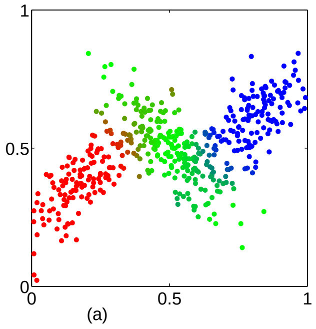
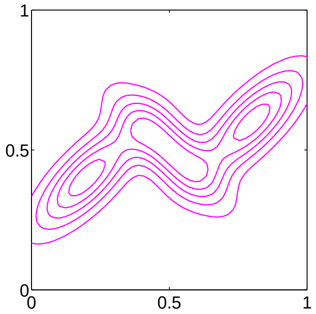
Mixture Models and the EM Algorithm

Christopher M. Bishop

Microsoft Research, Cambridge



2006 Advanced Tutorial
Lecture Series, CUED



Applications of Machine Learning

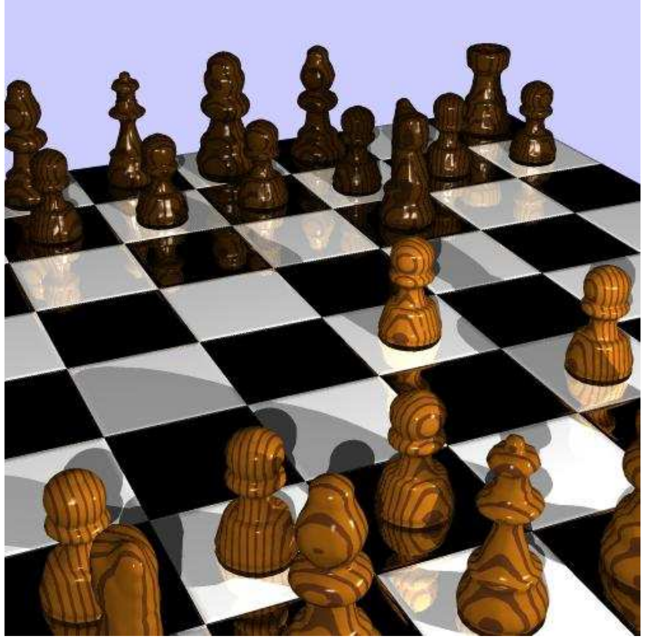
- Web search, email spam detection, collaborative filtering, game player ranking, video games, real-time stereo, protein folding, image editing, jet engine anomaly detection, fluorescence in-situ hybridisation, signature verification, satellite scatterometer, cervical smear screening, human genome analysis, compiler optimization, handwriting recognition, breast X-ray screening, fingerprint recognition, fast spectral analysis, one-touch microwave oven, monitoring of premature babies, text/graphics discrimination, event selection in high energy physics, electronic nose, real-time tokamak control, crash log analysis, QSAR, backgammon, sleep EEG staging, fMRI analysis, speech recognition, natural language processing, face detection, data visualization, computer Go, satellite track removal, iris recognition, …

Three Important Developments

- 1. Adoption of a Bayesian framework
- 2. Probabilistic graphical models
- 3. Efficient techniques for approximate inference

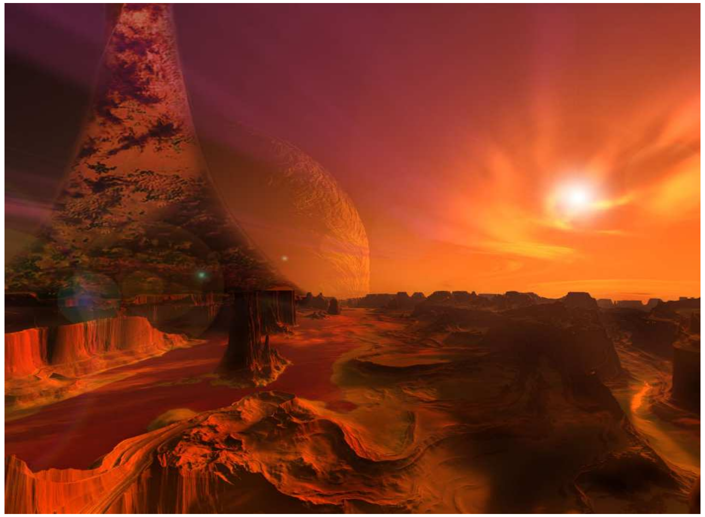
Illustration: Bayesian Ranking

- Goal is to rank player skill from outcome of games
- Conventional approach: Elo (used in chess)
  - maintains a single strength value for each player
  - cannot handle team games, or more than 2 players

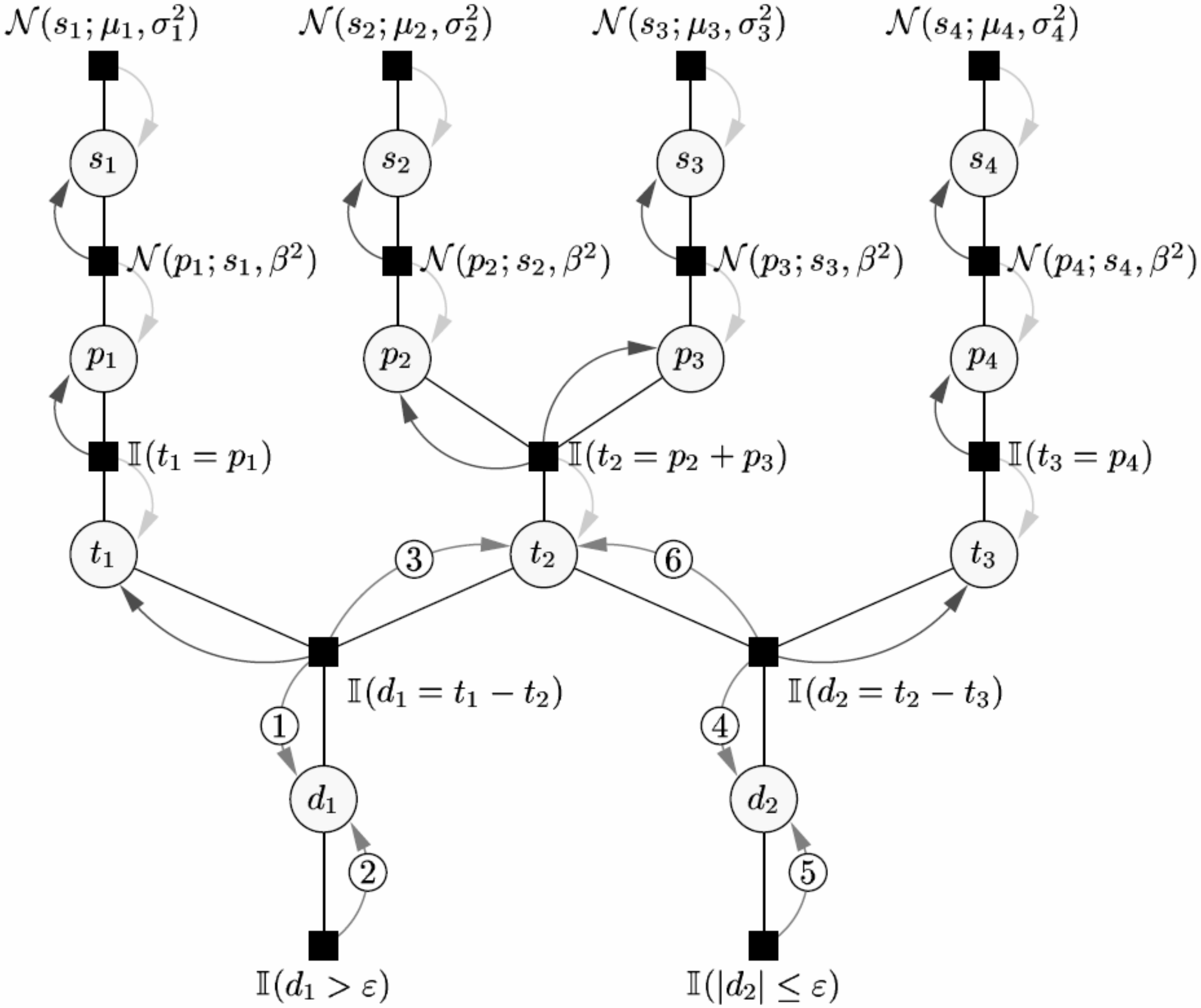


Bayesian Ranking: TrueSkill^{TM}

- Ralf Herbrich, Thore Graepel, Tom Minka
- Xbox 360 Live (November 2005)
  - millions of players
  - billions of service-hours
  - hundreds of thousands of game outcomes per day
- First "planet-scale" application of Bayesian methods?
- NIPS (2006) oral



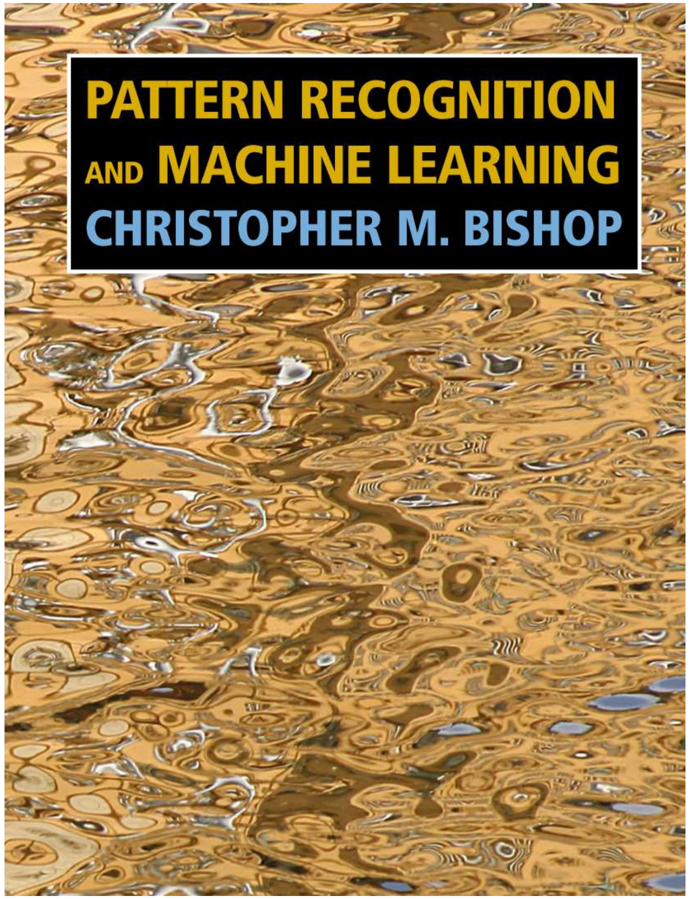
Expectation Propagation on a Factor Graph



$\mathcal{N}(s_1; \mu_1, \sigma_1^2)$
$\mathcal{N}(s_2; \mu_2, \sigma_2^2)$
$\mathcal{N}(s_3; \mu_3, \sigma_3^2)$
$\mathcal{N}(s_4; \mu_4, \sigma_4^2)$
s_1
s_2
s_3
s_4
$\mathcal{N}(p_1; s_1, \beta^2)$
$\mathcal{N}(p_2; s_2, \beta^2)$
$\mathcal{N}(p_3; s_3, \beta^2)$
$\mathcal{N}(p_4; s_4, \beta^2)$
p_1
p_2
p_3
p_4
$\mathbb{I}(t_1 = p_1)$
$\mathbb{I}(t_2 = p_2 + p_3)$
$\mathbb{I}(t_3 = p_4)$
t_1
3
t_2
6
t_3
$\mathbb{I}(d_1 = t_1 - t_2)$
$\mathbb{I}(d_2 = t_2 - t_3)$
1
4
d_1
d_2
2
5
$\mathbb{I}(d_1 > \varepsilon)$
$\mathbb{I}(|d_2| \leq \varepsilon)$

New Book

- Springer (2006)
- 738 pages, hardcover
- Full colour
- Low price
- 431 exercises + solutions
- Matlab software and companion text with Ian Nabney



http://research.microsoft.com/~cmbishop/PRML

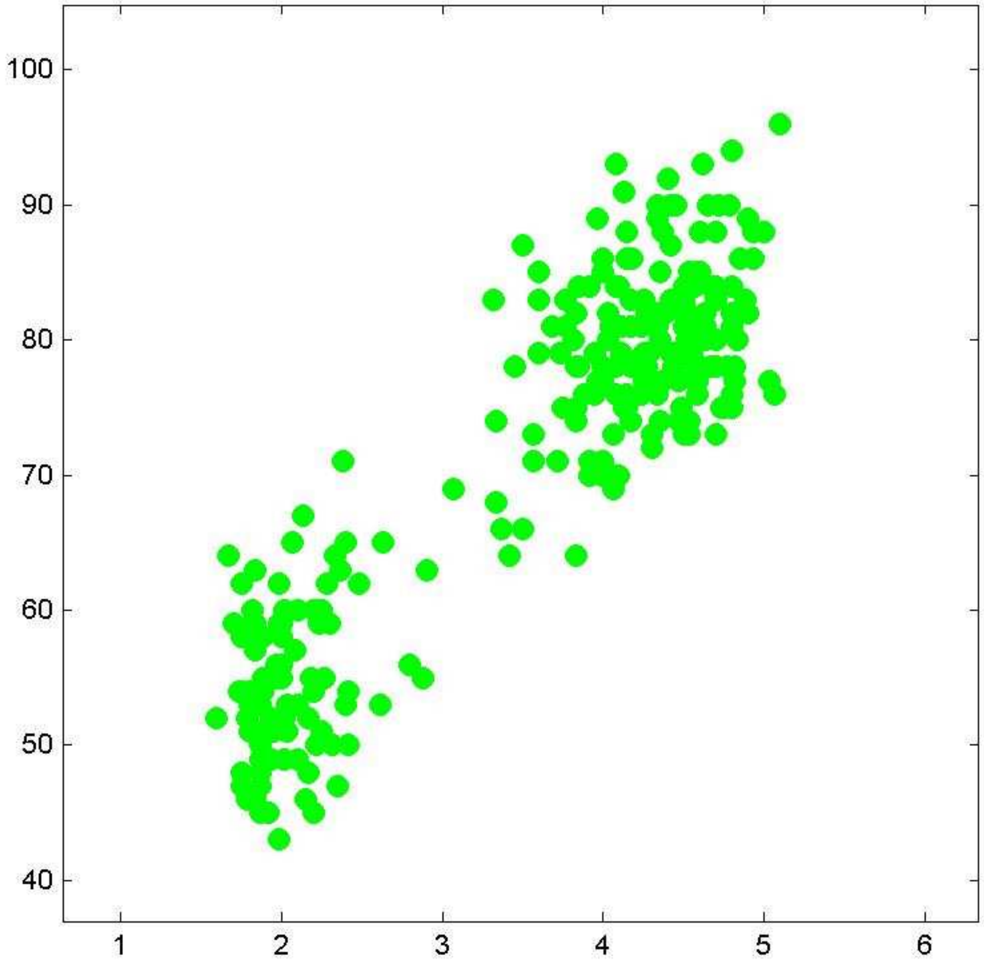
Mixture Models and EM

- K-means clustering
- Gaussian mixture model
- Maximum likelihood and EM
- Bayesian GMM and variational inference

Please ask questions!

