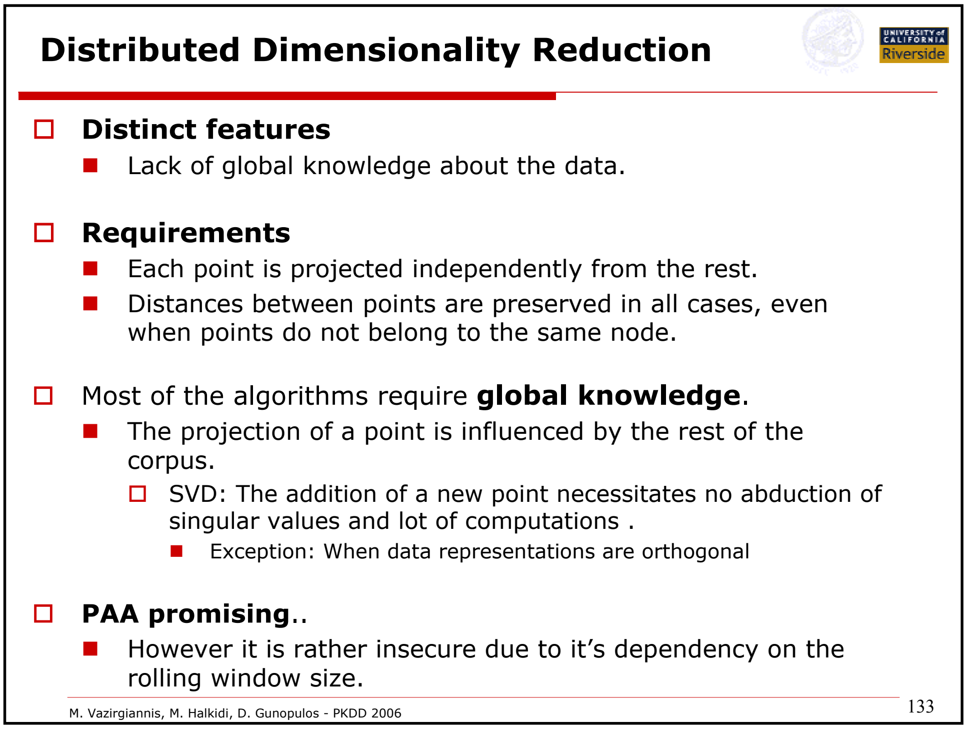


- □ FastMap (*Faloutsos et al. 1995*)
  - Projects all data to a hyperplane perpendicular to the line defined by the two most distant points of the dataset.
  - One of the fastest available methods
  - Algorithmic complexity: O(Nk)
- □ Piecewise Aggregate Approximation – PAA (*E.Keogh et al.2001)*
  - Replace a set of coordinates with their mean value.
  - Algorithmic complexity O(n)



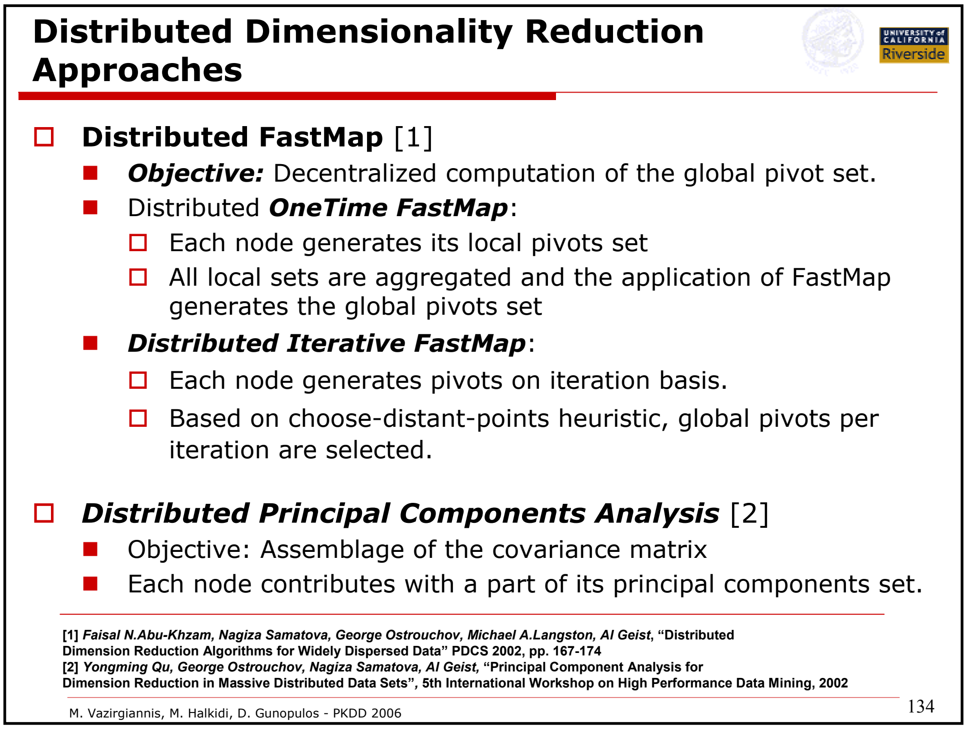
$x_1+ x_2/2$	$x_3+ x_4/2$	$x_5+ x_6/2$	$x_7+ x_8/2$	$x_9+ x_{10}/2$

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}



Distributed Dimensionality Reduction

- □ **Distinct features**
  - Lack of global knowledge about the data.
- □ **Requirements**
  - Each point is projected independently from the rest.
  - Distances between points are preserved in all cases, even when points do not belong to the same node.
- □ Most of the algorithms require **global knowledge**.
  - The projection of a point is influenced by the rest of the corpus.
    - □ SVD: The addition of a new point necessitates no abduction of singular values and lot of computations .
      - Exception: When data representations are orthogonal
- □ **PAA promising**..
  - However it is rather insecure due to it's dependency on the rolling window size.

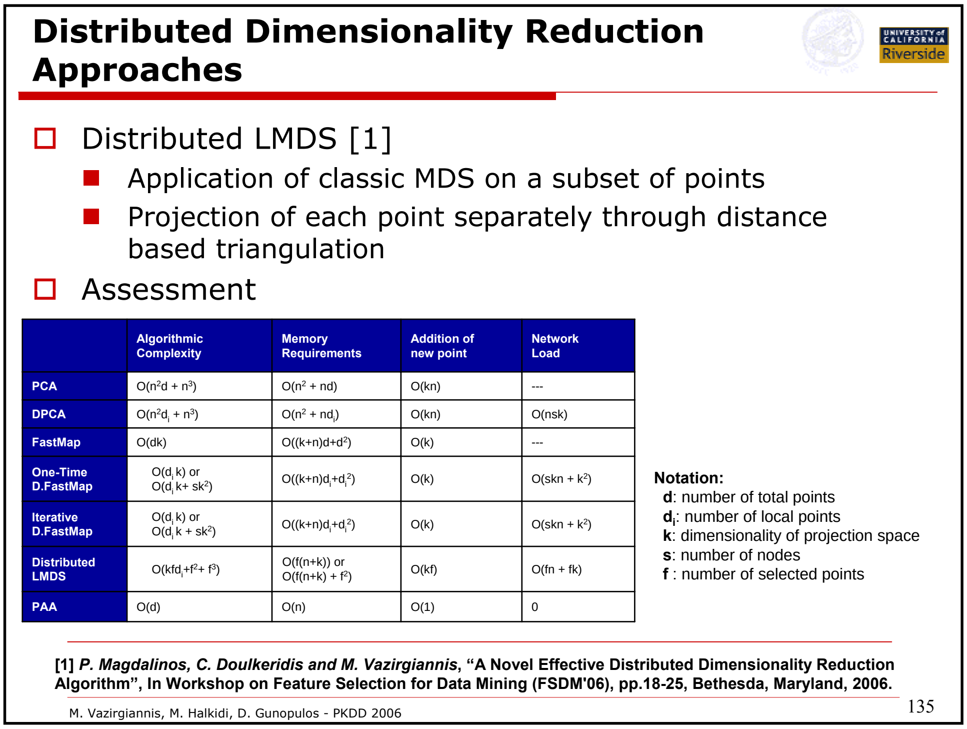


Distributed Dimensionality Reduction Approaches

- □ **Distributed FastMap** [1]
  - ***Objective:*** Decentralized computation of the global pivot set.
  - Distributed ***OneTime FastMap***:
    - □ Each node generates its local pivots set
    - □ All local sets are aggregated and the application of FastMap generates the global pivots set
  - ***Distributed Iterative FastMap***:
    - □ Each node generates pivots on iteration basis.
    - □ Based on choose-distant-points heuristic, global pivots per iteration are selected.
- □ ***Distributed Principal Components Analysis*** [2]
  - Objective: Assemblage of the covariance matrix
  - Each node contributes with a part of its principal components set.

