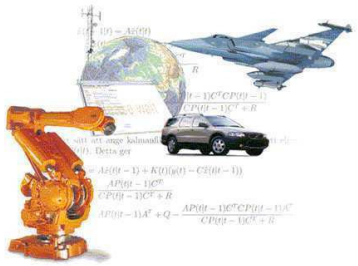
Machine Learning, Lecture 3
Expectation Maximization (EM) and Clustering



Thomas Schön

Division of Automatic Control
Linköping University
Linköping, Sweden.

Email: schon@isy.liu.se,
Phone: 013 - 281373,
Office: House B, Entrance 27.

Outline Lecture 3

1. Summary of Lecture 2
2. Expectation Maximization
   - General derivation
   - Example - identification of a linear state-space model
   - Example - identification of a Wiener system
3. Gaussian mixtures
   - Standard construction
   - Equivalent construction using latent variables
   - ML estimation using EM
4. Connections to the K-means algorithm for clustering

(Chapter 9)

Summary of Lecture 2 (I/IV)

Using the posterior mean $m_N = \beta S_N \Phi^T T$, the linear regression model is given by

$$y(x, m_N) = m_N^T \phi(x) = \beta \phi(x)^T S_N \Phi^T T = \sum_{n=1}^{N} \underbrace{\beta \phi(x)^T S_N \phi(x_n)}_{k(x,x_n)} t_n$$

Hence, the mean of the predictive distribution at a point x is a linear combination of the training target variables t_n, $y(x, m_N) = \sum_{n=1}^{N} k(x, x_n) t_n$, where $k(x, x') = \beta \phi(x)^T S_N \phi(x')$ is called the *equivalent kernel*.

An alternative approach to regression is to directly make use of a localized kernel, rather than starting from basis functions. This leads to the so called *kernel methods*.

Summary of Lecture 2 (II/IV)

Investigated one linear discriminant (a function that takes an input and assigns it to one of K classes) method in detail (least squares)

Modelled each class as $y_k(x) = w_k^T x + w_{k,0}$ and solved the LS problem, resulting in $\widehat{w} = (X^T X)^{-1} X^T T$

Showed how probabilistic generative models could be built for classification using the strategy,

1. Model $p(x \mid \mathcal{C}_k)$ (a.k.a. class-conditional density)
2. Model $p(\mathcal{C}_k)$
3. Use ML to find the parameters in $p(x \mid \mathcal{C}_k)$ and $p(\mathcal{C}_k)$.
4. Use Bayes' rule to find $p(\mathcal{C}_k \mid x)$

Summary of Lecture 2 (III/IV)

The "direct" method called logistic regression was introduced. Start by stating the model

$$p(\mathcal{C}_1 \mid \phi) = \sigma(w^T\phi) = \frac{1}{1+e^{-w^T\phi}},$$

which results in a log-likelihood function according to

$$L(w) = -\ln p(T \mid w) = -\sum_{n=1}^{N} \left(t_n \ln(y_n) + (1-t_n)\ln(1-y_n)\right),$$

where $y_n = p(\mathcal{C}_1 \mid \phi) = \sigma(w^T\phi)$. Note that this is a nonlinear, but concave function of w.

Hence, we can easily find the global minimum using Newton's method (resulting in an algorithm known as IRLS).

Summary of Lecture 2 (IV/IV)

The likelihood function for logistic regression is

$$p(T \mid w) = \prod_{n=1}^{N} \sigma(w^T\phi_n)^{t_n} \left(1-\sigma(w^T\phi_n)\right)^{1-t_n}$$

Hence, computing the posterior density $p(w \mid T) = \frac{p(T|w)p(w)}{p(T)}$ is intractable and we considered the Laplace approximation for solving this.

The Laplace approximation is a simple (local) approximation that is obtained by fitting a Gaussian centered around the (MAP) mode of the distribution.

(Very) Brief on Evaluating Classifiers

Let us define the following concepts,

- **True positives (tp):** Items correctly classified as belonging to the class.
- **True negatives (tn):** Items correctly classified as not belonging to the class.
- **False positives (fp):** Items incorrectly classified as belonging to the class. In other words, a false alarm.
- **False negatives (fn):** Items that are not classified in the correct class, even though they should have. In other words, a missed detection.

$$\text{Precision} = \frac{\text{tp}}{\text{tp}+\text{fp}} \qquad \text{Recall} = \frac{\text{tp}}{\text{tp}+\text{fn}}$$

Latent Variables - Example

A latent variable is a variable that is not directly observed. Other common names are hidden variables, unobserved variables or missing data.