Old Faithful

Old Faithful Data Set

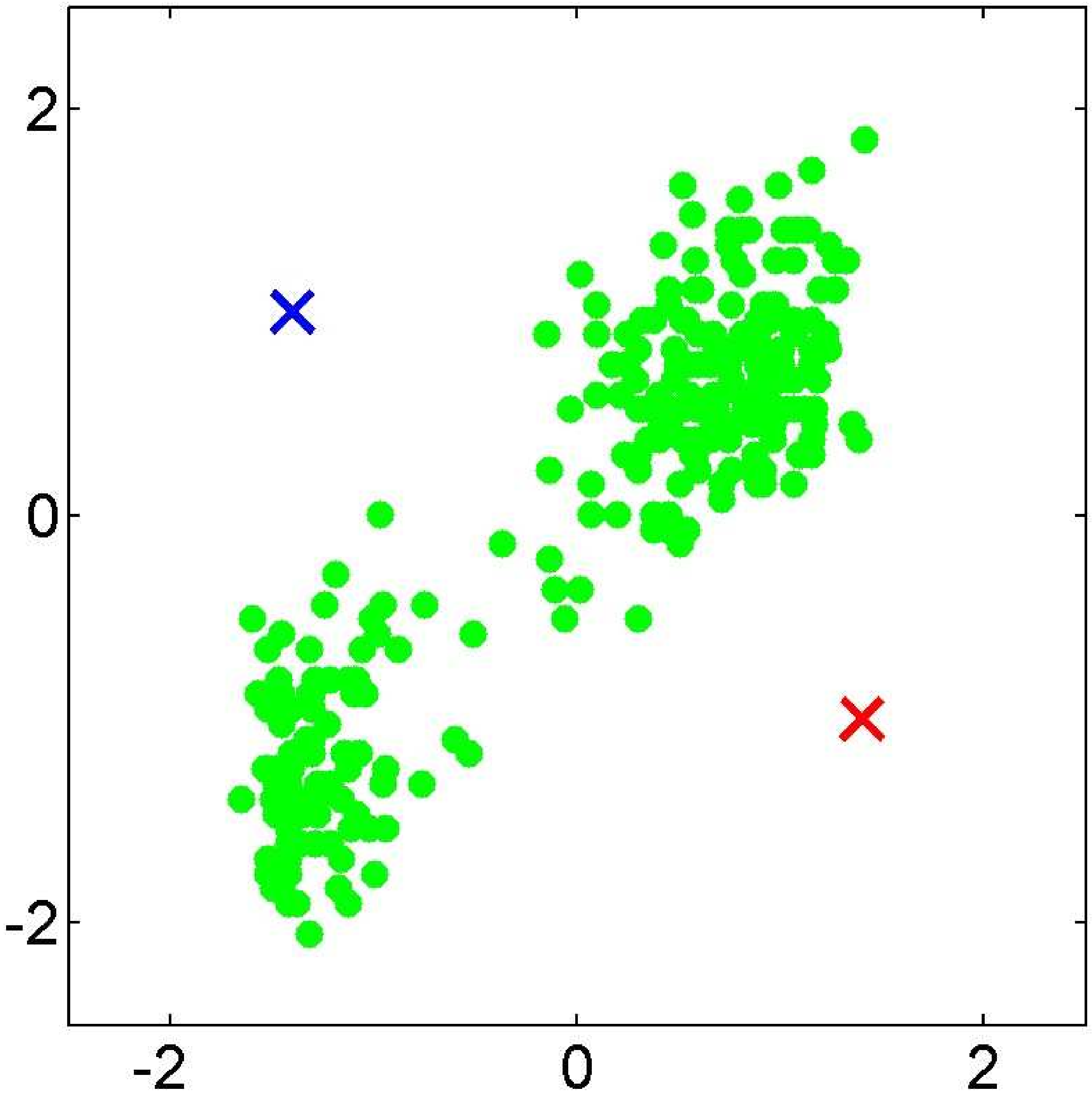


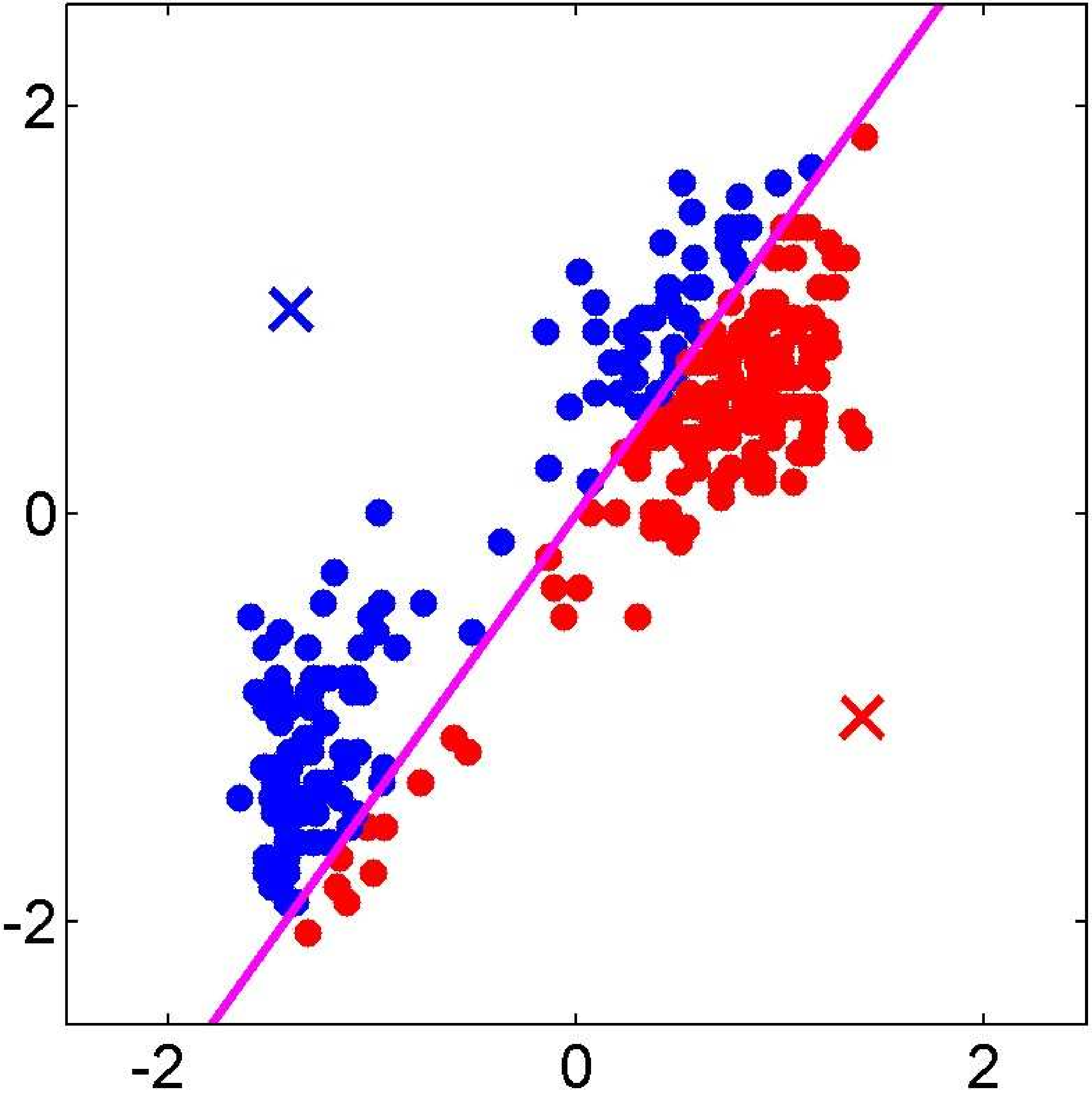
Time between eruptions (minutes)

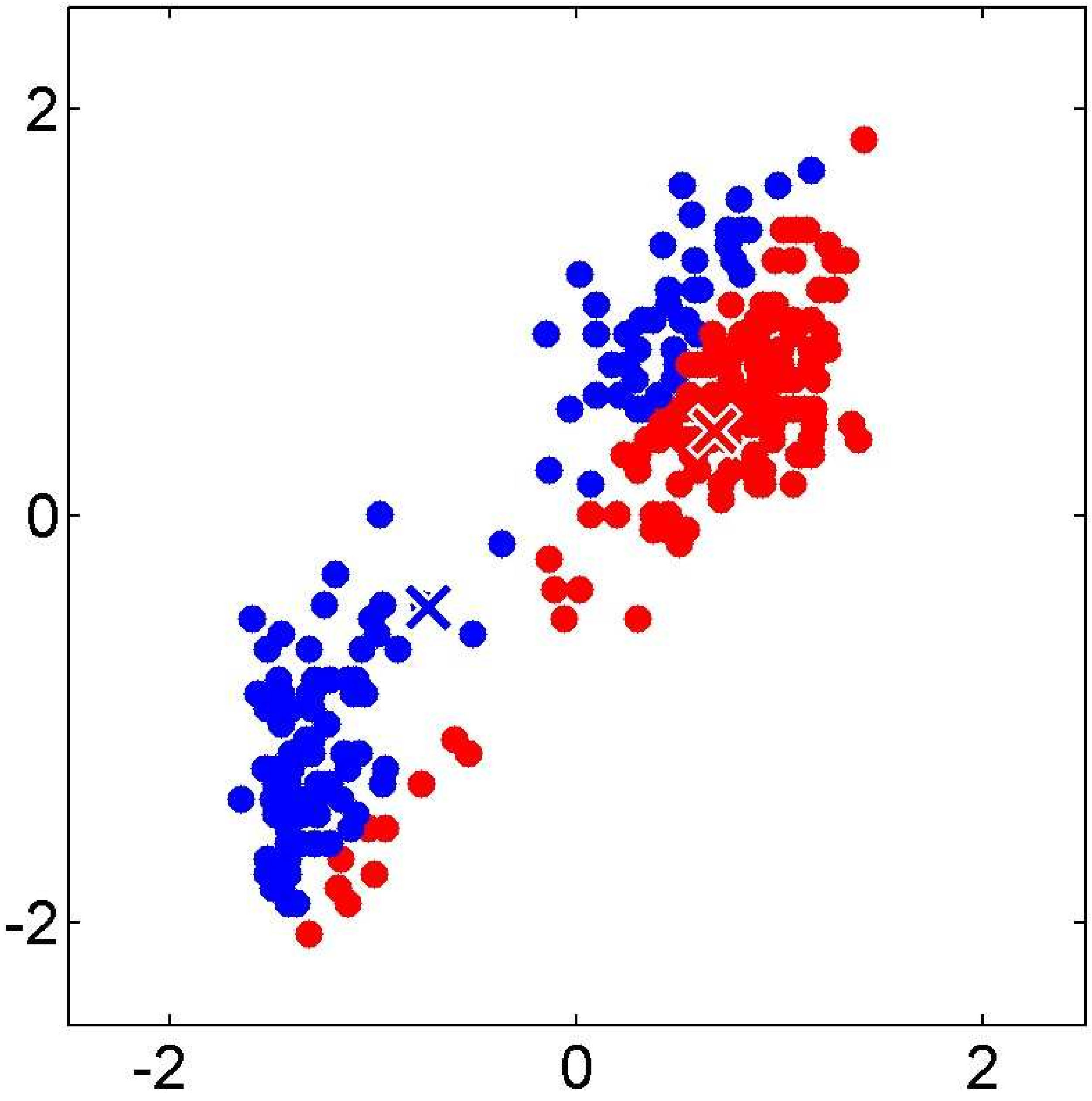
Duration of eruption (minutes)

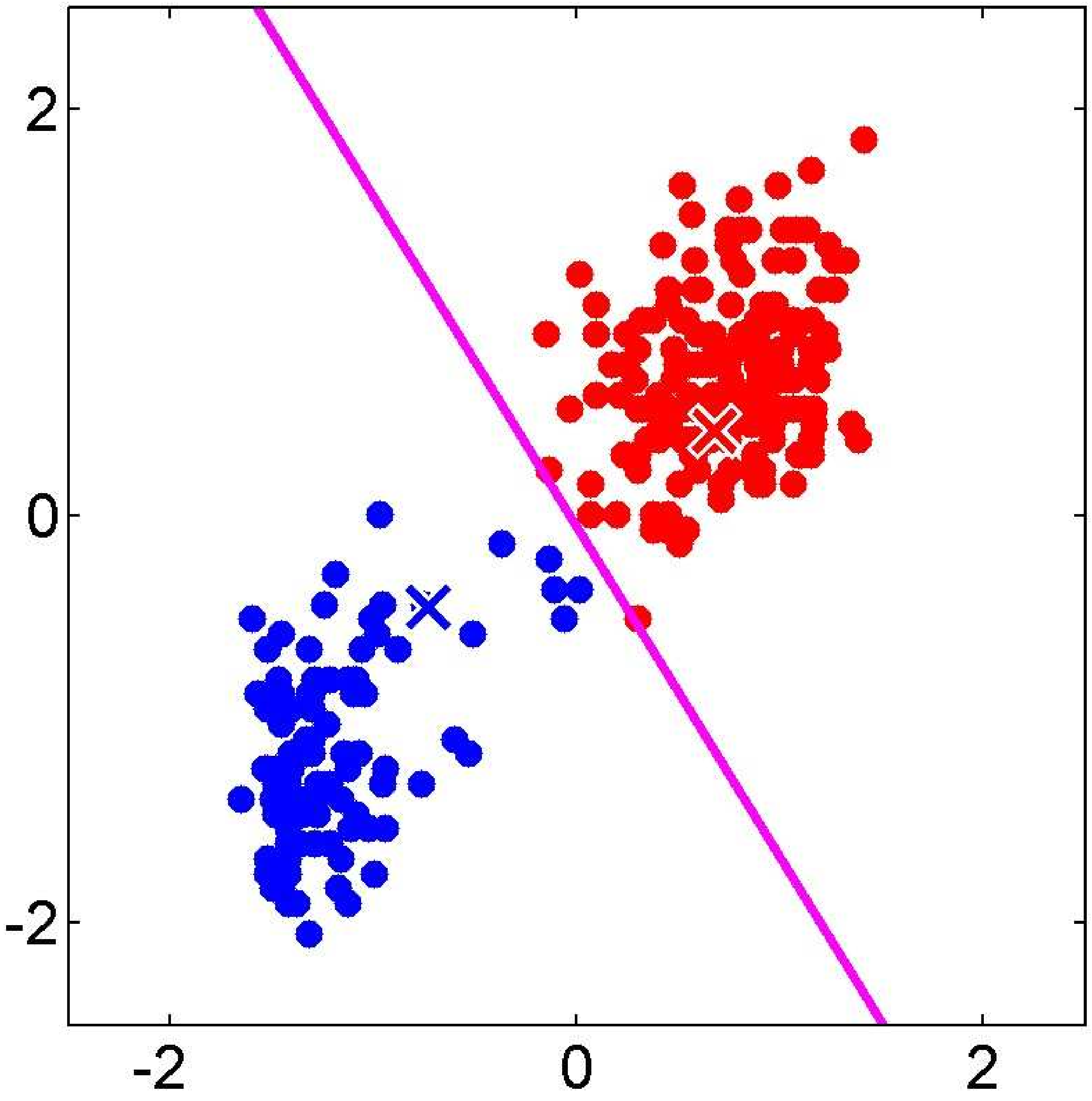
K-means Algorithm

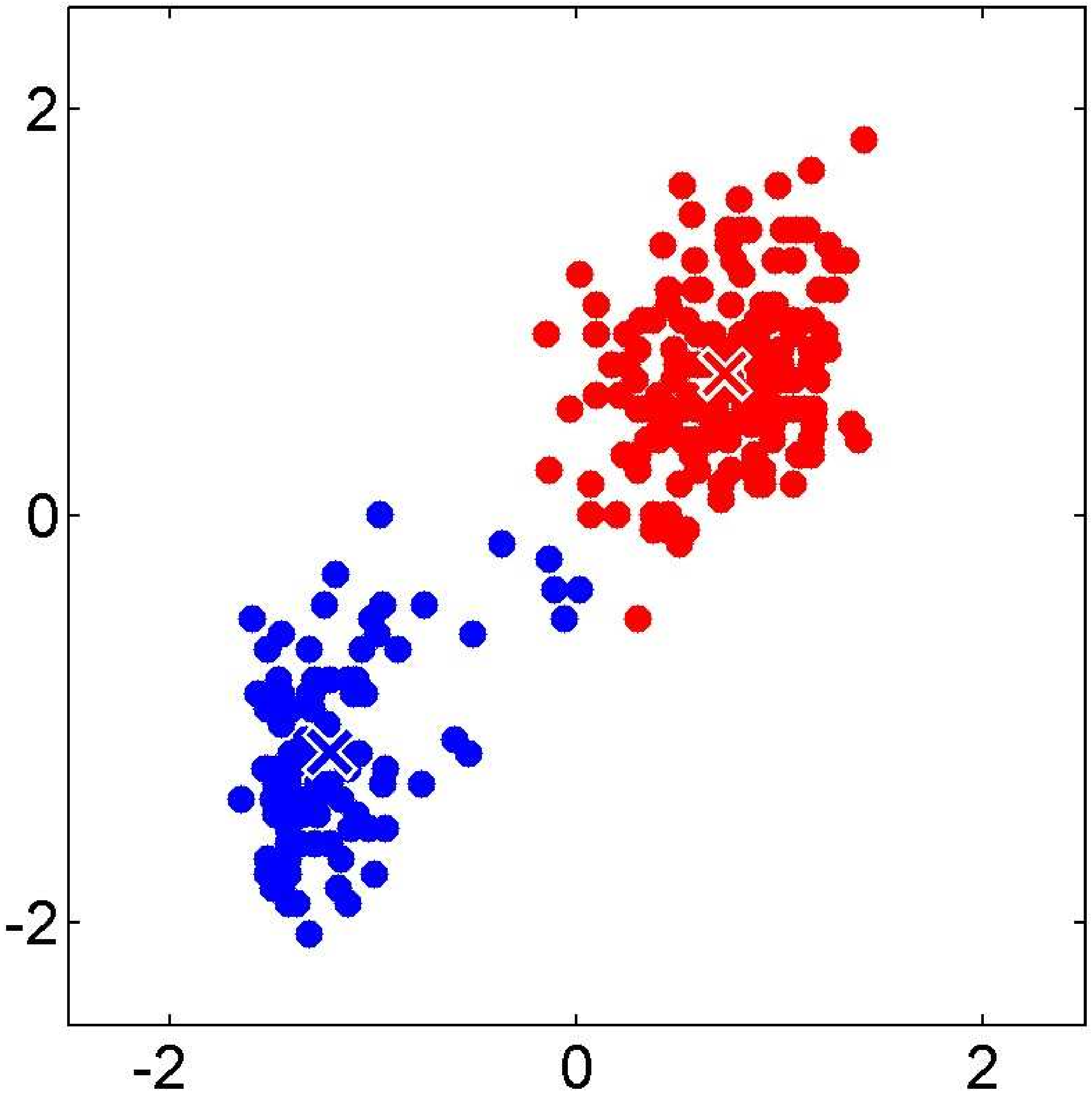
- Goal: represent a data set in terms of K clusters each of which is summarized by a prototype $\boldsymbol{\mu}_k$
- Initialize prototypes, then iterate between two phases:
  - E-step: assign each data point to nearest prototype
  - M-step: update prototypes to be the cluster means
- Simplest version is based on Euclidean distance
  - re-scale Old Faithful data