[1] ***Faisal N.Abu-Khzam, Nagiza Samatova, George Ostrouchov, Michael A.Langston, Al Geist***, **"Distributed Dimension Reduction Algorithms for Widely Dispersed Data" PDCS 2002, pp. 167-174**

[2] ***Yongming Qu, George Ostrouchov, Nagiza Samatova, Al Geist***, **"Principal Component Analysis for Dimension Reduction in Massive Distributed Data Sets", 5th International Workshop on High Performance Data Mining, 2002**



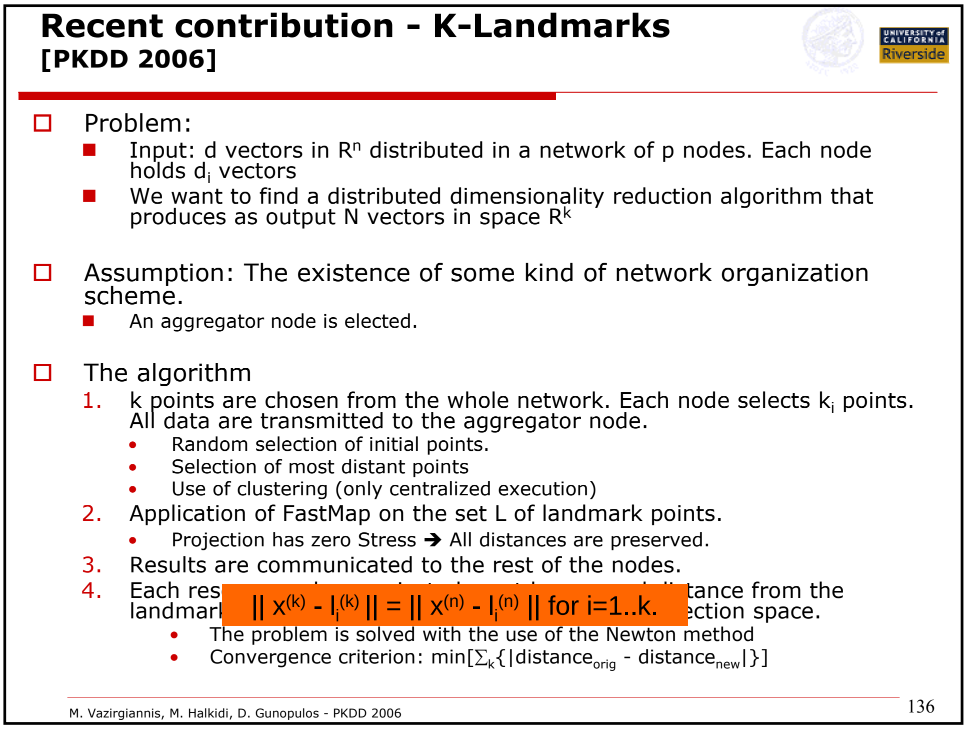
Distributed Dimensionality Reduction Approaches

- □ Distributed LMDS [1]
  - Application of classic MDS on a subset of points
  - Projection of each point separately through distance based triangulation
- □ Assessment

	Algorithmic Complexity	Memory Requirements	Addition of new point	Network Load
PCA	$O(n^2d + n^3)$	$O(n^2 + nd)$	$O(kn)$	---
DPCA	$O(n^2d_i + n^3)$	$O(n^2 + nd_i)$	$O(kn)$	$O(nsk)$
FastMap	$O(dk)$	$O((k+n)d+d^2)$	$O(k)$	---
One-Time D.FastMap	$O(d_i k)$ or $O(d_i k+ sk^2)$	$O((k+n)d_i+d_i^2)$	$O(k)$	$O(skn + k^2)$
Iterative D.FastMap	$O(d_i k)$ or $O(d_i k + sk^2)$	$O((k+n)d_i+d_i^2)$	$O(k)$	$O(skn + k^2)$
Distributed LMDS	$O(kfd_i+f^2+ f^3)$	$O(f(n+k))$ or $O(f(n+k) + f^2)$	$O(kf)$	$O(fn + fk)$
PAA	$O(d)$	$O(n)$	$O(1)$	0

Notation:
d: number of total points
d_i: number of local points
k: dimensionality of projection space
s: number of nodes
f : number of selected points

[1] ***P. Magdalinos, C. Doulkeridis and M. Vazirgiannis*, "A Novel Effective Distributed Dimensionality Reduction Algorithm", In Workshop on Feature Selection for Data Mining (FSDM'06), pp.18-25, Bethesda, Maryland, 2006.**



Recent contribution - K-Landmarks [PKDD 2006]

- □ Problem:
  - Input: d vectors in R^n distributed in a network of p nodes. Each node holds d_i vectors
  - We want to find a distributed dimensionality reduction algorithm that produces as output N vectors in space R^k
- □ Assumption: The existence of some kind of network organization scheme.
  - An aggregator node is elected.
- □ The algorithm
  1. k points are chosen from the whole network. Each node selects k_i points. All data are transmitted to the aggregator node.
     - Random selection of initial points.
     - Selection of most distant points
     - Use of clustering (only centralized execution)
  2. Application of FastMap on the set L of landmark points.
     - Projection has zero Stress ➔ All distances are preserved.
  3. Results are communicated to the rest of the nodes.
  4. Each res tance from the landmar ection space.

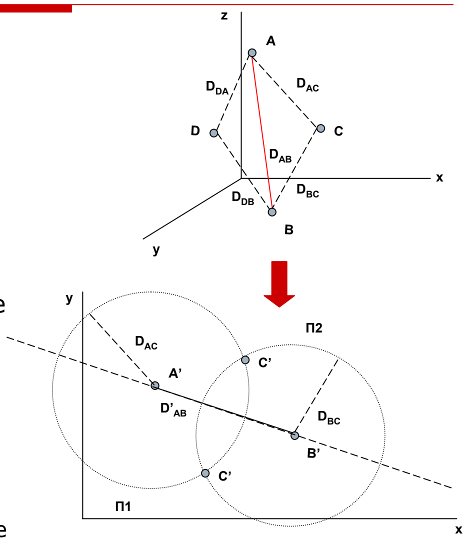
     $|| x^{(k)} - l_i^{(k)} || = || x^{(n)} - l_i^{(n)} ||$ for i=1..k.
     - The problem is solved with the use of the Newton method
     - Convergence criterion: $\min[\Sigma_k\{|distance_{orig} - distance_{new}|\}]$

K-Landmarks - I

- □ Geometric interpretation
  - Equation $||x^{(k)} - l^{(k)}|| = D = ||x^{(n)} - l^{(n)}||$ represents a hypersphere in R^k with center in $l^{(k)}$ radius D.
  - The algorithm searches for the common trace of the k hyperspheres.
- □ The algorithm always converges if the Euclidean metric holds true in the original space.
  - Criterion of non convergence:
    - □ $||A'B'^{\rightarrow}||>||CA^{\rightarrow}||+||CB^{\rightarrow}||$

      or

      $||A'B'^{\rightarrow}||<||CA^{\rightarrow}||-||CB^{\rightarrow}||$
  - Projection with zero stress:
    - □ $||A'B'^{\rightarrow}||=||AB^{\rightarrow}||$
  - The criterion of non-convergence in never satisfied



K-Landmarks II

- □ **Computational Cost:**
  - Choice of k_i points from each node:
    - □ Random: $O(k_i)$
    - □ Heuristic based : $O(d_ik_i)$
  - Distances' calculation between landmark points:
    - □ $O(k^2)$ – cost for the aggregator node only.
  - FastMap execution:
    - □ $O(k^2)$ - cost for the aggregator node only.
  - Calculation of the distances of the remaining d_i -k_i points from the landmark points:
    - □ $O\{(d_i -k_i)k\}$
  - Solution of $(d_i -k_i)$ non-linear equations system:
    - □ $O\{(d_i -k_i)k^3/3\}$
  - Eventually:
    - □ $O\{(d_i -k_i)k^3/3\}$ for each node
- □ **Network stress:**
  - Communication of k vectors of dimensionality n: O(nk)
  - Communication of the aforementioned vectors and their projections in the k dimensions space: $O(nk + k^2)$
  - Eventually:$O(nk + k^2)$