An example of a latent variable is the state x_t in a state-space model.

Consider the following linear scalar state-space model

$$\begin{aligned} x_{t+1} &= \theta x_t + v_t, \\ y_t &= \frac{1}{2}x_t + e_t, \end{aligned} \qquad \begin{pmatrix} v_t \\ e_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}\right).$$

Expectation Maximization (EM) - Strategy and Idea 9(40)

The strategy underlying the EM algorithm is to separate the original ML problem into two linked problems, each of which is hopefully easier to solve than the original problem.

This separation is accomplished by exploiting the structure inherent in the probabilistic model.

The key idea is to consider the joint log-likelihood function of both the observed variables X and the latent variables Z,

$$L_\theta(Z, X) = \ln p_\theta(Z, X).$$

Expectation Maximization (EM) Algorithm 10(40)

Algorithm (*Expectation Maximization (EM)*)

1. *Set $i = 0$ and choose an initial guess θ_0.*
2. ***Expectation (E) step:*** *Compute*

$$\begin{aligned} \mathcal{Q}(\theta, \theta_i) &= \mathbf{E}_{\theta_i} \left\{ \ln p_\theta(Z, X) \mid X \right\} \\ &= \int \ln p_\theta(Z, X) p_{\theta_i}(Z \mid X) dZ. \end{aligned}$$

3. ***Maximization (M) step:*** *Compute*

$$\theta_{i+1} = \arg\max_\theta \mathcal{Q}(\theta, \theta_i).$$

4. *If not converged, update $i := i + 1$ and return to step 2.*

EM Example 1 - Linear System Identification 11(40)

Consider the following linear scalar state-space model

$$\begin{aligned} x_{t+1} &= \theta x_t + v_t, \\ y_t &= \frac{1}{2} x_t + e_t, \end{aligned} \qquad \begin{pmatrix} v_t \\ e_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix} \right).$$

The initial state is fully known, $x_1 = 0$. Finally, the true θ-parameter is given by $\theta^\star = 0.9$.

The identification problem is now to determine the parameter θ on the basis of the observations $Y = \{y_1, \ldots, y_N\}$, using the EM algorithm.

The latent variables Z are given by the states $Z = X \triangleq \{x_1, \ldots, x_{N+1}\}$.

Note the difference in notation compared to Bishop! The observations are denoted Y and the latent variables are given by X.

EM Example 1 - Linear System Identification 12(40)

The resulting $\mathcal{Q}$-function is

$$\begin{aligned} \mathcal{Q}(\theta, \theta_i) &\propto -\mathbf{E}_{\theta_i} \left\{ \sum_{t=1}^{N} x_t^2 \mid Y \right\} \theta^2 + 2\mathbf{E}_{\theta_i} \left\{ \sum_{t=1}^{N} x_t x_{t+1} \mid Y \right\} \theta \\ &= -\varphi\theta^2 + 2\psi\theta, \end{aligned}$$

where we have defined

$$\varphi \triangleq \sum_{t=1}^{N} \mathbf{E}_{\theta_i} \left\{ x_t^2 \mid Y \right\}, \qquad \psi \triangleq \sum_{t=1}^{N} \mathbf{E}_{\theta_i} \left\{ x_t x_{t+1} \mid Y \right\}.$$

There exists explicit expressions for these expected values.

The maximization (M) step:

$$\theta_{i+1} = \arg\max_{\theta} \mathcal{Q}(\theta, \theta_i).$$

Hence, the M step simply amounts to solving the following quadratic problem,

$$\theta_{i+1} = \arg\max_{\theta} -\varphi\theta^2 + 2\psi\theta,$$

which results in

$$\theta_{i+1} = \frac{\psi}{\varphi}.$$

Algorithm

1. *Set $i = 0$ and choose an initial guess θ_0.*
2. ***Expectation (E) step:*** *Compute*

$$\varphi = \sum_{t=1}^{N} \mathbf{E}_{\theta_i}\left\{x_t^2 \mid Y\right\}, \qquad \psi = \sum_{t=1}^{N} \mathbf{E}_{\theta_i}\left\{x_t x_{t+1} \mid Y\right\}.$$