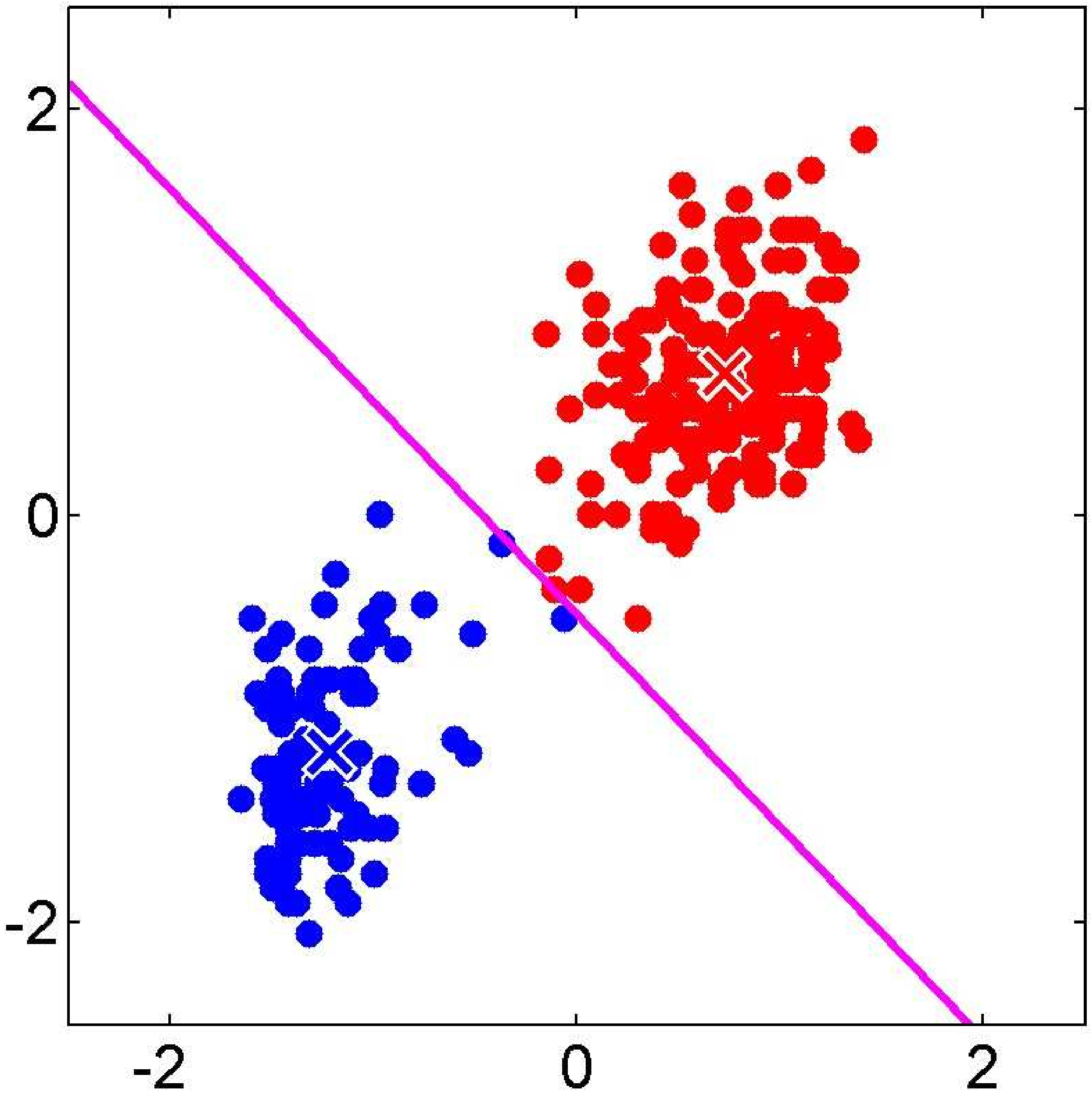

2
0
-2
-2
0
2

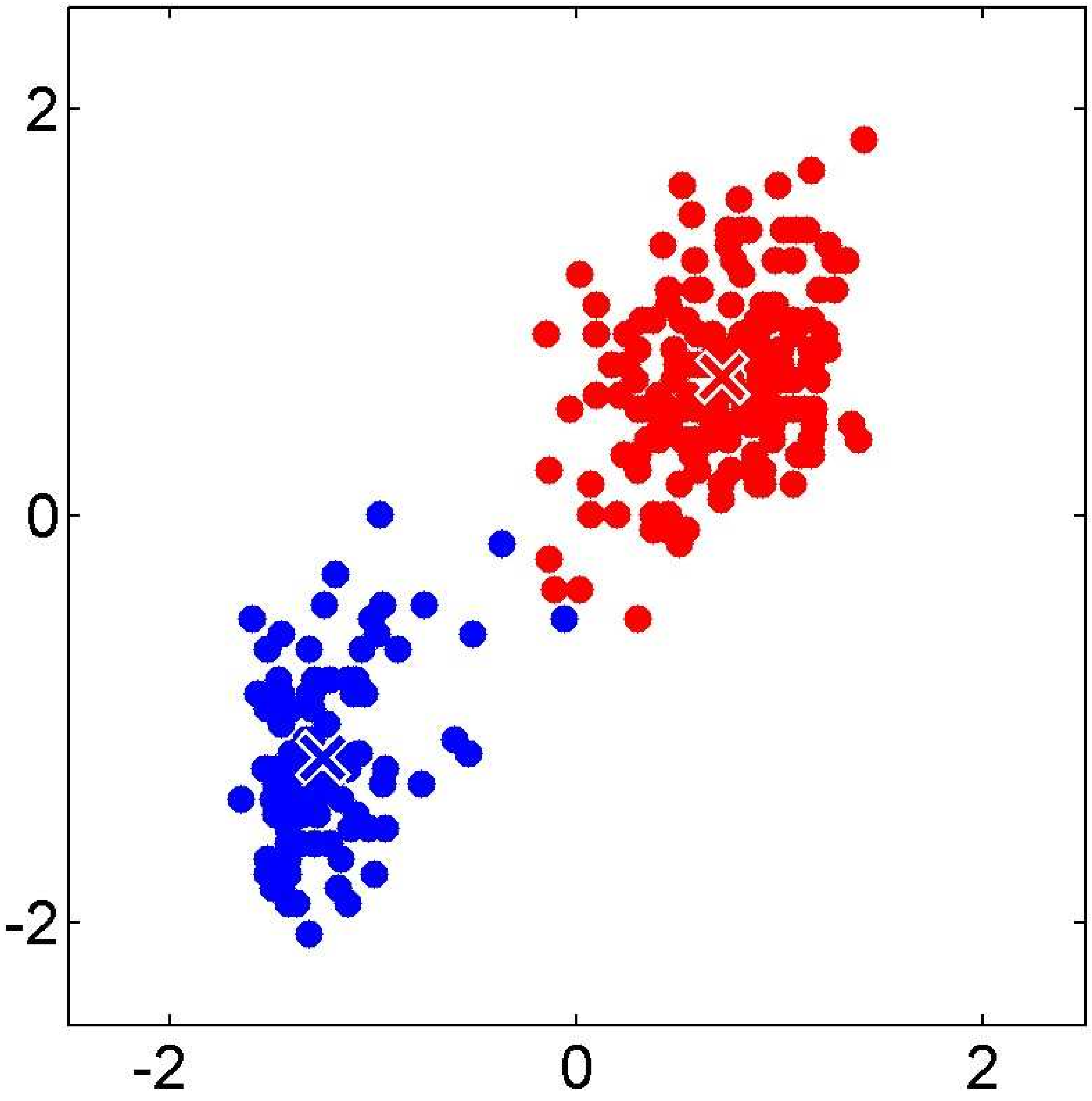

2
0
-2
-2
0
2

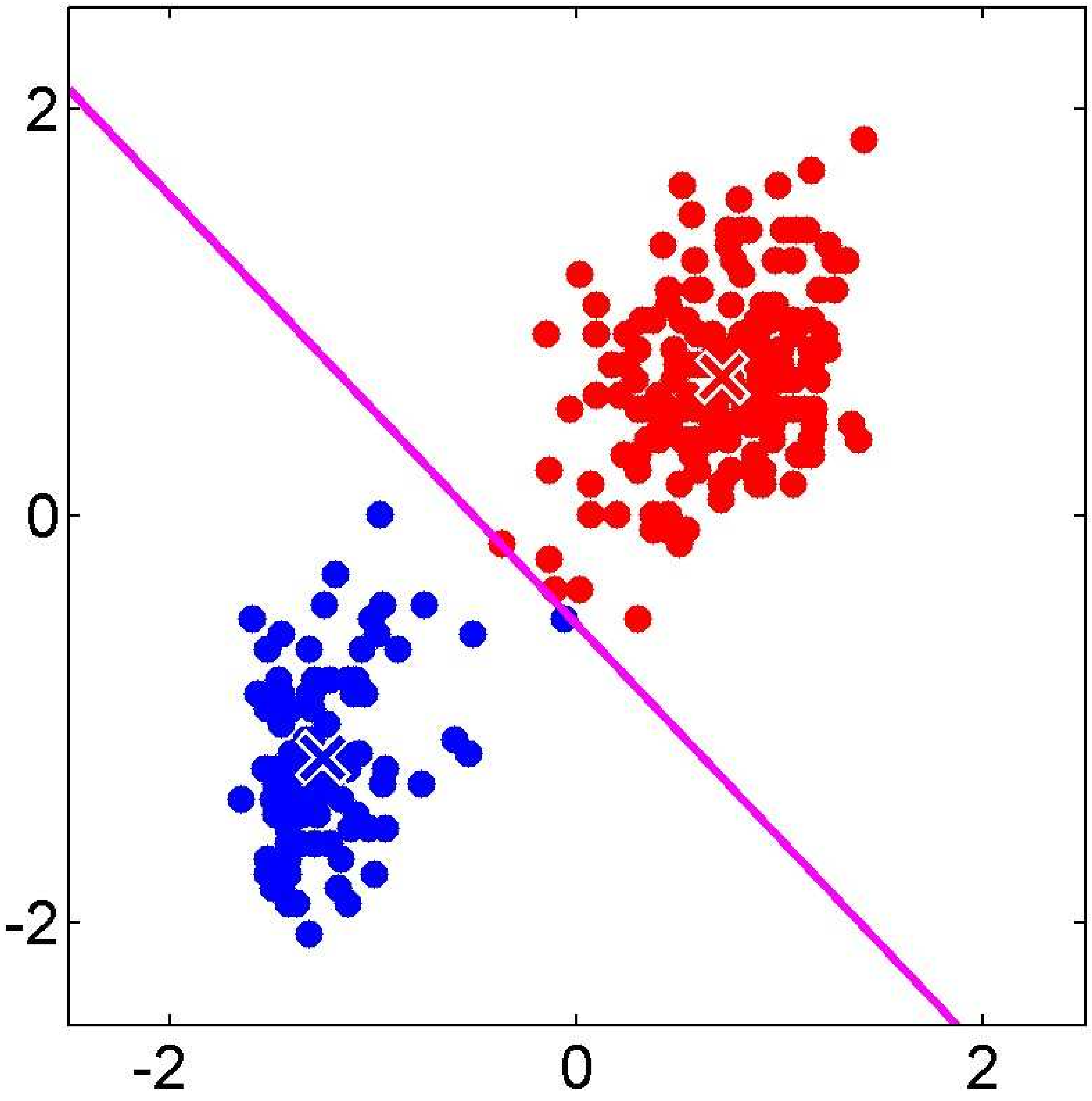

2
0
-2
-2
0
2

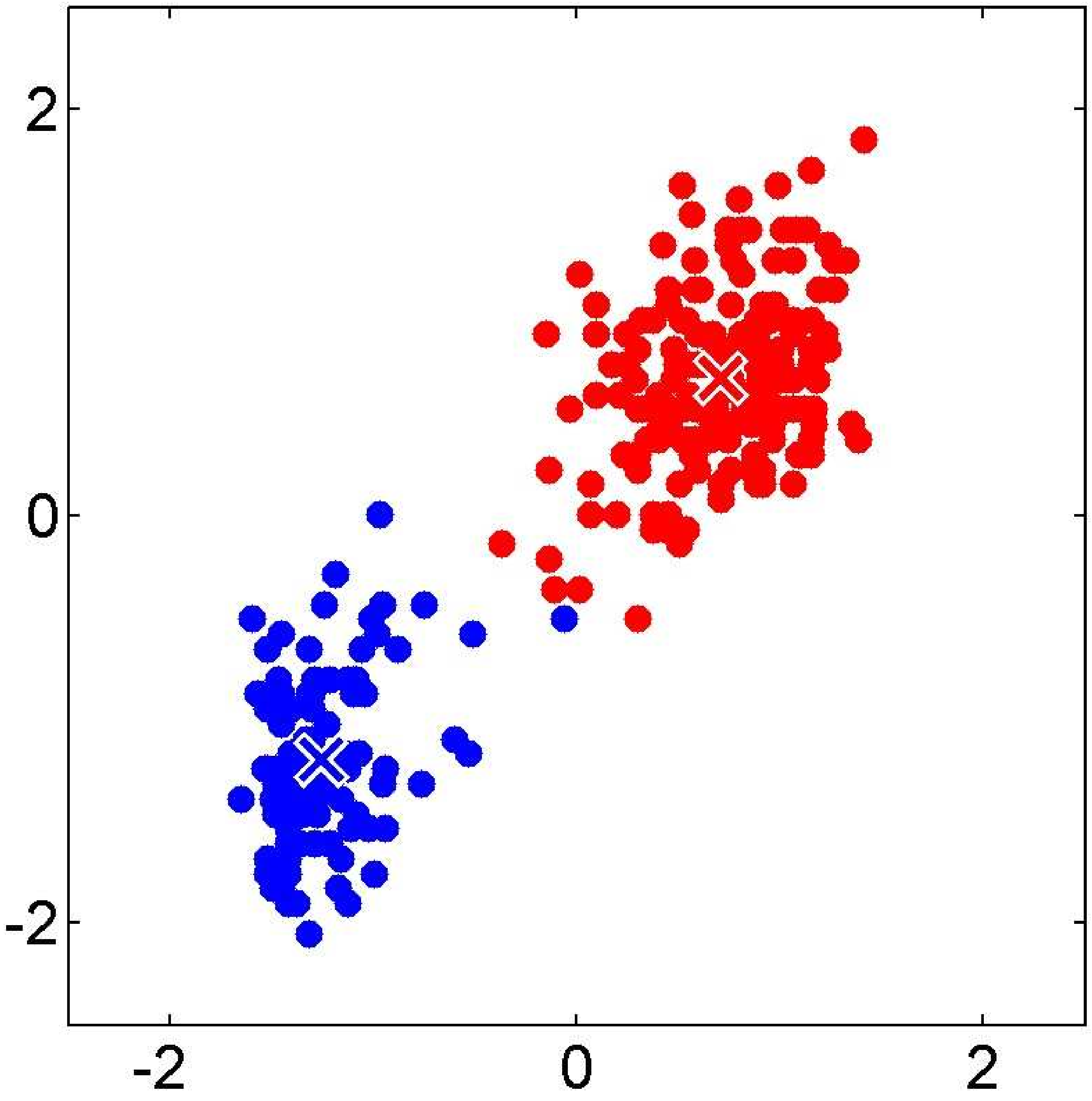

2
0
-2
-2
0
2


2
0
-2
-2
0
2


2
0
-2
-2
0
2


2
0
-2
-2
0
2


2
0
-2
-2
0
2


2
0
-2
-2
0
2

Responsibilities

- *Responsibilities* assign data points to clusters

$$r_{nk} \in \{0, 1\}$$

such that

$$\sum_k r_{nk} = 1$$

- Example: 5 data points and 3 clusters

$$(r_{nk}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

K-means Cost Function

data

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

responsibilities

prototypes

Minimizing the Cost Function

- E-step: minimize J w.r.t. r_{nk}

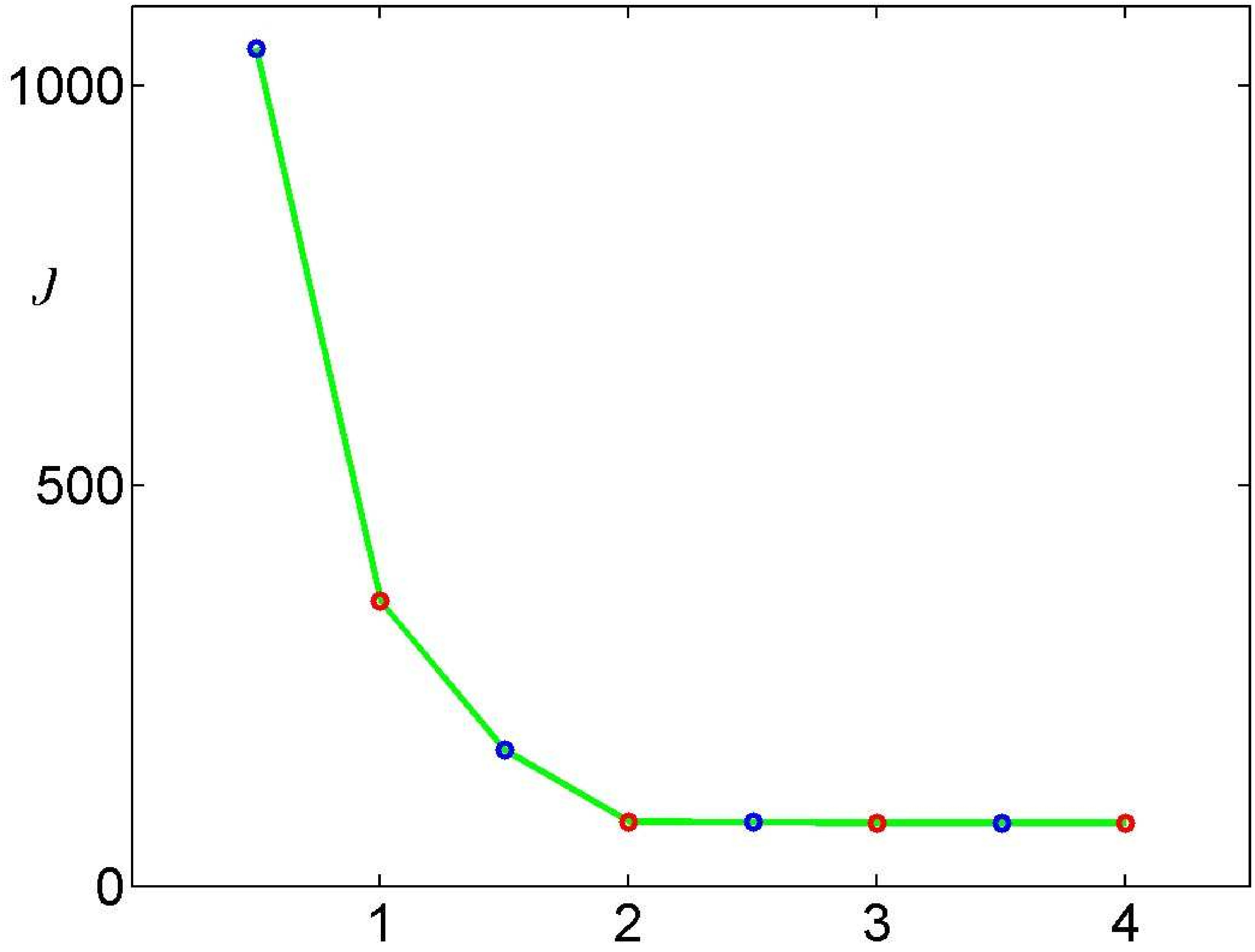
$$J = \sum_n \sum_k r_{nk} \|\underline{x}_n - \underline{\mu}_k\|^2$$

- M-step: minimize J w.r.t $\boldsymbol{\mu}_k$

$$\boxed{\underline{\mu}_j = \frac{\sum_n r_{nj} \underline{x}_n}{\sum_n r_{nj}}}$$

$$0 = 2\sum_{n=1}^{N} r_{nj} (\underline{x}_n - \underline{\mu}_j)$$

- Convergence guaranteed since there is a finite number of possible settings for the responsibilities


1000
500
0
J
1
2
3
4

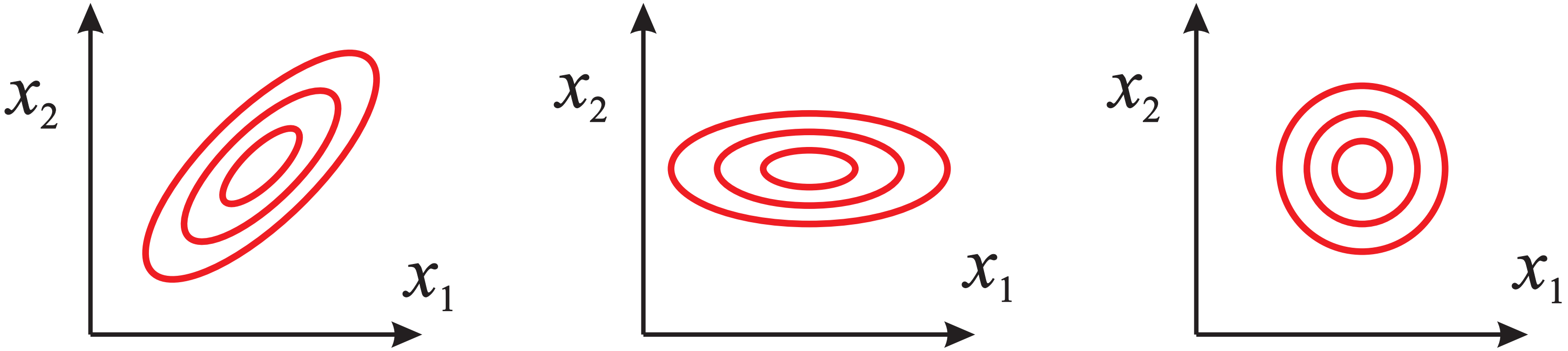
Probabilistic Clustering

- Represent the probability distribution of the data as a *mixture model*
  - captures uncertainty in cluster assignments
  - gives model for data distribution
  - *Bayesian* mixture model allows us to determine K
- Consider mixtures of *Gaussians*

The Gaussian Distribution

- Multivariate Gaussian

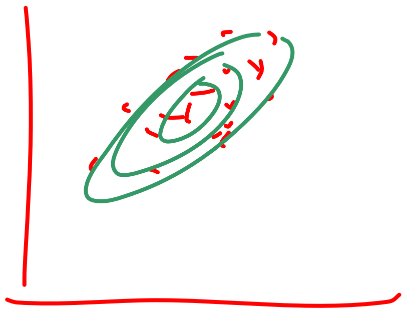
$$\mathcal{N}(\mathbf{x}|\underset{\substack{\nwarrow \\ \text{mean}}}{\boldsymbol{\mu}}, \underset{\substack{\nearrow \\ \text{covariance}}}{\boldsymbol{\Sigma}}) = \frac{1}{(2\pi)^{d/2}\,|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$