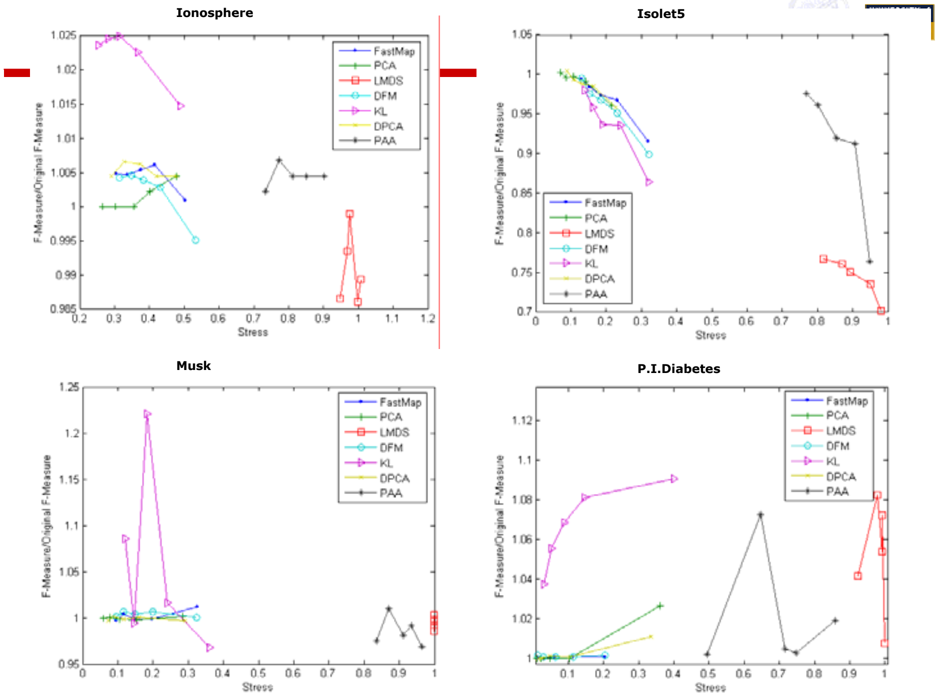
	Algorithmic Complexity	Memory Requirements	Addition of new point	Network Load
K-Landmarks	$O((d_i-k_i)\ k^3/3)$	$O(kn+k^2)$	$O(k^3/3)$	$O(nk + k^2)$

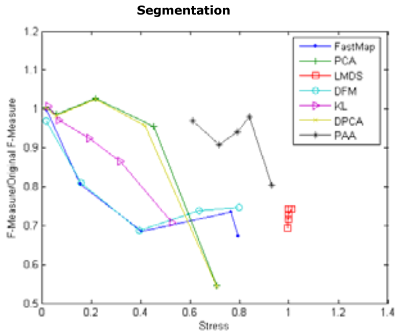
Experiments

- Experiments with a selection from UCI datasets
  - Projection from 2% up to 10% of original dimensions
- We measure:
  - Stress: distance preservation while projecting
  - Relative clustering quality preservation: discovering clusters before vs. after projecting
    - F-Measure-k /F-Measure-n
- Datasets:

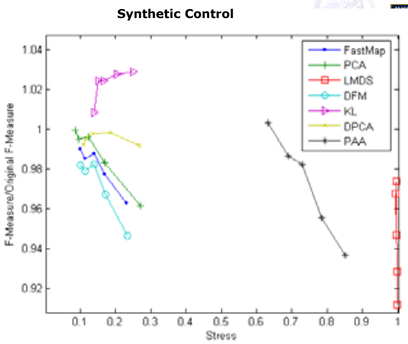
Dataset Name	Objects	Dimensions	Classes	Description
Ionosphere	351	34	2	Radar observations.
Isolet5	1559	617	26	Letters of the alphabet.
Musk	476	166	2	Molecules descriptions.
P.I.Diabetes	768	8	2	Medical observations.
Segmentation	2000	19	7	Outdoor images segments.
Synthetic control	600	60	6	Randomly generated data ([1]).

[1] ***Alcock R.J. and Manolopoulos Y***, **"Time-Series Similarity Queries Employing a Feature-Based Approach", 7th Hellenic Conference on Informatics. August 27-29. Ioannina,Greece 1999.**





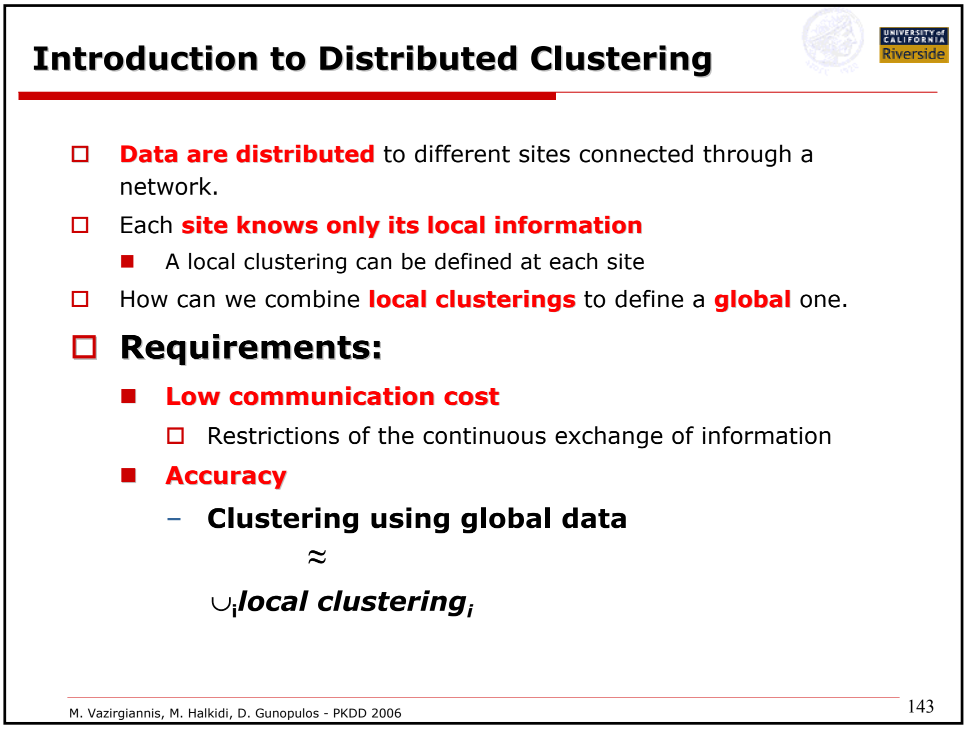
Segmentation



Synthetic Control

- Remarks
  - KLandmarks exhibits high clustering quality preservation and low stress value
  - Distributed LMDS proves rather unstable, with extremely high stress value
  - Although PAA retains clusters it fails in distances preservation

Distributed Clustering approaches

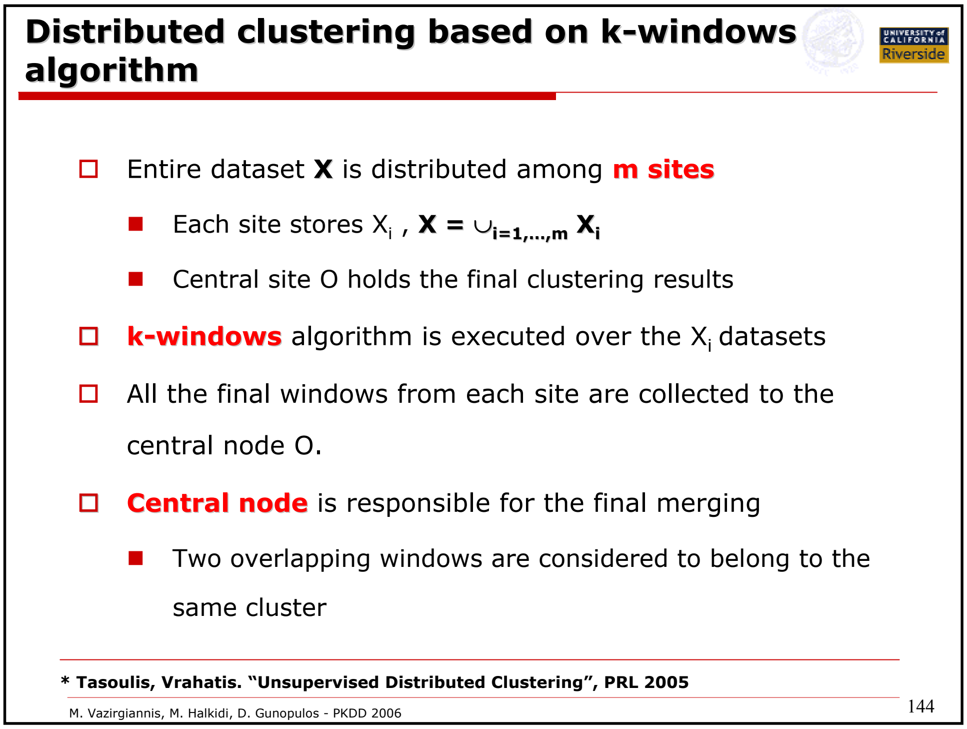


Introduction to Distributed Clustering

- □ **Data are distributed** to different sites connected through a network.
- □ Each **site knows only its local information**
  - ■ A local clustering can be defined at each site
- □ How can we combine **local clusterings** to define a **global** one.
- □ **Requirements:**
  - ■ **Low communication cost**
    - □ Restrictions of the continuous exchange of information
  - ■ **Accuracy**
    - – **Clustering using global data**

$$\approx$$

$$\cup_i \textbf{\textit{local clustering}}_i$$



Distributed clustering based on k-windows algorithm

- □ Entire dataset **X** is distributed among **m sites**
  - ■ Each site stores X_i , $\mathbf{X} = \cup_{i=1,\dots,m} \mathbf{X_i}$
  - ■ Central site O holds the final clustering results
- □ **k-windows** algorithm is executed over the X_i datasets
- □ All the final windows from each site are collected to the central node O.
- □ **Central node** is responsible for the final merging
  - ■ Two overlapping windows are considered to belong to the same cluster

