CSC 411: Lecture 13: Mixtures of Gaussians and EM

Richard Zemel, Raquel Urtasun and Sanja Fidler

University of Toronto

Today

- Mixture of Gaussians
- EM algorithm
- Latent Variables

A Generative View of Clustering

- Last time: hard and soft k-means algorithm

A Generative View of Clustering

- Last time: hard and soft k-means algorithm
- Today: statistical formulation of clustering $\rightarrow$ principled, justification for updates

A Generative View of Clustering

- Last time: hard and soft k-means algorithm
- Today: statistical formulation of clustering $\rightarrow$ principled, justification for updates
- We need a sensible measure of what it means to cluster the data well
  - This makes it possible to judge different methods
  - It may help us decide on the number of clusters

A Generative View of Clustering

- Last time: hard and soft k-means algorithm
- Today: statistical formulation of clustering $\rightarrow$ principled, justification for updates
- We need a sensible measure of what it means to cluster the data well
  - This makes it possible to judge different methods
  - It may help us decide on the number of clusters
- An obvious approach is to imagine that the data was produced by a generative model

A Generative View of Clustering

- Last time: hard and soft k-means algorithm
- Today: statistical formulation of clustering $\rightarrow$ principled, justification for updates
- We need a sensible measure of what it means to cluster the data well
  - This makes it possible to judge different methods
  - It may help us decide on the number of clusters
- An obvious approach is to imagine that the data was produced by a generative model
  - Then we adjust the model parameters to maximize the probability that it would produce exactly the data we observed

Gaussian Mixture Model (GMM)

- A Gaussian mixture model represents a **distribution** as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

Gaussian Mixture Model (GMM)

- A Gaussian mixture model represents a **distribution** as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

- GMM is a density estimator

Gaussian Mixture Model (GMM)

- A Gaussian mixture model represents a **distribution** as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

- GMM is a density estimator
- Where have we already used a density estimator?

Gaussian Mixture Model (GMM)

- A Gaussian mixture model represents a **distribution** as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

with π_k the mixing coefficients, where:

$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

- GMM is a density estimator
- Where have we already used a density estimator?
- We know that neural nets are universal approximators of functions

Gaussian Mixture Model (GMM)

- A Gaussian mixture model represents a **distribution** as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

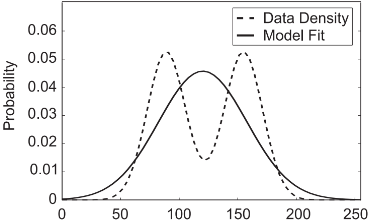
with π_k the mixing coefficients, where:

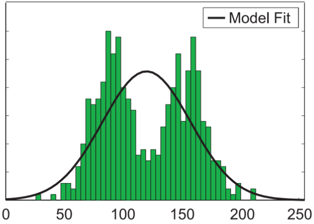
$$\sum_{k=1}^{K} \pi_k = 1 \quad \text{and} \quad \pi_k \geq 0 \quad \forall k$$

- GMM is a density estimator
- Where have we already used a density estimator?
- We know that neural nets are universal approximators of functions
- GMMs are **universal approximators of densities** (if you have enough Gaussians). Even diagonal GMMs are universal approximators.

Visualizing a Mixture of Gaussians – 1D Gaussians

- In the beginning of class, we tried to fit a Gaussian to data:

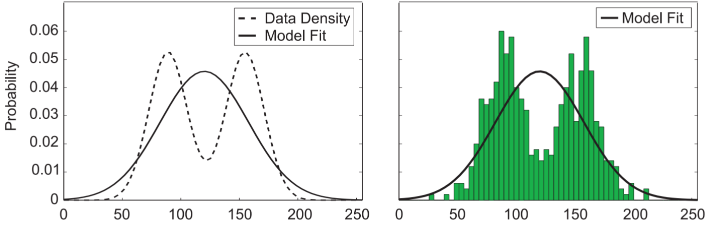




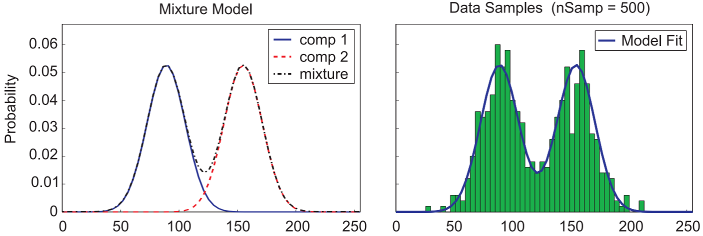
[Slide credit: K. Kutulakos]

