# 8. Clustering (cont.)

# EM Algorithm for GMM

1. Initialize means $\mu_k$ covariance matrices $\Sigma_k$ and mixing coefficients $\pi_k$
2. Compute the initial log-likelihood $\log p(X \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma)$
3. **E-Step.** Compute the responsibilities:

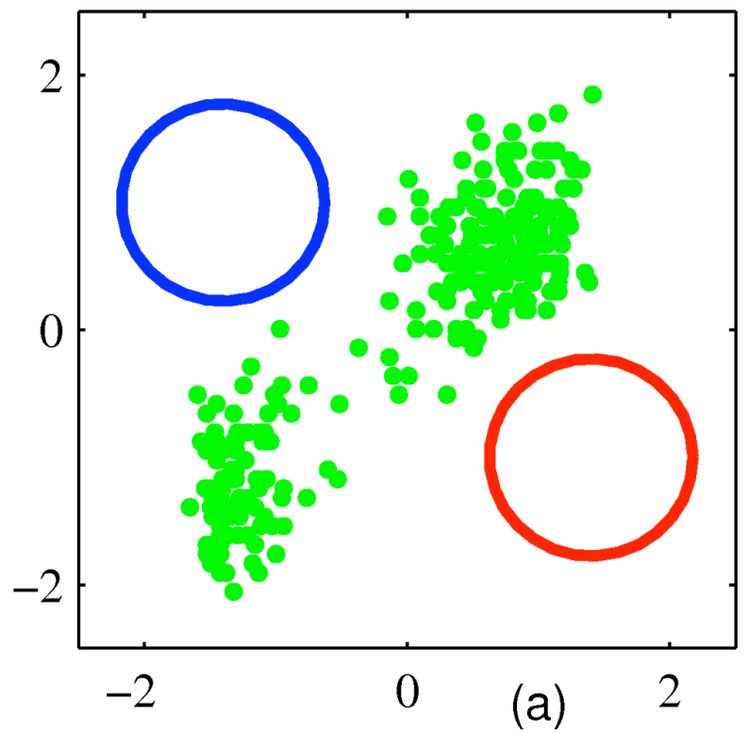
$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \Sigma_j)}$$

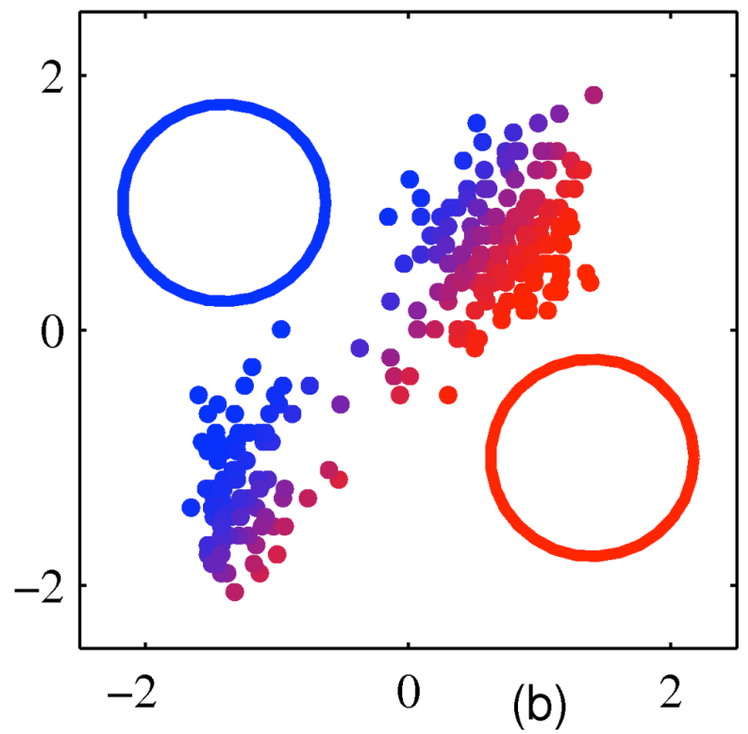
4. **M-Step.** Update the parameters:

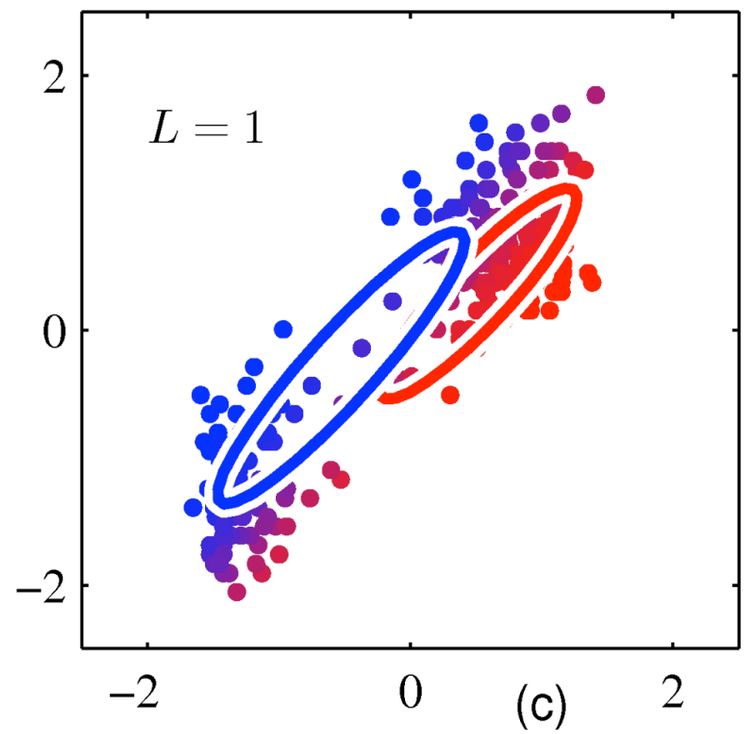
$$\boldsymbol{\mu}_k^{\text{new}} = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^{N} \gamma(z_{nk})} \quad \Sigma_k^{\text{new}} = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^T}{\sum_{n=1}^{N} \gamma(z_{nk})} \quad \pi_k^{\text{new}} = \frac{1}{N} \sum_{n=1}^{N} \gamma(z_{nk})$$

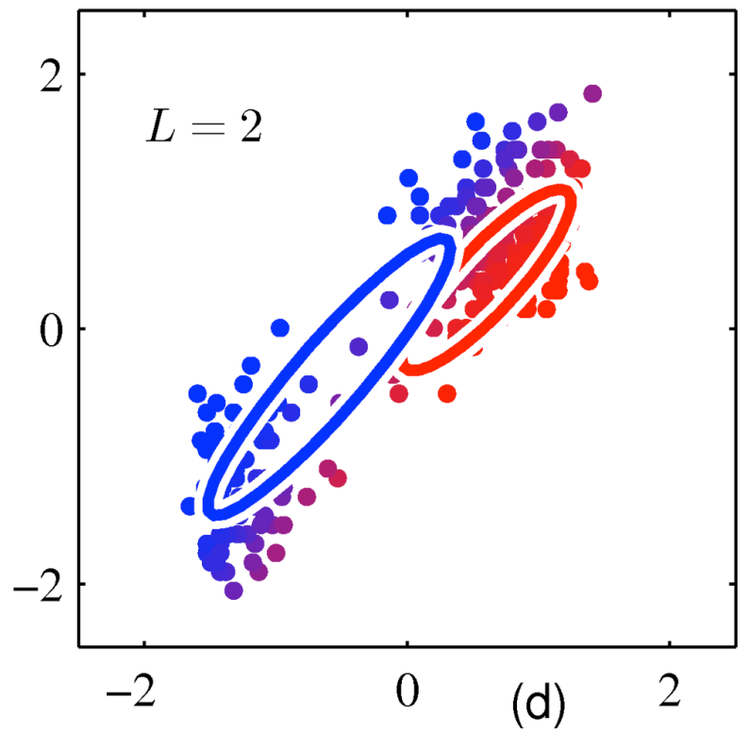
5. Compute log-likelihood; if not converged go to 3.