(a) (b) (c)

Likelihood Function

- Data set



$$D = \{\mathbf{x}_n\} \qquad n = 1, \ldots, N$$

- Consider first a single Gaussian
- Assume observed data points generated independently

$$p(D|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Viewed as a function of the parameters, this is known as the *likelihood function*

Maximum Likelihood

- Set the parameters by maximizing the likelihood function
- Equivalently maximize the log likelihood

$$\frac{\partial}{\partial \underline{\mu}}$$

$$\begin{aligned}\ln p(D|\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= -\frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{Nd}{2}\ln(2\pi) \\ &\quad -\frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu})\end{aligned}$$

$$0 = \sum_{n=1}^{N} (\underline{x}_n - \underline{\mu}) \Rightarrow \boxed{\underline{\mu} = \frac{1}{N}\sum_{n=1}^{N} \underline{x}_n}$$

Maximum Likelihood Solution

- Maximizing w.r.t. the mean gives the *sample mean*

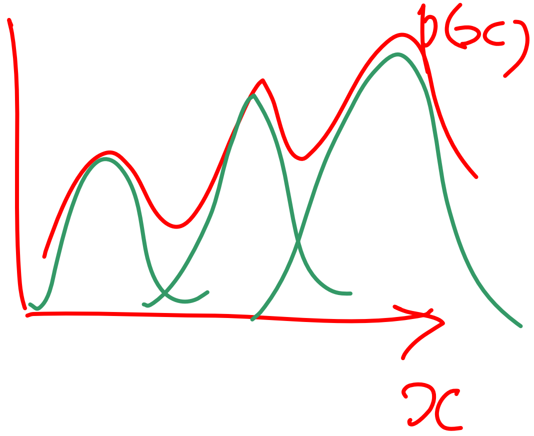
$$\boldsymbol{\mu}_{\mathsf{ML}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n$$

- Maximizing w.r.t covariance gives the *sample covariance*

$$\boldsymbol{\Sigma}_{\mathsf{ML}} = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu}_{\mathsf{ML}})(\mathbf{x}_n - \boldsymbol{\mu}_{\mathsf{ML}})^{\mathsf{T}}$$

Gaussian Mixtures

- Linear super-position of Gaussians



$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

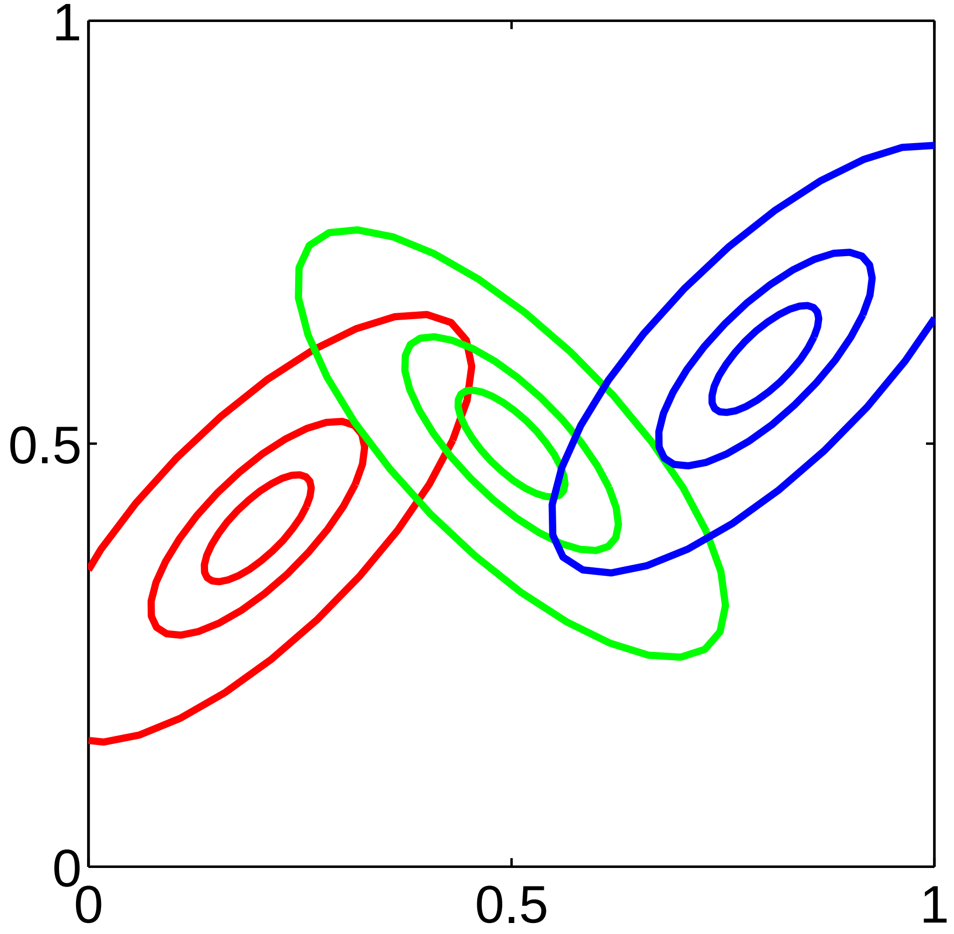
- Normalization and positivity require

$$\sum_{k=1}^{K} \pi_k = 1 \qquad 0 \leqslant \pi_k \leqslant 1$$

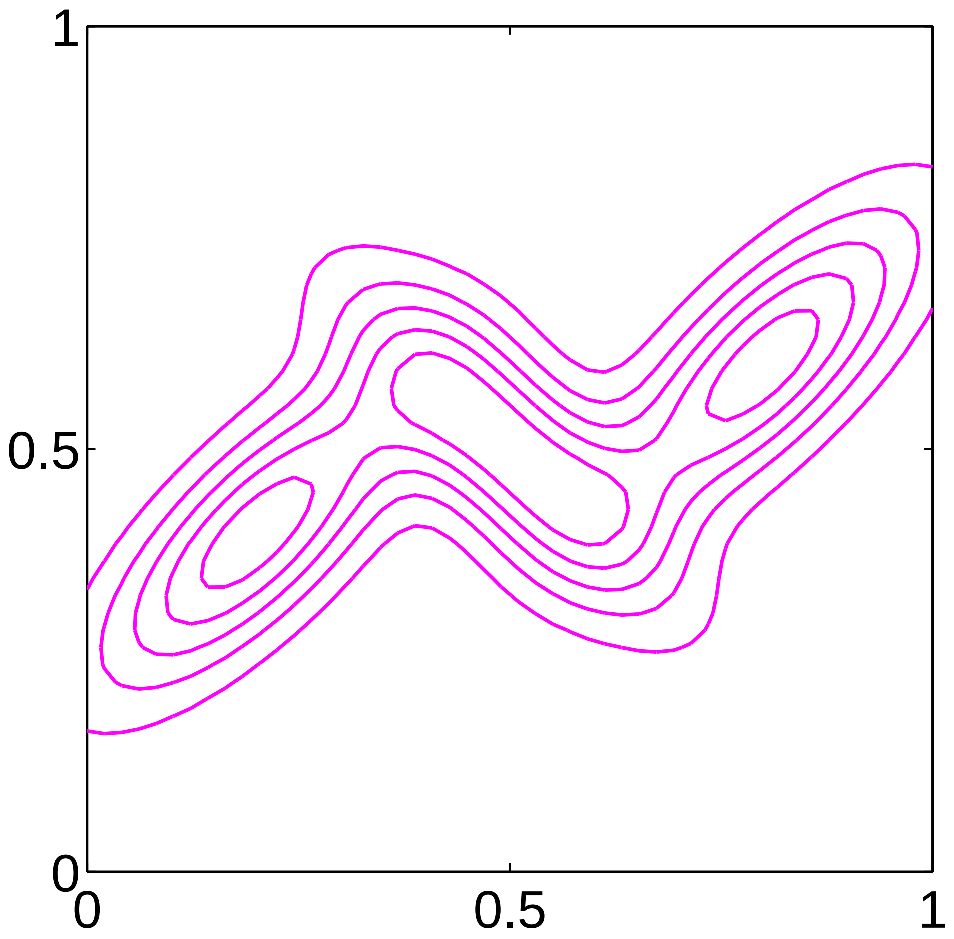
- Can interpret the mixing coefficients as prior probabilities