Visualizing a Mixture of Gaussians – 1D Gaussians

- In the beginning of class, we tried to fit a Gaussian to data:

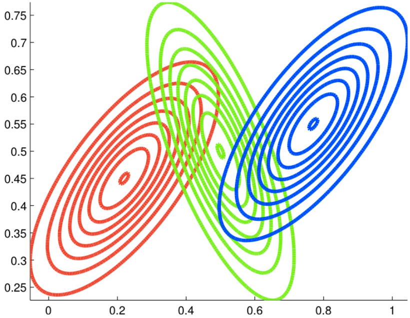


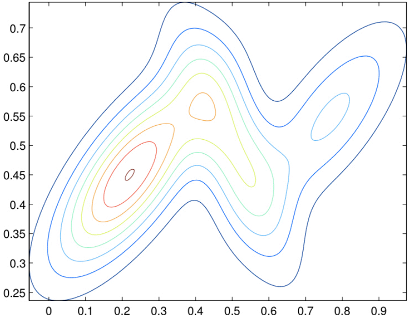
- Now, we are trying to fit a GMM (with $K = 2$ in this example):

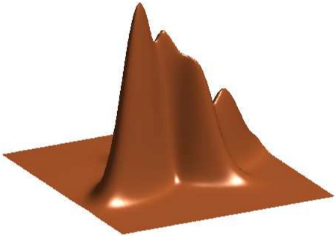


[Slide credit: K. Kutulakos]

Visualizing a Mixture of Gaussians – 2D Gaussians







Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

  w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point
  - Identifiability: Solution is up to permutations

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

  w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point
  - Identifiability: Solution is up to permutations

- How would you optimize this?

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point
  - Identifiability: Solution is up to permutations
- How would you optimize this?
- Can we have a closed form update?

Fitting GMMs: Maximum Likelihood

- Maximum likelihood maximizes

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

  w.r.t $\Theta = \{\pi_k, \mu_k, \Sigma_k\}$

- Problems:
  - Singularities: Arbitrarily large likelihood when a Gaussian explains a single point
  - Identifiability: Solution is up to permutations
- How would you optimize this?
- Can we have a closed form update?
- Don't forget to satisfy the constraints on π_k

Trick: Introduce a Latent Variable

- Introduce a hidden variable such that its knowledge would simplify the maximization

Trick: Introduce a Latent Variable

- Introduce a hidden variable such that its knowledge would simplify the maximization
- We could introduce a hidden (latent) variable z which would represent which Gaussian generated our observation $\mathbf{x}$, with some probability

Trick: Introduce a Latent Variable

- Introduce a hidden variable such that its knowledge would simplify the maximization
- We could introduce a hidden (latent) variable z which would represent which Gaussian generated our observation $\mathbf{x}$, with some probability
- Let $z \sim \text{Categorical}(\boldsymbol{\pi}) \quad$ (where $\pi_k \geq 0, \quad \sum_k \pi_k = 1$)

Trick: Introduce a Latent Variable

- Introduce a hidden variable such that its knowledge would simplify the maximization
- We could introduce a hidden (latent) variable z which would represent which Gaussian generated our observation $\mathbf{x}$, with some probability
- Let $z \sim \mathrm{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Then:

$$
\begin{aligned}
p(\mathbf{x}) &= \sum_{k=1}^{K} p(\mathbf{x}, z = k) \\
&= \sum_{k=1}^{K} \underbrace{p(z = k)}_{\pi_k} \underbrace{p(\mathbf{x}|z = k)}_{\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}
\end{aligned}
$$

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets
- Variables which are always unobserved are called latent variables, or sometimes hidden variables

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets
- Variables which are always unobserved are called latent variables, or sometimes hidden variables
- We may want to intentionally introduce latent variables to model complex dependencies between variables – this can actually simplify the model

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets
- Variables which are always unobserved are called latent variables, or sometimes hidden variables

- We may want to intentionally introduce latent variables to model complex dependencies between variables – this can actually simplify the model
- Form of divide-and-conquer: use simple parts to build complex models

Latent Variable Models

- Some model variables may be unobserved, either at training or at test time, or both
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets
- Variables which are always unobserved are called latent variables, or sometimes hidden variables

- We may want to intentionally introduce latent variables to model complex dependencies between variables – this can actually simplify the model
- Form of divide-and-conquer: use simple parts to build complex models