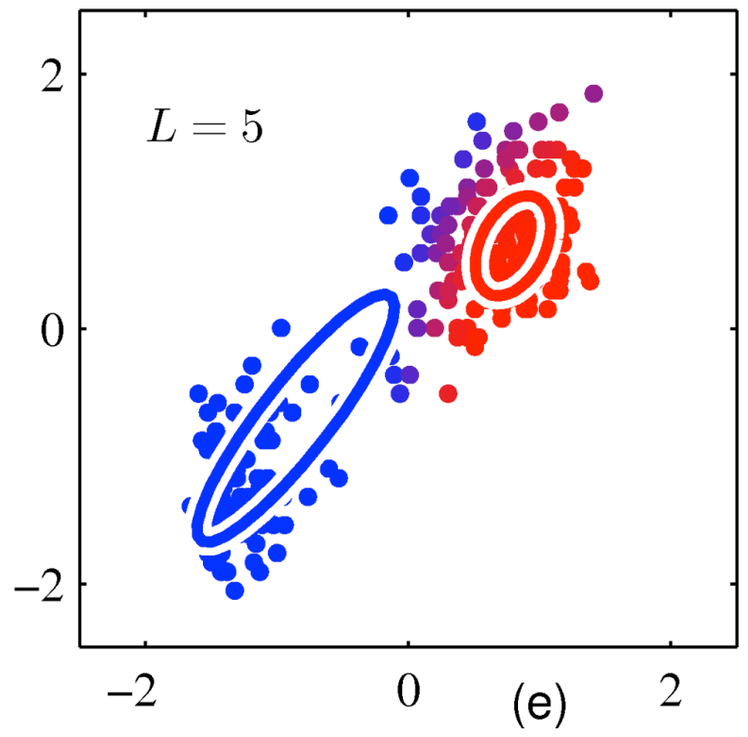
# EM for GMM: Example

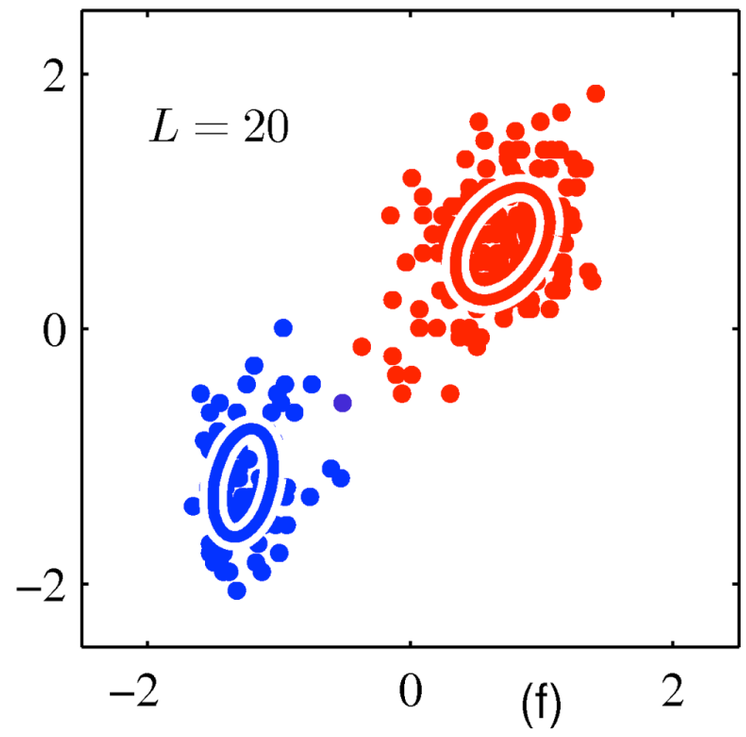


# Why is it Called “EM”?

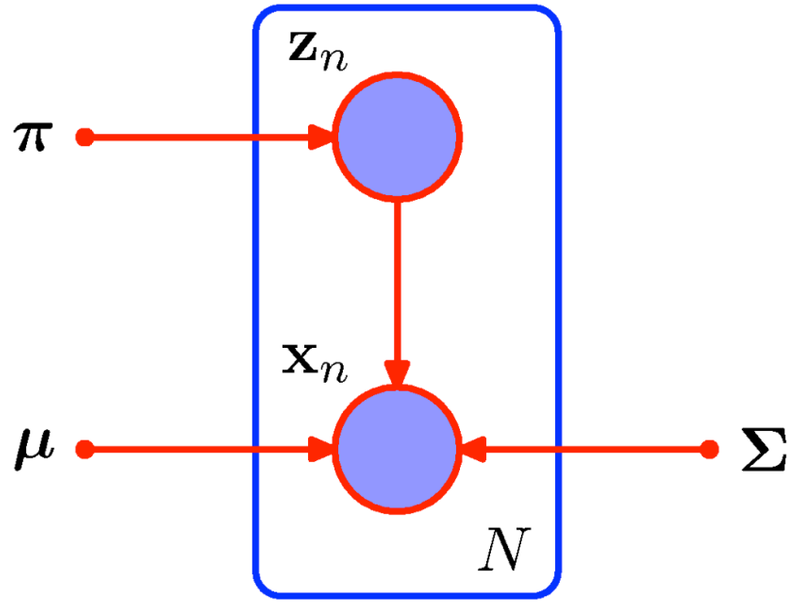
Assume for a moment that we observe $X$ and the binary latent variables $Z$. The likelihood is then:

$$p(X, Z \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma) = \prod_{n=1}^{N} p(\mathbf{z}_n \mid \boldsymbol{\pi}) p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\mu}, \Sigma)$$

**“Complete-data log-likelihood”**

where $p(\mathbf{z}_n \mid \boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{z_{nk}}$ and

$$p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\mu}, \Sigma) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k)^{z_{nk}}$$




which leads to the log-formulation:

$$\log p(X, Z \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} (\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k))$$

# Why is it Called “EM”?

Instead of maximizing the joint log-likelihood, we maximize its **expectation** under the latent variable distribution:

$$\mathbb{E}_Z[\log p(X, Z \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma)] = \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}_Z[z_{nk}](\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k))$$

# Why is it Called “EM”?

Instead of maximizing the joint log-likelihood, we maximize its **expectation** under the latent variable distribution:

$$\mathbb{E}_Z[\log p(X, Z \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma)] = \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}_Z[z_{nk}](\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \Sigma_k))$$

where the latent variable distribution per point is:

$$p(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\theta}) = \frac{p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\theta}) p(\mathbf{z}_n \mid \boldsymbol{\theta})}{p(\mathbf{x}_n \mid \boldsymbol{\theta})} \qquad \boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \Sigma)$$

$$= \frac{\prod_{l=1}^{K} (\pi_l \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_l, \Sigma_l))^{z_{nl}}}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \Sigma_j)}$$

# Observations

- Compared to K-means, points can now belong to both clusters (**soft assignment**)
- In addition to the cluster center, a covariance is estimated by EM
- Initialization is the same as used for K-means
- Number of iterations needed for EM is much higher
- Also: each cycle requires much more computation
- Therefore: start with K-means and run EM on the result of K-means (covariances can be initialized to the sample covariances of K-means)
- EM only finds a **local** maximum of the likelihood!

# Rep.: From MLE to MAP (Regression)

In MLE, we searched for parameters $\mathbf{w}$, that maximize the data likelihood. Now, we assume a Gaussian ***prior***:

$$p(\mathbf{w} \mid \sigma_2) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \sigma_2 I)$$

Using this, we can compute the ***posterior*** (Bayes):

$$p(\mathbf{w} \mid \mathbf{x}, \mathbf{t}) \propto p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}) p(\mathbf{w})$$

Posterior — Likelihood — Prior

**“Maximum A-Posteriori Estimation (MAP)”**

# Generalization: The Bayesian Approach

This idea can be generalized:

- Given a data-dependent **likelihood** term
- Find an appropriate **prior** distribution
- Multiply both and obtain the (unnormalized) **posterior** from Bayes rule

- Main benefit: less overfitting

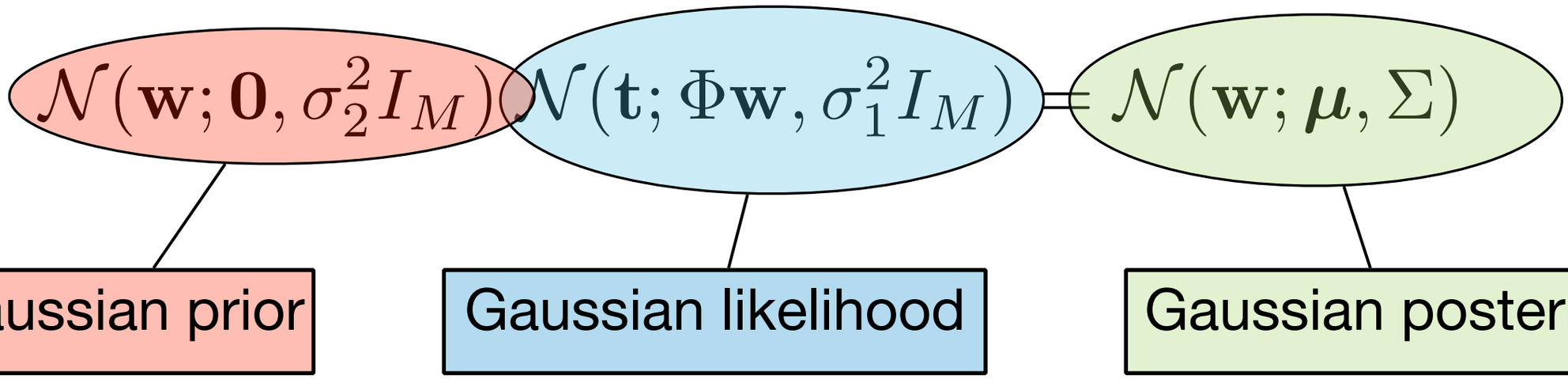
However:

- How should we define the prior?

