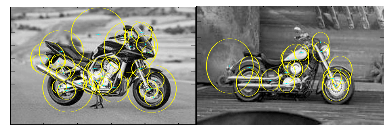
Expectation Maximization

Greg Mori - CMPT 419/726

Bishop PRML Ch. 9

Learning Parameters to Probability Distributions

- We discussed probabilistic models at length
- In assignment 3 you showed that given fully observed training data, setting parameters $\boldsymbol{\theta}_i$ to probability distributions is straight-forward
- However, in many settings not all variables are observed (labelled) in the training data: $\boldsymbol{x}_i = (\boldsymbol{x}_i, \boldsymbol{h}_i)$
  - e.g. Speech recognition: have speech signals, but not phoneme labels
  - e.g. Object recognition: have object labels (car, bicycle), but not part labels (wheel, door, seat)
  - Unobserved variables are called latent variables



figs from Fergus et al.

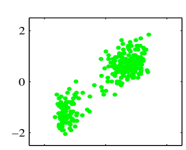
Outline

K-Means

Gaussian Mixture Models

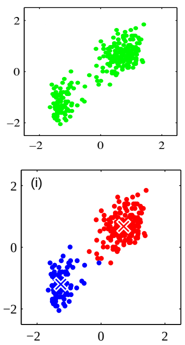
Expectation-Maximization

Unsupervised Learning



- We will start with an unsupervised learning (clustering) problem:
- Given a dataset $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, each $\boldsymbol{x}_i \in \mathbb{R}^D$, partition the dataset into K clusters
  - Intuitively, a cluster is a group of points, which are close together and far from others

Distortion Measure



- Formally, introduce prototypes (or cluster centers) $\boldsymbol{\mu}_k \in \mathbb{R}^D$
- Use binary r_{nk}, 1 if point n is in cluster k, 0 otherwise (1-of-K coding scheme again)
- Find $\{\boldsymbol{\mu}_k\}$, $\{r_{nk}\}$ to minimize distortion measure:

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

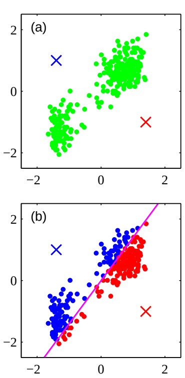
Minimizing Distortion Measure

- Minimizing J directly is hard

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

- However, two things are easy
  - If we know $\boldsymbol{\mu}_k$, minimizing J wrt r_{nk}
  - If we know r_{nk}, minimizing J wrt $\boldsymbol{\mu}_k$
- This suggests an iterative procedure
  - Start with initial guess for $\boldsymbol{\mu}_k$
  - Iteration of two steps:
    - Minimize J wrt r_{nk}
    - Minimize J wrt $\boldsymbol{\mu}_k$
  - Rinse and repeat until convergence

Determining Membership Variables



- Step 1 in an iteration of K-means is to minimize distortion measure J wrt cluster membership variables r_{nk}

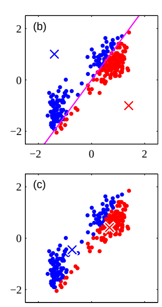
$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

- Terms for different data points $\boldsymbol{x}_n$ are independent, for each data point set r_{nk} to minimize

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

- Simply set $r_{nk} = 1$ for the cluster center $\boldsymbol{\mu}_k$ with smallest distance

Determining Cluster Centers



- Step 2: fix r_{nk}, minimize J wrt the cluster centers $\boldsymbol{\mu}_k$

$$J = \sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2 \text{ switch order of sums}$$

- So we can minimze wrt each $\boldsymbol{\mu}_k$ separately
- Take derivative, set to zero:

$$2 \sum_{n=1}^{N} r_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0$$

$$\Leftrightarrow \boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$$

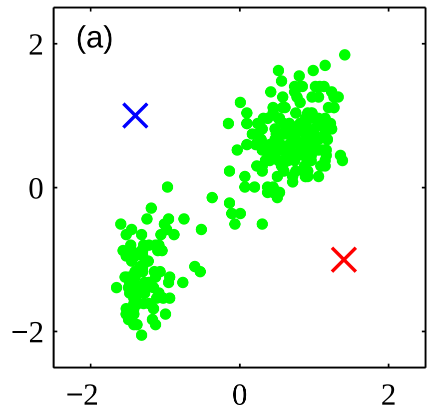
i.e. mean of datapoints $\boldsymbol{x}_n$ assigned to cluster k