- In a mixture model, the identity of the component that generated a given datapoint is a latent variable

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \mathrm{Categorical}(\boldsymbol{\pi})$ $\quad$ (where $\pi_k \geq 0, \quad \sum_k \pi_k = 1$)

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \mathrm{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \text{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$
- Log-likelihood:

$$\begin{aligned} \ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) &= \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma) \\ &= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma) p(z^{(n)}|\, \boldsymbol{\pi}) \end{aligned}$$

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \mathrm{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$
- Log-likelihood:

$$\begin{aligned}\ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) &= \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma) \\ &= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma) p(z^{(n)}|\, \boldsymbol{\pi})\end{aligned}$$

- Note: We have a hidden variable $z^{(n)}$ for every observation

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \text{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$
- Log-likelihood:

$$\begin{aligned} \ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) &= \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma) \\ &= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma)p(z^{(n)}|\, \boldsymbol{\pi}) \end{aligned}$$

- Note: We have a hidden variable $z^{(n)}$ for every observation
- General problem: sum inside the log

Back to GMM

- A Gaussian mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

- We had: $z \sim \mathrm{Categorical}(\boldsymbol{\pi})$ (where $\pi_k \geq 0$, $\sum_k \pi_k = 1$)
- Joint distribution: $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$
- Log-likelihood:

$$\begin{aligned}\ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) &= \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma) \\ &= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma)p(z^{(n)}|\, \boldsymbol{\pi})\end{aligned}$$

- Note: We have a hidden variable $z^{(n)}$ for every observation
- General problem: sum inside the log
- How can we optimize this?

Maximum Likelihood

- **If we knew** $z^{(n)}$ for every $x^{(n)}$, the maximum likelihood problem is easy:

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(x^{(n)}, z^{(n)} | \pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)} | \, z^{(n)}; \mu, \Sigma) + \ln p(z^{(n)} | \, \boldsymbol{\pi})$$

Maximum Likelihood

- **If we knew** $z^{(n)}$ for every $x^{(n)}$, the maximum likelihood problem is easy:

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(x^{(n)}, z^{(n)} | \pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)} | \, z^{(n)}; \mu, \Sigma) + \ln p(z^{(n)} | \, \boldsymbol{\pi})$$

- We have been optimizing something similar for Gaussian bayes classifiers

Maximum Likelihood

- **If we knew** $z^{(n)}$ for every $x^{(n)}$, the maximum likelihood problem is easy:

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(x^{(n)}, z^{(n)} | \pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)} | \, z^{(n)}; \mu, \Sigma) + \ln p(z^{(n)} | \, \boldsymbol{\pi})$$

- We have been optimizing something similar for Gaussian bayes classifiers
- We would get this (check old slides):

$$\begin{aligned}
\mu_k &= \frac{\sum_{n=1}^{N} 1_{[z^{(n)}=k]} \, \mathbf{x}^{(n)}}{\sum_{n=1}^{N} 1_{[z^{(n)}=k]}} \\
\Sigma_k &= \frac{\sum_{n=1}^{N} 1_{[z^{(n)}=k]} \, (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T}{\sum_{n=1}^{N} 1_{[z^{(n)}=k]}} \\
\pi_k &= \frac{1}{N} \sum_{n=1}^{N} 1_{[z^{(n)}=k]}
\end{aligned}$$

Intuitively, How Can We Fit a Mixture of Gaussians?

- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:

Intuitively, How Can We Fit a Mixture of Gaussians?

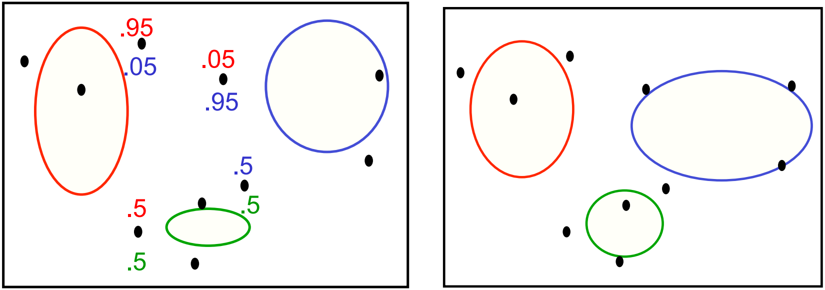
- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:
    1. E-step: Compute the posterior probability that each Gaussian generates each datapoint (as this is unknown to us)

Intuitively, How Can We Fit a Mixture of Gaussians?

- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:
  1. E-step: Compute the posterior probability that each Gaussian generates each datapoint (as this is unknown to us)
  2. M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.

Intuitively, How Can We Fit a Mixture of Gaussians?

- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:
  1. E-step: Compute the posterior probability that each Gaussian generates each datapoint (as this is unknown to us)
  2. M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.



Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

  1. E-step:
     - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?

Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

  1. E-step:
     - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
     - We cannot be sure, so it's a distribution over all possibilities.

$$\gamma_k^{(n)} = p(z^{(n)} = k | \mathbf{x}^{(n)}; \pi, \mu, \Sigma)$$

Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

1. E-step:
   - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
   - We cannot be sure, so it's a distribution over all possibilities.

$$\gamma_k^{(n)} = p(z^{(n)} = k|\mathbf{x}^{(n)}; \pi, \mu, \Sigma)$$

2. M-step:
   - Each Gaussian gets a certain amount of posterior probability for each datapoint.

Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

1. E-step:
   - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
   - We cannot be sure, so it's a distribution over all possibilities.

$$\gamma_k^{(n)} = p(z^{(n)} = k|\mathbf{x}^{(n)}; \pi, \mu, \Sigma)$$

2. M-step:
   - Each Gaussian gets a certain amount of posterior probability for each datapoint.
   - At the optimum we shall satisfy

$$\frac{\partial \ln p(\mathbf{X}|\pi, \mu, \Sigma)}{\partial \Theta} = 0$$

Expectation Maximization

- Elegant and powerful method for finding maximum likelihood solutions for models with latent variables

1. E-step:
   - In order to adjust the parameters, we must first solve the inference problem: Which Gaussian generated each datapoint?
   - We cannot be sure, so it's a distribution over all possibilities.

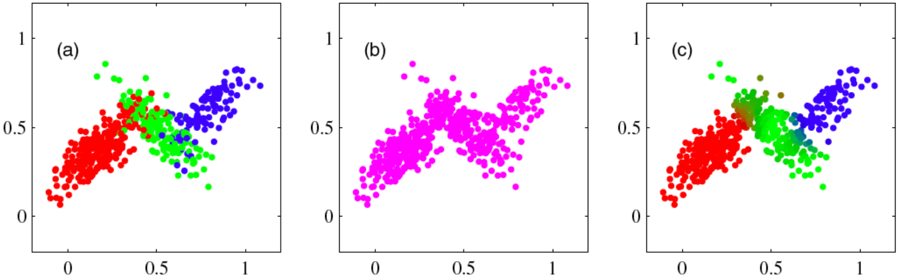
$$\gamma_k^{(n)} = p(z^{(n)} = k | \mathbf{x}^{(n)}; \pi, \mu, \Sigma)$$

2. M-step:
   - Each Gaussian gets a certain amount of posterior probability for each datapoint.
   - At the optimum we shall satisfy

$$\frac{\partial \ln p(\mathbf{X} | \pi, \mu, \Sigma)}{\partial \Theta} = 0$$

   - We can derive closed form updates for all parameters

Visualizing a Mixture of Gaussians



E-Step: Responsabilities

- Conditional probability (using Bayes rule) of $\mathbf{z}$ given $\mathbf{x}$

$$\gamma_k = p(z = k|\mathbf{x}) \quad =$$

E-Step: Responsabilities

- Conditional probability (using Bayes rule) of **z** given **x**

$$\gamma_k = p(z = k|\mathbf{x}) = \frac{p(z = k)p(\mathbf{x}|z = k)}{p(\mathbf{x})}$$

$$=$$

E-Step: Responsabilities

- Conditional probability (using Bayes rule) of **z** given **x**

$$\begin{aligned}
\gamma_k = p(z = k|\mathbf{x}) &= \frac{p(z = k)p(\mathbf{x}|z = k)}{p(\mathbf{x})} \\
&= \frac{p(z = k)p(\mathbf{x}|z = k)}{\sum_{j=1}^{K} p(z = j)p(\mathbf{x}|z = j)} \\
&=
\end{aligned}$$

E-Step: Responsabilities

- Conditional probability (using Bayes rule) of **z** given **x**

$$
\begin{aligned}
\gamma_k = p(z = k|\mathbf{x}) &= \frac{p(z = k)p(\mathbf{x}|z = k)}{p(\mathbf{x})} \\
&= \frac{p(z = k)p(\mathbf{x}|z = k)}{\sum_{j=1}^{K} p(z = j)p(\mathbf{x}|z = j)} \\
&= \frac{\pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j)}
\end{aligned}
$$