*** Tasoulis, Vrahatis. "Unsupervised Distributed Clustering", PRL 2005**

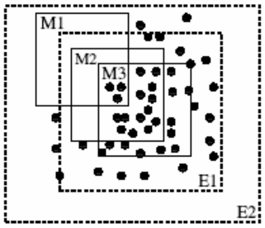
Unsupervised k-windows algorithm

- □ Tries to place a ***d***-dimensional window containing all patterns that belong to a single cluster.
- □ Two step approach based on
  - sequential **movement** and **enlargements** of windows
- □ Windows aim to capture patterns that belong to the same cluster
- □ After the **clustering procedure**
  - windows that share a sufficiently large number of patterns are merged to form a single cluster

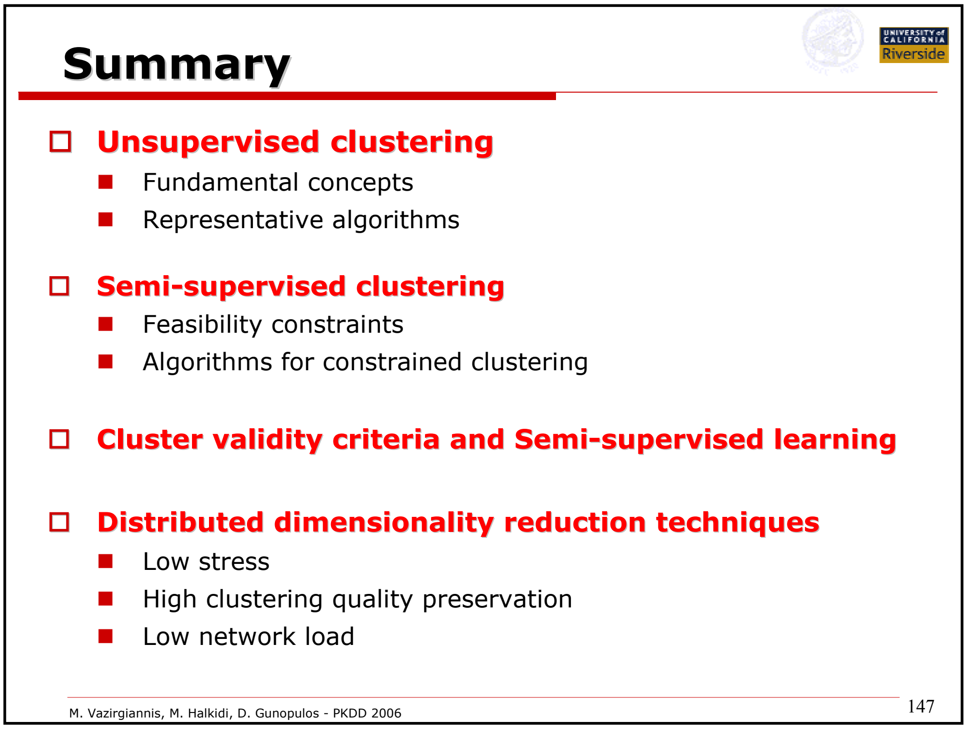
Unsupervised k-windows algorithm

1st step.

- The windows are moved to the Euclidean space without altering their size.
- Each window is moved by setting its center to the mean of the patterns currently included.
- **Termination:** further movement does not increase the number of patterns included

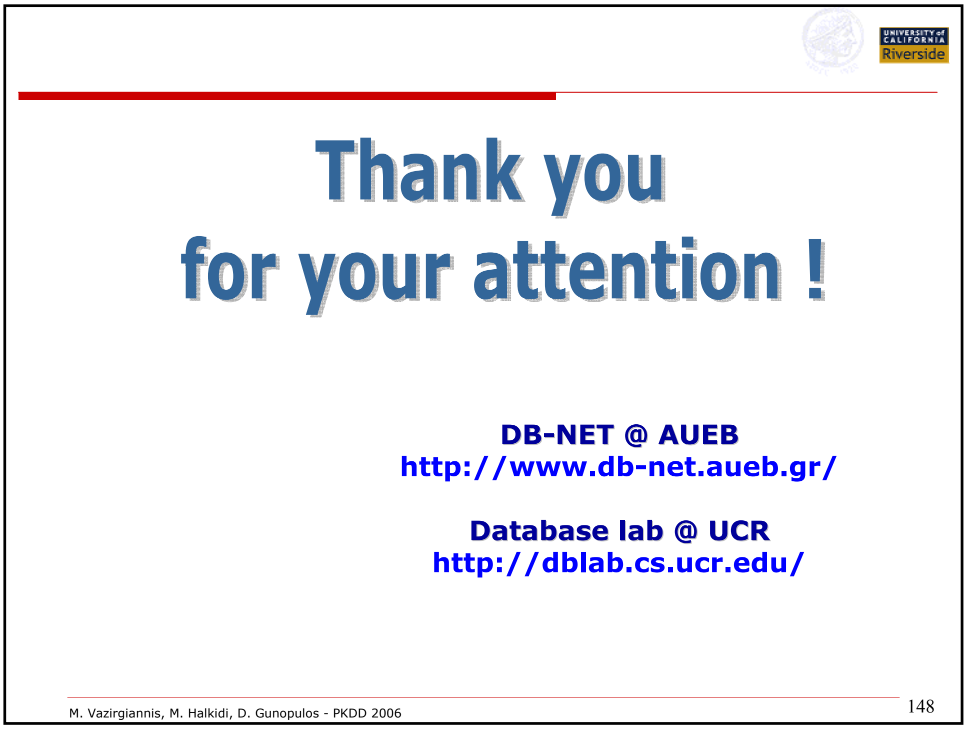


- □ **2nd step.**
  - The size of windows enlarged to capture as many patterns of the cluster as possible
  - **Termination:** the number of patterns included in the window no longer increases.



Summary

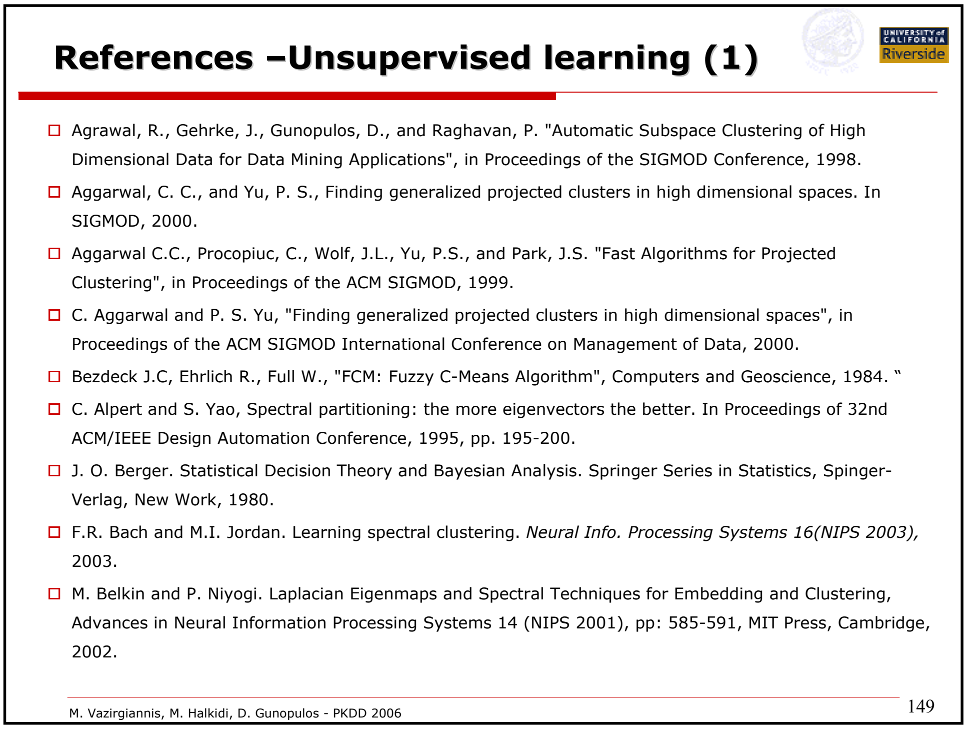
- □ **Unsupervised clustering**
  - Fundamental concepts
  - Representative algorithms
- □ **Semi-supervised clustering**
  - Feasibility constraints
  - Algorithms for constrained clustering
- □ **Cluster validity criteria and Semi-supervised learning**
- □ **Distributed dimensionality reduction techniques**
  - Low stress
  - High clustering quality preservation
  - Low network load



Thank you for your attention !