3. ***Maximization (M) step:*** *Find the next iterate according to*

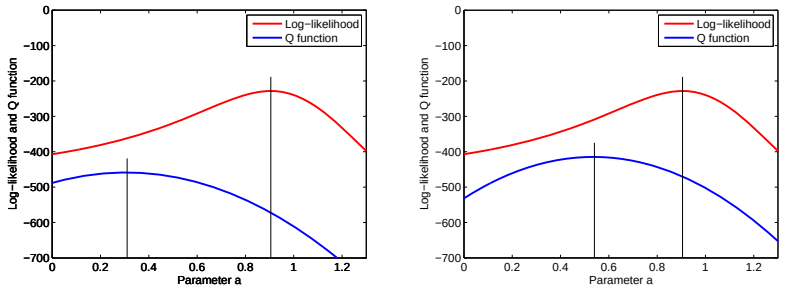
$$\theta_{i+1} = \frac{\psi}{\varphi}.$$

4. *If $|L_{\theta_i}(Y) - L_{\theta_{i-1}}(Y)| \geq 10^{-6}$, update $i := i + 1$ and return to step 2, otherwise terminate.*

- Different number of samples N used.
- Monte Carlo studies, each using 1000 realisations of data.
- Initial guess $\theta_0 = 0.1$.

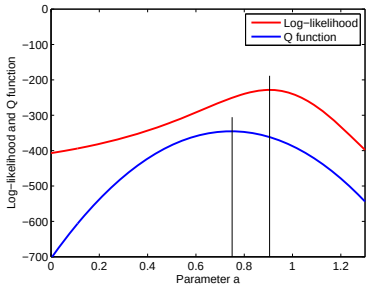
N	100	200	500	1000	2000	5000	10000
$\widehat{\theta}$	0.8716	0.8852	0.8952	0.8978	0.8988	0.8996	0.8998

No surprise, since ML is asymptotically efficient.

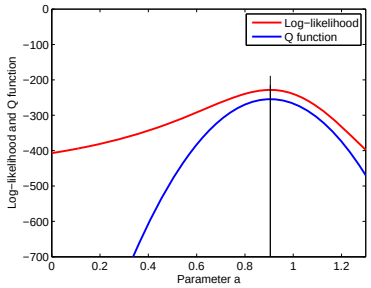


(a) Iteration 1 (b) Iteration 2

EM Example 1 - Linear System Identification



(c) Iteration 3



(d) Iteration 11

All details (including MATLAB code) are provided in

Thomas B. Schön, *An Explanation of the Expectation Maximization Algorithm*. Division of Automatic Control, Linköping University, Sweden, Technical Report nr: LiTH-ISY-R-2915, August 2009.

`http://www.rt.isy.liu.se/~schon/Publications/SchonEM2009.pdf`

Nonlinear System Identification Using EM (I/VI)

Now, consider the problem of estimating the parameters θ in

$$\begin{aligned} x_{t+1} &= f_t(x_t, \theta) + v_t(\theta), \\ y_t &= h_t(x_t, \theta) + e_t(\theta). \end{aligned}$$

Commonly also written as

$$\begin{aligned} x_{t+1} &\sim p_\theta(x_{t+1} \mid x_t), \\ y_t &\sim p_\theta(y_t \mid x_t). \end{aligned}$$

According to the above, the first step is to compute the $\mathcal{Q}$-function

$$\mathcal{Q}(\theta, \hat{\theta}_k) = \mathbf{E}_{\theta_k} \left\{ \log p_\theta(Z, Y) \mid Y \right\}$$

Nonlinear System Identification Using EM (II/VI)

Applying $\mathbf{E}_{\theta_k}\{\cdot \mid Y\}$ to

$$\begin{aligned} \log p_\theta(X, Y) &= \log p_\theta(Y \mid X) + \log p_\theta(X) \\ &= \log p_\theta(x_1) + \sum_{t=1}^{N-1} \log p_\theta(x_{t+1} \mid x_t) + \sum_{t=1}^{N} \log p_\theta(y_t \mid x_t) \end{aligned}$$

results in $\mathcal{Q}(\theta, \theta_k) = I_1 + I_2 + I_3$, where

$$\begin{aligned} I_1 &= \int \log p_\theta(x_1) p_{\theta_k}(x_1 \mid Y) dx_1, \\ I_2 &= \sum_{t=1}^{N-1} \int\int \log p_\theta(x_{t+1} \mid x_t) p_{\theta_k}(x_{t+1}, x_t \mid Y) dx_t dx_{t+1}, \\ I_3 &= \sum_{t=1}^{N} \int \log p_\theta(y_t \mid x_t) p_{\theta_k}(x_t \mid Y) dx_t. \end{aligned}$$