$$p(\mathbf{x}) = \sum_{k=1}^{K} p(k) p(\mathbf{x}|k)$$

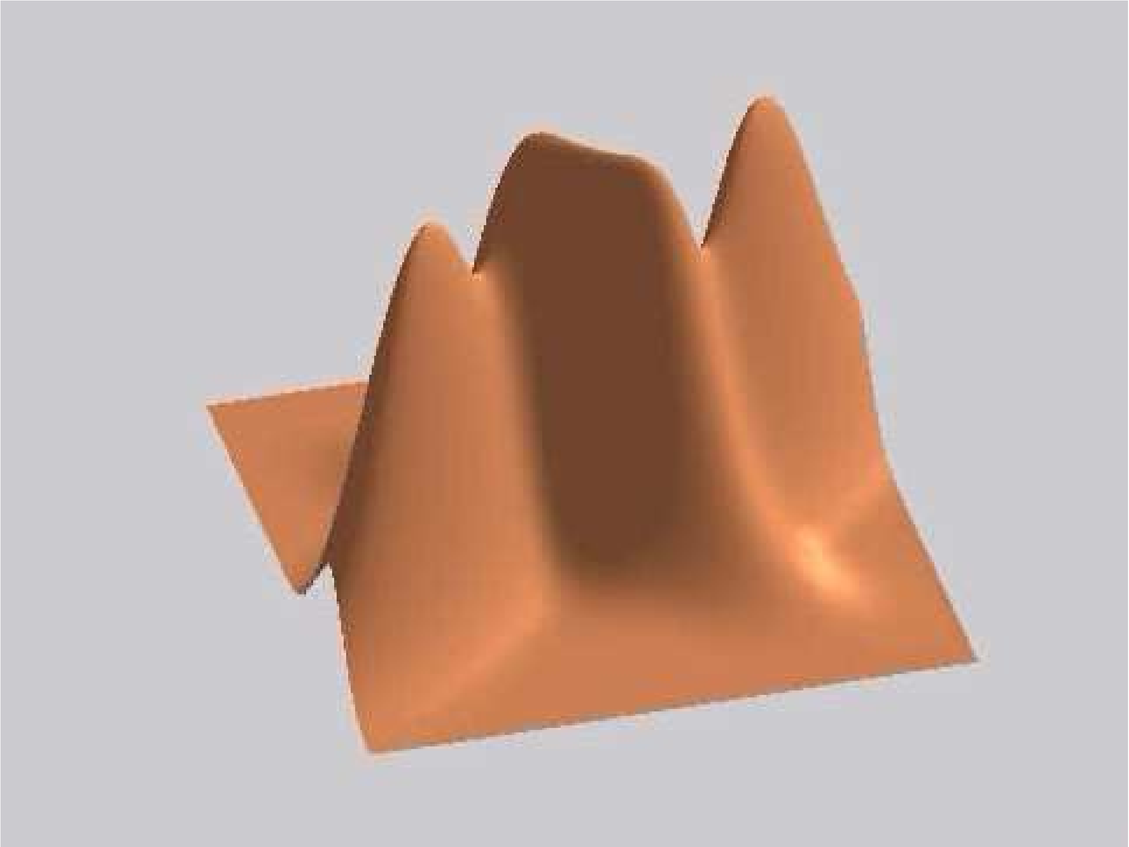
Example: Mixture of 3 Gaussians



Contours of Probability Distribution



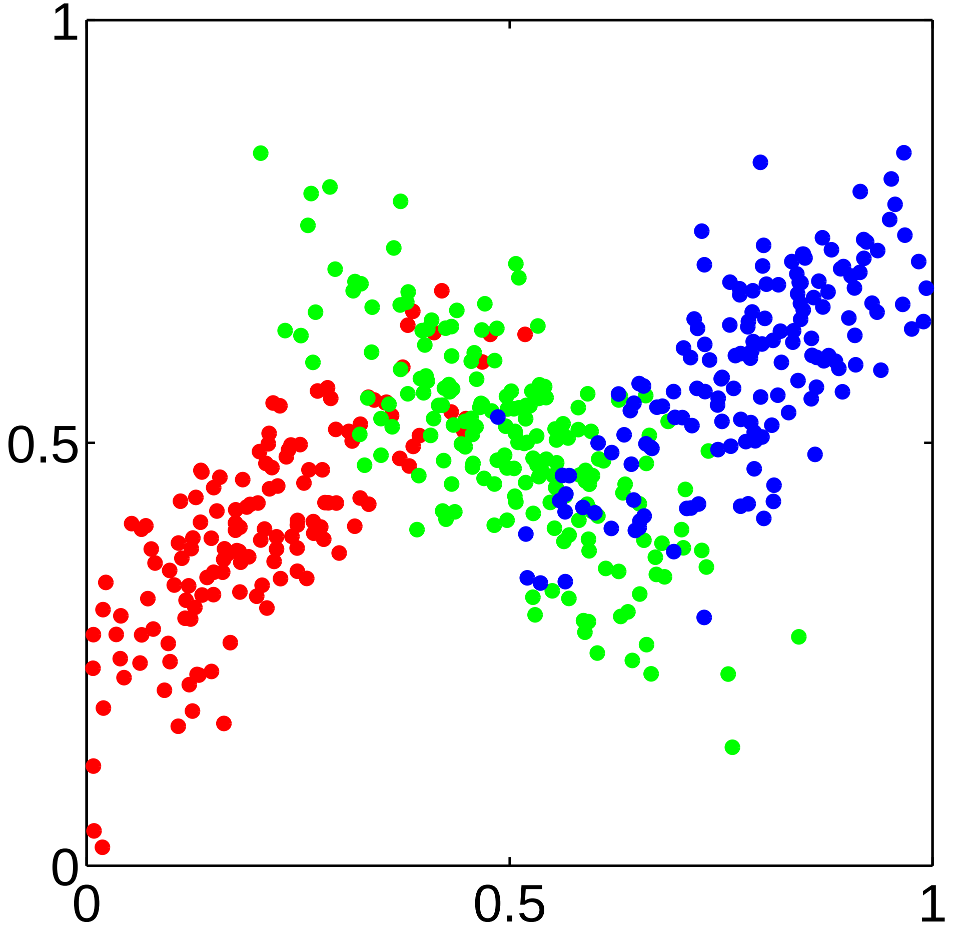
Surface Plot



Sampling from the Gaussian

- To generate a data point:
  - first pick one of the components with probability π_k
  - then draw a sample $\mathbf{x}_n$ from that component
- Repeat these two steps for each new data point

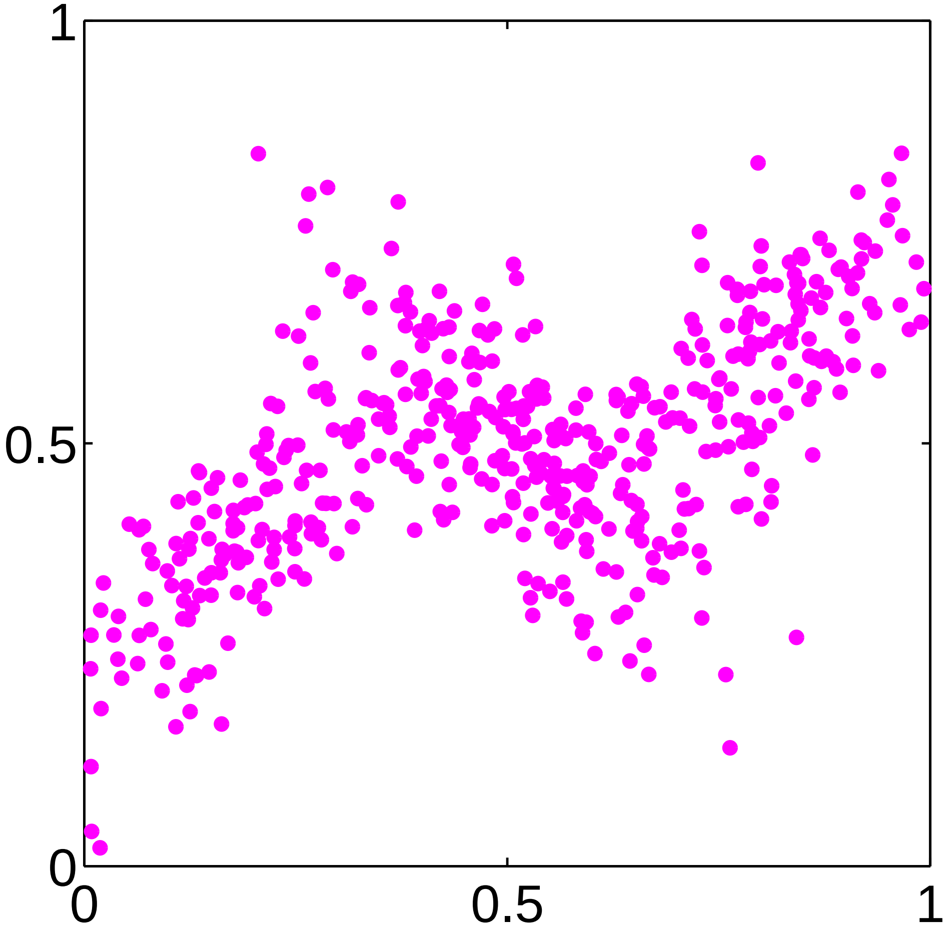
Synthetic Data Set



Fitting the Gaussian Mixture

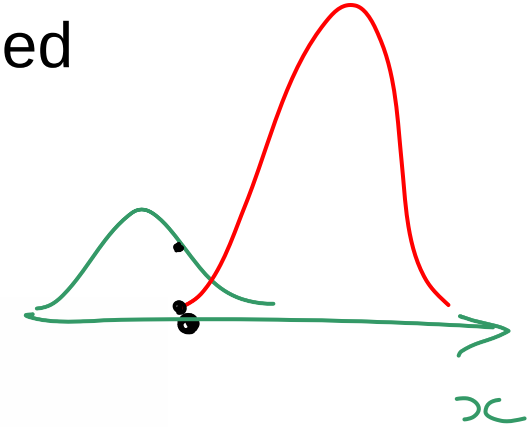
- We wish to invert this process – given the data set, find the corresponding parameters:
  - mixing coefficients
  - means
  - covariances
- If we knew which component generated each data point, the maximum likelihood solution would involve fitting each component to the corresponding cluster
- Problem: the data set is unlabelled
- We shall refer to the labels as *latent* (= hidden) variables

Synthetic Data Set Without Labels



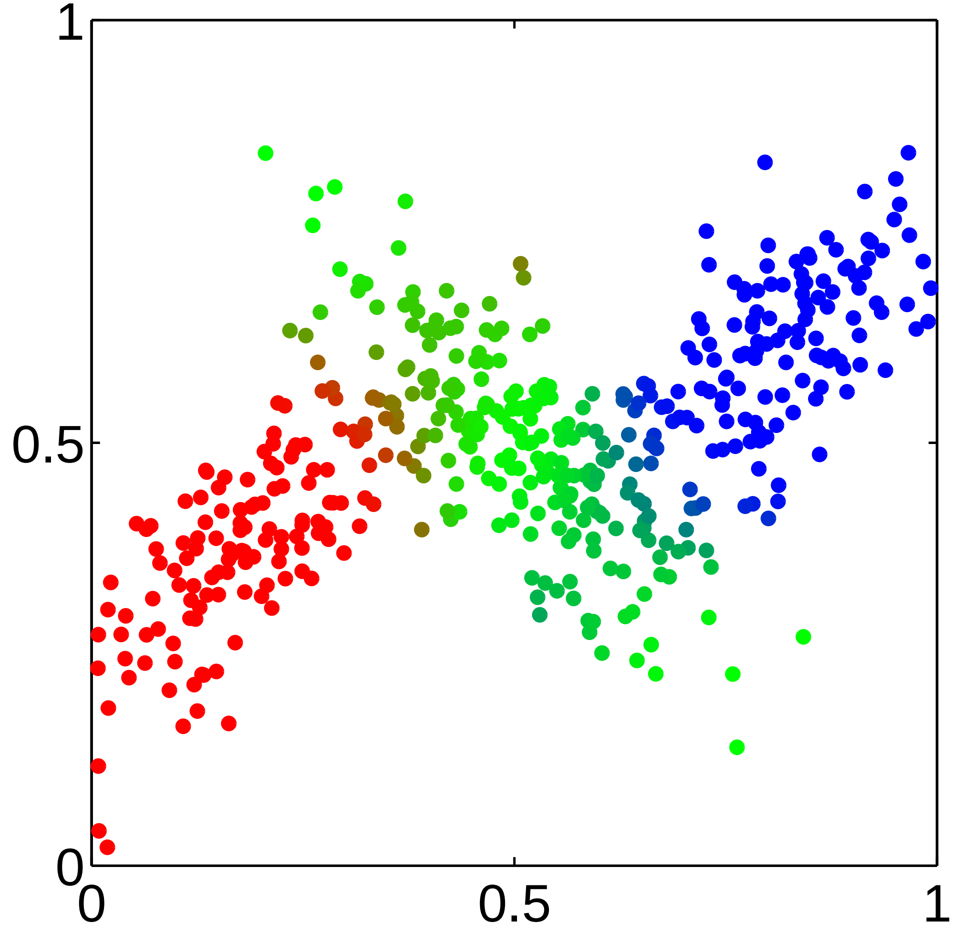
Posterior Probabilities

- We can think of the mixing coefficients as prior probabilities for the components
- For a given value of $\mathbf{x}$ we can evaluate the corresponding posterior probabilities, called *responsibilities*
- These are given from Bayes' theorem by

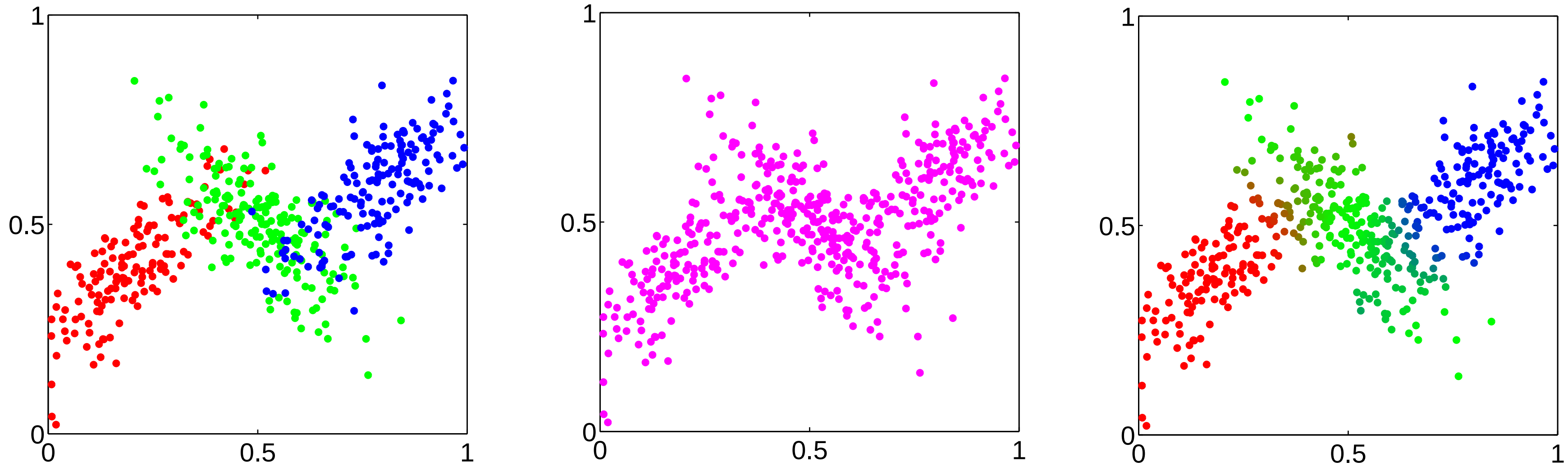


$$\begin{aligned}\gamma_k(\mathbf{x}) \equiv p(k|\mathbf{x}) &= \frac{p(k)p(\mathbf{x}|k)}{p(\mathbf{x})} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}\end{aligned}$$

Posterior Probabilities (colour coded)



Latent Variables



Maximum Likelihood for the GMM

- The log likelihood function takes the form

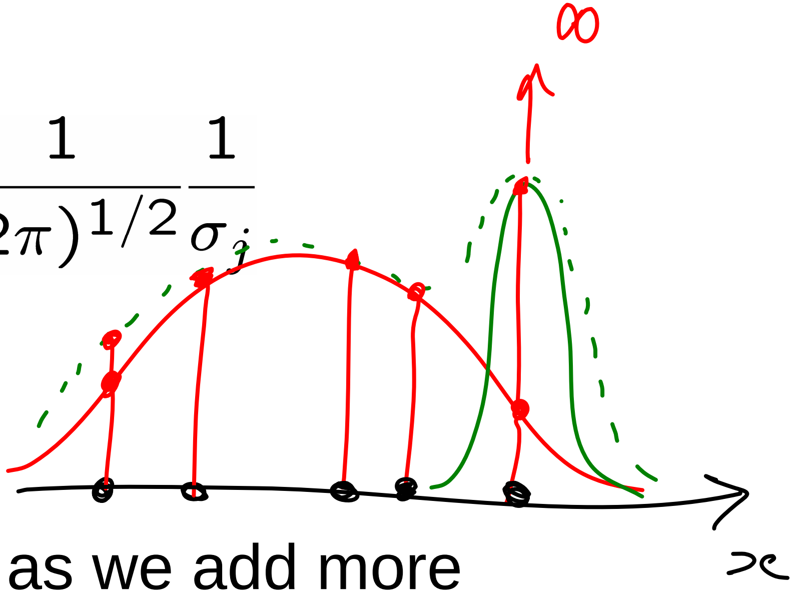
$$\ln p(D|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

- Note: sum over components appears *inside* the log
- There is no closed form solution for maximum likelihood

Over-fitting in Gaussian Mixture Models

- Singularities in likelihood function when a component ‘collapses’ onto a data point:

$$\mathcal{N}(\mathbf{x}_n|\mathbf{x}_n, \sigma_j^2\mathbf{I}) = \frac{1}{(2\pi)^{1/2}}\frac{1}{\sigma_j}$$



then consider $\sigma_j \to 0$

- Likelihood function gets larger as we add more components (and hence parameters) to the model
  - not clear how to choose the number K of components

Problems and Solutions

- How to maximize the log likelihood
  - solved by expectation-maximization (EM) algorithm
- How to avoid singularities in the likelihood function
  - solved by a Bayesian treatment
- How to choose number K of components
  - also solved by a Bayesian treatment

EM Algorithm – Informal Derivation

- Let us proceed by simply differentiating the log likelihood

$$\ln p(D|\mu,\pi,\Sigma) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k N_{nk}\right\} \qquad N_{nk} \equiv \mathcal{N}(\underline{x}_n|\underline{\mu}_k, \underline{\Sigma}_k)$$

$$0 = \sum_{n=1}^{N} \underbrace{\boxed{\frac{\pi_j N_{nj}}{\sum_k \pi_k N_{nk}}}}_{\gamma_j(\underline{x}_n)} \cancel{\underline{\Sigma}^{-1}} (\underline{x}_n - \underline{\mu}_j)$$

$$\boxed{\underline{\mu}_j = \frac{\sum_n \gamma_{jn} \underline{x}_n}{\sum_n \gamma_{jn}}}$$