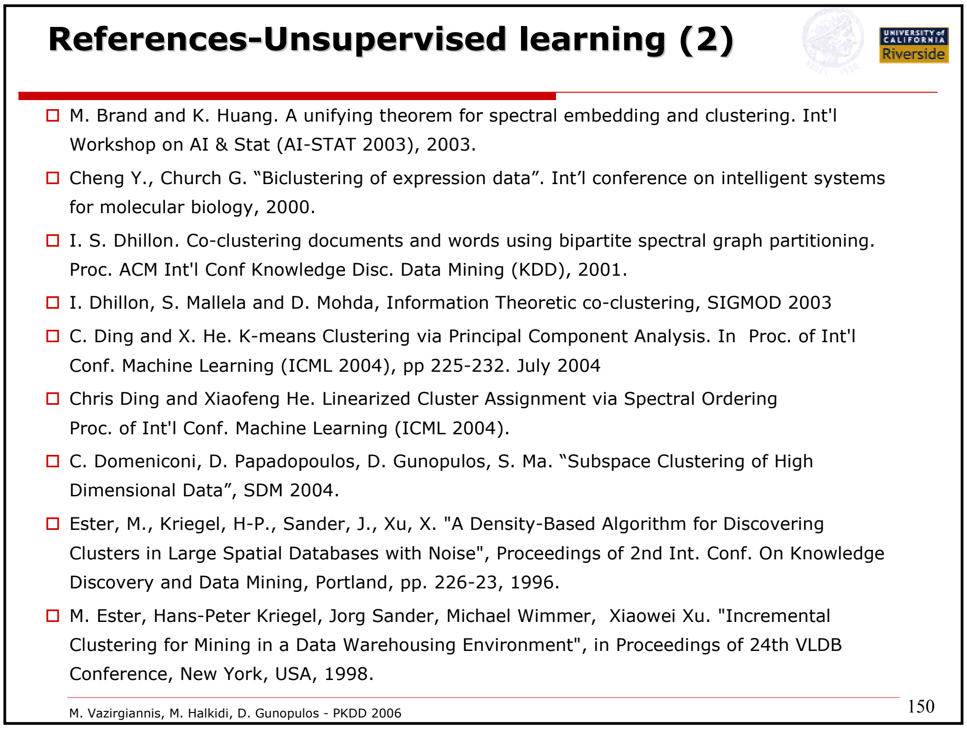
DB-NET @ AUEB
http://www.db-net.aueb.gr/

Database lab @ UCR
http://dblab.cs.ucr.edu/



References –Unsupervised learning (1)

- Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications", in Proceedings of the SIGMOD Conference, 1998.
- Aggarwal, C. C., and Yu, P. S., Finding generalized projected clusters in high dimensional spaces. In SIGMOD, 2000.
- Aggarwal C.C., Procopiuc, C., Wolf, J.L., Yu, P.S., and Park, J.S. "Fast Algorithms for Projected Clustering", in Proceedings of the ACM SIGMOD, 1999.
- C. Aggarwal and P. S. Yu, "Finding generalized projected clusters in high dimensional spaces", in Proceedings of the ACM SIGMOD International Conference on Management of Data, 2000.
- Bezdeck J.C, Ehrlich R., Full W., "FCM: Fuzzy C-Means Algorithm", Computers and Geoscience, 1984. “
- C. Alpert and S. Yao, Spectral partitioning: the more eigenvectors the better. In Proceedings of 32nd ACM/IEEE Design Automation Conference, 1995, pp. 195-200.
- J. O. Berger. Statistical Decision Theory and Bayesian Analysis. Springer Series in Statistics, Spinger-Verlag, New Work, 1980.
- F.R. Bach and M.I. Jordan. Learning spectral clustering. *Neural Info. Processing Systems 16(NIPS 2003),* 2003.
- M. Belkin and P. Niyogi. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering, Advances in Neural Information Processing Systems 14 (NIPS 2001), pp: 585-591, MIT Press, Cambridge, 2002.

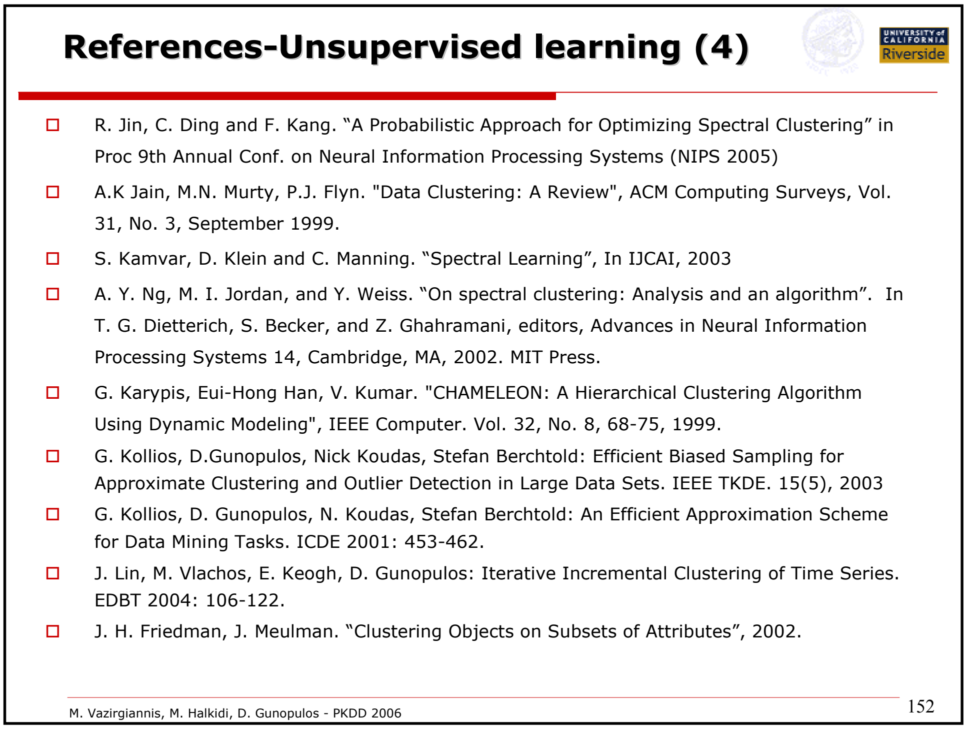


References-Unsupervised learning (2)

- M. Brand and K. Huang. A unifying theorem for spectral embedding and clustering. Int'l Workshop on AI & Stat (AI-STAT 2003), 2003.
- Cheng Y., Church G. “Biclustering of expression data”. Int’l conference on intelligent systems for molecular biology, 2000.
- I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. Proc. ACM Int'l Conf Knowledge Disc. Data Mining (KDD), 2001.
- I. Dhillon, S. Mallela and D. Mohda, Information Theoretic co-clustering, SIGMOD 2003
- C. Ding and X. He. K-means Clustering via Principal Component Analysis. In Proc. of Int'l Conf. Machine Learning (ICML 2004), pp 225-232. July 2004
- Chris Ding and Xiaofeng He. Linearized Cluster Assignment via Spectral Ordering Proc. of Int'l Conf. Machine Learning (ICML 2004).
- C. Domeniconi, D. Papadopoulos, D. Gunopulos, S. Ma. “Subspace Clustering of High Dimensional Data”, SDM 2004.
- Ester, M., Kriegel, H-P., Sander, J., Xu, X. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", Proceedings of 2nd Int. Conf. On Knowledge Discovery and Data Mining, Portland, pp. 226-23, 1996.
- M. Ester, Hans-Peter Kriegel, Jorg Sander, Michael Wimmer, Xiaowei Xu. "Incremental Clustering for Mining in a Data Warehousing Environment", in Proceedings of 24th VLDB Conference, New York, USA, 1998.

References-Unsupervised learning (3)

- S. Guha, R.Rastogi, K. Shim. "CURE: An Efficient Clustering Algorithm for Large Databases", in SIGMOD Conference, 1998.
- S. Guha, R. Rastogi, K. Shim. "ROCK: A Robust Clustering Algorithm for Categorical Attributes", in Proceedings of the IEEE Conference on Data Engineering, 1999.
- Ming Gu, Hongyuan Zha, Chris Ding, Xiaofeng He and Horst Simon. “Spectral Relaxation Models and Structure Analysis for K-way Graph Clustering and Bi-clustering”. Technical Report, 2001.
- J. Han, M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2001.
- R. J. Hathaway, J. C. Bezdek, John W. Davenport. "On relational data versions of c-means algorithm", Pattern Recognition Letters, Vol. 17, pp. 607-612, 1996.
- A. Hinneburg, D. Keim. "An Efficient Approach to Clustering in Large Multimedia Databases with Noise", in Proceedings of KDD Conference, 1998.
- Z. Huang. "A Fast Clustering Algorithm to Cluster very Large Categorical Data sets in Data Mining", DMKD, 1997.
- R. J. Hathaway, James C. Bezdek. "NERF c-Means: Non-Euclidean Relational Fuzzy Clustering", Pattern Recognition Letters, Vol. 27, No 3, pp. 428-437, 1994.
- L. Parsons, E. Haque, and H.Liu. “Subspace clustering for high dimensional data: a review”. SIGKDD Explor. Newsl., 6(1):90105, 2004.