K-means Algorithm

- Start with initial guess for $\boldsymbol{\mu}_k$
- Iteration of two steps:
  - Minimize J wrt r_{nk}
    - Assign points to nearest cluster center
  - Minimize J wrt $\boldsymbol{\mu}_k$
    - Set cluster center as average of points in cluster
- Rinse and repeat until convergence

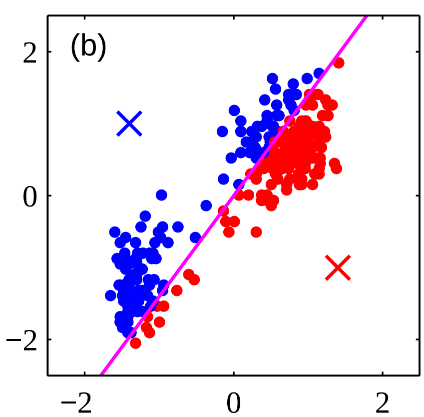
K-Means Gaussian Mixture Models Expectation-Maximization

K-means example



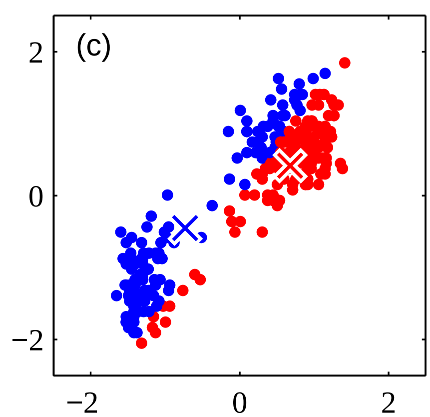
K-Means Gaussian Mixture Models Expectation-Maximization