EM Algorithm – Informal Derivation

- Similarly for the covariances

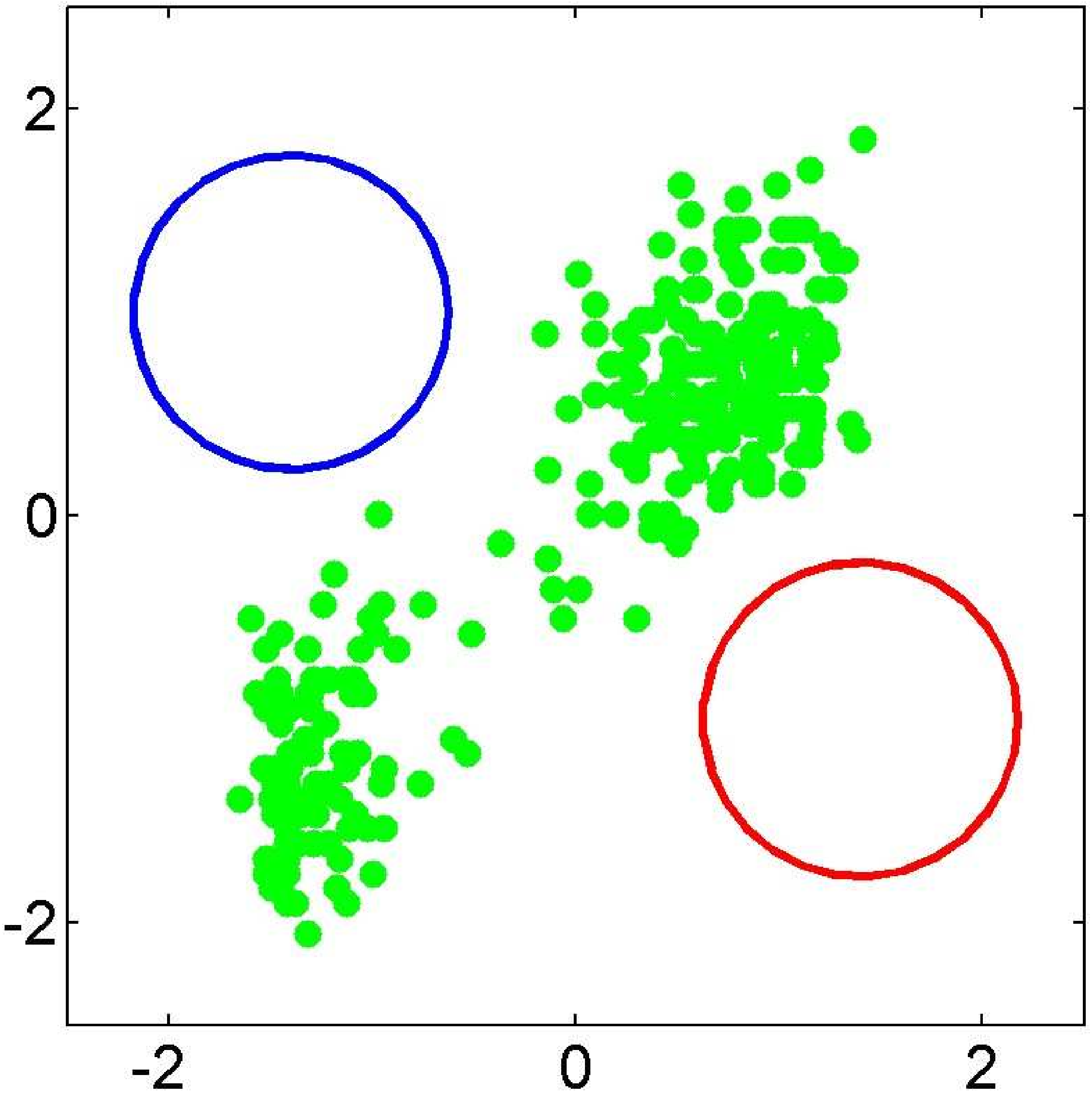
$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^{\top}}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

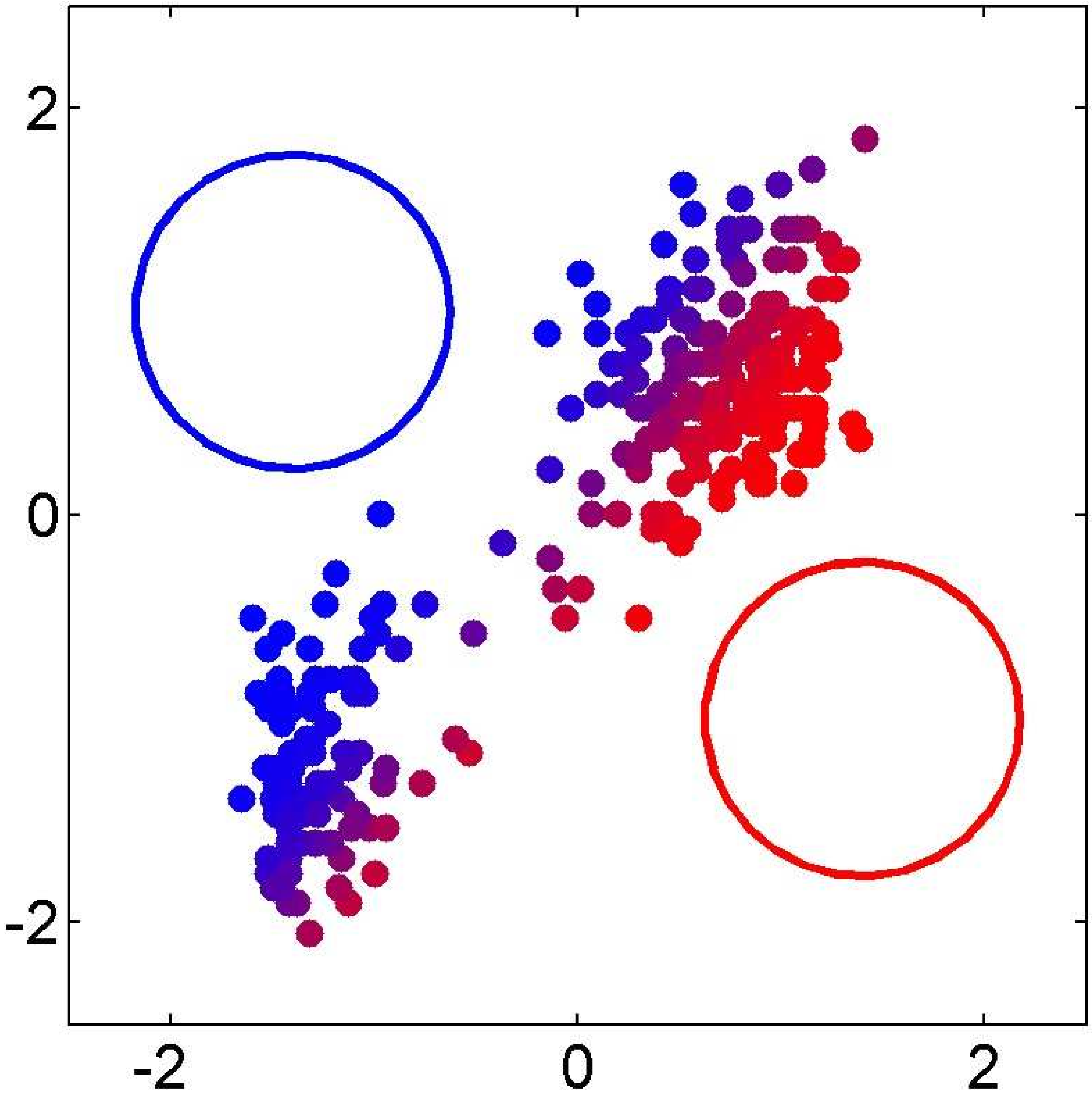
- For mixing coefficients use a Lagrange multiplier to give

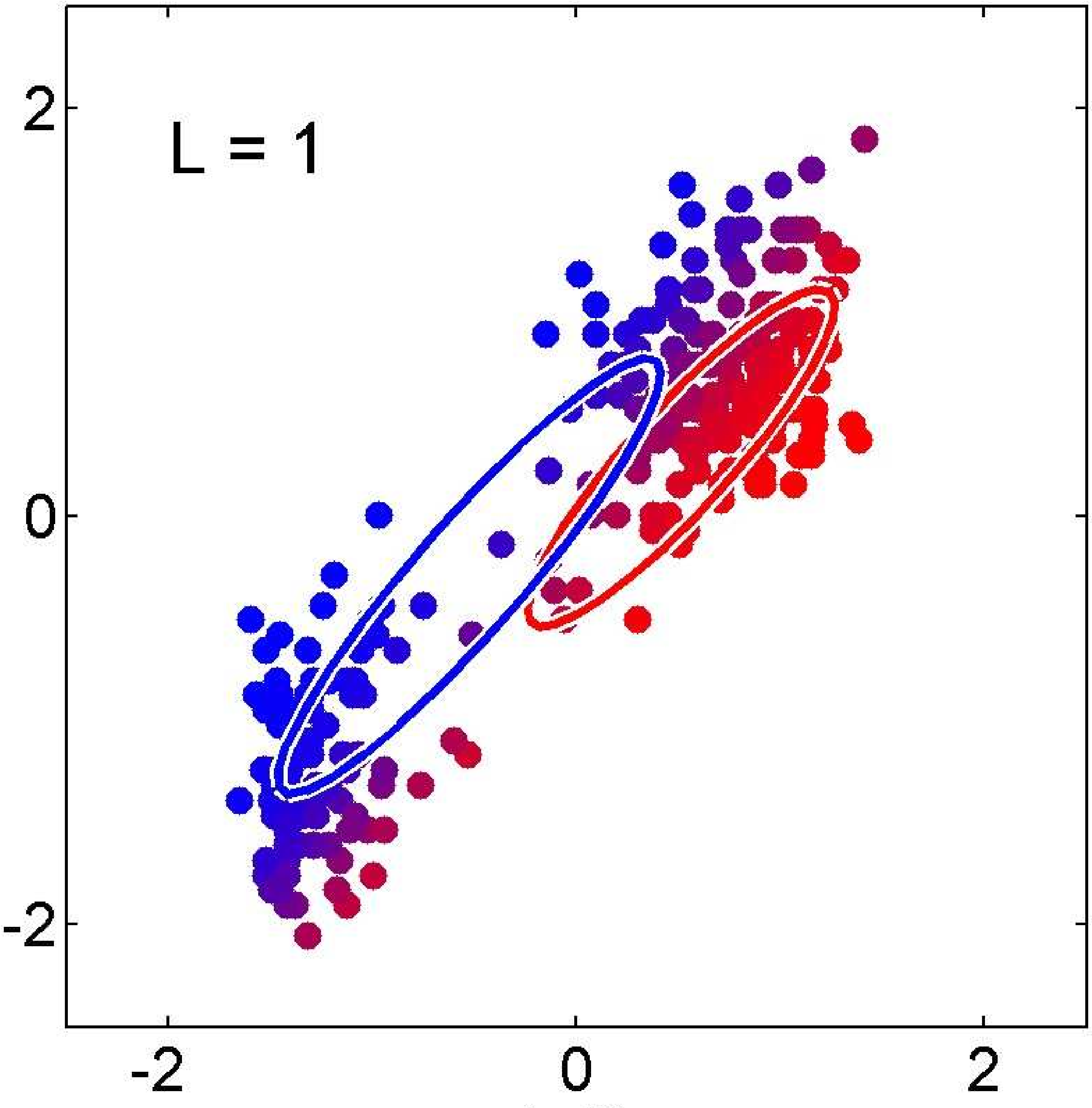
$$\pi_j = \frac{1}{N}\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$

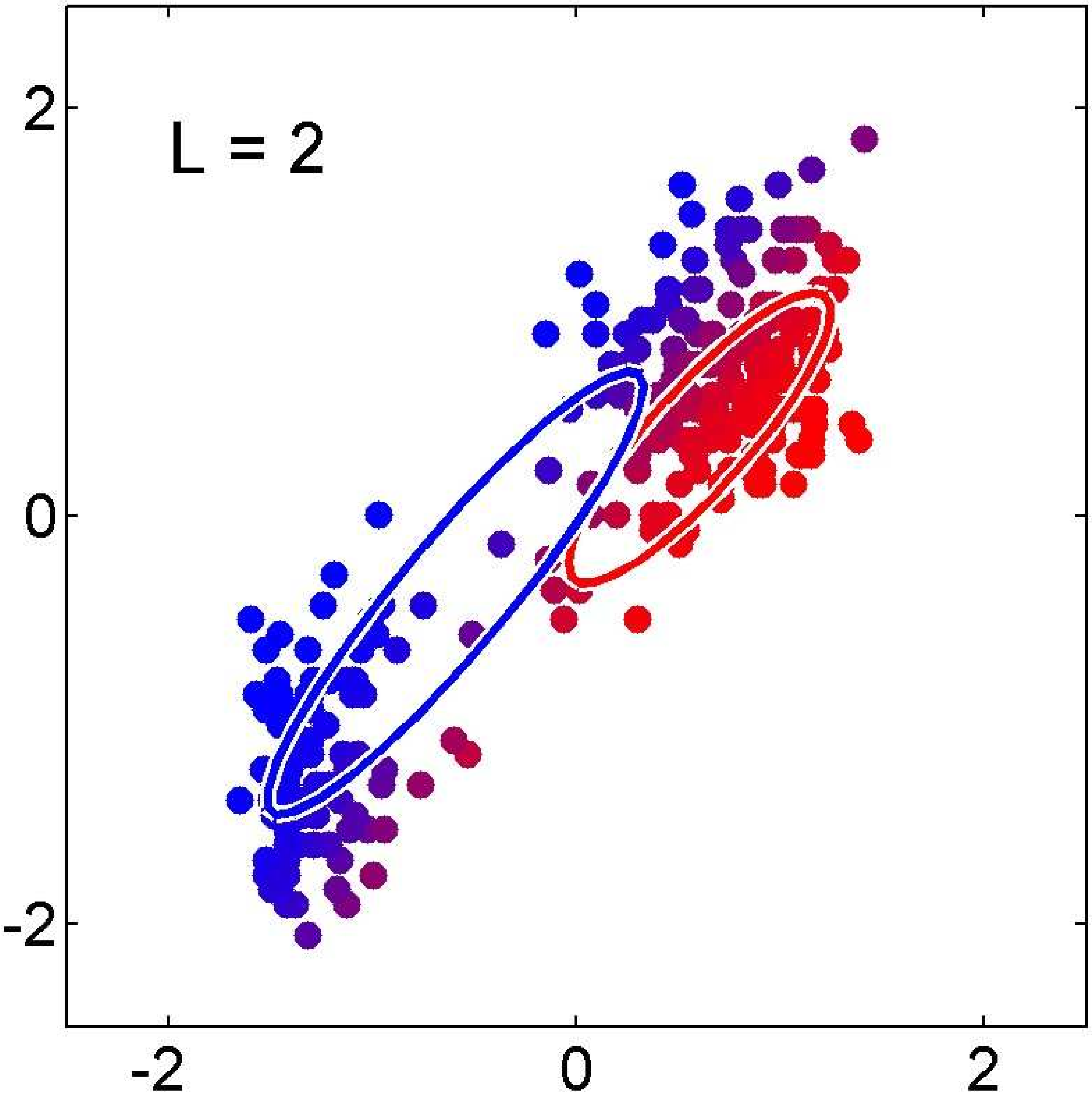
EM Algorithm – Informal Derivation

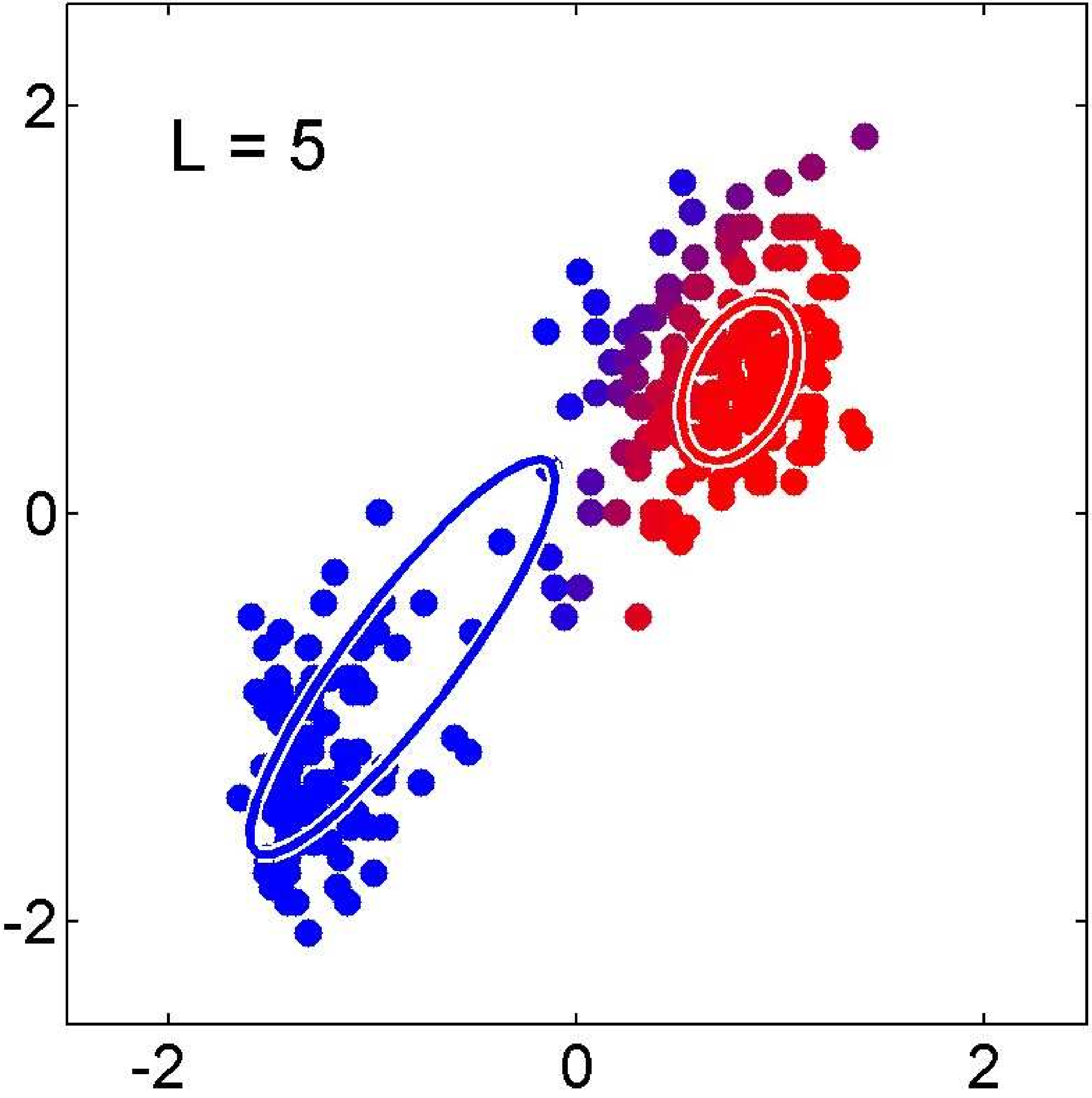
- The solutions are not closed form since they are coupled
- Suggests an iterative scheme for solving them:
  - make initial guesses for the parameters
  - alternate between the following two stages:
    1. E-step: evaluate responsibilities
    2. M-step: update parameters using ML results
- Each EM cycle guaranteed not to decrease the likelihood