References-Unsupervised learning (4)

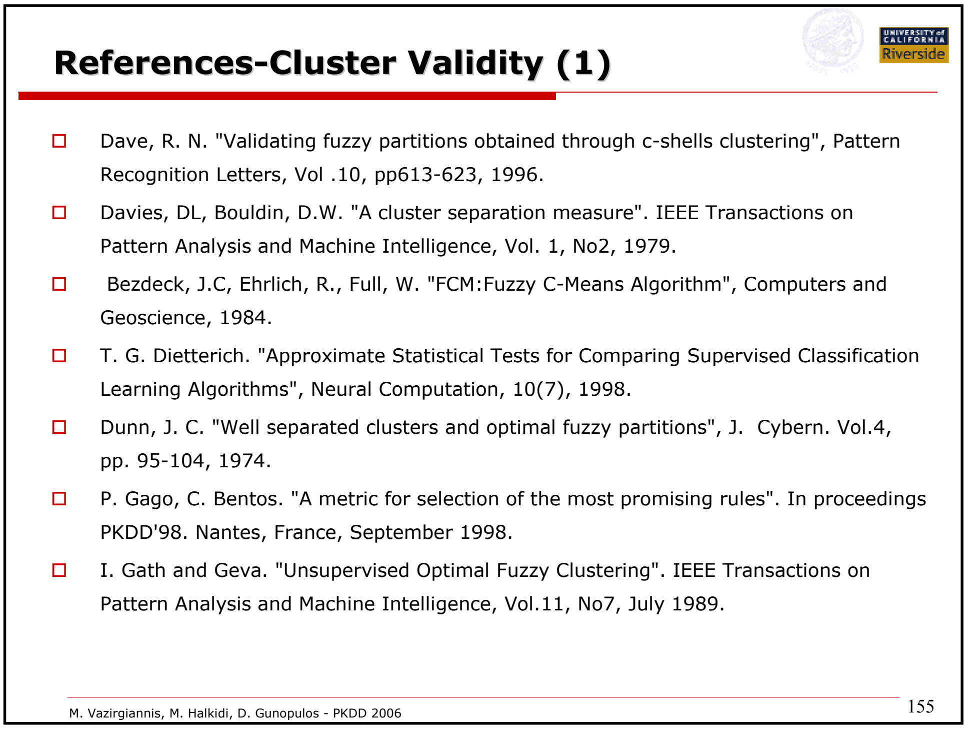
- R. Jin, C. Ding and F. Kang. “A Probabilistic Approach for Optimizing Spectral Clustering” in Proc 9th Annual Conf. on Neural Information Processing Systems (NIPS 2005)
- A.K Jain, M.N. Murty, P.J. Flyn. "Data Clustering: A Review", ACM Computing Surveys, Vol. 31, No. 3, September 1999.
- S. Kamvar, D. Klein and C. Manning. “Spectral Learning”, In IJCAI, 2003
- A. Y. Ng, M. I. Jordan, and Y. Weiss. “On spectral clustering: Analysis and an algorithm”. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, Advances in Neural Information Processing Systems 14, Cambridge, MA, 2002. MIT Press.
- G. Karypis, Eui-Hong Han, V. Kumar. "CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling", IEEE Computer. Vol. 32, No. 8, 68-75, 1999.
- G. Kollios, D.Gunopulos, Nick Koudas, Stefan Berchtold: Efficient Biased Sampling for Approximate Clustering and Outlier Detection in Large Data Sets. IEEE TKDE. 15(5), 2003
- G. Kollios, D. Gunopulos, N. Koudas, Stefan Berchtold: An Efficient Approximation Scheme for Data Mining Tasks. ICDE 2001: 453-462.
- J. Lin, M. Vlachos, E. Keogh, D. Gunopulos: Iterative Incremental Clustering of Time Series. EDBT 2004: 106-122.
- J. H. Friedman, J. Meulman. “Clustering Objects on Subsets of Attributes”, 2002.

References-Unsupervised learning (5)

- A. Nanopoulos, Y. Theodoridis, Y. Manolopoulos. "C2P: Clustering based on Closest Pairs", in Proceeding of the VLDB Conference, Roma, Italy, 2001.
- R. Ng, J.Han. "Efficient and Effective Clustering Methods for Spatial Data Mining", in Proceedings of the VLDB Conference, Santiago, Chile, 1994.
- Dimitris Papadopoulos, Carlotta Domeniconi, Dimitrios Gunopulos, Sheng Ma: Clustering gene expression data in SQL using locally adaptive metrics. DMKD 2003: 35-41
- P. Perona and W. Freeman, "A factorization approach to grouping," in Proc. ECCV '98, vol. 1, 1998, pp. 655--670.
- Procopiuc, C. M., Jones, M., Agarwal, P. K., and Murali, T. M. A Monte Carlo algorithm for fast projective clustering. In SIGMOD, 2002.
- G. Scott and H. Longuet-Higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. In British Conference on Machine Vision, pages 731--737, 1990
- C. Sheikholeslami, S. Chatterjee, A. Zhang. "WaveCluster: A-MultiResolution Clustering Approach for Very Large Spatial Database", in Proceedings of 24th VLDB Conference, New York, USA, 1998.

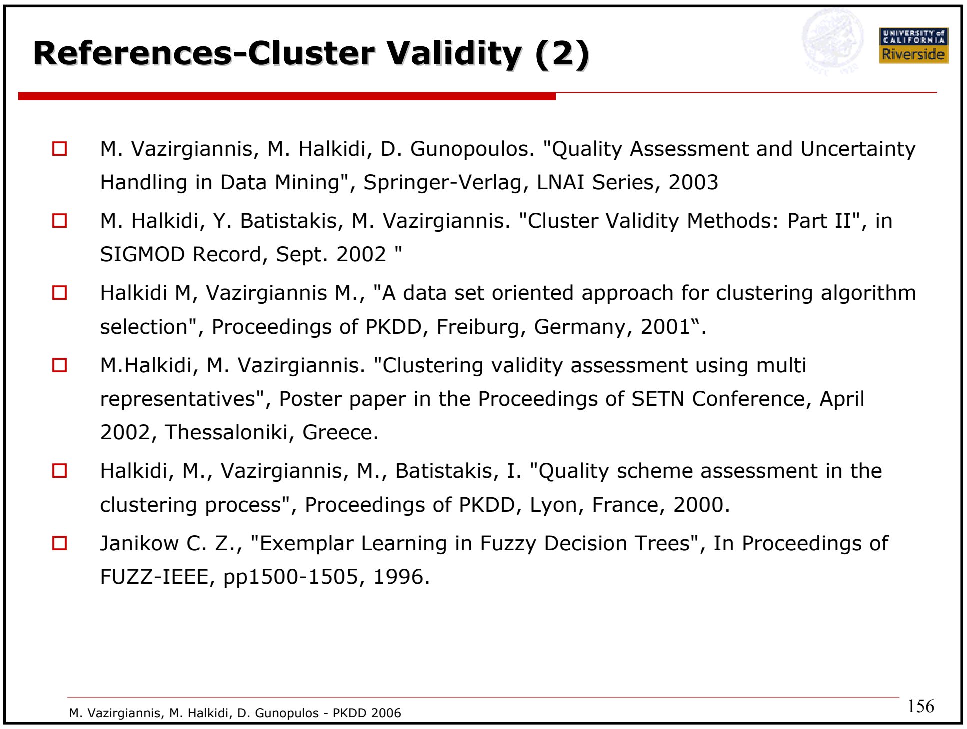
References-Unsupervised learning (6)

- Wei Wang, Jiorg Yang and Richard Muntz. "STING: A statistical information grid approach to spatial data mining", in proceedings of the VLDB Conference, 1997.
- Tian Zhang, Raghu Ramakrishnman, Miron Linvy. "BIRCH: An Efficient Method for Very Large Databases", SIGMOD Rec. 25, 2, 103-114. 1996.
- M. Meila and J. Shi. A random walks view of spectral segmentation. Int'l Workshop on AI & Stat (AI-STAT), 2001
- M. Meila and L. Xu. Multiway cuts and spectral clustering. U. Washington Tech Report, 2003.
- H. Zha, X. He, C. Ding, M. Gu & H. Simon. Bipartite Graph Partitioning and Data Clustering, Proc. of ACM 10th Int'l Conf. Information and Knowledge Management (CIKM 2001), pp.25-31, 2001, Atlanta.
- Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis. Univ. Minnesota, CS Dept. Tech Report 01-40, 2001.
- Yair Weiss. "Segmentation Using Eigenvectors: A Unifying View". ICCV, 1999.