K-means example

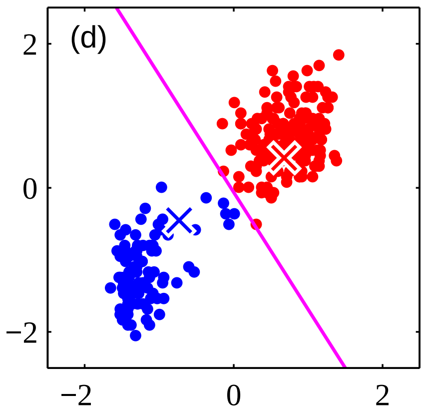


K-Means Gaussian Mixture Models Expectation-Maximization

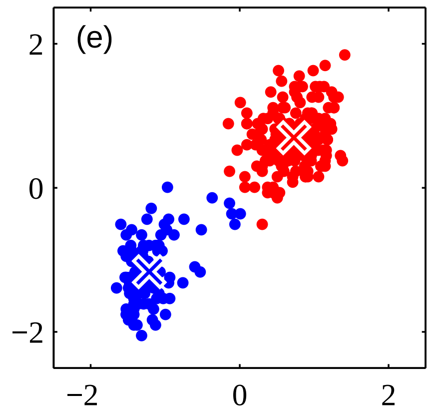
K-means example



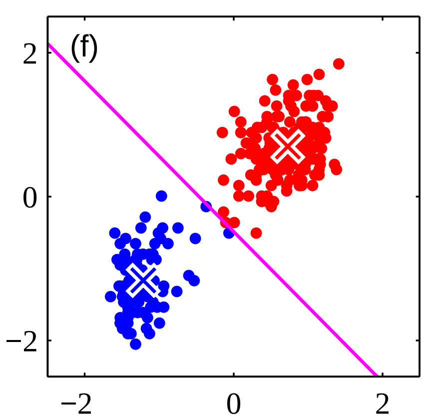
K-means example



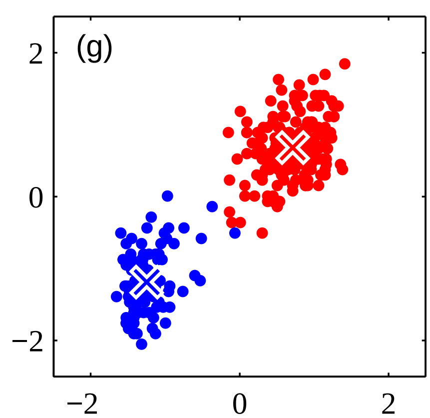
K-means example



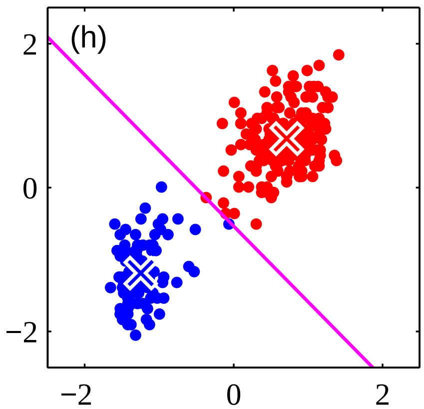
K-means example



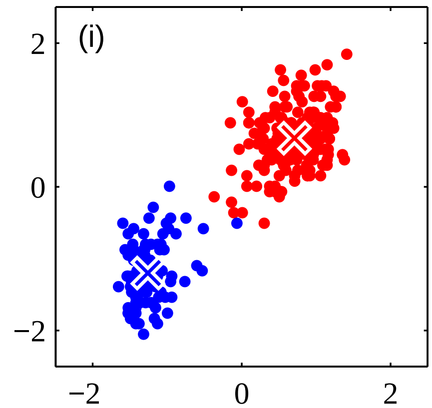
K-means example



K-means example



K-means example

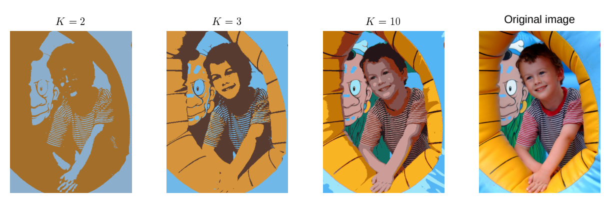


Next step doesn't change membership – stop

K-means Convergence

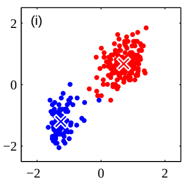
- Repeat steps until no change in cluster assignments
- For each step, value of J either goes down, or we stop
- Finite number of possible assignments of data points to clusters, so we are guarranteed to converge eventually
- Note it may be a local maximum rather than a global maximum to which we converge

K-means Example - Image Segmentation



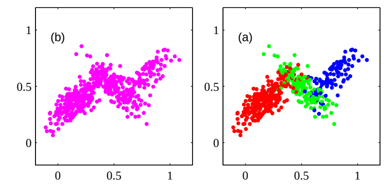
- K-means clustering on pixel colour values
- Pixels in a cluster are coloured by cluster mean
- Represent each pixel (e.g. 24-bit colour value) by a cluster number (e.g. 4 bits for $K = 10$), compressed version
- This technique known as vector quantization
  - Represent vector (in this case from RGB, $\mathbb{R}^3$) as a single discrete value

Hard Assignment vs. Soft Assignment



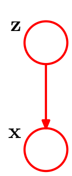
- In the K-means algorithm, a hard assignment of points to clusters is made
- However, for points near the decision boundary, this may not be such a good idea
- Instead, we could think about making a soft assignment of points to clusters

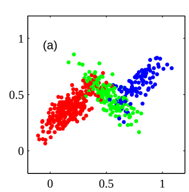
Gaussian Mixture Model



- The Gaussian mixture model (or mixture of Gaussians MoG) models the data as a combination of Gaussians
- Above shows a dataset generated by drawing samples from three different Gaussians

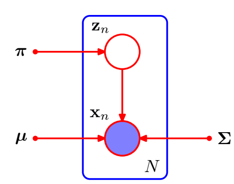
Generative Model

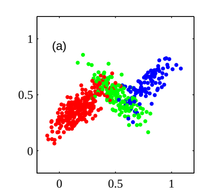




- The mixture of Gaussians is a generative model
- To generate a datapoint $\boldsymbol{x}_n$, we first generate a value for a discrete variable $z_n \in \{1, \ldots, K\}$
- We then generate a value $\boldsymbol{x}_n \sim \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for the corresponding Gaussian