E-Step: Responsabilities

- Conditional probability (using Bayes rule) of $\mathbf{z}$ given $\mathbf{x}$

$$\begin{aligned}
\gamma_k = p(z=k|\mathbf{x}) &= \frac{p(z=k)p(\mathbf{x}|z=k)}{p(\mathbf{x})} \\
&= \frac{p(z=k)p(\mathbf{x}|z=k)}{\sum_{j=1}^{K} p(z=j)p(\mathbf{x}|z=j)} \\
&= \frac{\pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j)}
\end{aligned}$$

- γ_k can be viewed as the responsibility

M-Step: Estimate Parameters

- Log-likelihood:

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

M-Step: Estimate Parameters

- Log-likelihood:

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

- Set derivatives to 0:

$$\frac{\partial \ln p(\mathbf{X}|\pi, \mu, \Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j)} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

M-Step: Estimate Parameters

- Log-likelihood:

$$\ln p(\mathbf{X}|\pi,\mu,\Sigma) = \sum_{n=1}^{N} \ln\left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)\right)$$

- Set derivatives to 0:

$$\frac{\partial \ln p(\mathbf{X}|\pi,\mu,\Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j,\Sigma_j)} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

- We used:

$$\mathcal{N}(\mathbf{x}|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$$

and:

$$\frac{\partial(\mathbf{x}^T A\mathbf{x})}{\partial \mathbf{x}} = \mathbf{x}^T(A + A^T)$$

M-Step: Estimate Parameters

$$\frac{\partial \ln p(\mathbf{X}|\pi,\mu,\Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j,\Sigma_j)}}_{\gamma_k^{(n)}} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

M-Step: Estimate Parameters

$$\frac{\partial \ln p(\mathbf{X}|\pi,\mu,\Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j,\Sigma_j)}}_{\gamma_k^{(n)}} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

- This gives

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)}$$

with N_k the effective number of points in cluster k

$$N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

M-Step: Estimate Parameters

$$\frac{\partial \ln p(\mathbf{X}|\pi,\mu,\Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j,\Sigma_j)}}_{\gamma_k^{(n)}} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

- This gives

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)}$$

  with N_k the effective number of points in cluster k

$$N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- We just take the center-of gravity of the data that the Gaussian is responsible for

M-Step: Estimate Parameters

$$\frac{\partial \ln p(\mathbf{X}|\pi,\mu,\Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k,\Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j,\Sigma_j)}}_{\gamma_k^{(n)}} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

- This gives

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)}$$

with N_k the effective number of points in cluster k

$$N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- We just take the center-of gravity of the data that the Gaussian is responsible for
- Just like in K-means, except the data is weighted by the posterior probability of the Gaussian.

M-Step: Estimate Parameters

$$\frac{\partial \ln p(\mathbf{X}|\pi, \mu, \Sigma)}{\partial \mu_k} = 0 = \sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j)}}_{\gamma_k^{(n)}} \Sigma_k(\mathbf{x}^{(n)} - \mu_k)$$

- This gives

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)}$$

with N_k the effective number of points in cluster k

$$N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- We just take the center-of gravity of the data that the Gaussian is responsible for
- Just like in K-means, except the data is weighted by the posterior probability of the Gaussian.
- Guaranteed to lie in the convex hull of the data (Could be big initial jump)

M-Step (variance, mixing coefficients)

- We can get similarly expression for the variance

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T$$

M-Step (variance, mixing coefficients)

- We can get similarly expression for the variance

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T$$

- We can also minimize w.r.t the mixing coefficients

$$\pi_k = \frac{N_k}{N}, \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

M-Step (variance, mixing coefficients)

- We can get similarly expression for the variance

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T$$

- We can also minimize w.r.t the mixing coefficients

$$\pi_k = \frac{N_k}{N}, \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- The optimal mixing proportion to use (given these posterior probabilities) is just the fraction of the data that the Gaussian gets responsibility for.

M-Step (variance, mixing coefficients)

- We can get similarly expression for the variance

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T$$

- We can also minimize w.r.t the mixing coefficients

$$\pi_k = \frac{N_k}{N}, \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- The optimal mixing proportion to use (given these posterior probabilities) is just the fraction of the data that the Gaussian gets responsibility for.
- Note that this is not a closed form solution of the parameters, as they depend on the responsibilities $\gamma_k^{(n)}$, which are complex functions of the parameters

M-Step (variance, mixing coefficients)

- We can get similarly expression for the variance

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T$$

- We can also minimize w.r.t the mixing coefficients

$$\pi_k = \frac{N_k}{N}, \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)}$$

- The optimal mixing proportion to use (given these posterior probabilities) is just the fraction of the data that the Gaussian gets responsibility for.
- Note that this is not a closed form solution of the parameters, as they depend on the responsibilities $\gamma_k^{(n)}$, which are complex functions of the parameters
- But we have a simple iterative scheme to optimize

EM Algorithm for GMM

- Initialize the means μ_k, covariances Σ_k and mixing coefficients π_k
- Iterate until convergence:
  - E-step: Evaluate the responsibilities given current parameters

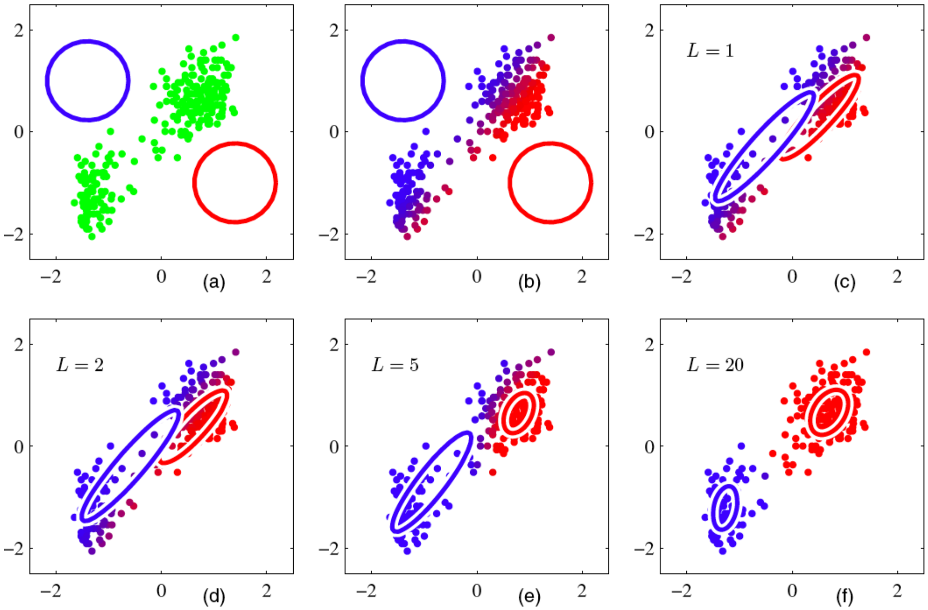
$$\gamma_k^{(n)} = p(z^{(n)}|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}^{(n)}|\mu_j, \Sigma_j)}$$

  - M-step: Re-estimate the parameters given current responsibilities

$$\begin{aligned} \mu_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)} \\ \Sigma_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T \\ \pi_k &= \frac{N_k}{N} \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)} \end{aligned}$$

  - Evaluate log likelihood and check for convergence

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$



2
0
−2
−2
0
2
(a)
(b)
(c)
(d)
(e)
(f)
$L = 1$
$L = 2$
$L = 5$
$L = 20$

Mixture of Gaussians vs. K-means

- EM for mixtures of Gaussians is just like a soft version of K-means, with fixed priors and covariance

Mixture of Gaussians vs. K-means

- EM for mixtures of Gaussians is just like a soft version of K-means, with fixed priors and covariance
- Instead of hard assignments in the E-step, we do soft assignments based on the softmax of the squared Mahalanobis distance from each point to each cluster.

Mixture of Gaussians vs. K-means

- EM for mixtures of Gaussians is just like a soft version of K-means, with fixed priors and covariance
- Instead of hard assignments in the E-step, we do soft assignments based on the softmax of the squared Mahalanobis distance from each point to each cluster.
- Each center moved by weighted means of the data, with weights given by soft assignments

Mixture of Gaussians vs. K-means

- EM for mixtures of Gaussians is just like a soft version of K-means, with fixed priors and covariance
- Instead of hard assignments in the E-step, we do soft assignments based on the softmax of the squared Mahalanobis distance from each point to each cluster.
- Each center moved by weighted means of the data, with weights given by soft assignments
- In K-means, weights are 0 or 1

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data
- If we knew z, then easy to maximize (replace sum over k with just the k where $z = k$)

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data
- If we knew z, then easy to maximize (replace sum over k with just the k where $z = k$)
- Unfortunately we are not given the complete data, but only the incomplete.