Often used principle: **Conjugacy**

# Conjugate Priors

A conjugate prior distribution allows to represent the posterior in the **same functional** (closed) form as the prior, e.g.:




Common pairs of likelihood and conjugate priors are:

| Likelihood | Conjugate Prior |
|---|---|
| Normal with known variance | Normal |
| Binomial | Beta |
| Multinomial | Dirichlet |
| Multivariate Normal | Normal-inverse Wishart |

# Multinomial

- Given $K$ clusters and probabilities of these clusters $\pi_1, \ldots, \pi_K$ where $\sum_{k=1}^{K} \pi_k = 1$
- The probability that out of $N$ samples $m_k$ are in cluster $k$ is:

$$p(m_1, \ldots, m_K \mid \boldsymbol{\pi}, N) = \binom{N}{m_1 \cdots m_K} \prod_{k=1}^{K} \mu_k^{m_k}$$

- This is called the **multinomial distribution**
- In our case:

$$p(Z \mid \boldsymbol{\pi}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mu_k^{z_{nk}} = \prod_{k=1}^{K} \mu_k^{m_k}$$

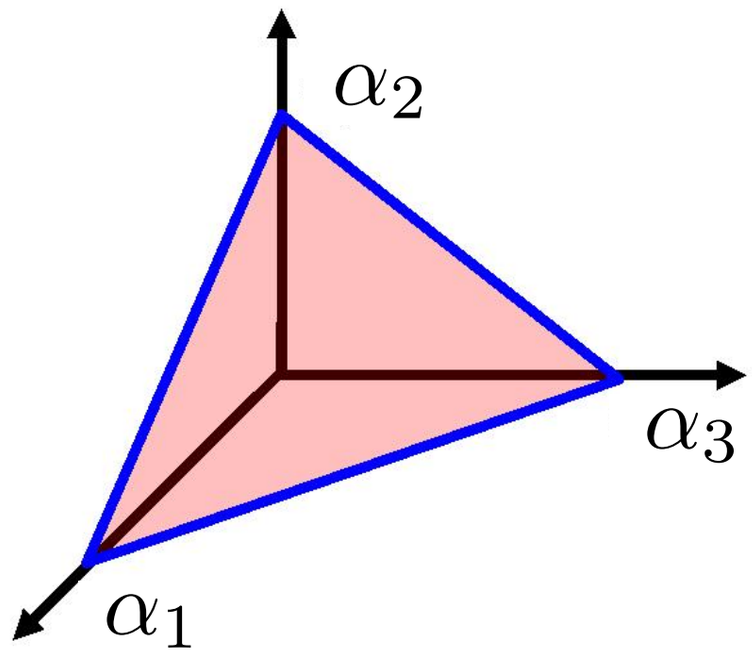
# The Dirichlet Distribution

- The Dirichlet distribution is defined as:

$$\mathrm{Dir}(\boldsymbol{\mu} \mid \boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \qquad \alpha_0 = \sum_{k=1}^{K} \alpha_k$$

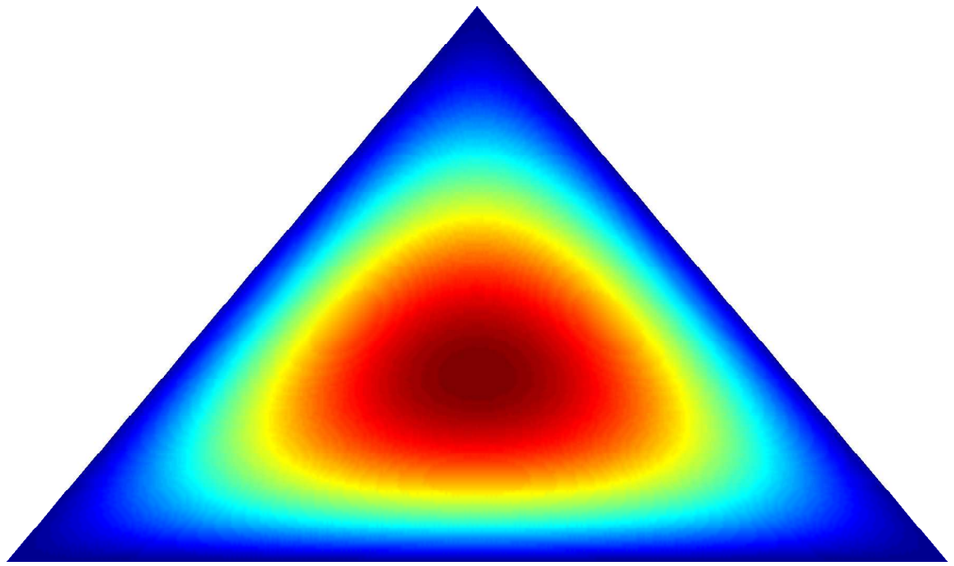
$$0 \leq \mu_k \leq 1 \qquad \sum_{k=1}^{K} \mu_k = 1$$

- It is the conjugate prior for the multinomial distribution
- There, the parameter $\alpha$ can be interpreted as the effective number of observations for every state