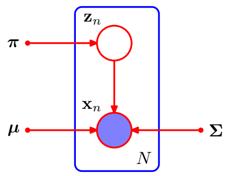
Graphical Model

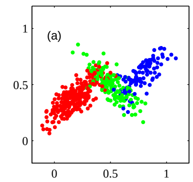




- Full graphical model using plate notation
  - Note z_n is a latent variable, unobserved
- Need to give conditional distributions $p(z_n)$ and $p(\boldsymbol{x}_n|z_n)$
- The one-of-K representation is helpful here: $z_{nk} \in \{0, 1\}$, $\boldsymbol{z}_n = (z_{n1}, \ldots, z_{nK})$

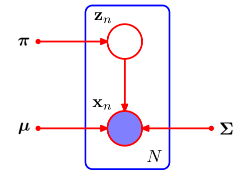
Graphical Model - Latent Component Variable

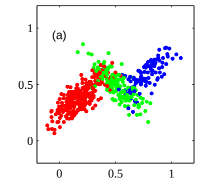




- Use a Bernoulli distribution for $p(z_n)$
  - i.e. $p(z_{nk} = 1) = \pi_k$
  - Parameters to this distribution $\{\pi_k\}$
  - Must have $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$
- $p(z_n) = \prod_{k=1}^{K} \pi_k^{z_{nk}}$

Graphical Model - Observed Variable

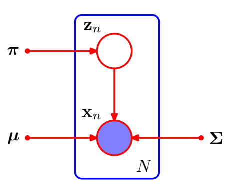


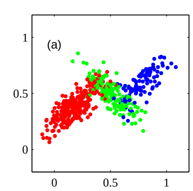


- Use a Gaussian distribution for $p(\boldsymbol{x}_n|z_n)$
  - Parameters to this distribution $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$

$$\begin{aligned} p(\boldsymbol{x}_n|z_{nk} = 1) &= \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ p(\boldsymbol{x}_n|z_n) &= \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}} \end{aligned}$$

Graphical Model - Joint distribution





- The full joint distribution is given by:

$$\begin{aligned} p(\boldsymbol{x}, z) &= \prod_{n=1}^{N} p(z_n) p(\boldsymbol{x}_n|z_n) \\ &= \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}} \end{aligned}$$

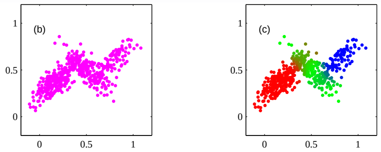
MoG Marginal over Observed Variables

- The marginal distribution $p(\boldsymbol{x}_n)$ for this model is:

$$\begin{aligned} p(\boldsymbol{x}_n) &= \sum_{z_n} p(\boldsymbol{x}_n, z_n) = \sum_{z_n} p(z_n) p(\boldsymbol{x}_n|z_n) \\ &= \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned}$$

- A mixture of Gaussians

MoG Conditional over Latent Variable



- The conditional $p(z_{nk} = 1|\boldsymbol{x}_n)$ will play an important role for learning
- It is denoted by $\gamma(z_{nk})$ can be computed as:

$$\begin{aligned}\gamma(z_{nk}) \equiv p(z_{nk} = 1|\boldsymbol{x}_n) &= \frac{p(z_{nk} = 1)p(\boldsymbol{x}_n|z_{nk} = 1)}{\sum_{j=1}^{K} p(z_{nj} = 1)p(\boldsymbol{x}_n|z_{nj} = 1)} \\ &= \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}\end{aligned}$$

- $\gamma(z_{nk})$ is the responsibility of component k for datapoint n

MoG Learning

- Given a set of observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, without the latent variables z_n, how can we learn the parameters?
  - Model parameters are $\boldsymbol{\theta} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
- Answer will be similar to k-means:
  - If we know the latent variables z_n, fitting the Gaussians is easy
  - If we know the Gaussians $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, finding the latent variables is easy
- Rather than latent variables, we will use responsibilities $\gamma(z_{nk})$

