

# Gaussian Mixture Models

Peter Bell

Automatic Speech Recognition— ASR Lecture 6  
28 January 2021

## GMMs

- Univariate and multivariate Gaussians
- Gaussian mixture models
- GMM estimation with the EM algorithm
- Using GMMs with HMMs

Consider a real valued random variable  $X$

- Cumulative distribution function (cdf)  $F(x)$  for  $X$ :

$$F(x) = P(X \leq x)$$

- To obtain the probability of falling in an interval we can do the following:

$$\begin{aligned} P(a < X \leq b) &= P(X \leq b) - P(X \leq a) \\ &= F(b) - F(a) \end{aligned}$$

- The rate of change of the cdf gives us the *probability density function* (pdf),  $p(x)$ :

$$p(x) = \frac{d}{dx}F(x) = F'(x)$$

$$F(x) = \int_{-\infty}^x p(x)dx$$

- $p(x)$  is **not** the probability that  $X$  has value  $x$ . But the pdf is proportional to the probability that  $X$  lies in a small interval centred on  $x$ .
- Notation:  $p$  for pdf,  $P$  for probability

# The Gaussian distribution (univariate)

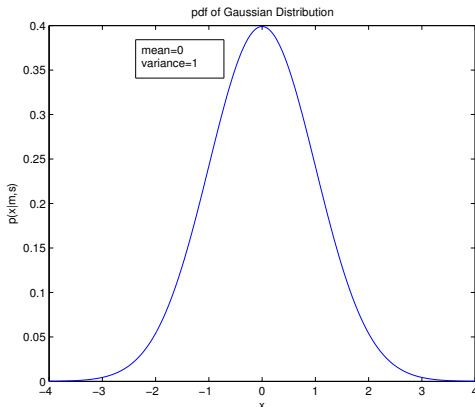
- The **Gaussian** (or **Normal**) distribution is the most common (and easily analysed) continuous distribution
- It is also a reasonable model in many situations (the famous “bell curve”)
- If a (scalar) variable has a Gaussian distribution, then it has a probability density function with this form:

$$p(x | \mu, \sigma^2) = \mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x - \mu)^2}{2\sigma^2}\right)$$

- The Gaussian is described by two parameters:
  - the mean  $\mu$  (location)
  - the variance  $\sigma^2$  (dispersion)

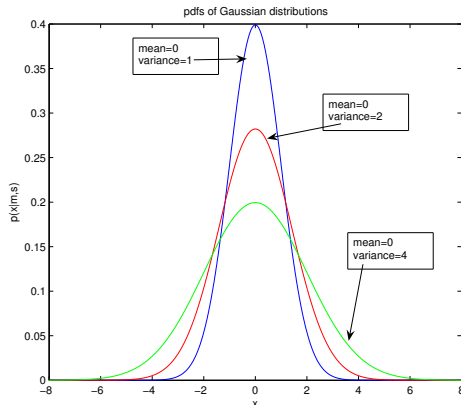
# Plot of Gaussian distribution

- Gaussians have the same shape, with the location controlled by the mean, and the spread controlled by the variance
- One-dimensional Gaussian with zero mean and unit variance ( $\mu = 0, \sigma^2 = 1$ ):



# Properties of the Gaussian distribution

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(x - \mu)^2}{2\sigma^2}\right)$$



# Parameter estimation

- Estimate mean and variance parameters of a Gaussian from data  $x_1, x_2, \dots, x_N$
- Use the following as the estimates:

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n \quad (\text{mean})$$

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})^2 \quad (\text{variance})$$



## Example: ML estimation of the mean

Consider the log likelihood of a set of  $N$  training data points  $\{x_1, \dots, x_N\}$  being generated by a Gaussian with mean  $\mu$  and variance  $\sigma^2$ :

$$\begin{aligned} L = \ln p(\{x_1, \dots, x_N\} | \mu, \sigma^2) &= -\frac{1}{2} \sum_{n=1}^N \left( \frac{(x_n - \mu)^2}{\sigma^2} - \ln \sigma^2 - \ln(2\pi) \right) \\ &= -\frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2 - \frac{N}{2} \ln \sigma^2 - \frac{N}{2} \ln(2\pi) \end{aligned}$$

By maximising the the log likelihood function with respect to  $\mu$  we can show that the maximum likelihood estimate for the mean is indeed the sample mean:

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n.$$

# The multivariate Gaussian distribution

- The  $D$ -dimensional vector  $\mathbf{x} = (x_1, \dots, x_D)^T$  follows a multivariate Gaussian (or normal) distribution if it has a probability density function of the following form:

$$p(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

The pdf is parameterised by the mean vector  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_D)^T$  and the covariance matrix  $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \dots & \sigma_{1D} \\ \vdots & \ddots & \vdots \\ \sigma_{D1} & \dots & \sigma_{DD} \end{pmatrix}$ .