Nonlinear System Identification Using EM (III/VI

This leads us to a nonlinear state smoothing problem, which we can solve using particle smoothers.
The current particle smoothers provides us with the following approximation of the joint smoothing density

$$p(X \mid Y) \approx \frac{1}{M} \sum_{t=1}^{M} \delta\left(X - X^i\right),$$

which allows for the following approximations of the marginal smoothing densities that we need,

$$p_{\theta_k}(x_t \mid Y) \approx \hat{p}_{\theta_k}(x_t \mid Y) = \frac{1}{M} \sum_{i=1}^{M} \delta(x_t - x_t^i),$$

$$p_{\theta_k}(x_{t:t+1} \mid Y) \approx \hat{p}_{\theta_k}(x_{t:t+1} \mid Y) = \frac{1}{M} \sum_{i=1}^{M} \delta(x_{t:t+1} - x_{t:t+1}^i).$$

Nonlinear System Identification Using EM (IV/VI) 21(40)

Inserting the above approximations into the integrals straightforwardly yields the approximation we are looking for,

$$\hat{I}_1 = \int \log p_\theta(x_1) \sum_{i=1}^{M} \frac{1}{M} \delta(x_1 - x_1^i) dx_1 = \frac{1}{M} \sum_{i=1}^{M} \log p_\theta(x_1^i),$$

$$\hat{I}_2 = \sum_{t=1}^{N-1} \int \int \log p_\theta(x_{t+1} \mid x_t) \sum_{i=1}^{M} \frac{1}{M} \delta\left(x_{t:t+1} - x_{t:t+1}^i\right) dx_{t:t+1}$$

$$= \frac{1}{M} \sum_{t=1}^{N-1} \sum_{i=1}^{M} \log p_\theta(x_{t+1}^i \mid x_t^i),$$

$$\hat{I}_3 = \sum_{t=1}^{N} \int \log p_\theta(y_t \mid x_t) \sum_{i=1}^{M} \frac{1}{M} \delta(x_t - x_t^i) dx_t = \frac{1}{M} \sum_{t=1}^{N} \sum_{t=1}^{M} \log p_\theta(y_t \mid x_t^i)$$

Nonlinear System Identification Using EM (V/VI) 22(40)

It is straightforward to make use of the approximation of the Q-functions just derived in order to compute gradients of the Q-function,

$$\frac{\partial}{\partial \theta} \widehat{\mathcal{Q}}(\theta, \theta_k) = \frac{\partial \hat{I}_1}{\partial \theta} + \frac{\partial \hat{I}_2}{\partial \theta} + \frac{\partial \hat{I}_3}{\partial \theta}$$

For example (the other two terms analogously),

$$\hat{I}_3 = \frac{1}{M} \sum_{t=1}^{N} \sum_{t=1}^{M} \log p_\theta(y_t \mid x_t^i),$$

$$\frac{\partial \hat{I}_3}{\partial \theta} = \frac{1}{M} \sum_{t=1}^{N} \sum_{t=1}^{M} \frac{\partial \log p_\theta(y_t \mid x_t^i)}{\partial \theta}$$

With these gradients in place there are many algorithms that can be used in order to solve the maximization problem, we employ BFGS.

Nonlinear System Identification Using EM (VI/VI) 23(40)

Algorithm (*Nonlinear System Identification Using EM*)

1. *Set $i = 0$ and choose an initial guess θ_0.*
2. ***Expectation (E) step:***
   1. *Run the particle filter and the particle smoother*
   2. *Calculate an approximation of the $\mathcal{Q}$-function $\widehat{\mathcal{Q}}_M(\theta, \hat{\theta}_i) = \hat{I}_1 + \hat{I}_2 + \hat{I}_3$.*
3. ***Maximization (M) step:*** *Compute*

$$\theta_{i+1} = \arg\max_\theta \widehat{\mathcal{Q}}_M(\theta, \theta_i).$$

4. *If not converged, update $i := i + 1$ and return to step 2.*

Thomas B. Schön, Adrian Wills and Brett Ninness. System Identification of Nonlinear State-Space Models. *Automatica*, 47(1):39-49, January 2011.

EM Example 2 - Blind Wiener Identification (I/IV) 24(40)

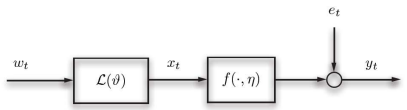


Figure: The blind (only the outputs y_t are available) Wiener (linear dynamic system followed by a static nonlinearity) problem.

$$\xi_{t+1} = A\xi_t + w_t, \qquad w_t \sim \mathcal{N}(0, Q),$$
$$x_t = C\xi_t,$$
$$y_t = f(x_t, \eta) + e_t, \qquad e_t \sim \mathcal{N}(0, R).$$

$$\theta = [\vartheta^T, \eta^T, \text{vec}\{R\}^T],$$
$$\vartheta = [\text{vec}\{A\}^T, \text{vec}\{C\}^T, \text{vec}\{Q\}]^T.$$

Most existing work deals with special cases, with assumptions like;

1. Invertible nonlinearities,
2. no measurement noise,
3. no process noise.

We are not forced to do any of these assumptions.

EM Example 2 - Blind Wiener Identification (II/IV) 25(40)

Recall that ($Z = \Xi$) $\mathcal{Q}(\theta, \theta_i) \triangleq \int \ln p_\theta(\Xi, Y) p_{\theta_i}(\Xi \mid Y) d\Xi$.
According to the model we have

$$\begin{aligned}\ln p_\theta(\Xi, Y) &= \ln p_\theta(Y \mid \Xi) + \ln p_\theta(\Xi) \\ &= \sum_{t=1}^{N} \ln p_\theta(y_t \mid \xi_t) + \sum_{t=1}^{N-1} \ln p_\theta(\xi_{t+1} \mid \xi_t),\end{aligned}$$

resulting in $\mathcal{Q}(\theta, \theta_i) = I_1 + I_2$,

$$I_1 = \sum_{t=1}^{N} \int \int \ln p_\theta(\xi_{t+1} \mid \xi_t) p_{\theta_i}(\xi_{t+1}, \xi_t \mid Y) d\xi_t d\xi_{t+1},$$

$$I_2 = \sum_{t=1}^{N} \ln p_\theta(y_t \mid \xi_t) p_{\theta_i}(\xi_t \mid Y) d\xi_t,$$

where $p_{\theta_i}(\xi_{t+1}, \xi_t \mid Y)$ and $p_{\theta_i}(\xi_t \mid Y)$ are provided by the particle smoother.

EM Example 2 - Blind Wiener Identification (III/IV) 26(40)

Wiener system:

$$H(q) = \frac{q^{-1} + 0.1q^{-2} - 0.49q^{-3} + 0.01q^{-4}}{1 + 0.3676q^{-1} + 0.88746q^{-2} + 0.52406q^{-3} + 0.55497q^{-4}},$$

$$f(x) = \begin{cases} 0.3 & : x > 0.3, \\ x & : -0.2 \leq x \leq 0.3, \\ -0.2 & : x < -0.2. \end{cases}$$

$N = 1000$ samples used, $w_t \sim \mathcal{N}(0, 0.1), e_t \sim \mathcal{N}(0, 0.001)$.

$M = 200$ particles are used and 100 Monte Carlo runs.

Initialize the linear system using a subspace method.

EM Example 2 - Blind Wiener Identification (IV/IV) 27(40)

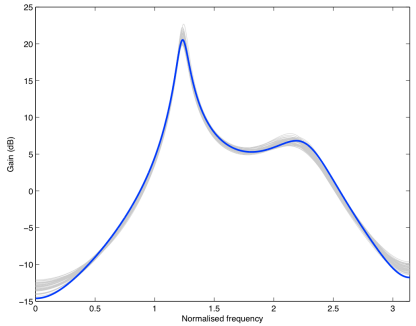


Figure: Bode plot of estimated (grey) and true (blue) systems.

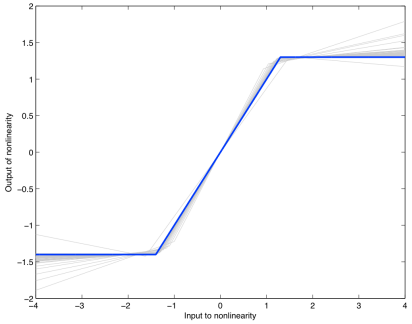


Figure: Estimated (grey) and true (blue) memoryless nonlinearity (saturation).

EM Example 2 - Blind Wiener Identification 28(40)

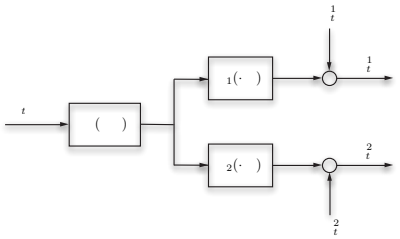


Figure: Block diagram of blind Wiener model with two outputs.

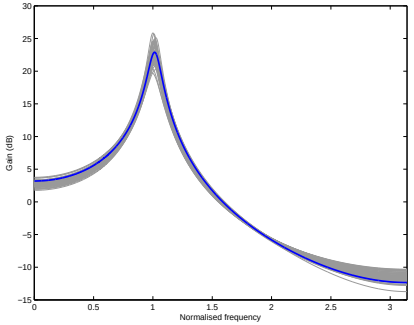


Figure: Bode plot of estimated (grey) and true (blue) systems.

EM Example 2 - Blind Wiener Identification

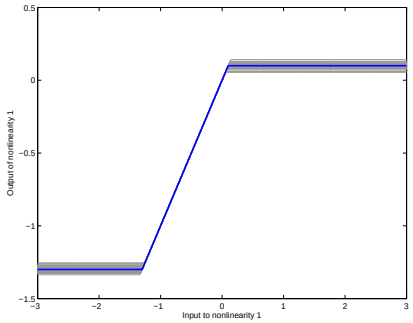


Figure: Estimated (grey) and true (blue) nonlinearity (saturation).

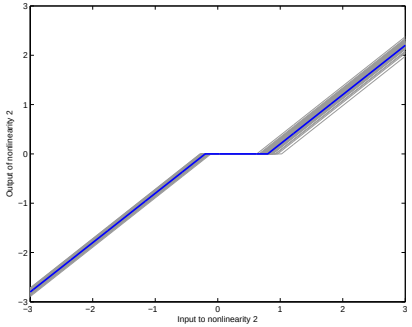


Figure: Estimated (grey) and true (blue) nonlinearity (dead band).

Adrian Wills, Thomas B. Schön, Lennart Ljung and Brett Ninness. Blind Identification of Wiener Models. *Proceedings of the 18th World Congress of the International Federation of Automatic Control (IFAC)*, Milan, Italy, September 2011. (accepted for publication)

Gaussian Mixture - Standard Construction

A linear superposition of Gaussians

$$p(x) = \sum_{k=1}^{K} \underbrace{\pi_k}_{p(k)} \underbrace{\mathcal{N}\left(x \mid \mu_k, \Sigma_k\right)}_{p(x|k)}$$

is called a Gaussian mixture. The mixture coefficients π_k satisfies

$$\sum_{k=1}^{K} \pi_k = 1, \qquad 0 \leq \pi_k \leq 1.$$

Interpretation: The density $p(x \mid k) = \mathcal{N}(x \mid \mu_k, \Sigma_k)$ is the probability of x, given that component k was chosen. The probability of choosing component k is given by the prior probability $p(k)$.

Gaussian Mixture - Example

Consider the following Gaussian mixture

$$p(x) = \underbrace{0.3}_{\pi_1} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 4 \\ 4.5 \end{pmatrix}}_{\mu_1}, \underbrace{\begin{pmatrix} 1.2 & 0.6 \\ 0.6 & 0.5 \end{pmatrix}}_{\Sigma_1}\Big) + \underbrace{0.5}_{\pi_2} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 8 \\ 1 \end{pmatrix}}_{\mu_2}, \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\Sigma_2}\Big) + \underbrace{0.2}_{\pi_3} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 9 \\ 8 \end{pmatrix}}_{\mu_3}, \underbrace{\begin{pmatrix} 0.6 & 0.5 \\ 0.5 & 1.5 \end{pmatrix}}_{\Sigma_3}\Big)$$

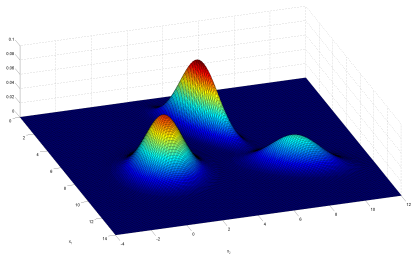


Figure: Probability density function.

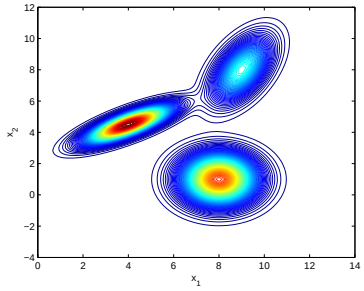


Figure: Contour plot.

Gaussian Mixture - Problem with Standard Construction

Given N independent observations $\{x_n\}_{n=1}^{N}$, the log-likelihood function if given by

$$\ln p(X; \pi_{1:K}, \mu_{1:K}, \Sigma_{1:K}) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(x \mid \mu_k, \Sigma_k\right) \right)$$

There is no closed form solution available (due to the sum inside the logarithm).

We will now see that this problem can be separated into two simple problems using the EM algorithm.

First we introduce an equivalent construction of the Gaussian mixture by introducing a latent variable.

EM for Gaussian Mixtures - Intuitive Preview

Based on

$$p(z_n) = \prod_{k=1}^{K} \pi_k^{z_{nk}} \qquad \text{and} \qquad p(x_n \mid z_n) = \prod_{k=1}^{K} \mathcal{N}(x_n \mid \mu_k, \Sigma_k)^{z_{nk}}$$

we have (for independent observations $\{x_n\}_{n=1}^N$)

$$p(X, Z) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(x_n \mid \mu_k, \Sigma_k)^{z_{nk}},$$

resulting in the following log-likelihood

$$\ln p(X, Z) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \left(\ln \pi_k + \ln \mathcal{N}(x \mid \mu_k, \Sigma_k) \right) \qquad (1)$$

Let us now use wishful thinking and assume that Z is known. Then, maximization of (1) is straightforward.

EM for Gaussian Mixtures - Explicit Algorithm

Algorithm (*EM for Gaussian Mixtures*)

1. *Initialize* $\mu_k^1, \Sigma_k^1, \pi_k^1$ *and set* $i = 1$.
2. ***Expectation (E) step:*** *Compute*

$$\gamma(z_{nk}) = \frac{\pi_k^i \mathcal{N}(x_n \mid \mu_k^i, \Sigma_k^i)}{\sum_{j=1}^{K} \pi_j^i \mathcal{N}(x_n \mid \mu_j^i, \Sigma_j^i)}, \quad n = 1, \ldots, N, k = 1, \ldots, K.$$

3. ***Maximization (M) step:*** *Compute*

$$\mu_k^{i+1} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n, \quad \pi_k^{i+1} = \frac{N_k}{N}, \quad N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

$$\Sigma_k^{i+1} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{i+1})(x_n - \mu_k^{i+1})^T$$

4. *If not converged, update* $i := i + 1$ *and return to step* 2.

Example - EM for Gaussian Mixtures (I/III)

Consider the same Gaussian mixture as before,

$$p(x) = \underbrace{0.3}_{\pi_1} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 4 \\ 4.5 \end{pmatrix}}_{\mu_1}, \underbrace{\begin{pmatrix} 1.2 & 0.6 \\ 0.6 & 0.5 \end{pmatrix}}_{\Sigma_1}\Big) + \underbrace{0.5}_{\pi_2} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 8 \\ 1 \end{pmatrix}}_{\mu_2}, \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\Sigma_2}\Big) + \underbrace{0.2}_{\pi_3} \mathcal{N}\Big(x \mid \underbrace{\begin{pmatrix} 9 \\ 8 \end{pmatrix}}_{\mu_3}, \underbrace{\begin{pmatrix} 0.6 & 0.5 \\ 0.5 & 1.5 \end{pmatrix}}_{\Sigma_3}\Big)$$

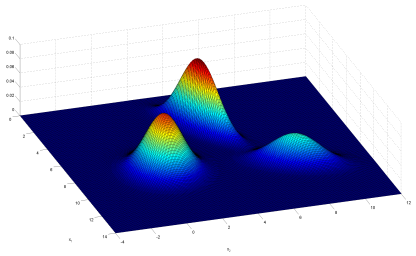


Figure: Probability density function.

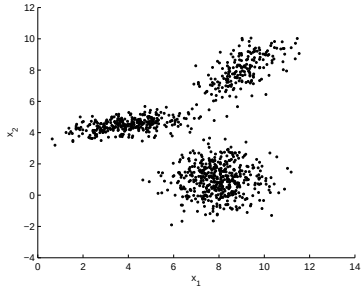


Figure: $N = 1000$ samples from the Gaussian mixture $p(x)$.

Example - EM for Gaussian Mixtures (II/III)

- Apply the EM algorithm to estimate a Gaussian mixture with $K = 3$ Gaussians, i.e. use the 1000 samples to compute estimates of $\pi_1, \pi_2, \pi_3, \mu_1, \mu_2, \mu_3, \Sigma_1, \Sigma_2, \Sigma_3$.
- 200 iterations.

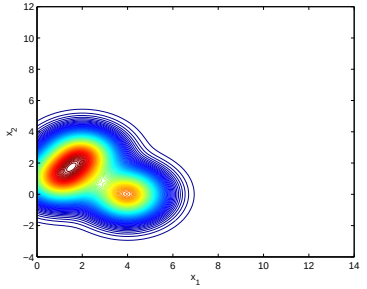


Figure: Initial guess.

Example - EM for Gaussian Mixtures (III/III)

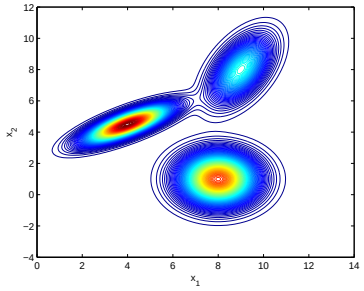


Figure: True PDF.

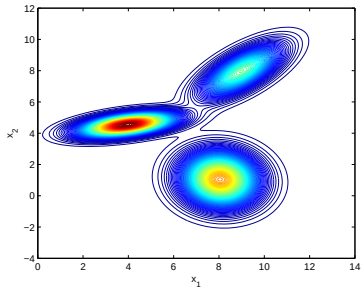


Figure: Estimate after 200 iterations of the EM algorithm.

The K-Means Algorithm (I/II)

Algorithm (K-means algorithm, a.k.a. Lloyd's algorithm)

1. *Initialize* μ_k^1 *and set* $i = 1$.
2. *Minimize* J *w.r.t.* r_{nk} *keeping* $\mu_k = \mu_k^i$ *fixed.*

$$r_{nk}^{i+1} = \begin{cases} 1 & \text{if } k = \arg\min_j \|x_n - \mu_j^i\|^2 \\ 0 & \text{otherwise} \end{cases}$$

3. *Minimize* J *w.r.t.* μ_k *keeping* $r_{nk} = r_{nk}^{i+1}$ *fixed.*

$$\mu_k^{i+1} = \frac{\sum_{n=1}^N r_{nk}^{i+1} x_n}{\sum_{n=1}^N r_{nk}^{i+1}}.$$

4. *If not converged, update* $i := i + 1$ *and return to step* 2.

The K-Means Algorithm (II/II)

The name K-means stems from the fact that in step 3 of the algorithm, u_k is give by the mean of all the data points assigned to cluster k.

Note the similarities between the K-means algorithm and the EM algorithm for Gaussian mixtures!

K-means is deterministic with "hard" assignment of data points to clusters (no uncertainty), whereas EM is a probabilistic method that provides a "soft" assignment.

If the Gaussian mixtures are modeled using covariance matrices

$$\Sigma_k = \epsilon I, \quad k = 1, \ldots, K,$$

it is straightforward to show that the EM algorithm for a mixture of K Gaussian's is equivalent to the K-means algorithm, when $\epsilon \to \infty$.

A Few Concepts to Summarize Lecture 6

Latent variable: A variable that is not directly observed. Sometimes also referred to as hidden variable or missing data.

Expectation Maximization (EM): The EM algorithm computes maximum likelihood estimates of unknown parameters in probabilistic models involving latent variables.

Jensen's inequality: States that if f is a convex function, then $\mathbf{E}(f(x)) \geq f(\mathbf{E}(x))$.

Clustering: Unsupervised learning, where a set of observations is divided into clusters. The observations belonging to a certain cluster are similar in some sense.

K-means algorithm (a.k.a. Lloyd's algorithm): A clustering algorithm that assigns N observations into K clusters, such that each observation belongs to the cluster with nearest (in the Euclidean sense) mean.