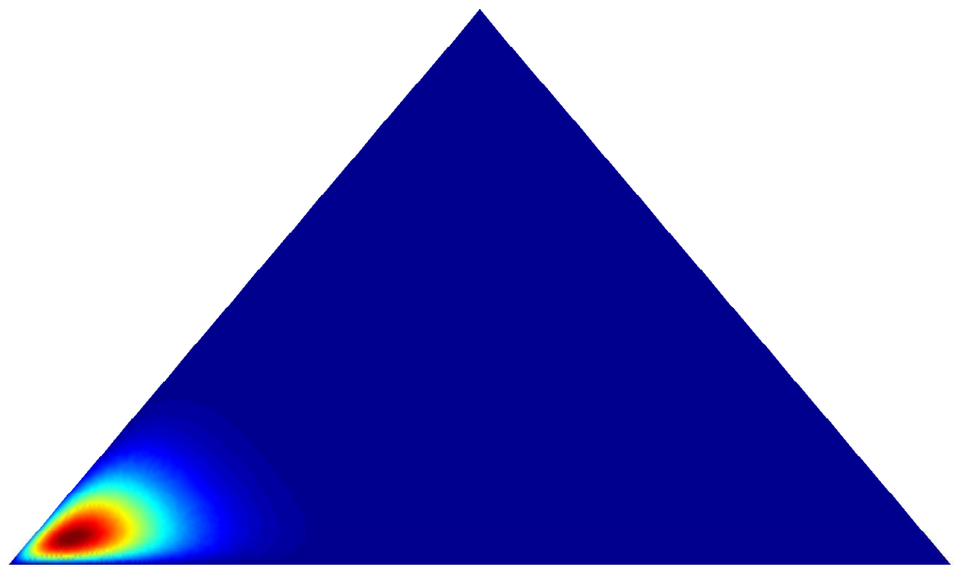



The simplex for K=3

# Some Examples

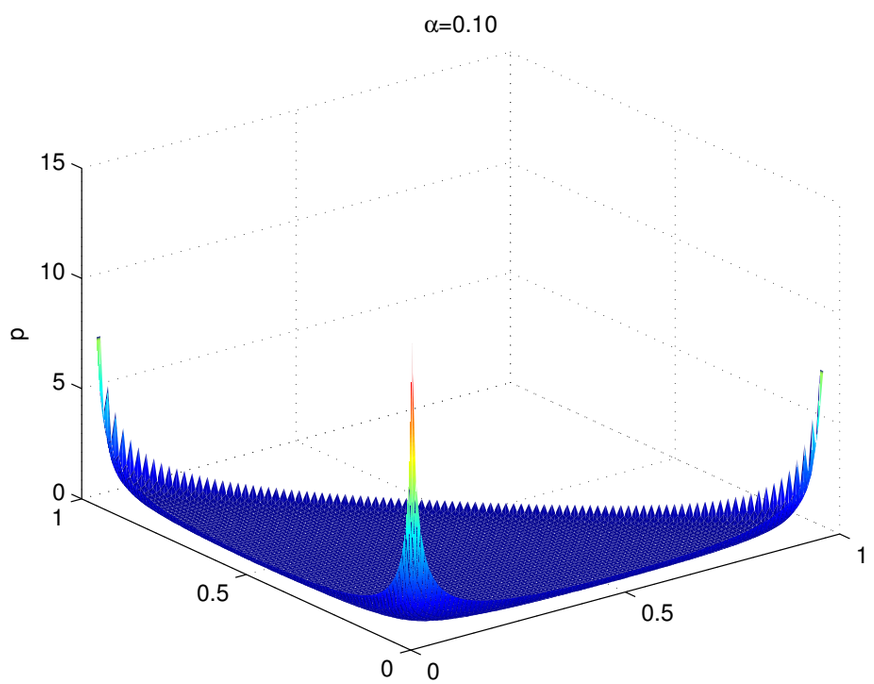


$\alpha = (2, 2, 2)$



$\alpha = (20, 2, 2)$

- $\alpha_0$ controls the strength of the distribution (“peakedness”)
- $\alpha_k$ control the location of the peak



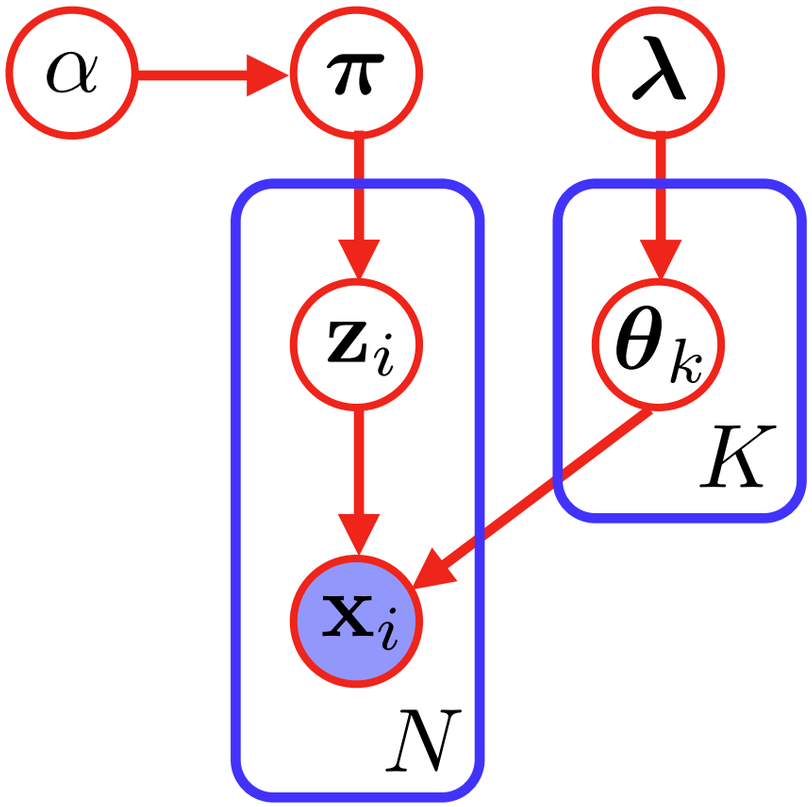

$\alpha = (0.1, 0.1, 0.1)$

# Clustering using Mixture Models

- The full posterior of the Gaussian Mixture Model is

$$p(X, Z, \boldsymbol{\mu}, \Sigma, \boldsymbol{\pi}) = p(X \mid Z, \boldsymbol{\mu}, \Sigma) p(Z \mid \boldsymbol{\pi}) p(\boldsymbol{\pi} \mid \alpha) p(\boldsymbol{\mu}, \Sigma \mid \boldsymbol{\lambda})$$

| data likelihood (Gaussian) | correspondence prob. (Multinomial) | mixture prior (Dirichlet) | parameter prior (Gauss-IW) |
|---|---|---|---|




Given this model, we can create new samples:

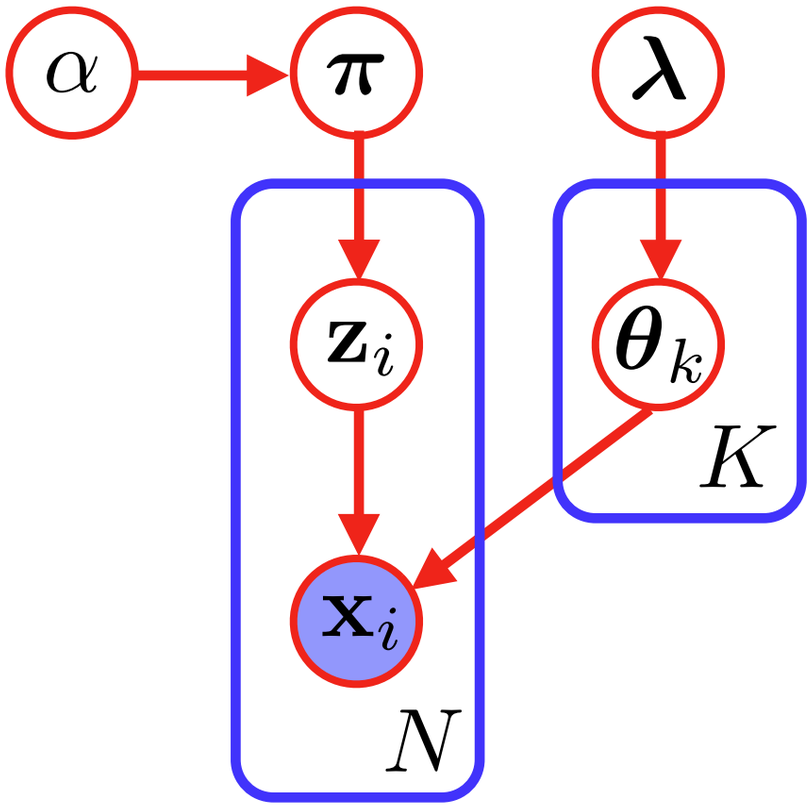
1. Sample $\boldsymbol{\pi}, \boldsymbol{\theta}_k$ from priors
2. Sample corresp. $\mathbf{z}_i$
3. Sample data point $\mathbf{x}_i$

# Clustering using Mixture Models

- The full posterior of the Gaussian Mixture Model is

$$p(X, Z, \boldsymbol{\mu}, \Sigma, \boldsymbol{\pi}) = p(X \mid Z, \boldsymbol{\mu}, \Sigma) p(Z \mid \boldsymbol{\pi}) p(\boldsymbol{\pi} \mid \alpha) p(\boldsymbol{\mu}, \Sigma \mid \boldsymbol{\lambda})$$

| data likelihood (Gaussian) | correspondence prob. (Multinomial) | mixture prior (Dirichlet) | parameter prior (Gauss-IW) |
|---|---|---|---|




$$\boldsymbol{\pi} \sim \mathrm{Dir}(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K})$$

$$\mathbf{z}_i \sim \mathrm{Mult}(\boldsymbol{\pi})$$

$$\boldsymbol{\theta}_k \sim \mathrm{NIW}(\boldsymbol{\lambda})$$

$$\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\theta}_{\mathbf{z}_i})$$