# The multivariate Gaussian distribution

- The  $D$ -dimensional vector  $\mathbf{x} = (x_1, \dots, x_D)^T$  follows a multivariate Gaussian (or normal) distribution if it has a probability density function of the following form:

$$p(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

The pdf is parameterised by the mean vector  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_D)^T$  and the covariance matrix  $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \dots & \sigma_{1D} \\ \vdots & \ddots & \vdots \\ \sigma_{D1} & \dots & \sigma_{DD} \end{pmatrix}$ .

- The 1-dimensional Gaussian is a special case of this pdf

# The multivariate Gaussian distribution

- The  $D$ -dimensional vector  $\mathbf{x} = (x_1, \dots, x_D)^T$  follows a multivariate Gaussian (or normal) distribution if it has a probability density function of the following form:

$$p(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

The pdf is parameterised by the mean vector  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_D)^T$  and the covariance matrix  $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \dots & \sigma_{1D} \\ \vdots & \ddots & \vdots \\ \sigma_{D1} & \dots & \sigma_{DD} \end{pmatrix}$ .

- The 1-dimensional Gaussian is a special case of this pdf
- The argument to the exponential  $0.5(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})$  is referred to as a *quadratic form*.

# Covariance matrix

- The mean vector  $\mu$  is the expectation of  $x$ :

$$\mu = E[x]$$

- The covariance matrix  $\Sigma$  is the expectation of the deviation of  $x$  from the mean:

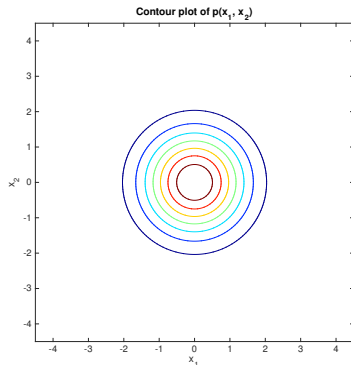
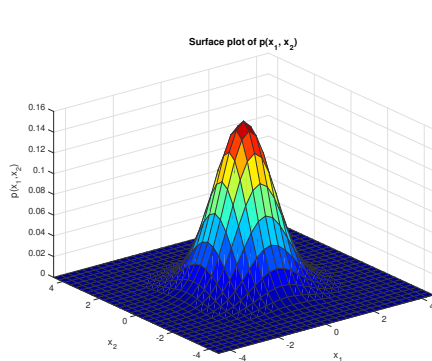
$$\Sigma = E[(x - \mu)(x - \mu)^T]$$

- $\Sigma$  is a  $D \times D$  symmetric matrix:

$$\sigma_{ij} = E[(x_i - \mu_i)(x_j - \mu_j)] = E[(x_j - \mu_j)(x_i - \mu_i)] = \sigma_{ji}$$

- The sign of the covariance helps to determine the relationship between two components:
  - If  $x_j$  is large when  $x_i$  is large, then  $(x_i - \mu_i)(x_j - \mu_j)$  will tend to be positive;
  - If  $x_j$  is small when  $x_i$  is large, then  $(x_i - \mu_i)(x_j - \mu_j)$  will tend to be negative.

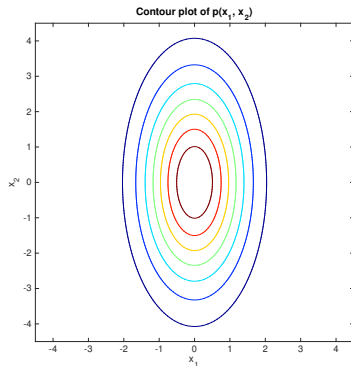
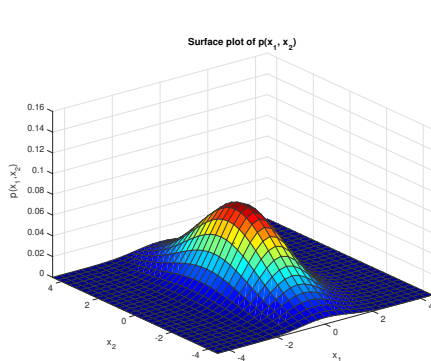
# Spherical Gaussian