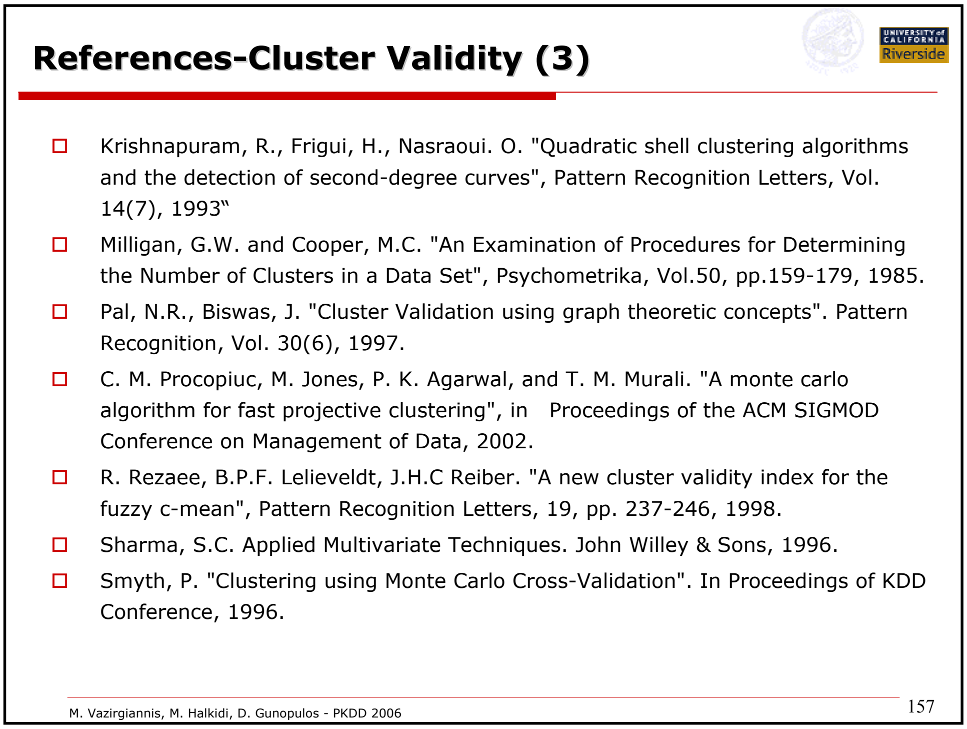
References-Cluster Validity (1)

- Dave, R. N. "Validating fuzzy partitions obtained through c-shells clustering", Pattern Recognition Letters, Vol .10, pp613-623, 1996.
- Davies, DL, Bouldin, D.W. "A cluster separation measure". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 1, No2, 1979.
- Bezdeck, J.C, Ehrlich, R., Full, W. "FCM:Fuzzy C-Means Algorithm", Computers and Geoscience, 1984.
- T. G. Dietterich. "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms", Neural Computation, 10(7), 1998.
- Dunn, J. C. "Well separated clusters and optimal fuzzy partitions", J. Cybern. Vol.4, pp. 95-104, 1974.
- P. Gago, C. Bentos. "A metric for selection of the most promising rules". In proceedings PKDD'98. Nantes, France, September 1998.
- I. Gath and Geva. "Unsupervised Optimal Fuzzy Clustering". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.11, No7, July 1989.



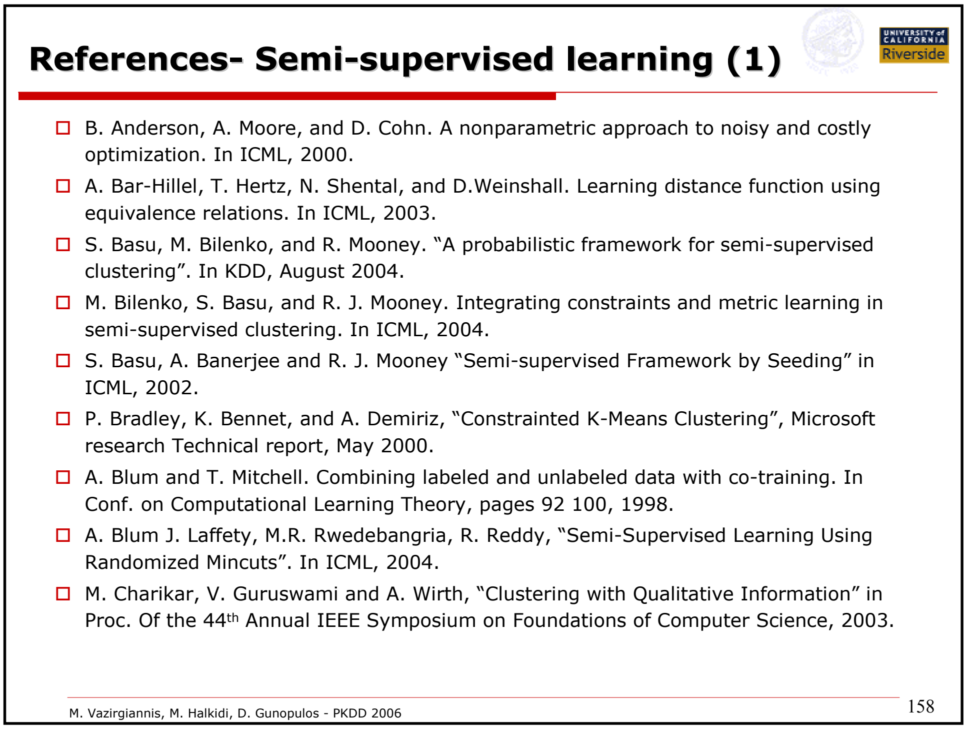
References-Cluster Validity (2)

- M. Vazirgiannis, M. Halkidi, D. Gunopoulos. "Quality Assessment and Uncertainty Handling in Data Mining", Springer-Verlag, LNAI Series, 2003
- M. Halkidi, Y. Batistakis, M. Vazirgiannis. "Cluster Validity Methods: Part II", in SIGMOD Record, Sept. 2002 "
- Halkidi M, Vazirgiannis M., "A data set oriented approach for clustering algorithm selection", Proceedings of PKDD, Freiburg, Germany, 2001".
- M.Halkidi, M. Vazirgiannis. "Clustering validity assessment using multi representatives", Poster paper in the Proceedings of SETN Conference, April 2002, Thessaloniki, Greece.
- Halkidi, M., Vazirgiannis, M., Batistakis, I. "Quality scheme assessment in the clustering process", Proceedings of PKDD, Lyon, France, 2000.
- Janikow C. Z., "Exemplar Learning in Fuzzy Decision Trees", In Proceedings of FUZZ-IEEE, pp1500-1505, 1996.



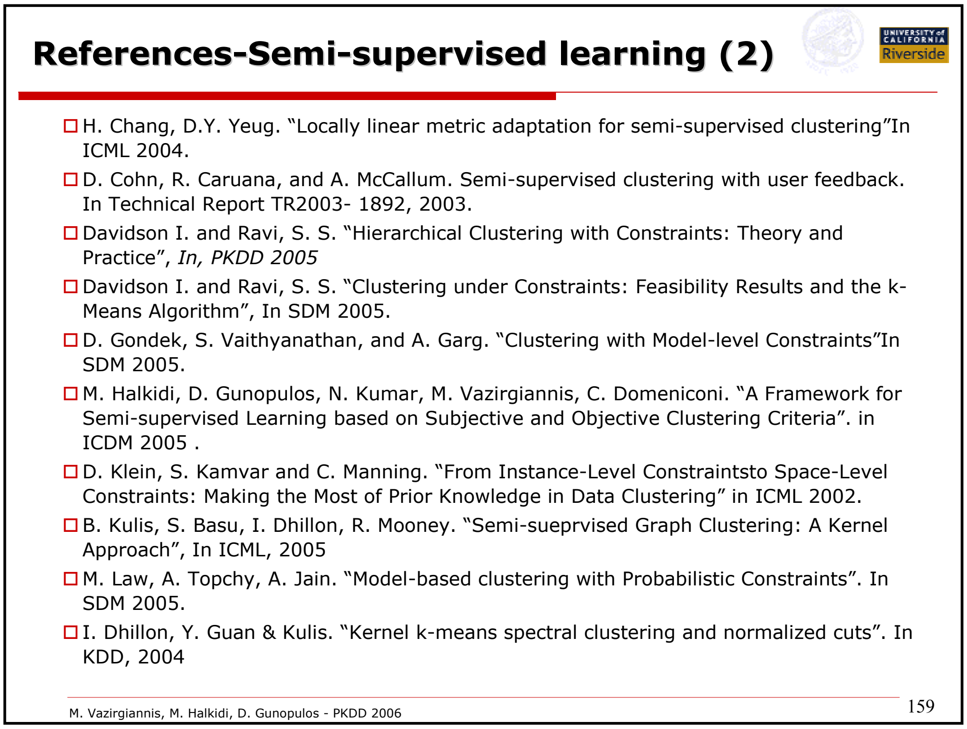
References-Cluster Validity (3)

- Krishnapuram, R., Frigui, H., Nasraoui. O. "Quadratic shell clustering algorithms and the detection of second-degree curves", Pattern Recognition Letters, Vol. 14(7), 1993‟
- Milligan, G.W. and Cooper, M.C. "An Examination of Procedures for Determining the Number of Clusters in a Data Set", Psychometrika, Vol.50, pp.159-179, 1985.
- Pal, N.R., Biswas, J. "Cluster Validation using graph theoretic concepts". Pattern Recognition, Vol. 30(6), 1997.
- 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 SIGMOD Conference on Management of Data, 2002.
- R. Rezaee, B.P.F. Lelieveldt, J.H.C Reiber. "A new cluster validity index for the fuzzy c-mean", Pattern Recognition Letters, 19, pp. 237-246, 1998.
- Sharma, S.C. Applied Multivariate Techniques. John Willey & Sons, 1996.
- Smyth, P. "Clustering using Monte Carlo Cross-Validation". In Proceedings of KDD Conference, 1996.