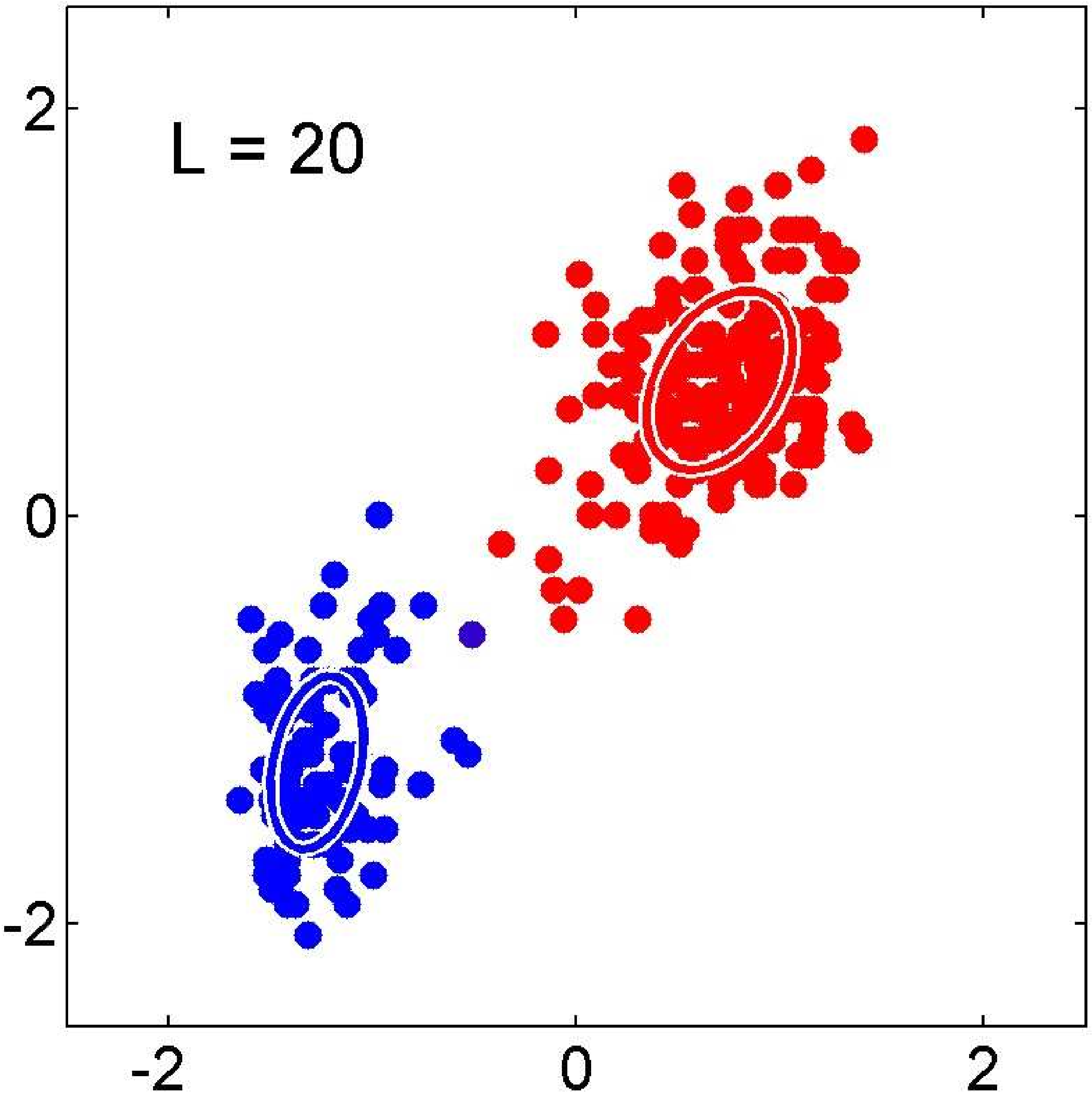

2
0
-2
-2
0
2


2
0
-2
-2
0
2


L = 1
2
0
-2
-2
0
2


L = 2
2
0
-2
-2
0
2


L = 5
2
0
-2
-2
0
2


L = 20
2
0
-2
-2
0
2

Relation to K-means

- Consider GMM with common covariances $\epsilon \underline{\underline{I}}$
- Take limit $\epsilon \to 0$
- Responsibilities become binary

$$\gamma_i(\mathbf{x}_n) = \frac{\pi_i \exp\left\{-\|\mathbf{x}_n - \boldsymbol{\mu}_i\|^2/2\epsilon\right\}}{\sum_j \pi_j \exp\left\{-\|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2/2\epsilon\right\}} \to r_{ni} \in \{0, 1\}$$

- EM algorithm is precisely equivalent to K-means

Bayesian Mixture of Gaussians

- Include prior distribution over parameters

$$\underline{\underline{\Lambda}} = \underline{\underline{\Sigma}}^{-1}$$

$$p(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\pi})$$

- Make predictions by *marginalizing* over parameters
  - c.f. point estimate from maximum likelihood

Bayesian Mixture of Gaussians

- Conjugate priors for the parameters:
  - Dirichlet prior for mixing coefficients

$$p(\boldsymbol{\pi}) = C(\boldsymbol{\alpha}_0) \prod_{k=1}^{K} \pi_k^{\alpha_0 - 1}$$

  - Normal-Wishart prior for means and precisions

$$p(\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{\mu}_k | \mathbf{m}_0, \beta_0^{-1} \boldsymbol{\Lambda}_k^{-1}) \, \mathcal{W}(\boldsymbol{\Lambda}_k | \mathbf{W}_0, \nu_0)$$

where the Wishart distribution is given by

$$\mathcal{W}(\boldsymbol{\Lambda} | \mathbf{W}, \nu) \propto |\boldsymbol{\Lambda}|^{(\nu - d - 1)/2} \exp\left(-\frac{1}{2} \mathrm{Tr}(\mathbf{W}^{-1} \boldsymbol{\Lambda}) \right)$$

$$\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$$

Variational Inference

- Exact solution is intractable
- *Variational inference:*
  - extension of EM
  - alternate between updating posterior over parameters and posterior over latent variables
  - again convergence is guaranteed

Illustration: a single Gaussian

- Convenient to work with precision

$$\tau = 1/\sigma^2$$

- Likelihood function

$$p(D|\mu,\tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp\left\{-\frac{\tau}{2}\sum_{n=1}^{N}(x_n-\mu)^2\right\}$$

- Prior over parameters

$$p(\mu,\tau)$$

Variational Inference

$$p(D) = \int\int p(D|\mu,\tau)p(\mu,\tau)d\mu\, d\tau$$

- Goal is to find true posterior distribution

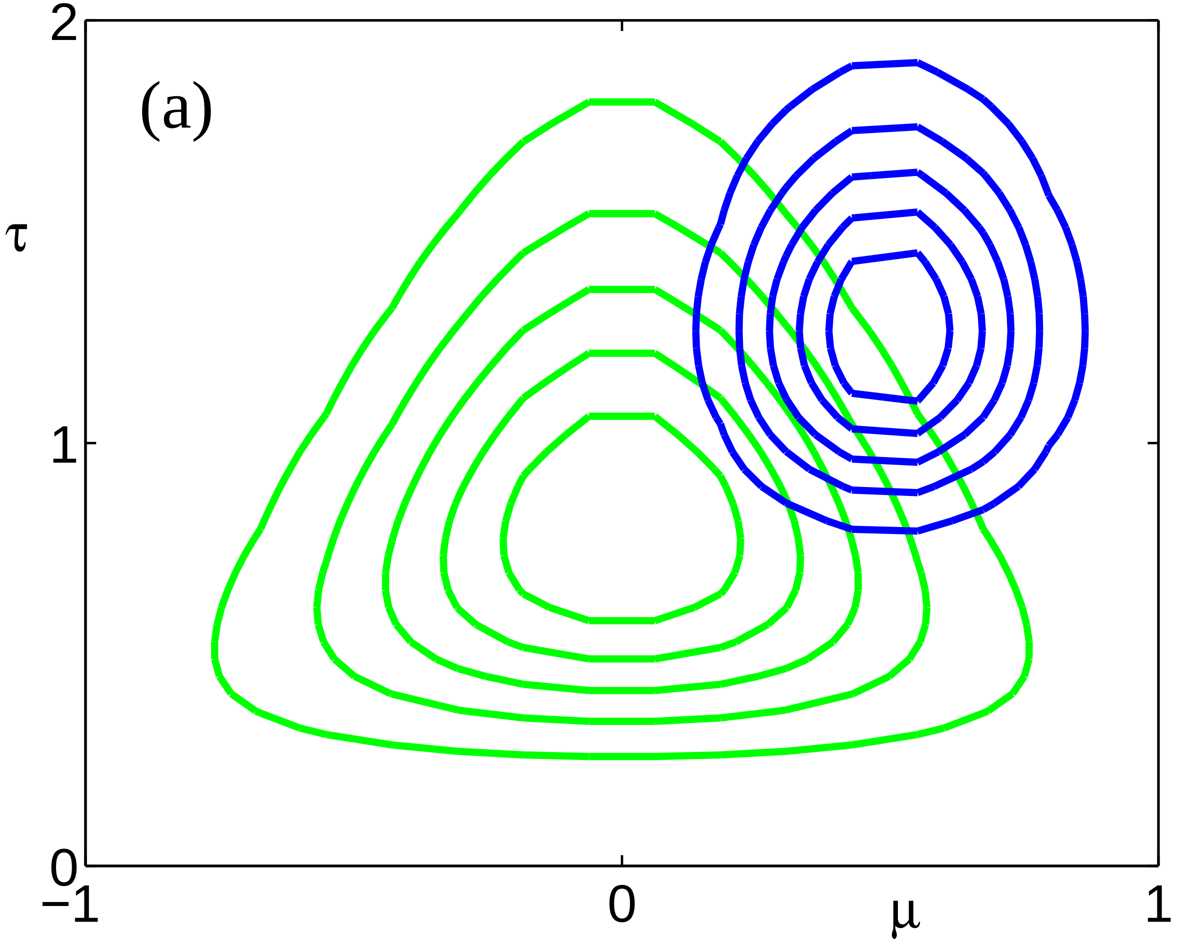
$$p(\mu,\tau|D) = \frac{p(D|\mu,\tau)p(\mu,\tau)}{p(D)}$$

- Factorized approximation

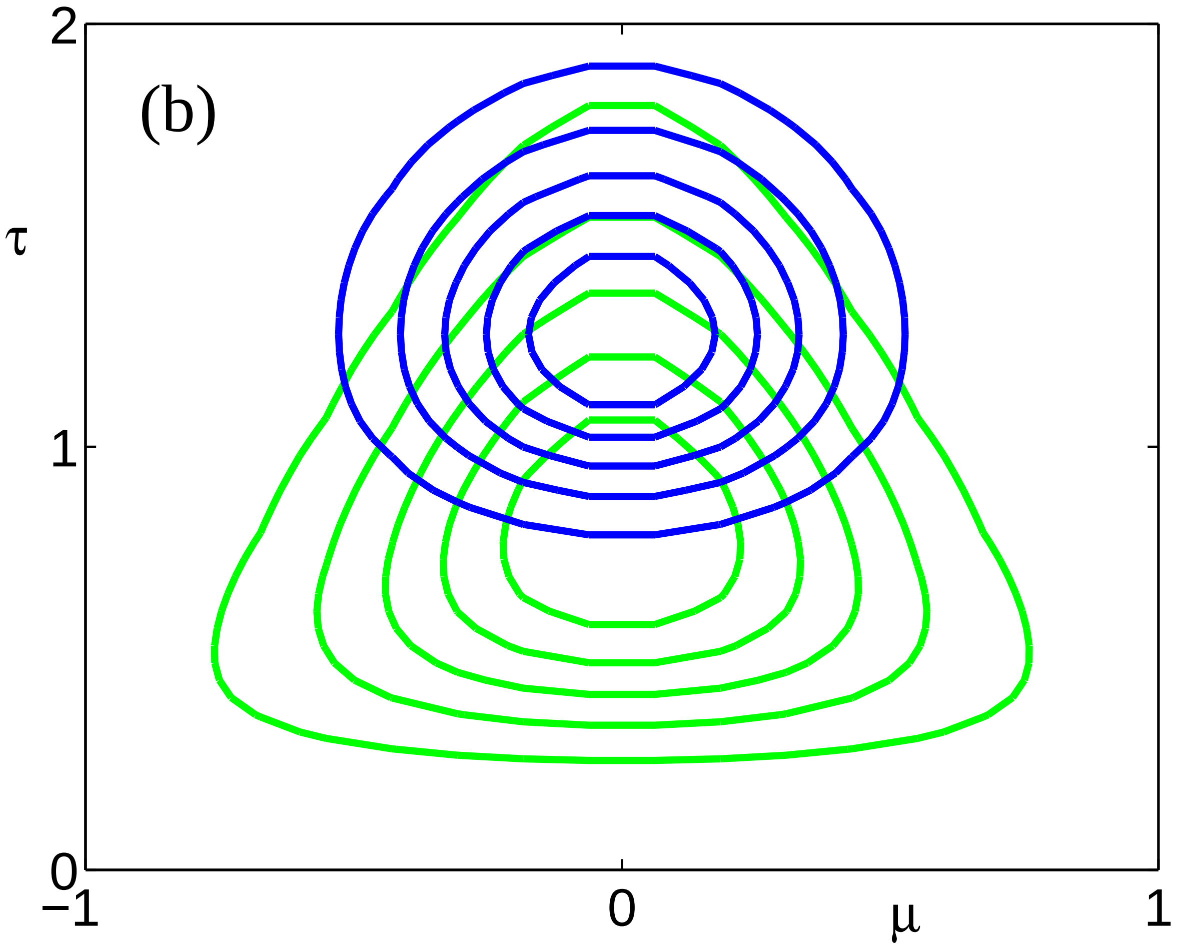
$$q(\mu,\tau) = q(\mu)q(\tau)$$

- Alternately update each factor to minimize a measure of closeness between the true and approximate distributions