MoG Maximum Likelihood Learning

- Given a set of observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, without the latent variables z_n, how can we learn the parameters?
  - Model parameters are $\boldsymbol{\theta} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
- We can use the maximum likelihood criterion:

$$\begin{aligned}\boldsymbol{\theta}_{ML} &= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}\end{aligned}$$

- Unfortunately, closed-form solution not possible this time – log of sum rather than log of product

MoG Maximum Likelihood Learning - Problem

- Maximum likelihood criterion, 1-D:

$$\boldsymbol{\theta}_{ML} = \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-(x_n - \mu_k)^2/(2\sigma^2)\right\} \right\}$$

- Suppose we set $\mu_k = x_n$ for some k and n, then we have one term in the sum:

$$\begin{aligned}&\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left\{-(x_n - \mu_k)^2/(2\sigma^2)\right\} \\ =\ &\pi_k \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left\{-(0)^2/(2\sigma^2)\right\}\end{aligned}$$

- In the limit as $\sigma_k \rightarrow 0$, this goes to ∞
  - So ML solution is to set some $\mu_k = x_n$, and $\sigma_k = 0$!

ML for Gaussian Mixtures

- Keeping this problem in mind, we will develop an algorithm for ML estimation of the parameters for a MoG model
  - Search for a local optimum
- Consider the log-likelihood function

$$\ell(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

- We can try taking derivatives and setting to zero, even though no closed form solution exists

Maximizing Log-Likelihood - Means

$$\begin{aligned} \ell(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \\ \frac{\partial}{\partial \mu_k} \ell(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) \\ &= \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) \end{aligned}$$

- Setting derivative to 0, and multiply by $\boldsymbol{\Sigma}_k$

$$\sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{\mu}_k = \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n$$

$$\Leftrightarrow \boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n \text{ where } N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

Maximizing Log-Likelihood - Means and Covariances

- Note that the mean $\boldsymbol{\mu}_k$ is a weighted combination of points $\boldsymbol{x}_n$, using the responsibilities $\gamma(z_{nk})$ for the cluster k

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n$$

  - $N_k = \sum_{n=1}^{N} \gamma(z_{nk})$ is the effective number of points in the cluster
- A similar result comes from taking derivatives wrt the covariance matrices $\boldsymbol{\Sigma}_k$:

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T$$

Maximizing Log-Likelihood - Mixing Coefficients

- We can also maximize wrt the mixing coefficients π_k
- Note there is a constraint that $\sum_k \pi_k = 1$
  - Use Lagrange multipliers, c.f. Chapter 7
- End up with:

$$\pi_k = \frac{N_k}{N}$$

average responsibility that component k takes

Three Parameter Types and Three Equations

- These three equations a solution does not make

$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

- All depend on $\gamma(z_{nk})$, which depends on all 3!
- But an iterative scheme can be used

EM for Gaussian Mixtures

- Initialize parameters, then iterate:
  - **E step**: Calculate responsibilities using current parameters

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

  - **M step**: Re-estimate parameters using these $\gamma(z_{nk})$

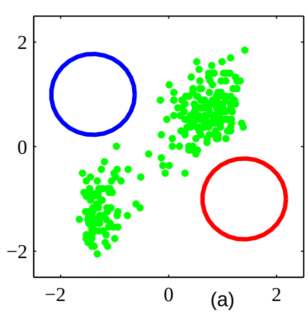
$$\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{x}_n-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

- This algorithm is known as the expectation-maximization algorithm (EM)
  - Next we describe its general form, why it works, and why it's called EM (but first an example)

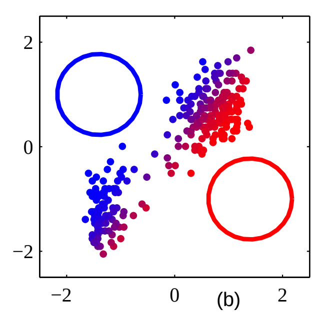
MoG EM - Example



(a)

- Same initialization as with K-means before
  - Often, K-means is actually used to initialize EM