$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \rho_{12} = 0$$

NB: Correlation coefficient  $\rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \quad (-1 \leq \rho_{ij} \leq 1)$

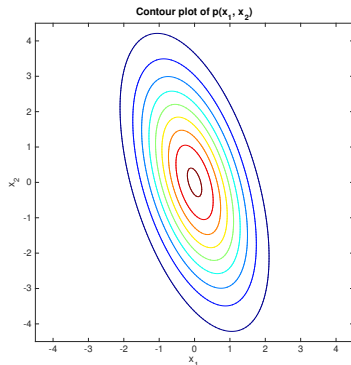
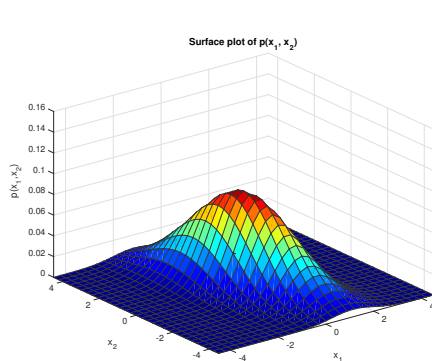
# Diagonal Covariance Gaussian



$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} \quad \rho_{12} = 0$$

NB: Correlation coefficient  $\rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \quad (-1 \leq \rho_{ij} \leq 1)$

# Full covariance Gaussian



$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \Sigma = \begin{pmatrix} 1 & -1 \\ -1 & 4 \end{pmatrix} \quad \rho_{12} = -0.5$$

NB: Correlation coefficient  $\rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \quad (-1 \leq \rho_{ij} \leq 1)$



# Parameter estimation of a multivariate Gaussian distribution

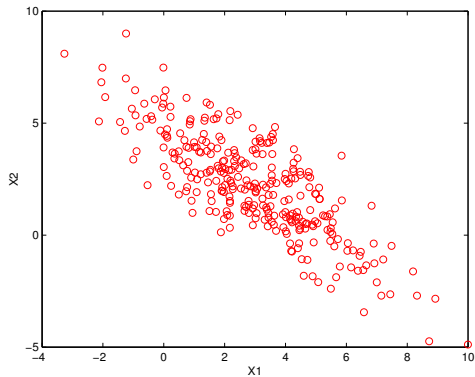
- It is possible to show that the mean vector  $\hat{\mu}$  and covariance matrix  $\hat{\Sigma}$  that maximise the likelihood of the training data are given by:

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

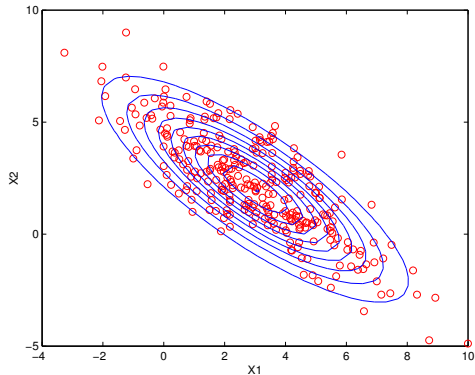
where  $\mathbf{x}_n = (x_{n,1}, \dots, x_{n,D})^T$ .

NB:  $T$  denotes either the number of samples or vector transpose depending on context.