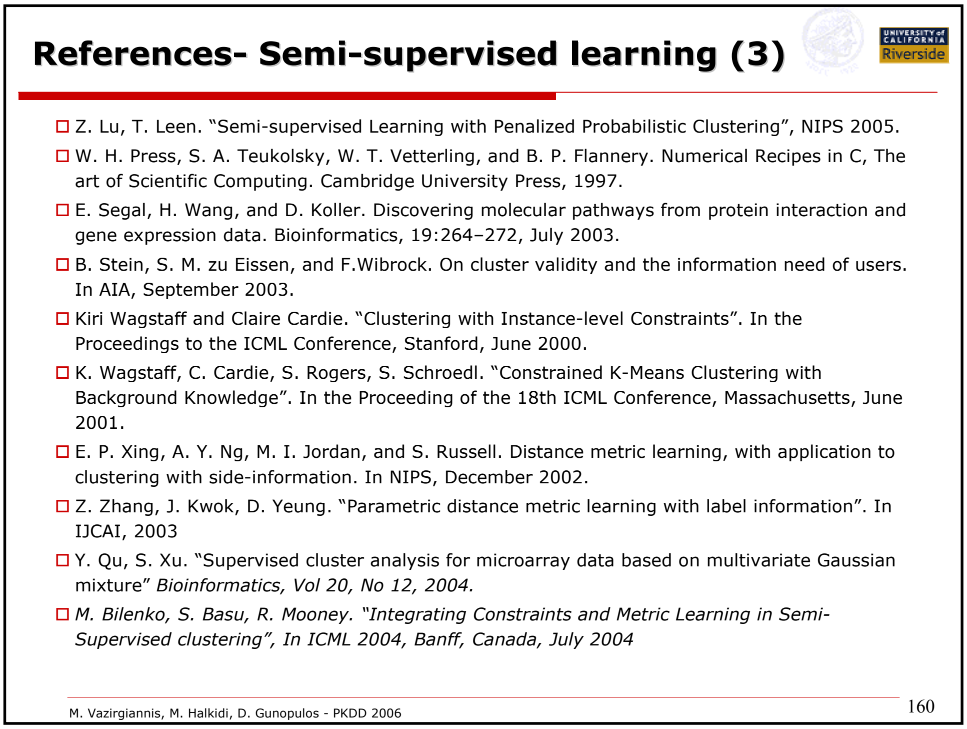
References- Semi-supervised learning (1)

- B. Anderson, A. Moore, and D. Cohn. A nonparametric approach to noisy and costly optimization. In ICML, 2000.
- A. Bar-Hillel, T. Hertz, N. Shental, and D.Weinshall. Learning distance function using equivalence relations. In ICML, 2003.
- S. Basu, M. Bilenko, and R. Mooney. "A probabilistic framework for semi-supervised clustering". In KDD, August 2004.
- M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In ICML, 2004.
- S. Basu, A. Banerjee and R. J. Mooney "Semi-supervised Framework by Seeding" in ICML, 2002.
- P. Bradley, K. Bennet, and A. Demiriz, "Constrainted K-Means Clustering", Microsoft research Technical report, May 2000.
- A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In Conf. on Computational Learning Theory, pages 92 100, 1998.
- A. Blum J. Laffety, M.R. Rwedebangria, R. Reddy, "Semi-Supervised Learning Using Randomized Mincuts". In ICML, 2004.
- M. Charikar, V. Guruswami and A. Wirth, "Clustering with Qualitative Information" in Proc. Of the 44th Annual IEEE Symposium on Foundations of Computer Science, 2003.



References-Semi-supervised learning (2)

- H. Chang, D.Y. Yeug. "Locally linear metric adaptation for semi-supervised clustering"In ICML 2004.
- D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. In Technical Report TR2003- 1892, 2003.
- Davidson I. and Ravi, S. S. "Hierarchical Clustering with Constraints: Theory and Practice", *In, PKDD 2005*
- Davidson I. and Ravi, S. S. "Clustering under Constraints: Feasibility Results and the k-Means Algorithm", In SDM 2005.
- D. Gondek, S. Vaithyanathan, and A. Garg. "Clustering with Model-level Constraints"In SDM 2005.
- M. Halkidi, D. Gunopulos, N. Kumar, M. Vazirgiannis, C. Domeniconi. "A Framework for Semi-supervised Learning based on Subjective and Objective Clustering Criteria". in ICDM 2005 .
- D. Klein, S. Kamvar and C. Manning. "From Instance-Level Constraintsto Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering" in ICML 2002.
- B. Kulis, S. Basu, I. Dhillon, R. Mooney. "Semi-sueprvised Graph Clustering: A Kernel Approach", In ICML, 2005
- M. Law, A. Topchy, A. Jain. "Model-based clustering with Probabilistic Constraints". In SDM 2005.
- I. Dhillon, Y. Guan & Kulis. "Kernel k-means spectral clustering and normalized cuts". In KDD, 2004



References- Semi-supervised learning (3)

- Z. Lu, T. Leen. "Semi-supervised Learning with Penalized Probabilistic Clustering", NIPS 2005.
- W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. Numerical Recipes in C, The art of Scientific Computing. Cambridge University Press, 1997.
- E. Segal, H. Wang, and D. Koller. Discovering molecular pathways from protein interaction and gene expression data. Bioinformatics, 19:264–272, July 2003.
- B. Stein, S. M. zu Eissen, and F.Wibrock. On cluster validity and the information need of users. In AIA, September 2003.
- Kiri Wagstaff and Claire Cardie. "Clustering with Instance-level Constraints". In the Proceedings to the ICML Conference, Stanford, June 2000.
- K. Wagstaff, C. Cardie, S. Rogers, S. Schroedl. "Constrained K-Means Clustering with Background Knowledge". In the Proceeding of the 18th ICML Conference, Massachusetts, June 2001.
- E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In NIPS, December 2002.
- Z. Zhang, J. Kwok, D. Yeung. "Parametric distance metric learning with label information". In IJCAI, 2003
- Y. Qu, S. Xu. "Supervised cluster analysis for microarray data based on multivariate Gaussian mixture" *Bioinformatics, Vol 20, No 12, 2004.*
- *M. Bilenko, S. Basu, R. Mooney. "Integrating Constraints and Metric Learning in Semi-Supervised clustering", In ICML 2004, Banff, Canada, July 2004*

References –
Distributed approaches, Dimensionality reduction

- □ Vin de Silva, Joshua B. "Sparse Multidimensional Scaling Using landmark points", Tenenbaum, 2004
- □ Vin de Silva, Joshua B. Tenenbaum"Global versus local methods in nonlinear dimensionality reduction",NIPS 2003
- □ I.K. Fodor, "A Survey of Dimension Reduction Techniques", US Department Of Energy, 2002
- □ Faisal N.Abu-Khzam, Nagiza Samatova, George Ostrouchov, Michael A.Langston, Al Geist, "Distributed Dimension Reduction Algorithms for Widely Dispersed Data" PDCS 2002, pp. 167-174
- □ Yongming Qu, George Ostrouchov, Nagiza Samatova, Al Geist, "Principal Component Analysis forDimension Reduction in Massive Distributed Data Sets", 5th International Workshop on High Performance Data Mining, 2002
- □ P. Magdalinos, C. Doulkeridis and M. Vazirgiannis, "A Novel Effective Distributed Dimensionality ReductionAlgorithm", In Workshop on Feature Selection for Data Mining (FSDM'06), pp.18-25, Bethesda, Maryland, 2006.
- □ Tasoulis, Vrahatis. "Unsupervised Distributed Clustering", PRL 2005
- □ M.N. Vrahatis, B. Boutsinas, P. Alevizos, G. Pavlides, "The new k-windows algorithm for improving the k-means clustering algorithm", *Journal of Complexity, 18:375-391, 2002*
- □ H. Kargupta, W. Huang, K. Sivakumar, E. Johnson. "Distributed clustering using collective principal component analysis". *Knowledge and Information Systems,* 3(4), 2001.