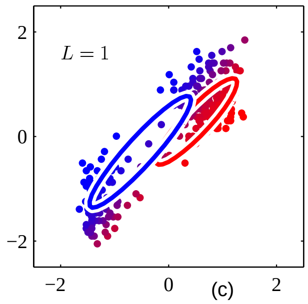
MoG EM - Example



(b)

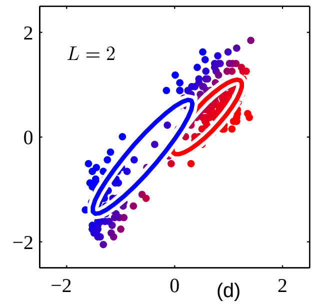
- Calculate responsibilities $\gamma(z_{nk})$

MoG EM - Example



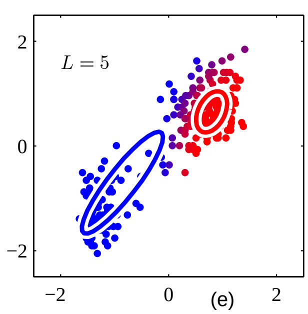
- Calculate model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ using these responsibilities

MoG EM - Example



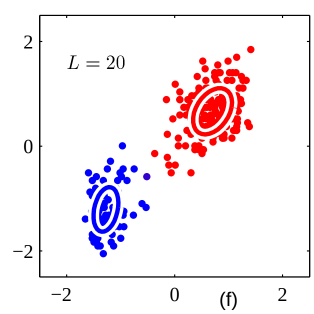
- Iteration 2

MoG EM - Example



- Iteration 5

MoG EM - Example



- Iteration 20 - converged

General Version of EM

- In general, we are interested in maximizing the likelihood

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

where $\mathbf{X}$ denotes all observed variables, and $\mathbf{Z}$ denotes all latent (hidden, unobserved) variables

- Assume that maximizing $p(\mathbf{X}|\boldsymbol{\theta})$ is difficult (e.g. mixture of Gaussians)
- But maximizing $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is tractable (everything observed)
  - $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is referred to as the complete-data likelihood function, which we don't have

A Lower Bound

- The strategy for optimization will be to introduce a lower bound on the likelihood
  - This lower bound will be based on the complete-data likelihood, which is easy to optimize
- Iteratively increase this lower bound
- Make sure we're increasing the likelihood while doing so

A Decomposition Trick

- To obtain the lower bound, we use a decomposition:

$$\begin{aligned}
\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) &= \ln p(\mathbf{X}|\boldsymbol{\theta}) + \ln p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}) \text{ product rule} \\
\ln p(\mathbf{X}|\boldsymbol{\theta}) &= \mathcal{L}(q, \boldsymbol{\theta}) + KL(q||p) \\
\mathcal{L}(q, \boldsymbol{\theta}) &\equiv \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \\
KL(q||p) &\equiv -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Z})} \right\}
\end{aligned}$$

- $KL(q||p)$ is known as the Kullback-Leibler divergence (KL-divergence), and is ≥ 0 (see p.55 PRML, next slide)
  - Hence $\ln p(\mathbf{X}|\boldsymbol{\theta}) \geq \mathcal{L}(q, \boldsymbol{\theta})$

Kullback-Leibler Divergence

- $KL(p(x)||q(x))$ is a measure of the difference between distributions $p(x)$ and $q(x)$:

$$KL(p(x)||q(x)) = -\sum_x p(x) \log \frac{q(x)}{p(x)}$$

  - Motivation: average additional amount of information required to encode x using code assuming distribution $q(x)$ when x actually comes from $p(x)$
  - Note it is not symmetric: $KL(q(x)||p(x)) \neq KL(p(x)||q(x))$ in general
  - It is non-negative:
    - Jensen's inequality: $-\ln(\sum_x x p(x)) \leq -\sum_x p(x) \ln x$
    - Apply to KL:

$$KL(p||q) = -\sum_x p(x) \log \frac{q(x)}{p(x)} \geq -\ln \left(\sum_x \frac{q(x)}{p(x)} p(x) \right) = -ln \sum_x q(x) = 0$$