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data
- If we knew z, then easy to maximize (replace sum over k with just the k where $z = k$)
- Unfortunately we are not given the complete data, but only the incomplete.
- Our knowledge about the latent variables is $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data
- If we knew z, then easy to maximize (replace sum over k with just the k where $z = k$)
- Unfortunately we are not given the complete data, but only the incomplete.
- Our knowledge about the latent variables is $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$
- In the E-step we compute $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

An Alternative View of EM

- Hard to maximize (log-)likelihood of data directly
- General problem: sum inside the log

$$\ln p(\mathbf{x}|\Theta) = \ln \sum_{z} p(\mathbf{x}, \mathbf{z}|\Theta)$$

- Complete data $\{\mathbf{x}, \mathbf{z}\}$, and $\mathbf{x}$ is the incomplete data
- If we knew z, then easy to maximize (replace sum over k with just the k where $z = k$)
- Unfortunately we are not given the complete data, but only the incomplete.
- Our knowledge about the latent variables is $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$
- In the E-step we compute $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$
- In the M-step we maximize w.r.t Θ

$$Q(\Theta, \Theta^{old}) = \sum_{z} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta)$$

General EM Algorithm

1. Initialize Θ^{old}
2. E-step: Evaluate $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$
3. M-step:
$$\Theta^{new} = arg \max_{\Theta} Q(\Theta, \Theta^{old})$$
where
$$Q(\Theta, \Theta^{old}) = \sum_{z} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta)$$
4. Evaluate log likelihood and check for convergence (or the parameters). If not converged, $\Theta^{old} = \Theta$, Go to step 2

- Beyond this slide, read if you are interested in more details

How do we know that the updates improve things?

- Updating each Gaussian definitely improves the probability of generating the data if we generate it from the same Gaussians after the parameter updates.

How do we know that the updates improve things?

- Updating each Gaussian definitely improves the probability of generating the data if we generate it from the same Gaussians after the parameter updates.
  - But we know that the posterior will change after updating the parameters.

How do we know that the updates improve things?

- Updating each Gaussian definitely improves the probability of generating the data if we generate it from the same Gaussians after the parameter updates.
  - But we know that the posterior will change after updating the parameters.
- A good way to show that this is OK is to show that there is a single function that is improved by both the E-step and the M-step.

How do we know that the updates improve things?

- Updating each Gaussian definitely improves the probability of generating the data if we generate it from the same Gaussians after the parameter updates.
  - But we know that the posterior will change after updating the parameters.
- A good way to show that this is OK is to show that there is a single function that is improved by both the E-step and the M-step.
  - The function we need is called Free Energy.

Why EM converges

- Free energy F is a cost function that is reduced by both the E-step and the M-step.

$$F = \text{expected energy} - \text{entropy}$$

Why EM converges

- Free energy F is a cost function that is reduced by both the E-step and the M-step.

$$F = \text{expected energy} - \text{entropy}$$

- The expected energy term measures how difficult it is to generate each datapoint from the Gaussians it is assigned to. It would be happiest assigning each datapoint to the Gaussian that generates it most easily (as in K-means).

Why EM converges

- Free energy F is a cost function that is reduced by both the E-step and the M-step.

$$F = \text{expected energy} - \text{entropy}$$

- The expected energy term measures how difficult it is to generate each datapoint from the Gaussians it is assigned to. It would be happiest assigning each datapoint to the Gaussian that generates it most easily (as in K-means).
- The entropy term encourages "soft" assignments. It would be happiest spreading the assignment probabilities for each datapoint equally between all the Gaussians.

Free Energy

- Our goal is to maximize

$$p(\mathbf{X}|\Theta) = \sum_{\mathbf{z}} p(\mathbf{X}, \mathbf{z}|\Theta)$$

- Typically optimizing $p(\mathbf{X}|\Theta)$ is difficult, but $p(\mathbf{X}, \mathbf{Z}|\Theta)$ is easy
- Let $q(\mathbf{Z})$ be a distribution over the latent variables. For any distribution $q(\mathbf{Z})$ we have

$$\ln p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + KL(q||p(\mathbf{Z}|\mathbf{X}, \Theta))$$

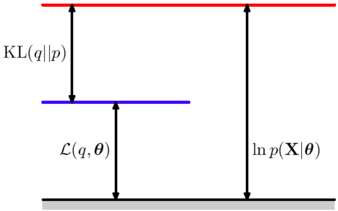
where

$$\begin{aligned} \mathcal{L}(q, \Theta) &= \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \\ KL(q||p) &= -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\} \end{aligned}$$