# Clustering using Mixture Models

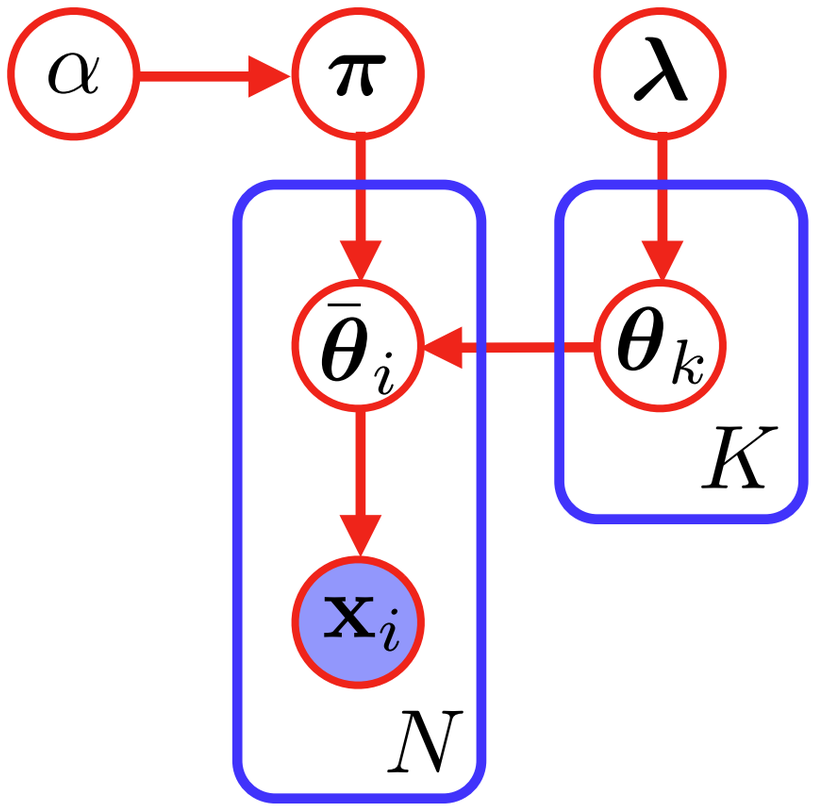
- The full posterior of the Gaussian Mixture Model is

$$p(X, Z, \boldsymbol{\mu}, \Sigma, \boldsymbol{\pi}) = p(X \mid Z, \boldsymbol{\mu}, \Sigma) p(Z \mid \boldsymbol{\pi}) p(\boldsymbol{\pi} \mid \alpha) p(\boldsymbol{\mu}, \Sigma \mid \boldsymbol{\lambda})$$

| data likelihood (Gaussian) | correspondence prob. (Multinomial) | mixture prior (Dirichlet) | parameter prior (Gauss-IW) |
|---|---|---|---|

An equivalent formulation of this model is this:

1. Sample $\pi, \theta_k$ from priors
2. Sample params $\bar{\theta}_i$ from a discrete dist. G
3. Sample data point $\mathbf{x}_i$



# Clustering using Mixture Models

What is the difference in that model?

- there is one parameter $\bar{\theta}_i$ for each observation $\mathbf{x}_i$
- intuitively: we first sample the location of the cluster and then the data that corresponds to it

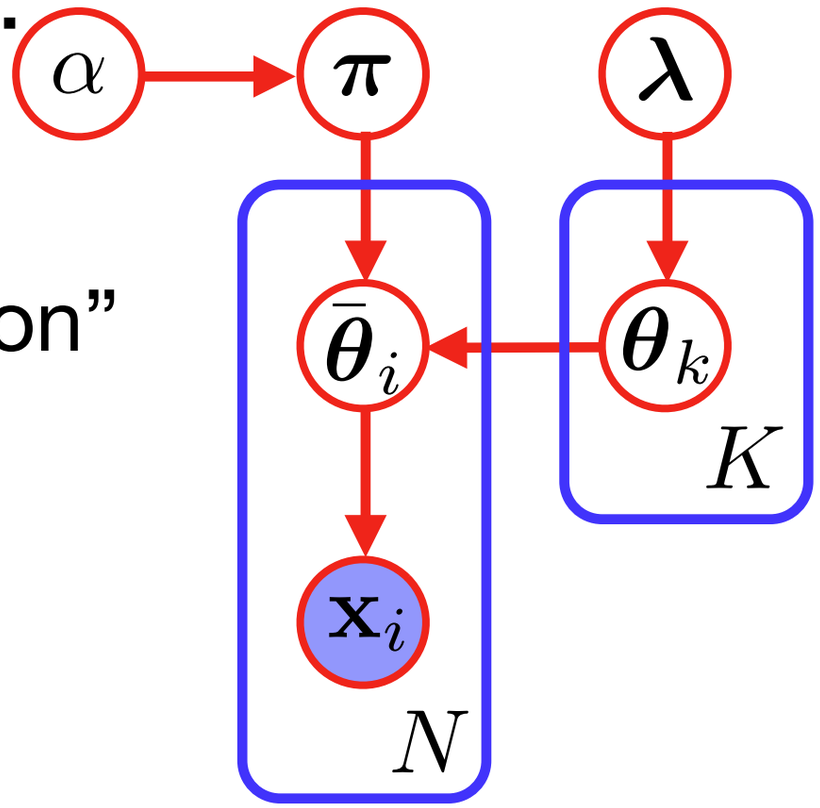
In general, we use the notation:

$$\boldsymbol{\pi} \sim \mathrm{Dir}(\frac{\alpha}{K}\mathbf{1})$$

$$\boldsymbol{\theta}_k \sim \mathrm{H}(\boldsymbol{\lambda}) \quad \text{“Base distribution”}$$

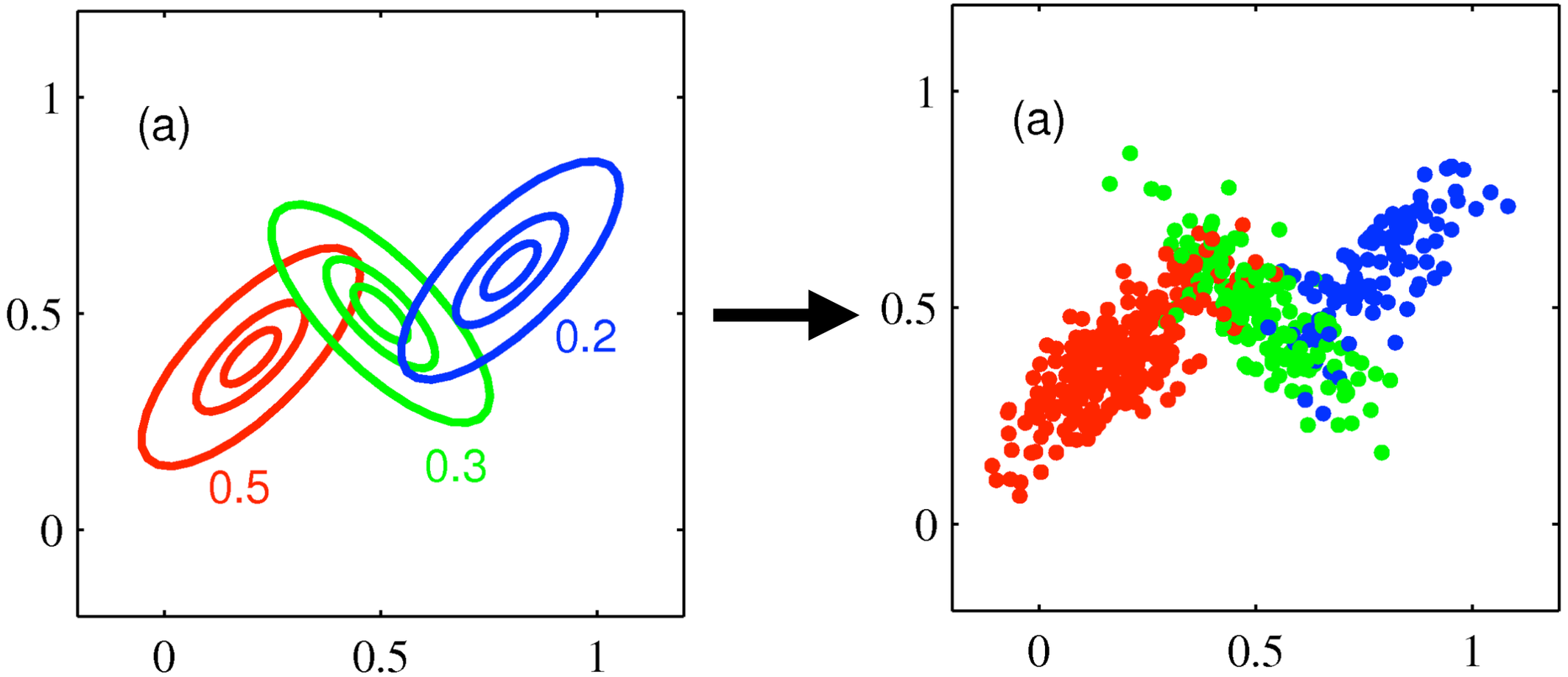
$$\bar{\boldsymbol{\theta}}_i \sim \mathrm{G}(\boldsymbol{\pi}, \boldsymbol{\theta}_k) \quad \text{where}$$

$$G(\boldsymbol{\pi}, \boldsymbol{\theta}_k) = \sum_{k=1}^{K} \pi_k \delta(\boldsymbol{\theta}_k, \bar{\boldsymbol{\theta}}_i)$$