Increasing the Lower Bound - E step

- EM is an iterative optimization technique which tries to maximize this lower bound: $\ln p(\boldsymbol{X}|\boldsymbol{\theta}) \geq \mathcal{L}(q, \boldsymbol{\theta})$
- **E step**: Fix $\boldsymbol{\theta}^{old}$, maximize $\mathcal{L}(q, \boldsymbol{\theta}^{old})$ wrt q
  - i.e. Choose distribution q to maximize $\mathcal{L}$
  - Reordering bound:

$$\mathcal{L}(q, \boldsymbol{\theta}^{old}) = \ln p(\boldsymbol{X}|\boldsymbol{\theta}^{old}) - KL(q||p)$$

  - $\ln p(\boldsymbol{X}|\boldsymbol{\theta}^{old})$ does not depend on q
  - Maximum is obtained when $KL(q||p)$ is as small as possible
    - Occurs when $q = p$, i.e. $q(\boldsymbol{Z}) = p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta})$
    - This is the posterior over $\boldsymbol{Z}$, recall these are the responsibilities from MoG model

Increasing the Lower Bound - M step

- **M step**: Fix q, maximize $\mathcal{L}(q, \boldsymbol{\theta})$ wrt $\boldsymbol{\theta}$
- The maximization problem is on

$$\mathcal{L}(q, \boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}) - \sum_{\boldsymbol{Z}} q(\boldsymbol{Z}) \ln q(\boldsymbol{Z})$$
$$= \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}) - \sum_{\boldsymbol{Z}} p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old}) \ln p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$$

- Second term is constant with respect to $\boldsymbol{\theta}$
- First term is ln of complete data likelihood, which is assumed easy to optimize
  - Expected complete log likelihood – what we think complete data likelihood will be

Why does EM work?

- In the M-step we changed from $\boldsymbol{\theta}^{old}$ to $\boldsymbol{\theta}^{new}$
- This increased the lower bound $\mathcal{L}$, unless we were at a maximum (so we would have stopped)
- This must have caused the log likelihood to increase
  - The E-step set q to make the KL-divergence 0:

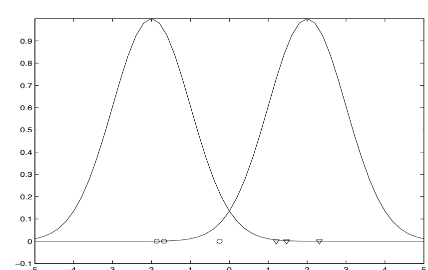
$$\ln p(\boldsymbol{X}|\boldsymbol{\theta}^{old}) = \mathcal{L}(q, \boldsymbol{\theta}^{old}) + KL(q||p) = \mathcal{L}(q, \boldsymbol{\theta}^{old})$$

  - Since the lower bound $\mathcal{L}$ increased when we moved from $\boldsymbol{\theta}^{old}$ to $\boldsymbol{\theta}^{new}$:

$$\begin{aligned} \ln p(\boldsymbol{X}|\boldsymbol{\theta}^{old}) &= \mathcal{L}(q, \boldsymbol{\theta}^{old}) < \mathcal{L}(q, \boldsymbol{\theta}^{new}) \\ &= \ln p(\boldsymbol{X}|\boldsymbol{\theta}^{new}) - KL(q||p^{new}) \end{aligned}$$

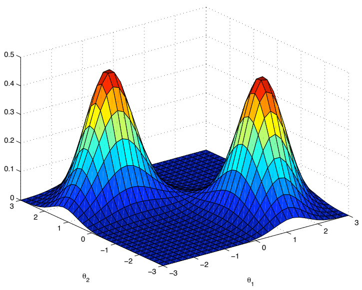
  - So the log-likelihood has increased going from $\boldsymbol{\theta}^{old}$ to $\boldsymbol{\theta}^{new}$

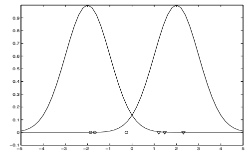
Bounding Example



Consider 2 component 1-D MoG with known variances (example from F. Dellaert)