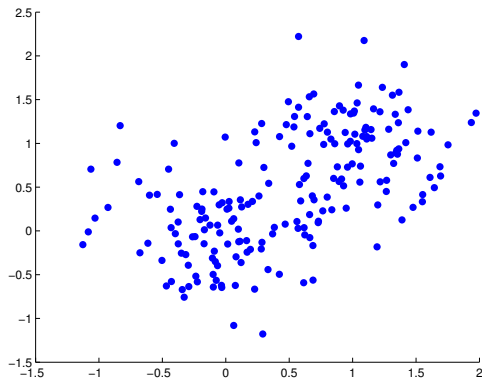
# Example data



# Maximum likelihood fit to a Gaussian

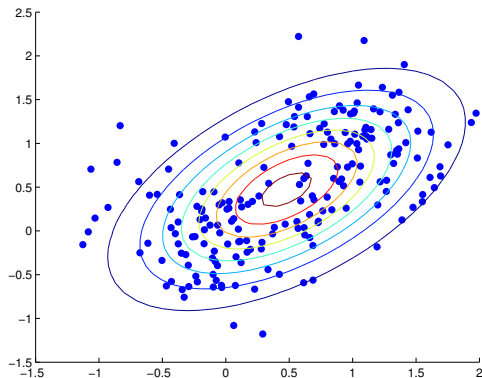


# Data in clusters (example 1)



$$\mu_1 = (0, 0)^T \quad \mu_2 = (1, 1)^T \quad \Sigma_1 = \Sigma_2 = 0.2I$$

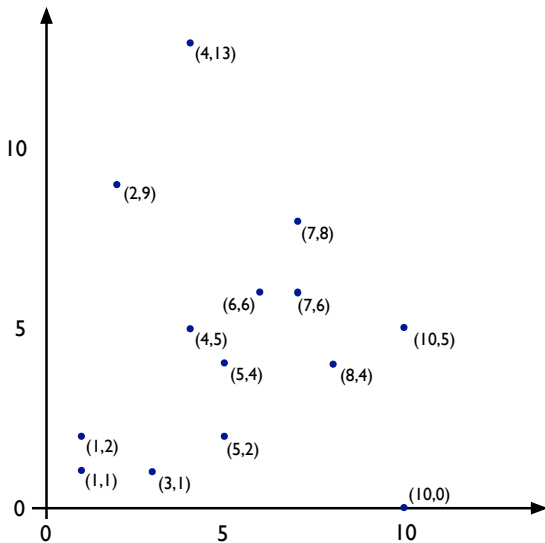
## Example 1 fit by a Gaussian



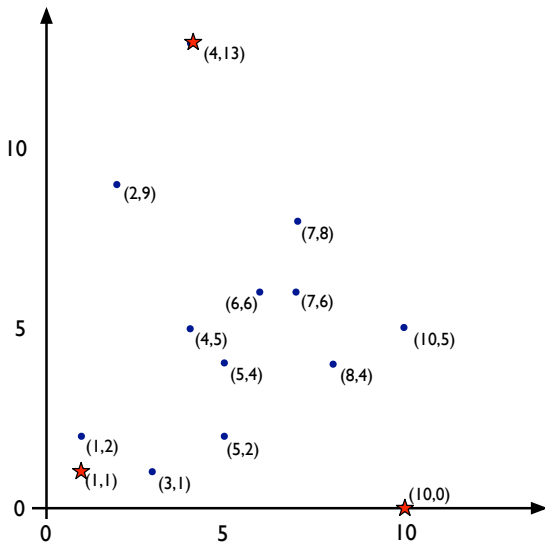
$$\mu_1 = (0, 0)^T \quad \mu_2 = (1, 1)^T \quad \Sigma_1 = \Sigma_2 = 0.2I$$

- k-means is an automatic procedure for clustering unlabelled data
- Requires a prespecified number of clusters
- Clustering algorithm chooses a set of clusters with the minimum within-cluster variance
- Guaranteed to converge (eventually)
- Clustering solution is dependent on the initialisation