**However: We need to know K**

# Remember: Generating GMM Data



# The Dirichlet Process

- So far, we assumed that K is known
- To extend that to infinity, we use a trick:

**Definition:** A Dirichlet process (DP) is a distribution over probability measures $G$, i.e. $G(\boldsymbol{\theta}) \geq 0$ and $\int G(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1$. If for any partition $(T_1, \ldots, T_K)$ it holds:

$$(G(T_1), \ldots, G(T_K)) \sim \mathrm{Dir}(\alpha H(T_1), \ldots, \alpha H(T_K))$$

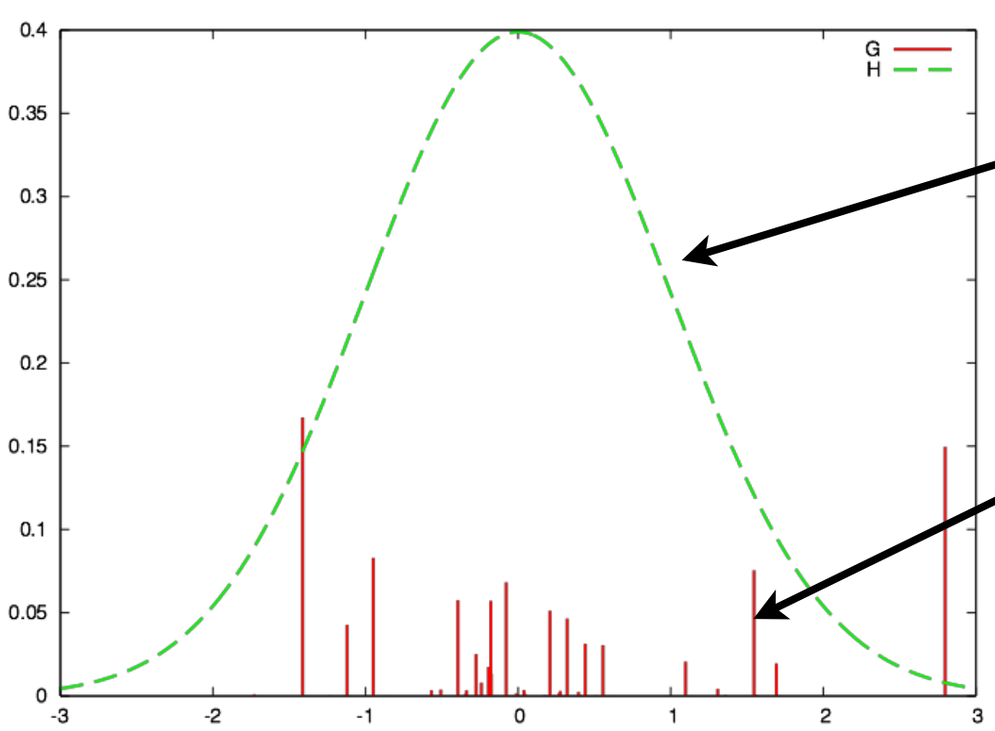
then $G$ is sampled from a Dirichlet process.

**Notation:** $G \sim \mathrm{DP}(\alpha, H)$

where $\alpha$ is the **concentration parameter** and $H$ is the **base measure**

# Intuitive Interpretation

- Every sample from a Dirichlet distribution is a vector of $K$ positive values that sum up to 1, i.e. the sample itself is a finite distribution
- Accordingly, a sample from a Dirichlet process is an infinite (but still discrete!) distribution



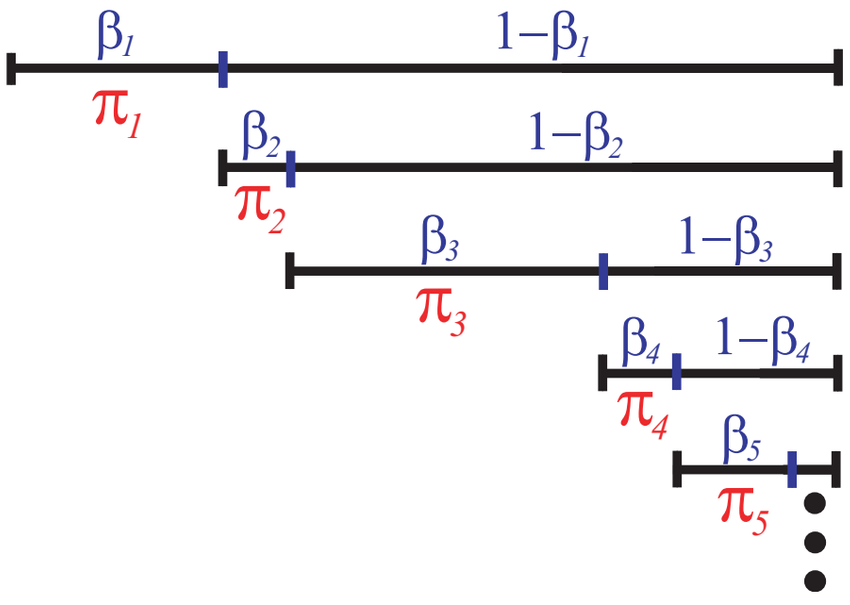

Base distribution (here Gaussian)

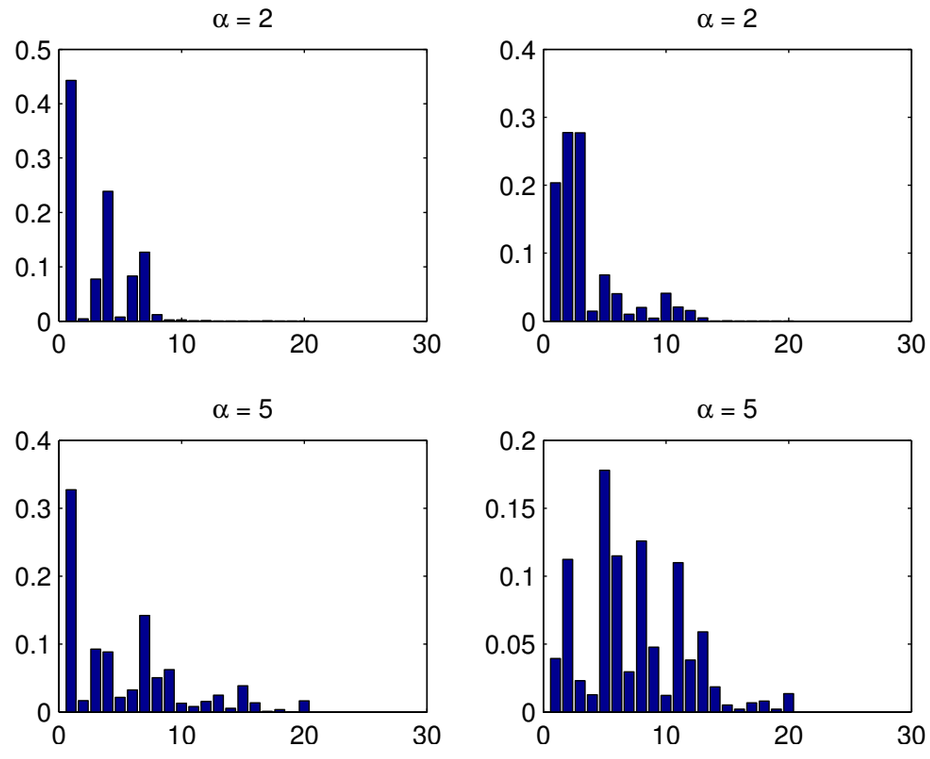
Infinitely many samples (sum up to 1)

# Construction of a Dirichlet Process

- The Dirichlet process is only defined **implicitly**, i.e. we can test whether a given probability measure is sampled from a DP, but we can not yet construct one.
- A DP can be constructed using the “stick-breaking” analogy:
  - imagine a stick of length 1
  - we select a random number $\beta$ between 0 and 1 from a Beta-distribution
  - we break the stick at π = β * length-of-stick
  - we repeat this infinitely often