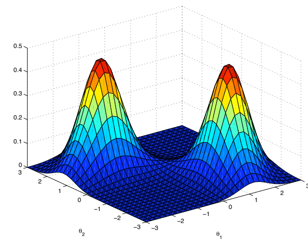
Bounding Example

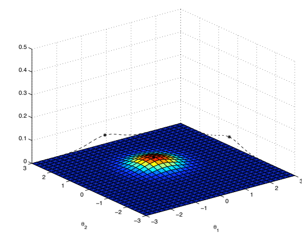




- True likelihood function
  - Recall we're fitting means θ_1, θ_2

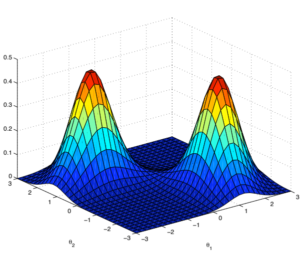
Bounding Example

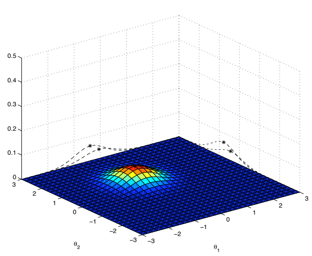




- Lower bound the likelihood function using averaging distribution $q(Z)$
  - $\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{Z})||p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}))$
  - Since $q(\mathbf{Z}) = p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$, bound is tight (equal to actual likelihood) at $\boldsymbol{\theta} = \boldsymbol{\theta}^{old}$

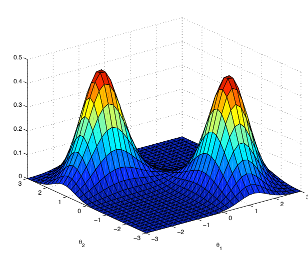
Bounding Example

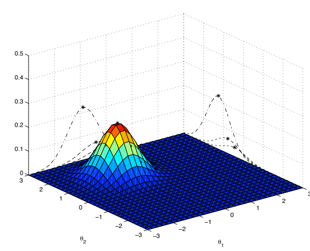




- Lower bound the likelihood function using averaging distribution $q(Z)$
  - $\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{Z})||p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}))$
  - Since $q(\mathbf{Z}) = p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$, bound is tight (equal to actual likelihood) at $\boldsymbol{\theta} = \boldsymbol{\theta}^{old}$

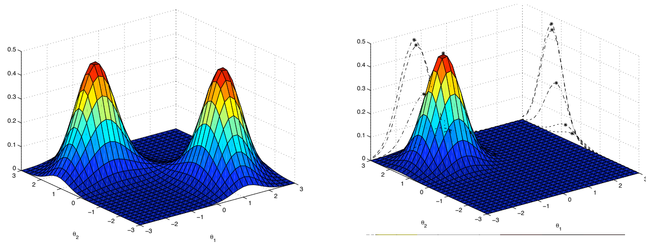
Bounding Example





- Lower bound the likelihood function using averaging distribution $q(Z)$
  - $\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{Z})||p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}))$
  - Since $q(\mathbf{Z}) = p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$, bound is tight (equal to actual likelihood) at $\boldsymbol{\theta} = \boldsymbol{\theta}^{old}$

Bounding Example



- Lower bound the likelihood function using averaging distribution $q(Z)$
  - $\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{Z})||p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}))$
  - Since $q(\mathbf{Z}) = p(\mathbf{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old})$, bound is tight (equal to actual likelihood) at $\boldsymbol{\theta} = \boldsymbol{\theta}^{old}$

EM - Summary

- EM finds local maximum to likelihood

$$p(\boldsymbol{X}|\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta})$$

- Iterates two steps:
  - **E step** “fills in” the missing variables $\boldsymbol{Z}$ (calculates their distribution)
  - **M step** maximizes expected complete log likelihood (expectation wrt **E step** distribution)
- This works because these two steps are performing a coordinate-wise hill-climbing on a lower bound on the likelihood $p(\boldsymbol{X}|\boldsymbol{\theta})$

Conclusion

- Readings: Ch. 9.1, 9.2, 9.4
- K-means clustering
- Gaussian mixture model
- What about K?
  - Model selection: either cross-validation or Bayesian version (average over all values for K)
- Expectation-maximization, a general method for learning parameters of models when not all variables are observed