More on Free Energy

- Since the KL-divergence is always positive and have value 0 only if $q(Z) = p(\mathbf{Z}|\mathbf{X}, \Theta)$
- Thus $\mathcal{L}(q, \Theta)$ is a lower bound on the likelihood

$$\mathcal{L}(q, \Theta) \leq \ln p(\mathbf{X}|\Theta)$$



E-step and M-step

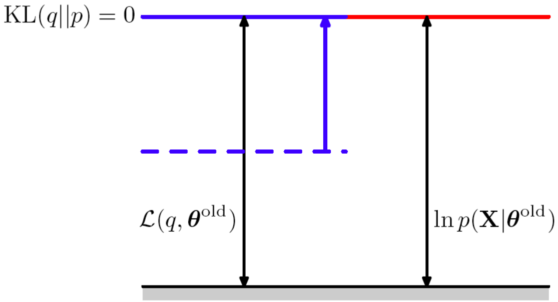
$$\ln p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + KL(q||p(\mathbf{Z}|\mathbf{X}, \Theta))$$

- In the E-step we maximize w.r.t $q(\mathbf{Z})$ the lower bound $\mathcal{L}(q, \Theta)$
- Since $\ln p(\mathbf{X}|\theta)$ does not depend on $q(\mathbf{Z})$, the maximum $\mathcal{L}$ is obtained when the KL is 0
- This is achieved when $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \Theta)$
- The lower bound $\mathcal{L}$ is then

$$\begin{aligned} \mathcal{L}(q, \Theta) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \\ &= Q(\Theta, \Theta^{old}) + \text{const} \end{aligned}$$

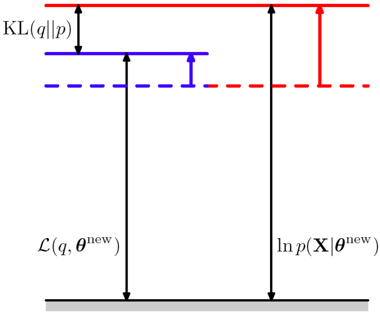
  with the content the entropy of the q distribution, which is independent of Θ
- In the M-step the quantity to be maximized is the expectation of the complete data log-likelihood
- Note that Θ is only inside the logarithm and optimizing the complete data likelihood is easier

Visualization of E-step



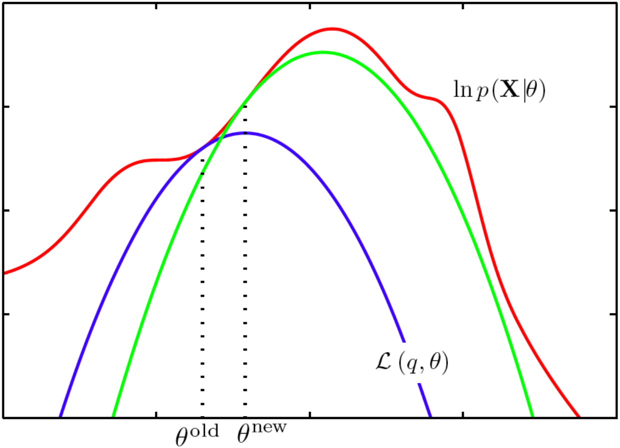
- The q distribution equal to the posterior distribution for the current parameter values Θ^{old}, causing the lower bound to move up to the same value as the log likelihood function, with the KL divergence vanishing.

Visualization of M-step



- The distribution $q(\mathbf{Z})$ is held fixed and the lower bound $\mathcal{L}(q, \Theta)$ is maximized with respect to the parameter vector Θ to give a revised value Θ^{new}. Because the KL divergence is nonnegative, this causes the log likelihood $\ln p(\mathbf{X}|\Theta)$ to increase by at least as much as the lower bound does.

Visualization of the EM Algorithm



- The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values. See the text for a full discussion.

Summary: EM is coordinate descent in Free Energy

$$
\begin{aligned}
\mathcal{L}(q, \Theta) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\Theta) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \ln p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \\
&= Q(\Theta, \Theta^{old}) + \text{const} \\
&= \text{expected energy} - \text{entropy}
\end{aligned}
$$

- The E-step minimizes F by finding the best distribution over hidden configurations for each data point.
- The M-step holds the distribution fixed and minimizes F by changing the parameters that determine the energy of a configuration.