# The Stick-Breaking Construction



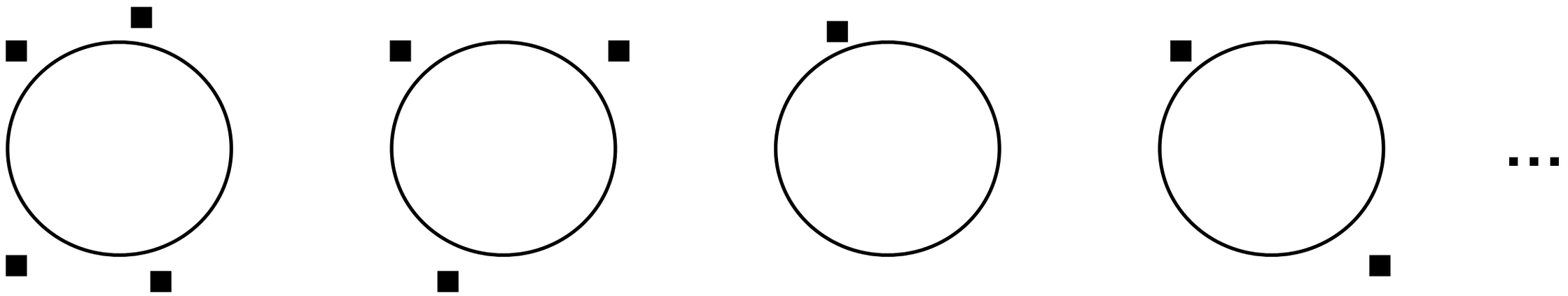

- formally, we have

$$\beta_k \sim \mathrm{Beta}(1, \alpha) \qquad \pi_k = \beta_k \prod_{l=1}^{k-1}(1-\beta_l) = \beta_k(1 - \sum_{l=1}^{k-1}\pi_l)$$

- now we define

$$G(\boldsymbol{\theta}) = \sum_{k=1}^{\infty} \pi_k \delta(\boldsymbol{\theta}_k, \boldsymbol{\theta}) \qquad \boldsymbol{\theta}_k \sim H \qquad \text{then:} \quad G \sim \mathrm{DP}(\alpha, H)$$

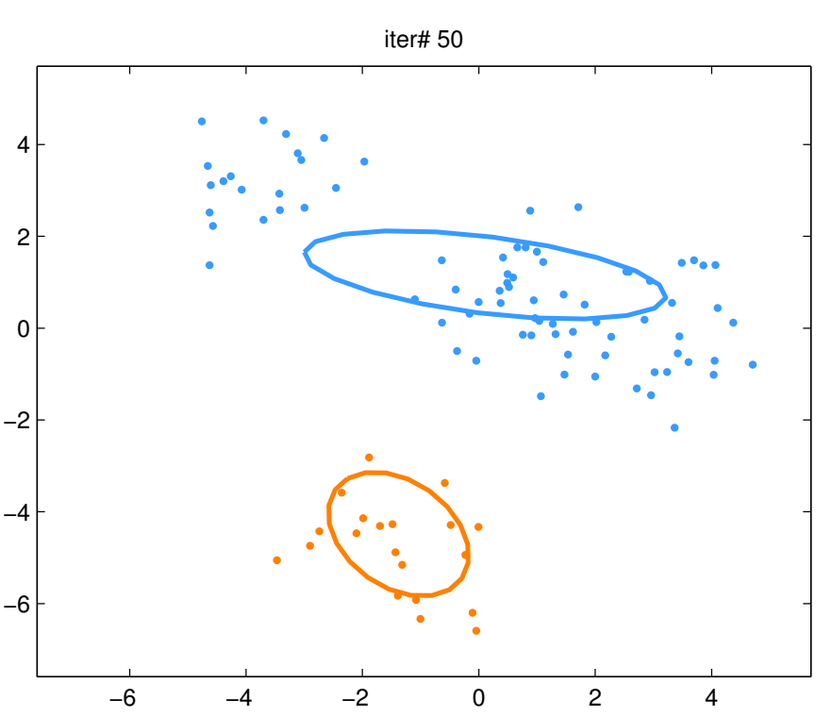
# The Chinese Restaurant Process

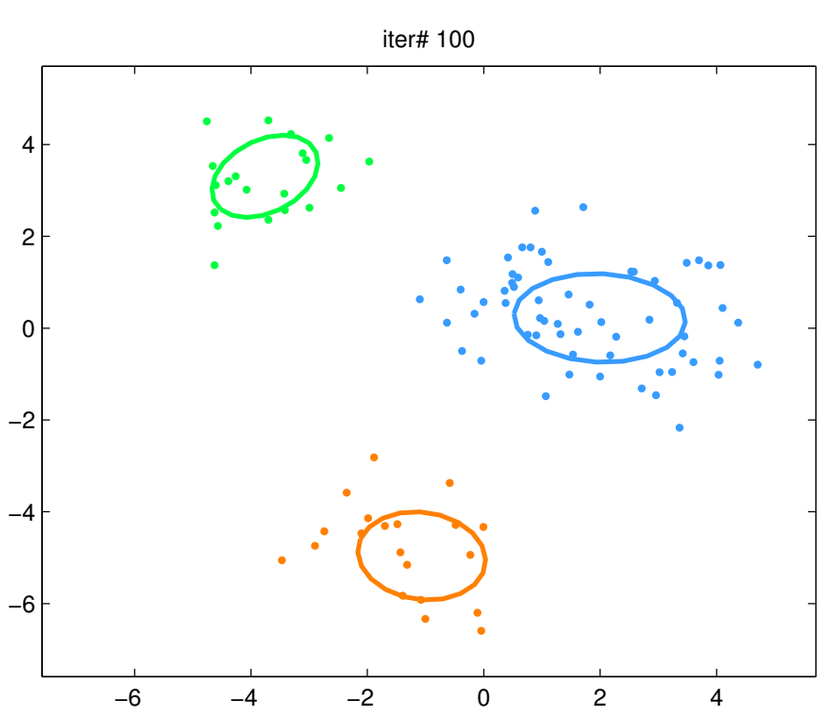


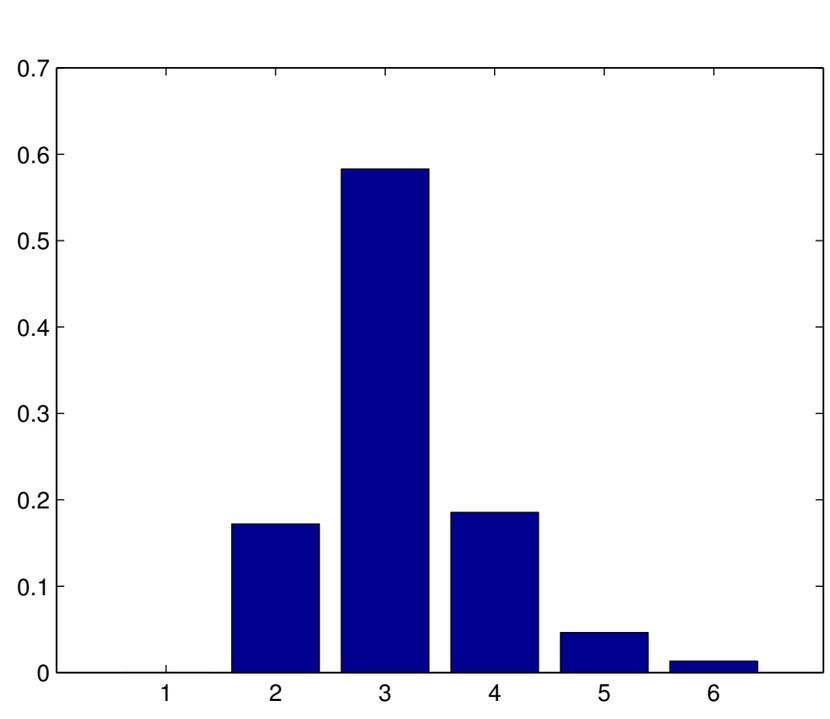
- Consider a restaurant with infinitely many tables
- Everytime a new customer comes in, he sits at an **occupied table** with probability **proportional to the number of people** sitting at that table, but he may choose to sit on a **new** table with **decreasing** probability as more customers enter the room.

# The DP for Mixture Modeling

- Using the stick-breaking construction, we see that we can extend the mixture model clustering to the situation where $K$ goes to infinity
- The algorithm can be implemented using Gibbs sampling



# Questions

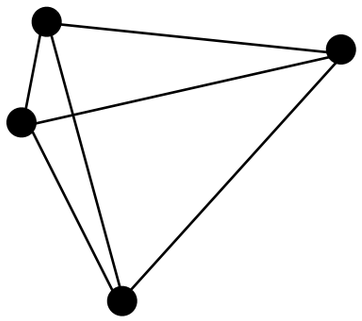
- What if the clusters can not be approximated well by Gaussians?

- Can we formulate an algorithm that only relies on pairwise similarities?