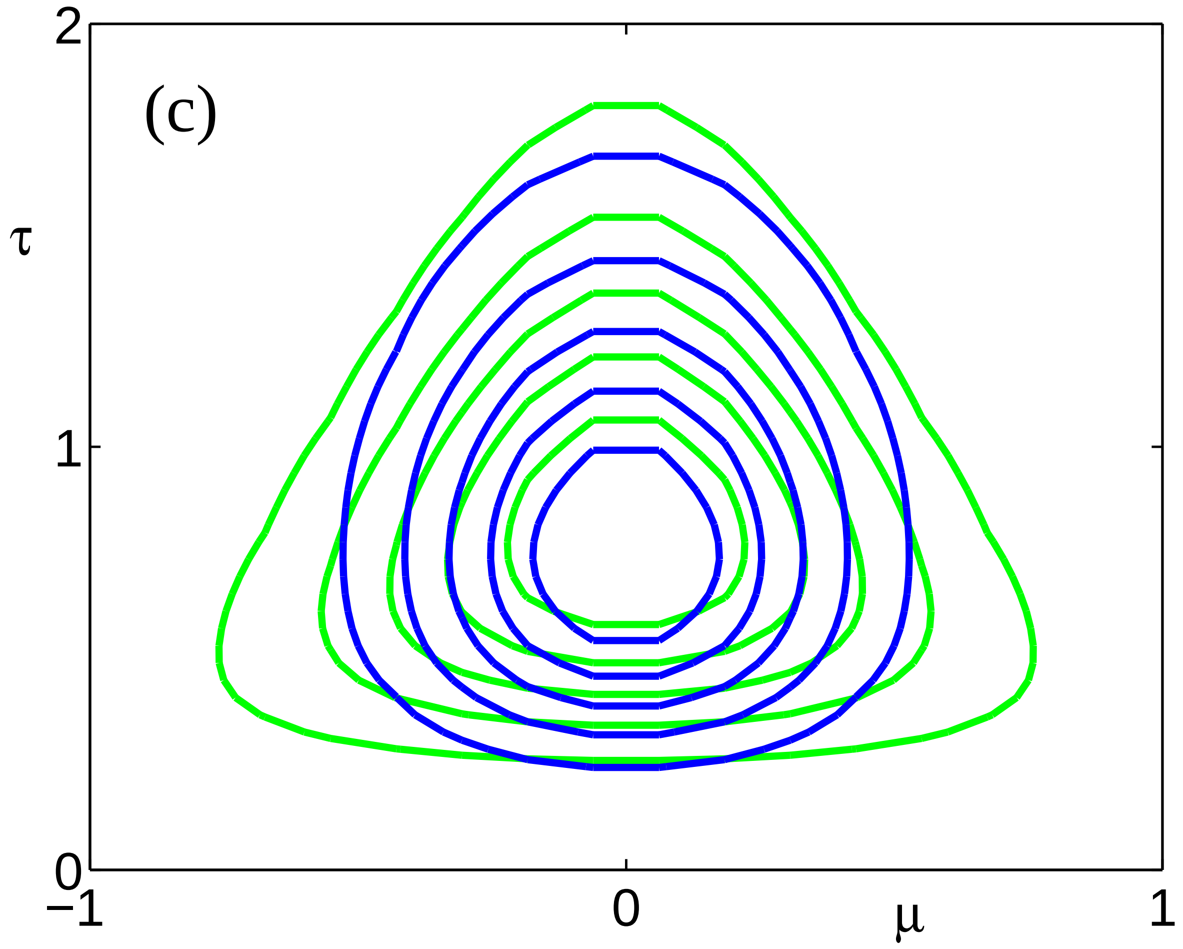
Initial Configuration



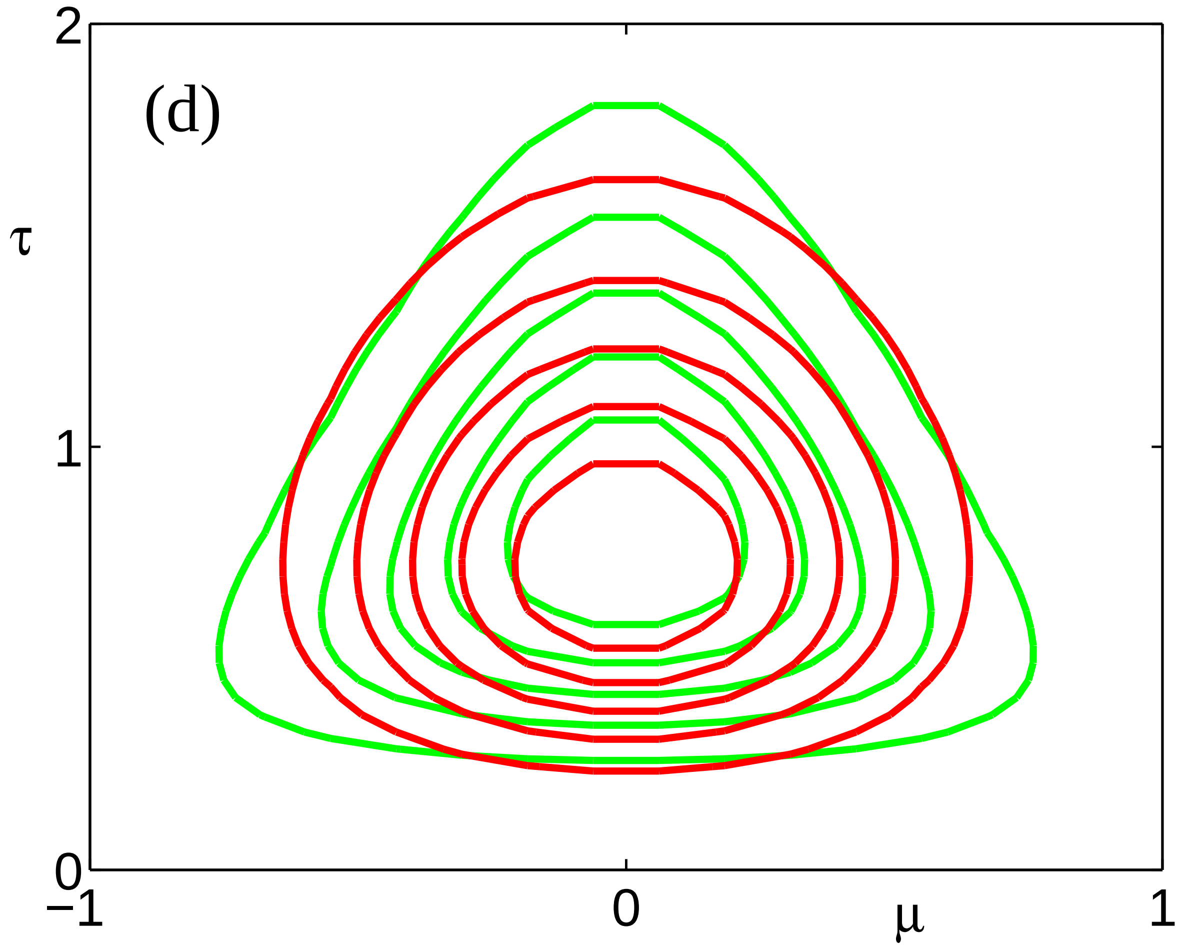
After Updating $q(\mu)$



After Updating $q(\tau)$



Converged Solution



Variational Equations for GMM

$$
\begin{aligned}
q^\star(\mathbf{z}) &= \prod_{n=1}^{N}\prod_{k=1}^{K} r_{nk}^{z_{nk}} \\
r_{nk} &\propto \widetilde{\pi}_k \widetilde{\Lambda}_k^{1/2} \exp\left\{ -\frac{d}{2\beta_k} - \frac{\nu_k}{2}(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}} \mathbf{W}_k (\mathbf{x}_n - \mathbf{m}_k) \right\} \\
\ln \widetilde{\Lambda}_k &= \sum_{i=1}^{d} \psi\left(\frac{\nu_k + 1 - i}{2}\right) + d\ln 2 - \ln|\mathbf{W}_k| \\
\ln \widetilde{\pi}_k &= \psi(\alpha_k) - \psi(\overline{\alpha}) \\
q^\star(\boldsymbol{\pi}) &= \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \qquad \alpha_k = \alpha_0 + N_k \qquad \nu_k = \nu_0 + N_k \\
q^\star(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) &= \mathcal{N}(\boldsymbol{\mu}_k|\mathbf{m}_k, \beta_k^{-1}\boldsymbol{\Lambda}_k^{-1})\,\mathcal{W}(\boldsymbol{\Lambda}_k|\mathbf{W}_k, \nu_k) \\
\mathbf{m}_k &= \frac{1}{\beta_k}(\beta_0\mathbf{m}_0 + N_k\overline{\mathbf{x}}_k) \qquad \beta_k = \beta_0 + N_k \\
\mathbf{W}_k^{-1} &= \mathbf{W}_0^{-1} + N_k\mathbf{S}_k + \frac{\beta_0 N_k}{\beta_0 + N_k}(\overline{\mathbf{x}}_k - \mathbf{m}_0)(\overline{\mathbf{x}}_k - \mathbf{m}_0)^{\mathrm{T}}
\end{aligned}
$$

Sufficient Statistics

$$
\begin{aligned}
N_k &= \sum_{n=1}^{N} \langle z_{nk} \rangle \\
\overline{\mathbf{x}}_k &= \frac{1}{N_k} \sum_{n=1}^{N} \langle z_{nk} \rangle \mathbf{x}_n \\
\mathbf{S}_k &= \frac{1}{N_k} \sum_{n=1}^{N} \langle z_{nk} \rangle (\mathbf{x}_n - \overline{\mathbf{x}}_k)(\mathbf{x}_n - \overline{\mathbf{x}}_k)^{\mathsf{T}}
\end{aligned}
$$

- Small computational overhead compared to maximum likelihood EM

Bayesian Model Comparison

- Multiple models (e.g. different values of K) with priors

$$p(\mathcal{M}_i)$$

- Posterior probabilities

$$p(\mathcal{M}_i|D) \propto p(D|\mathcal{M}_i)p(\mathcal{M}_i)$$

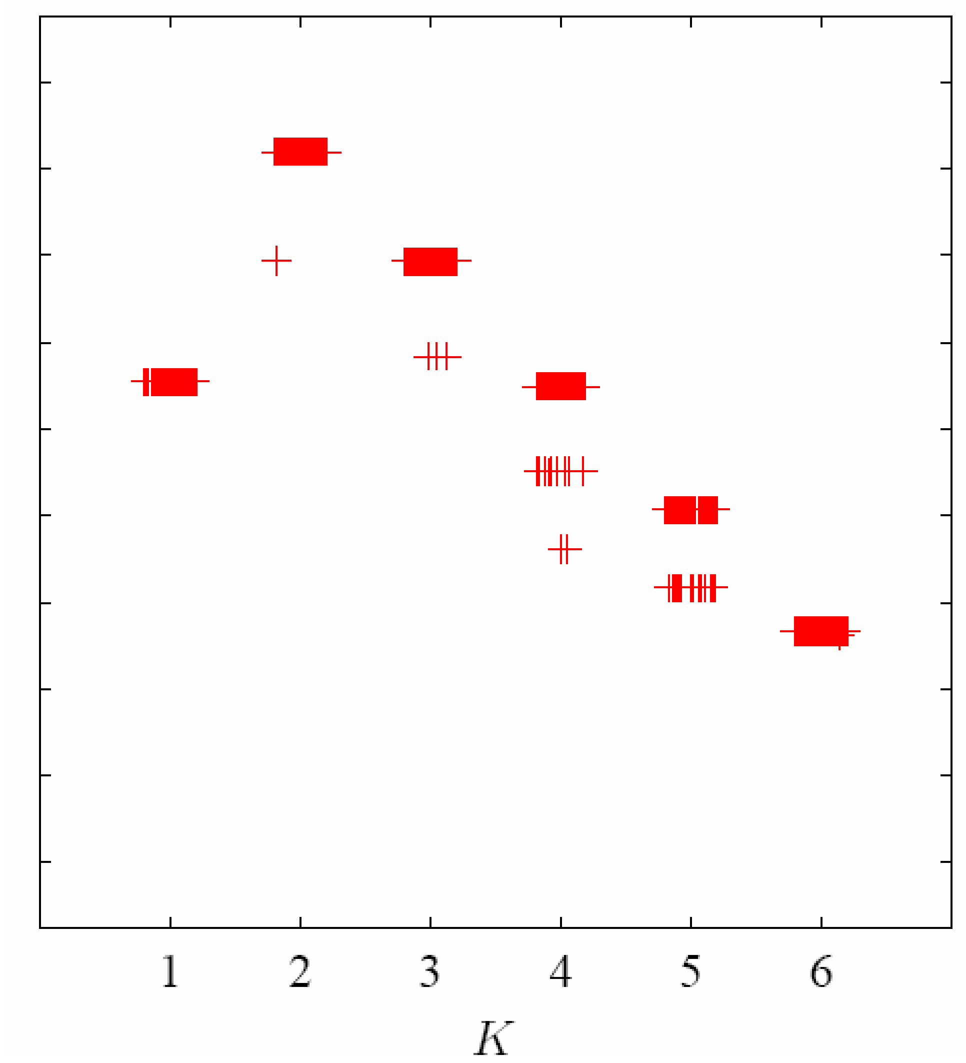
- For equal priors, models are compared using *evidence*

$$p(D|\mathcal{M}_i)$$

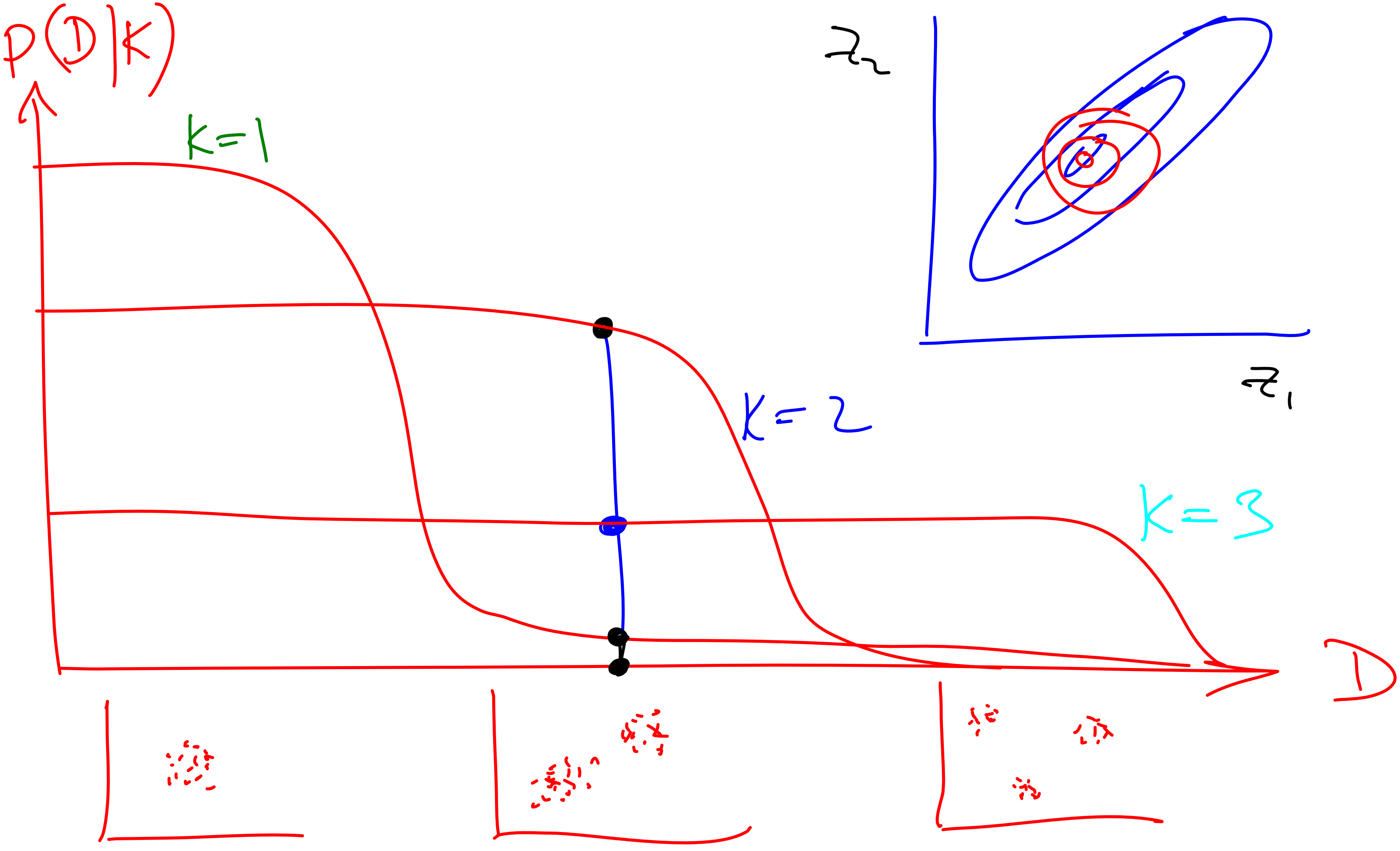
- Variational inference maximizes lower bound on $p(D|\mathcal{M}_i)$

Evidence vs. K for Old Faithful

$\mathcal{L} \leq \ln p(D|K)$



Bayesian Model Complexity

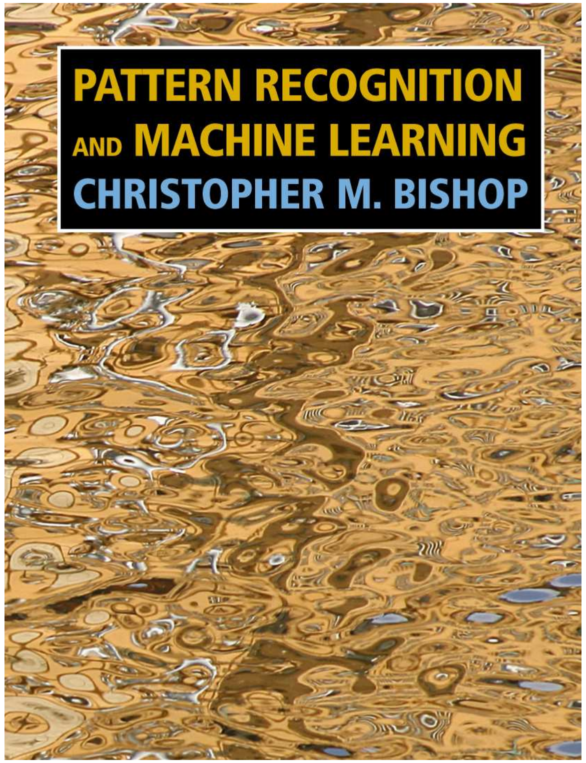


Take-home Messages

- Maximum likelihood gives severe over-fitting
  - singularities
  - favours ever larger numbers of components
- Bayesian mixture of Gaussians
  - no singularities
  - determines optimal number of components
- Variational inference
  - effective solution for Bayesian GMM
  - little computational overhead compared to EM

Viewgraphs, tutorials and publications available from:

http://research.microsoft.com/~cmbishop


PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP