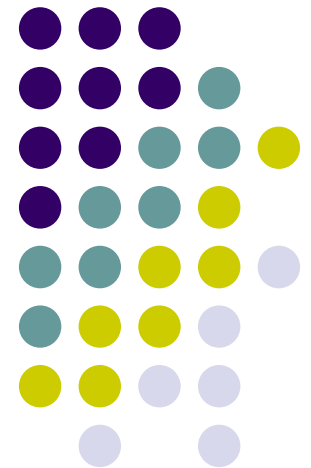


# Clustering

Zhang Hongxin  
zhx@cad.zju.edu.cn

State Key Lab of CAD&CG, ZJU  
2007-03-15





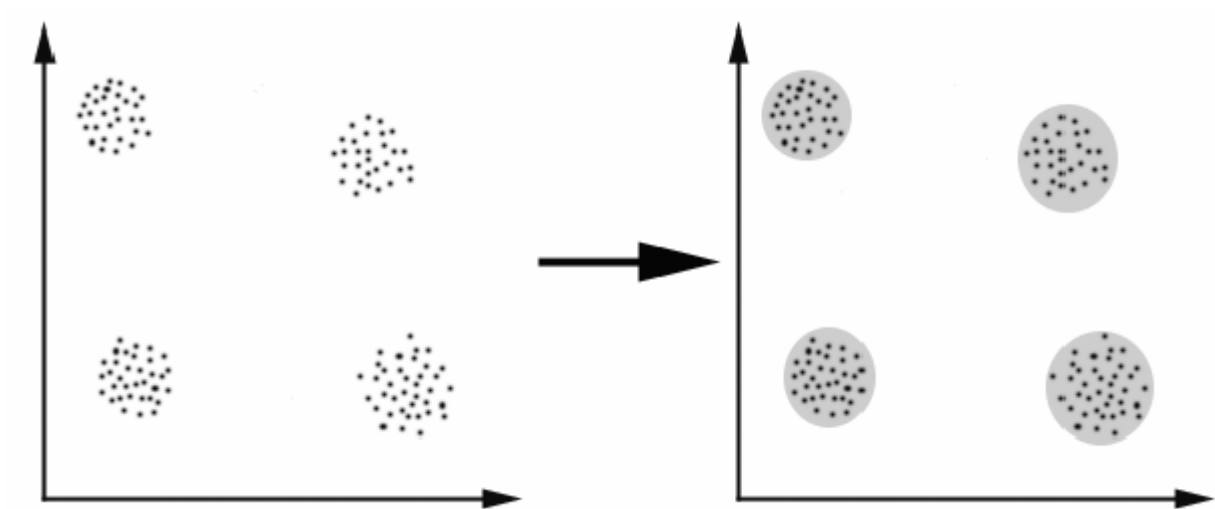
# Outline

- Flat clustering
  - Mixture of Gaussians
  - K-means
- Hierarchical clustering
  - bottom-up
- Spectral based clustering
- Applications



# Clustering

- Given set of data points, group them
- **Unsupervised** learning
- Learn the similarity. Which patient are similar?  
(or customers, faces, earthquakes, ...)





# Clustering vs. Classification

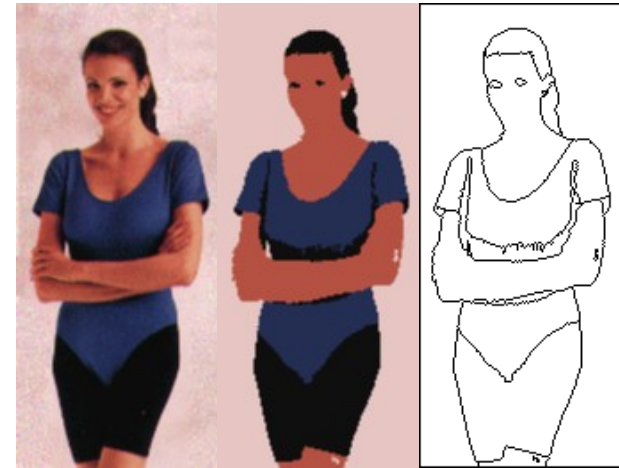
- **Clustering**

- **Instance:**  $\{\mathbf{x}_i\}_{i=1}^N$
- **Learn:**  $\langle \mathbf{x}_i, t_i \rangle$  and/or mapping from  $\mathbf{x}$  to  $t(\mathbf{x})$

- **Classification/Regression**

- **Instance:**  $\langle \mathbf{x}_i, t_i \rangle$
- **Learn:** mapping from  $\mathbf{x}$  to  $t(\mathbf{x})$

# Clustering: image segmentation



Mean-shift segmentation



# Mixtures of Gaussians

- Mixture distribution:
  - Assume  $P(x)$  is a mixture of  $K$  different Gaussians
  - Assume each data point,  $x$  is generated by 2-step process
    - Choose one of the  $K$  Gaussians as label  $z$
    - Generate  $x$  according to the Gaussian  $N(\mu_z, \Sigma_z)$

$$P(\mathbf{x}) = \sum_{z=1}^K P(Z = z | \pi) N(\mathbf{x} | \mu_z, \Sigma_z)$$

- What object function shall we optimize?
  - Maximize data likelihood



# Mixtures of Gaussians (cont.)

- Multivariate Gaussian model

$$p(\mathbf{x}|\mu, \Sigma) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)\right\}$$

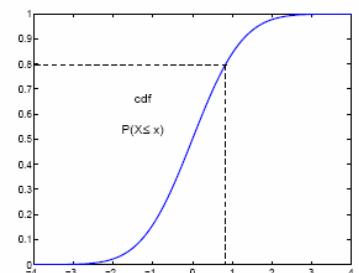
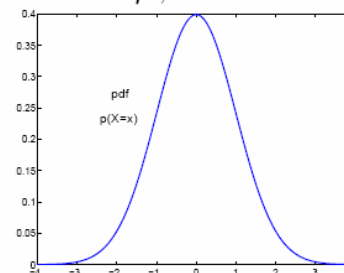
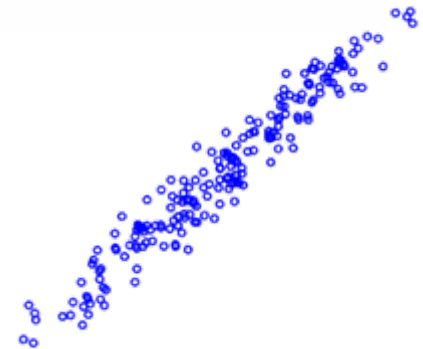
- How to generate it?

$$F_{\mu, \sigma^2}(x) = \int_{-\infty}^x p(z|\mu, \sigma^2) dz$$

$$u \sim \text{Uniform}(0, 1) \Rightarrow x = F_{\mu, \sigma^2}^{-1}(u) \sim p(x|\mu, \sigma^2)$$

$$z_i \sim p(z_i|\mu = 0, \sigma^2 = 1), \quad \mathbf{z} = [z_1, \dots, z_d]^T$$

$$\mathbf{x} = \Sigma^{1/2} \mathbf{z} + \mu$$



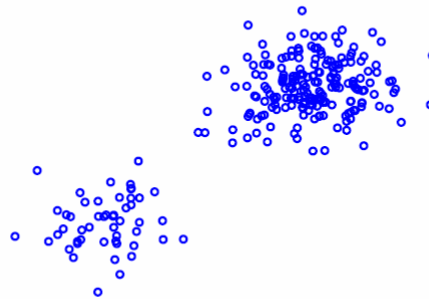


# Multi-variate density estimation

- A mixture of Gaussians model

$$p(\mathbf{x}|\theta) = \sum_{i=1}^k p_j p(\mathbf{x}|\mu_j, \Sigma_j)$$

where  $\theta = \{p_1, \dots, p_k, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k\}$  contains all the parameters of the mixture model.  $\{p_j\}$  are known as *mixing proportions or coefficients*.





# Mixtures of Gaussians: Wishart distribution



- A mixture of Gaussian Model:

$$p(\mathbf{x}|\theta) = \sum_{j=1}^k p_j p(\mathbf{x}|\mu_j, \Sigma_j)$$

High dimensional parameters

$$\theta = \{p_1, \dots, p_k, \mu_1, \dots, \mu_k, \underline{\Sigma_1, \dots, \Sigma_k}\}$$

- Wishart prior

$$P(\Sigma|S, n') \propto \frac{1}{|\Sigma|^{n'/2}} \exp\left(-\frac{n'}{2} \text{Trace}(\Sigma^{-1} S)\right)$$

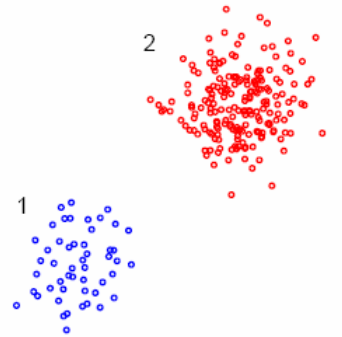
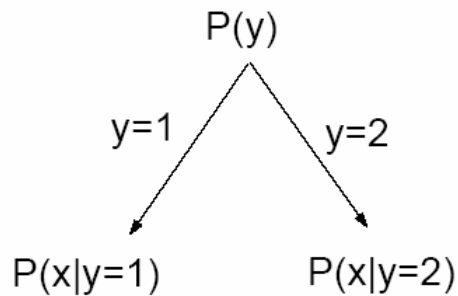
$S$  = “prior” covariance matrix

$n'$  = equivalent sample size

# Mixture density



- Data generation process:



$$\begin{aligned} p(\mathbf{x}|\theta) &= \sum_{j=1,2} P(y = j) \cdot p(\mathbf{x}|y = j) \quad (\text{generic mixture}) \\ &= \sum_{j=1,2} p_j \cdot p(\mathbf{x}|\mu_j, \Sigma_j) \quad (\text{mixture of Gaussians}) \end{aligned}$$

- Any data point  $\mathbf{x}$  could have been generated in two ways

# Mixture density



- If we are given just  $\mathbf{x}$  we don't know which mixture component this example came from

$$p(\mathbf{x}|\theta) = \sum_{j=1,2} p_j p(\mathbf{x}|\mu_j, \Sigma_j)$$

- We can evaluate the posterior probability that an observed  $\mathbf{x}$  was generated from the first mixture component

$$\begin{aligned} P(y = 1|\mathbf{x}, \theta) &= \frac{P(y = 1) \cdot p(\mathbf{x}|y = 1)}{\sum_{j=1,2} P(y = j) \cdot p(\mathbf{x}|y = j)} \\ &= \frac{p_1 p(\mathbf{x}|\mu_1, \Sigma_1)}{\sum_{j=1,2} p_j p(\mathbf{x}|\mu_j, \Sigma_j)} \end{aligned}$$

- This solves a *credit assignment* problem

# Mixture density: posterior sampling



- Consider sampling  $\mathbf{x}$  from the mixture density, then  $y$  from the posterior over the components given  $\mathbf{x}$ , and finally  $\mathbf{x}'$  from the component density indicated by  $y$ :

$$\mathbf{x} \sim p(\mathbf{x}|\theta)$$

$$y \sim P(y|\mathbf{x}, \theta)$$

$$\mathbf{x}' \sim p(\mathbf{x}'|y, \theta)$$

Is  $y$  a fair sample from the prior distribution  $P(y)$ ?

Is  $\mathbf{x}'$  a fair sample from the mixture density  $p(\mathbf{x}'|\theta)$ ?

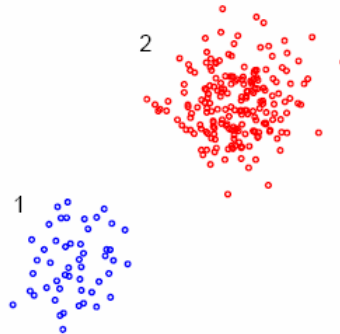
# Mixture density estimation



- Suppose we want to estimate a two component mixture of Gaussians model.

$$p(\mathbf{x}|\theta) = p_1 p(\mathbf{x}|\mu_1, \Sigma_1) + p_2 p(\mathbf{x}|\mu_2, \Sigma_2)$$

- If each example  $\mathbf{x}_i$  in the training set were labeled  $y_i = 1, 2$  according to which mixture component (1 or 2) had generated it, then the estimation would be easy.

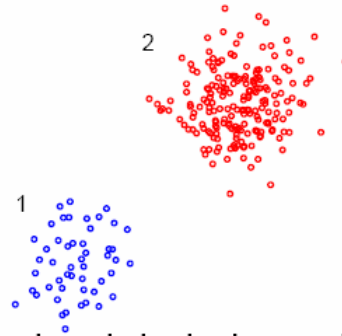


- Labeled examples  $\Rightarrow$  no credit assignment problem



# Mixture density estimation

When examples are already assigned to mixture components (labeled), we can estimate each Gaussian independently



- If  $\hat{n}_j$  is the number of examples labeled  $j$ , then for each  $j = 1, 2$  we set

$$\hat{p}_j \leftarrow \frac{\hat{n}_j}{n}$$

$$\hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} \mathbf{x}_i$$

$$\hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} (\mathbf{x}_i - \hat{\mu}_j)(\mathbf{x}_i - \hat{\mu}_j)^T$$

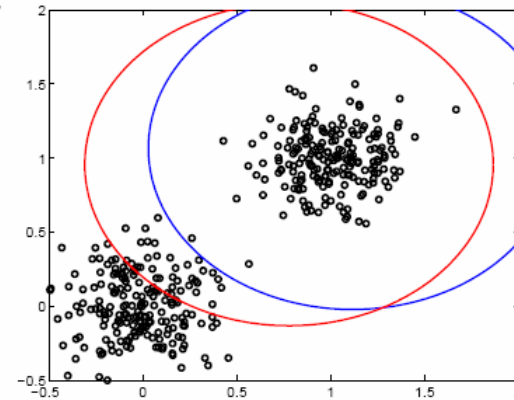
# Mixture density estimation: credit assignment



- Of course we don't have such labels ... but we can guess what the labels might be based on our current mixture distribution
- We get soft labels or posterior probabilities of which Gaussian generated which example:

$$\hat{p}(j|i) \leftarrow P(y_i = j | \mathbf{x}_i, \theta)$$

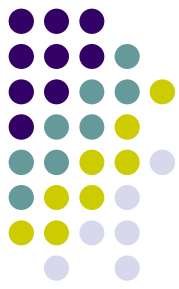
where  $\sum_{j=1,2} \hat{p}(j|i) = 1$  for all  $i = 1, \dots, n$ .



- When the Gaussians are almost identical (as in the figure),  $\hat{p}(1|i) \approx \hat{p}(2|i)$  for almost any available point  $\mathbf{x}_i$ .

Even slight differences can help us determine how we should modify the Gaussians.

# The Expectation-Maximization algorithm



**E-step:** softly assign examples to mixture components

$$\hat{p}(j|i) \leftarrow P(y_i = j | \mathbf{x}_i, \theta), \text{ for all } j = 1, 2 \text{ and } i = 1, \dots, n$$

**M-step:** re-estimate the parameters (separately for the two Gaussians) based on the soft assignments.

$$\hat{n}_j \leftarrow \sum_{i=1}^n \hat{p}(j|i) = \text{Soft \# of examples labeled } j$$

$$\hat{p}_j \leftarrow \frac{\hat{n}_j}{n}$$

$$\hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^n \hat{p}(j|i) \mathbf{x}_i$$

$$\hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^n \hat{p}(j|i) (\mathbf{x}_i - \hat{\mu}_j)(\mathbf{x}_i - \hat{\mu}_j)^T$$



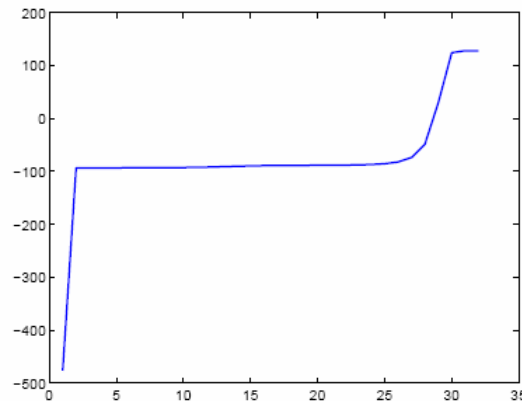


# The EM-algorithm

- Each iteration of the EM-algorithm *monotonically* increases the (log-)likelihood of the  $n$  training examples  $\mathbf{x}_1, \dots, \mathbf{x}_n$ :

$$\log p(\text{data} | \theta) = \sum_{i=1}^n \log \left( \overbrace{p_1 p(\mathbf{x}_i | \mu_1, \Sigma_1) + p_2 p(\mathbf{x}_i | \mu_2, \Sigma_2)}^{p(\mathbf{x}_i | \theta)} \right)$$

where  $\theta = \{p_1, p_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$  contains all the parameters of the mixture model.



# The EM algorithm



- The EM-algorithm finds a local maximum of  $l(\theta; D)$

**E-step:** evaluate the expected complete log-likelihood

$$\begin{aligned} J(\theta; \theta^{(t)}) &= \sum_{i=1}^n E_{j \sim P(j|\mathbf{x}_i, \theta^{(t)})} \log \left( p_j p(\mathbf{x}_i | \mu_j, \Sigma_j) \right) \\ &= \sum_{i=1}^n \sum_{j=1,2} P(j|\mathbf{x}_i, \theta^{(t)}) \log \left( p_j p(\mathbf{x}_i | \mu_j, \Sigma_j) \right) \end{aligned}$$

**M-step:** find the new parameters by maximizing the expected complete log-likelihood

$$\theta^{(t+1)} \leftarrow \operatorname{argmax}_{\theta} J(\theta; \theta^{(t)})$$



# Regularized EM algorithm

- To maximize a penalized (regularized) log-likelihood

$$l'(\theta; D) = \sum_{i=1}^n \log p(\mathbf{x}_i | \theta) + \log p(\theta)$$

we only need to modify the M-step of the EM-algorithm.

Specifically, in the M-step, we find  $\theta$  that maximize a penalized expected complete log-likelihood:

$$J(\theta; \theta^{(t)}) = \sum_{i=1}^n E_{j \sim P(j | \mathbf{x}_i, \theta^{(t)})} \log \left( p_j p(\mathbf{x}_i | \mu_j, \Sigma_j) \right) \\ + \log p(p_1, p_2) + \log p(\Sigma_1) + \log p(\Sigma_2)$$

where, for example,  $p(p_1, p_2)$  could be a *Dirichlet* and each  $p(\Sigma_j)$  a *Wishart* prior.

# Selecting the number of components

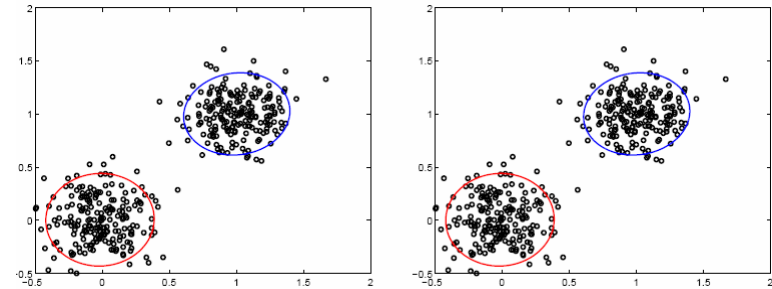
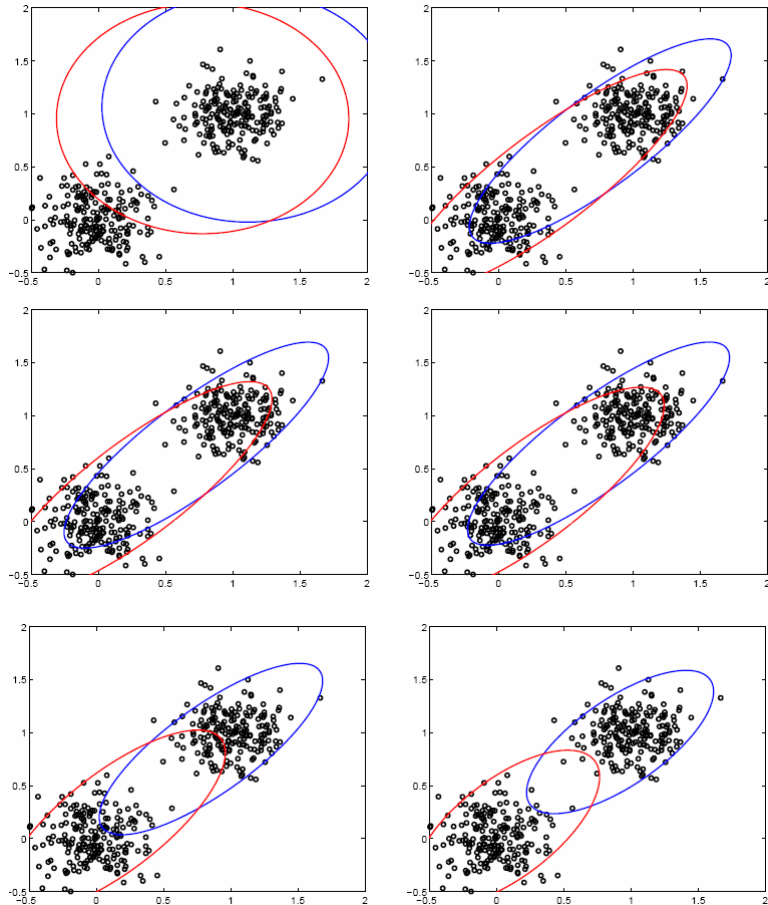


- As a simple strategy for selecting the appropriate number of mixture components, we can find  $k$  that minimize the following asymptotic approximation to the description length:

$$\text{DL} \approx -\log p(\text{data}|\hat{\theta}_k) + \frac{d_k}{2} \log(n)$$

where  $n$  is the number of training points,  $\hat{\theta}_k$  is the maximum likelihood parameter estimate for the  $k$ -component mixture, and  $d_k$  is the (effective) number of parameters in the  $k$ -mixture.

# Mixture density estimation: example





# K-means clustering

Given data  $\langle x_1 \dots x_n \rangle$ , and  $K$ , assign each  $x_i$  to one of  $K$  clusters,

$$C_1 \dots C_K, \text{ minimizing } J = \sum_{j=1}^K \sum_{x_i \in C_j} \|x_i - \mu_j\|^2$$

Where  $\mu_j$  is mean over all points in cluster  $C_j$

## K-Means Algorithm:

Initialize  $\mu_1 \dots \mu_K$  randomly

Repeat until convergence:

1. Assign each point  $x_i$  to the cluster with the closest mean  $\mu_j$
2. Calculate the new mean for each cluster

$$\mu_j \leftarrow \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

# K-Means vs. Mixture of Gaussians



- Both are iterative algorithms to assign points to clusters

- Objective function

- K Means: minimize

$$J = \sum_{j=1}^K \sum_{x_i \in C_j} \|x_i - \mu_j\|^2$$

- MoG: maximize likelihood  $P(X|\theta)$

- MoG the more general formulation

- Equivalent to K Means when  $\Sigma_k = \sigma I$ , and  $\sigma \rightarrow 0$

# Hierarchical (bottom-up) clustering



- Hierarchical agglomerative clustering: we sequentially merge the pair of “closest” points/clusters
- The procedure
  1. Find two closest points (clusters) and merge them
  2. Proceed until we have a single cluster (all the points)
- Two prerequisites:
  1. distance measure  $d(x_i, x_j)$  between two points
  2. distance measure between clusters (cluster linkage)



# Hierarchical (bottom-up) clustering



- A *linkage* method: we have to be able to measure distances between clusters of examples  $C_k$  and  $C_l$ 
  - a) Single linkage:

$$d_{kl} = \min_{i \in C_k, j \in C_l} d(\mathbf{x}_i, \mathbf{x}_j)$$

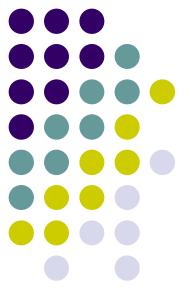
- b) Average linkage:

$$d_{kl} = \frac{1}{|C_l| |C_k|} \sum_{i \in C_k, j \in C_l} d(\mathbf{x}_i, \mathbf{x}_j)$$

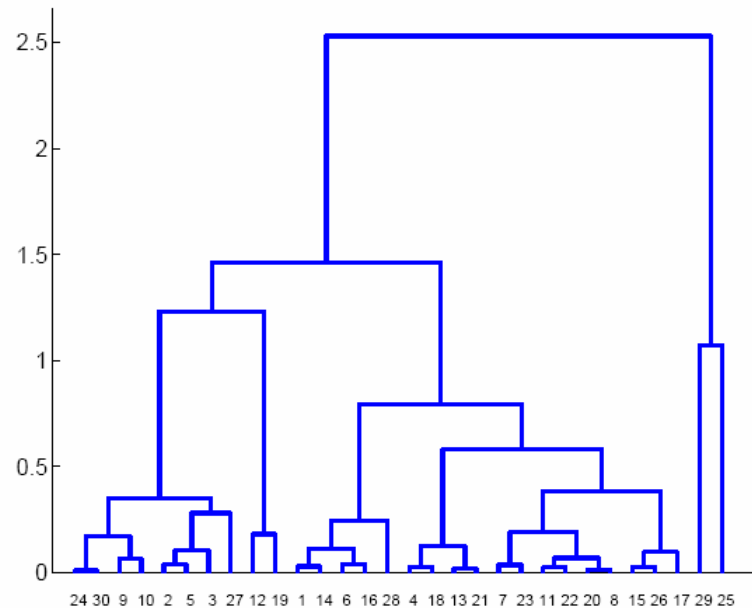
- c) Centroid linkage:

$$d_{kl} = d(\bar{\mathbf{x}}_k, \bar{\mathbf{x}}_l), \quad \bar{\mathbf{x}}_l = \frac{1}{|C_l|} \sum_{i \in C_l} \mathbf{x}_i$$

# Hierarchical (bottom-up) clustering



- A dendrogram representation of hierarchical clustering



The height of each pair represents the distance between the merged clusters; the specific linear ordering of points is chosen for clarity

# Spectral clustering



- The spectral clustering method we define relies on a random walk representation over the points. We construct this in three steps

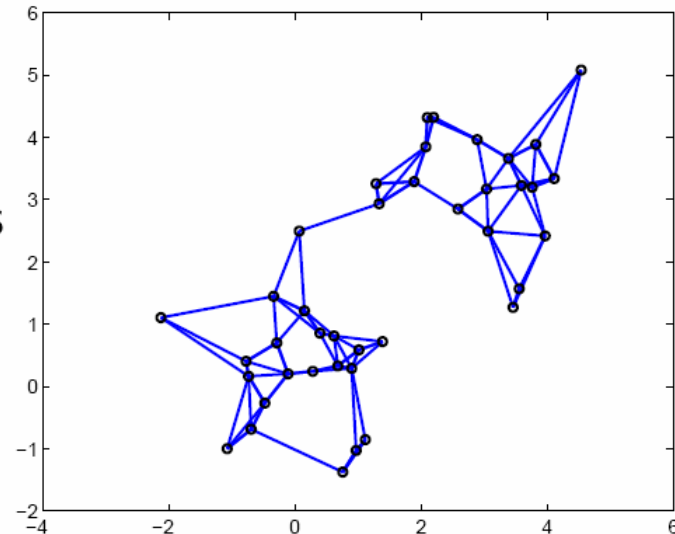
1. a nearest neighbor graph
2. similarity weights on the edges:

$$W_{ij} = \exp\{-\beta\|\mathbf{x}_i - \mathbf{x}_j\|\}$$

where  $W_{ii} = 1$  and the weight is zero for non-edges.

3. transition probability matrix

$$P_{ij} = W_{ij} / \sum_{j'} W_{ij'}$$



# Properties of the random walk



- If we start from  $i_0$ , the distribution of points  $i_t$  that we end up in after  $t$  steps is given by

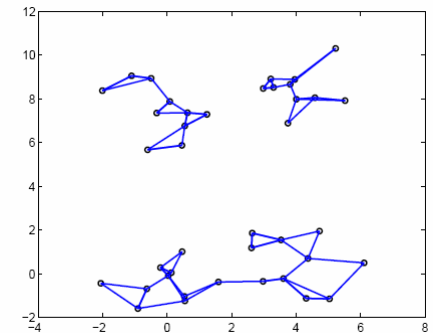
$$i_1 \sim P_{i_0 i_1}, \quad P_{ij} = \frac{W_{ij}}{W_{i\cdot}}, \quad \text{where } W_{i\cdot} = \sum_j W_{ij}$$

$$i_2 \sim \sum_{i_1} P_{i_0, i_1} P_{i_1 i_2} = [P^2]_{i_0 i_2},$$

$$i_3 \sim \sum_{i_1} \sum_{i_2} P_{i_0, i_1} P_{i_1 i_2} P_{i_2 i_3} = [P^3]_{i_0 i_3},$$

...

$$i_t \sim [P^t]_{i_0 i_t}$$



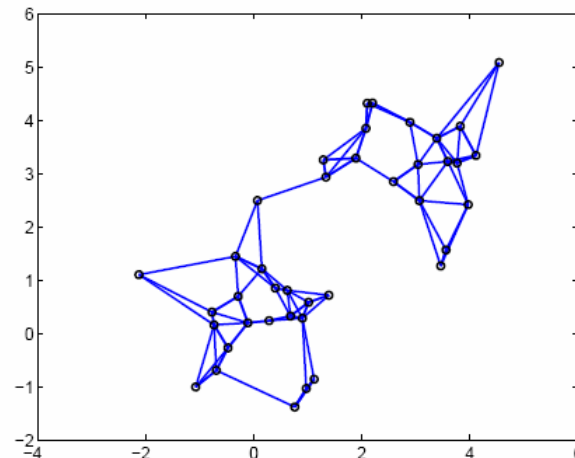
where  $P^t = PP \dots P$  ( $t$  matrix products) and  $[\cdot]_{ij}$  denotes the  $i, j$  component of the matrix.



# Random walk and clustering

- The distributions of points we end up in after  $t$  steps converge as  $t$  increases. If the graph is connected, the resulting distribution is independent of the starting point

Even for large  $t$ , the transition probabilities  $[P^t]_{ij}$  have a slightly higher probability of transitioning within “clusters” than across; we want to recover this effect from eigenvalues/vectors



# Eigenvalues/vectors and spectral clustering



- Let  $W$  be the matrix with components  $W_{ij}$  and  $D$  a diagonal matrix such that  $D_{ii} = \sum_j W_{ij}$ . Then

$$P = D^{-1}W$$

- To find out how  $P^t$  behaves for large  $t$  it is useful to examine the eigen-decomposition of the following symmetric matrix

$$D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = \lambda_1\mathbf{z}_1\mathbf{z}_1^T + \lambda_2\mathbf{z}_2\mathbf{z}_2^T + \dots + \lambda_n\mathbf{z}_n\mathbf{z}_n^T$$

where the ordering is such that  $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n|$ .



# Eigenvalues/vectors cont'd

- The symmetric matrix is related to  $P^t$  since

$$(D^{-\frac{1}{2}}WD^{-\frac{1}{2}}) \dots (D^{-\frac{1}{2}}WD^{-\frac{1}{2}}) = D^{\frac{1}{2}}(P \dots P)D^{-\frac{1}{2}}$$

This allows us to write the  $t$  step transition probability matrix in terms of the eigenvalues/vectors of the symmetric matrix

$$\begin{aligned} P^t &= D^{-\frac{1}{2}} \left( D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \right)^t D^{\frac{1}{2}} \\ &= D^{-\frac{1}{2}} \left( \lambda_1^t \mathbf{z}_1 \mathbf{z}_1^T + \lambda_2^t \mathbf{z}_2 \mathbf{z}_2^T + \dots + \lambda_n^t \mathbf{z}_n \mathbf{z}_n^T \right) D^{\frac{1}{2}} \end{aligned}$$

where  $\lambda_1 = 1$  and

$$P^\infty = D^{-\frac{1}{2}} \left( \mathbf{z}_1 \mathbf{z}_1^T \right) D^{\frac{1}{2}}$$

# Eigenvalues/vectors and spectral clustering



- We are interested in the largest correction to the asymptotic limit

$$P^t \approx P^\infty + D^{-\frac{1}{2}} \left( \lambda_2^t \mathbf{z}_2 \mathbf{z}_2^T \right) D^{\frac{1}{2}}$$

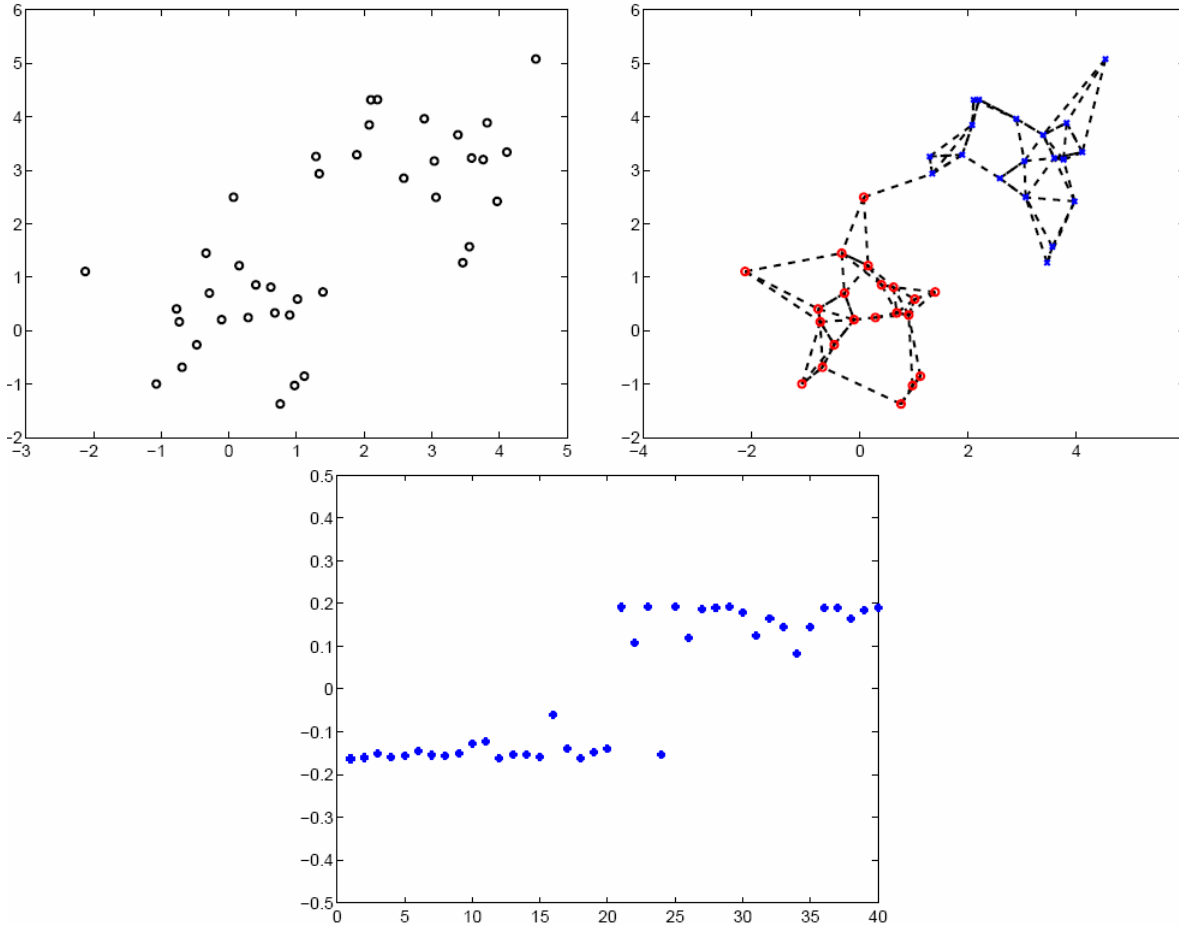
Note:  $[\mathbf{z}_2 \mathbf{z}_2^T]_{ij} = z_{2i} z_{2j}$  and thus the largest correction term increases the probability of transitions between points that share the same sign of  $z_{2i}$  and decreases transitions across points with different signs

- Binary spectral clustering: we divide the points into clusters based on the sign of the elements of  $\mathbf{z}_2$

$$z_{2j} > 0 \Rightarrow \text{cluster 1, otherwise cluster 0}$$



# Spectral clustering: example



Components of the eigenvector corresponding to the second largest eigenvalue



# Reference papers of SC

- A. Y. Ng, M. I. Jordan, and Y. Weiss, *On spectral clustering: Analysis and an algorithm*, NIPS, (2001)
- Y. Weiss, *Segmentation using eigenvectors: a unifying view*. ICCV, (1999)
- J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE TPAMI, 22 (2000)
- And more about image segmentations ...
  - Graph cut
  - Mean-shift





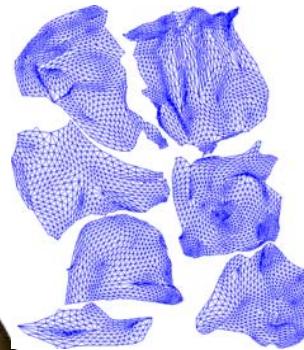
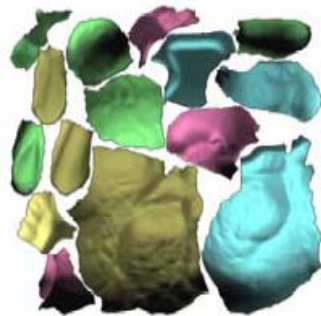
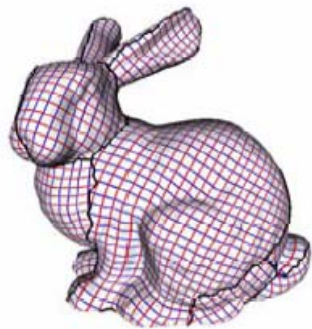
# An example: ISO/BLE-charts

- ISO-Charts:

- ISOMAP + Spectral Clustering + Stretch Minimization

- BLE-Charts:

- Statistical Embedding + Spectral Clustering + Stretch Minimization



它山之石，可以为错。它山之石，可以攻玉