**One example for such an algorithm is Spectral Clustering**

# Spectral Clustering

- Consider an undirected graph that connects all data points
- The edge weights are the similarities (“closeness”)
- We define the weighted degree $d_i$ of a node as the sum of all outgoing edges



$$W = \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

$$d_i = \sum_{j=1}^{N} w_{ij}$$

$$D = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & d_3 & \\ & & & d_4 \end{bmatrix}$$

# Spectral Clustering

- The Graph Laplacian is defined as:

$$L = D - W$$

- This matrix has the following properties:
  - the 1 vector is eigenvector with eigenvalue 0

# Spectral Clustering

- The Graph Laplacian is defined as:

$$L = D - W$$

- This matrix has the following properties:
  - the 1 vector is eigenvector with eigenvector 0
  - the matrix is symmetric and positive semi-definite

# Spectral Clustering

- The Graph Laplacian is defined as:

$$L = D - W$$

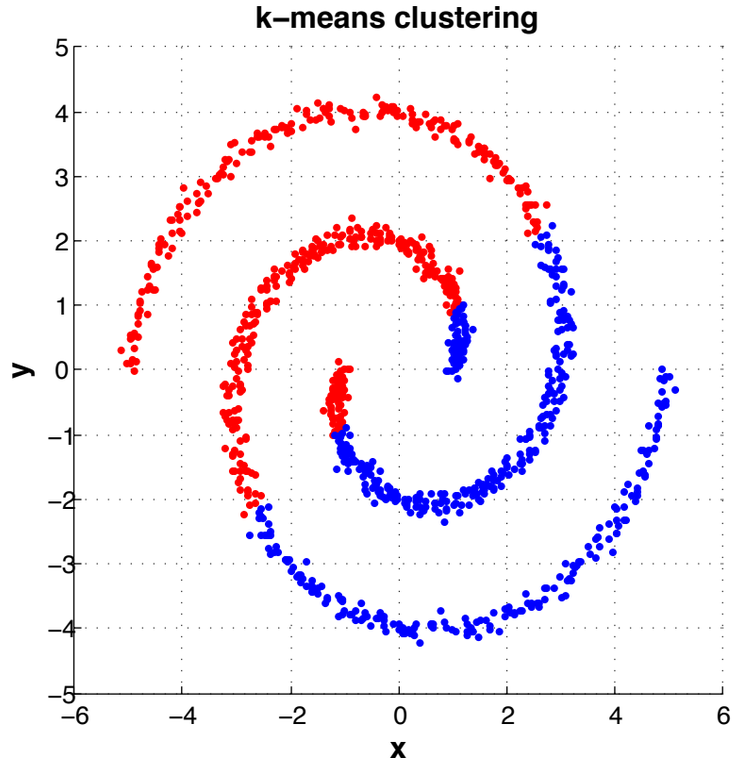
- This matrix has the following properties:
  - the 1 vector is eigenvector with eigenvector 0
  - the matrix is symmetric and positive semi-definite
- With these properties we can show:

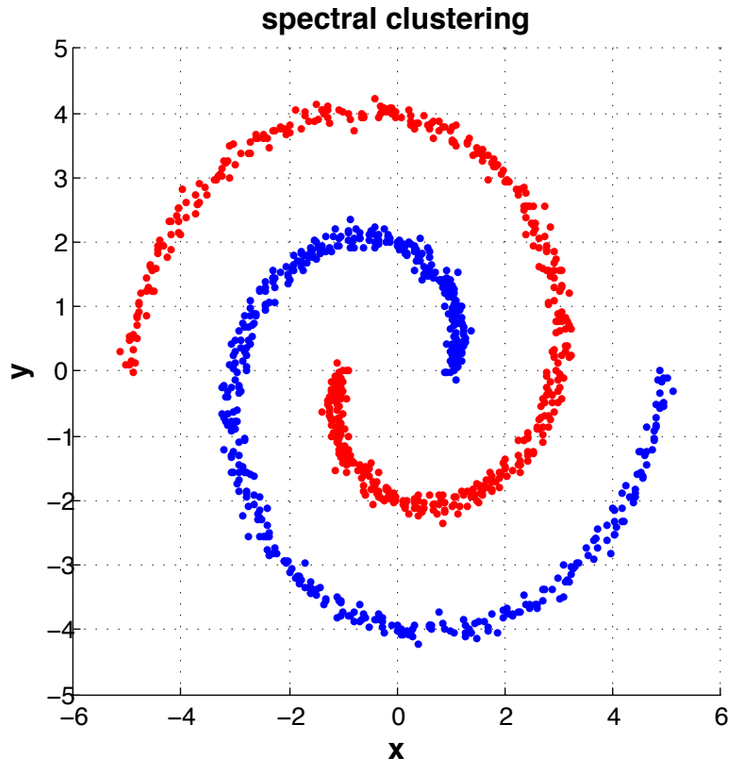
**Theorem:** The set of eigenvectors of $L$ with eigenvalue 0 is spanned by the indicator vectors $1_{A_1}, \ldots, 1_{A_K}$, where $A_k$ are the $K$ connected components of the graph.

# The Algorithm

- Input: Similarity matrix W
- Compute L = D - W
- Compute the eigenvectors that correspond to the K smallest eigenvalues
- Stack these vectors as columns in a matrix U
- Treat each row of U as a K-dim data point
- Cluster the N rows with K-means clustering
- The indices of the rows that correspond to the resulting clusters are those of the original data points.

# An Example



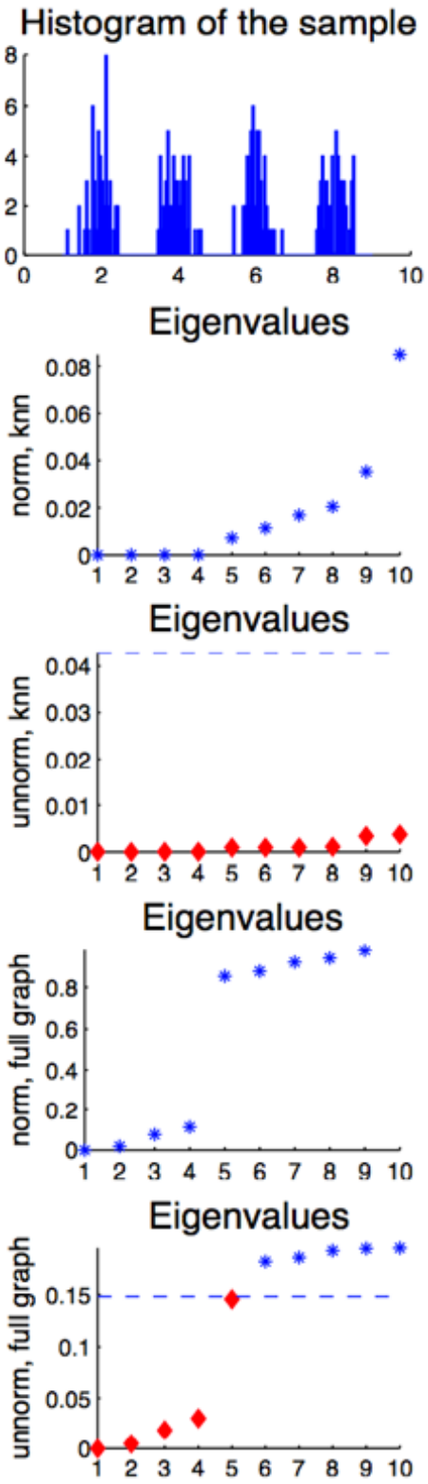




- Spectral clustering can handle complex problems such as this one
- The complexity of the algorithm is $O(N^3)$, because it has to solve an eigenvector problem
- But there are efficient variants of the algorithm

# Further Remarks

- To account for nodes that are highly connected, we can use a normalized version of the graph Laplacian
- Two different methods exist:
  - $L_{rw} = D^{-1}L = I - D^{-1}W$
  - $L_{sym} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$
- These have similar eigenspaces than the original Laplacian L
- Clustering results tend to be better than with the unnormalized Laplacian
- The number of clusters K can be found using the “eigen-gap heuristic”

# Eigen-Gap Heuristic




- Compute all eigen values of the graph Laplacian
- Sort them in increasing order
- Usually, there is a big “jump” between two consecutive eigen values
- The corresponding number $K$ is a good choice for the estimated number of clusters

# Summary

- Several Clustering methods exist:
  - K-means clustering and Expectation-Maximization, both based on Gaussian Mixture Models
  - K-means uses hard assignments, whereas EM uses soft assignments and estimates also the covariances
  - The Dirichlet Process is a non-parametric model to perform clustering without specifying K
  - Spectral clustering uses the graph Laplacian and performs an eigenvector analysis
- Major Problem:
  - most clustering algorithms require the number of clusters to be given