## k-means example: data set

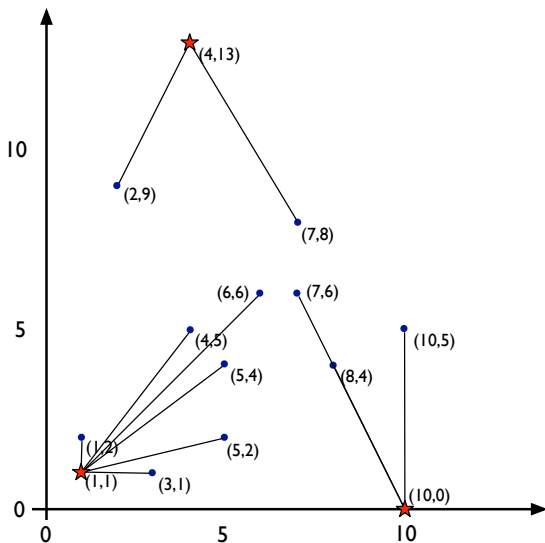


## k-means example: initialisation

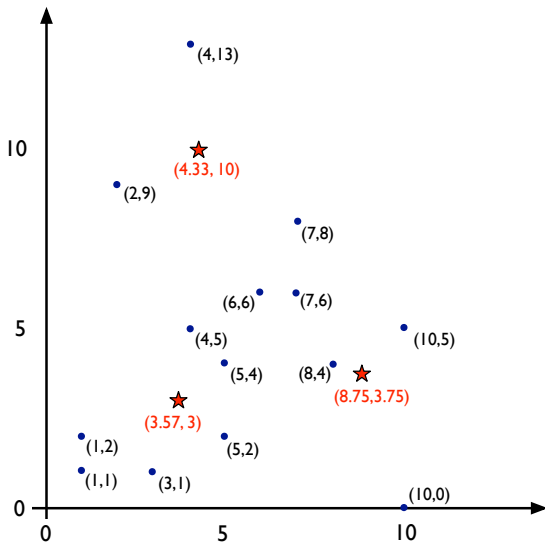




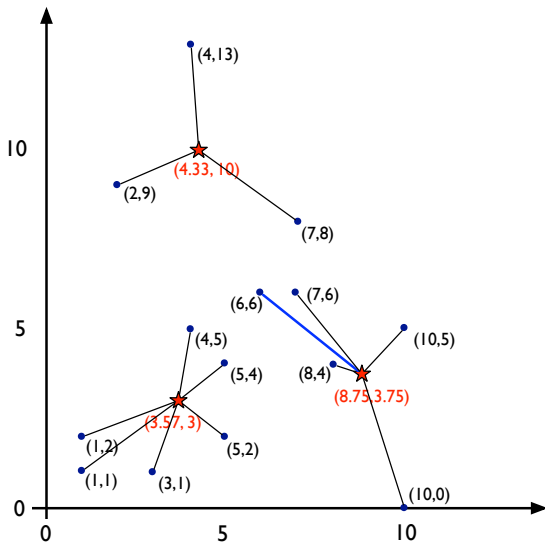
## k-means example: iteration 1 (assign points to clusters)



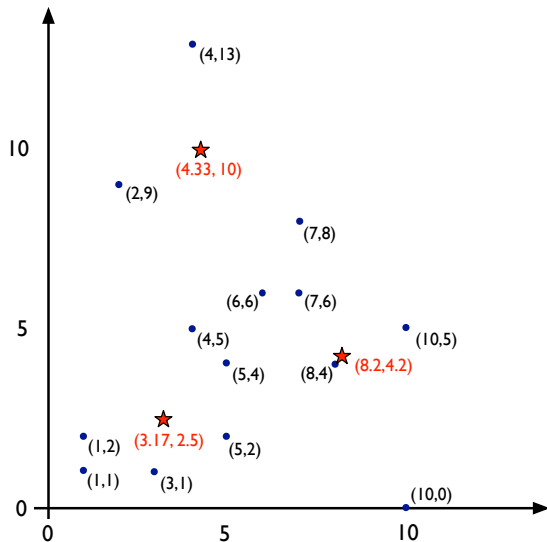
## k-means example: iteration 1 (recompute centres)



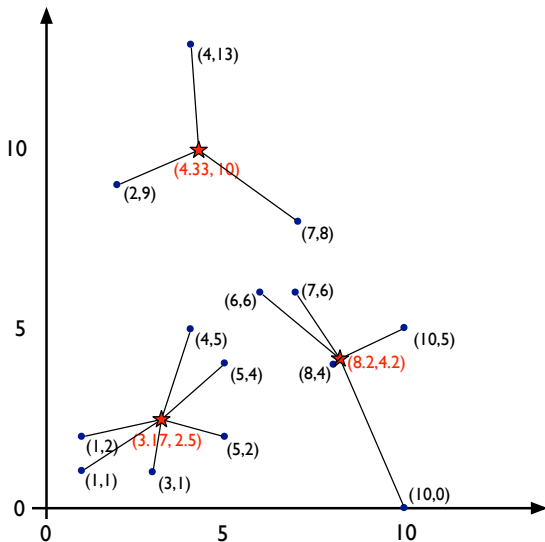
## k-means example: iteration 2 (assign points to clusters)



## k-means example: iteration 2 (recompute centres)



## k-means example: iteration 3 (assign points to clusters)



No changes, so converged

- A more flexible form of density estimation is made up of a linear combination of component densities:

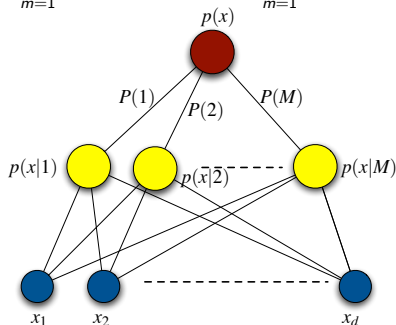
$$p(x) = \sum_{m=1}^M P(m)p(x|m)$$

- This is called a *mixture model* or a *mixture density*
- $p(x|m)$  : component densities
- $P(m)$  : mixing parameters
- Generative model:
  - 1 Choose a mixture component based on  $P(m)$
  - 2 Generate a data point  $x$  from the chosen component using  $p(x|m)$

# Gaussian mixture model

- The most important mixture model is the *Gaussian Mixture Model* (GMM), where the component densities are Gaussians
- Consider a GMM, where each component Gaussian  $\mathcal{N}(x; \mu_m, \Sigma_m)$  has mean  $\mu_m$  and a *spherical covariance*  $\Sigma_m = \sigma_m^2 \mathbf{I}$

$$p(x) = \sum_{m=1}^M P(m) p(x|m) = \sum_{m=1}^M P(m) \mathcal{N}(x; \mu_m, \sigma_m^2 \mathbf{I})$$



# GMM Parameter estimation when we know which component generated the data

- Define the indicator variable  $z_{mn} = 1$  if component  $m$  generated data point  $\mathbf{x}_n$  (and 0 otherwise)
- If  $z_{mn}$  wasn't hidden then we could count the number of observed data points generated by  $m$ :

$$N_m = \sum_{n=1}^N z_{mn}$$

- And estimate the mean, variance and mixing parameters as:

$$\hat{\boldsymbol{\mu}}_m = \frac{\sum_n z_{mn} \mathbf{x}_n}{N_m}$$

$$\hat{\sigma}_m^2 = \frac{\sum_n z_{mn} \|\mathbf{x}_n - \hat{\boldsymbol{\mu}}_m\|^2}{N_m}$$

$$\hat{P}(m) = \frac{1}{N} \sum_n z_{mn} = \frac{N_m}{N}$$



# GMM Parameter estimation when we don't know which component generated the data

- *Problem:* we don't know  $z_{mn}$  - which mixture component a data point comes from...
- Instead we use the EM algorithm: estimate the posterior probability  $P(m|x)$ , which gives the probability that component  $m$  was responsible for generating data point  $x$ , using an initial set of parameters,  $\lambda_0$
- At each iteration, we maximise

$$\sum_m P(m|x, \lambda_0) \log P(x|m, \lambda)$$

$$P(m|x, \lambda_0) = \frac{p(x|m) P(m)}{p(x)} = \frac{p(x|m) P(m)}{\sum_{m'=1}^M p(x|m') P(m')}$$

(dropping the dependence on  $\lambda_0$  for clarity)

# Soft assignment

- We can view the EM algorithm as estimating “*soft counts*” for the data points, based on the component occupation probabilities  $P(m|\mathbf{x}_n)$ :

$$N_m^* = \sum_{n=1}^N P(m|\mathbf{x}_n)$$

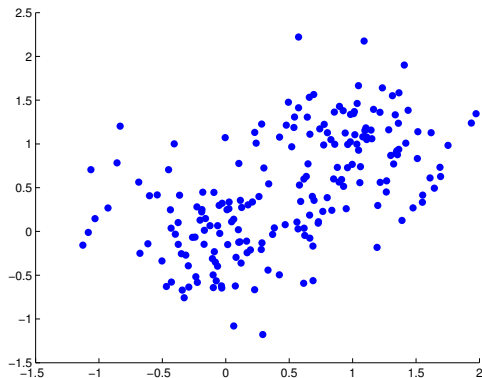
- We can imagine this as assigning data points to component  $m$  weighted by the component occupation probability  $P(m|\mathbf{x}_t)$
- Estimate the mean, variance and prior probabilities as:

$$\hat{\mu}_m = \frac{\sum_n P(m|\mathbf{x}_n)\mathbf{x}_n}{\sum_n P(m|\mathbf{x}_n)} = \frac{\sum_n P(m|\mathbf{x}_n)\mathbf{x}_n}{N_m^*}$$

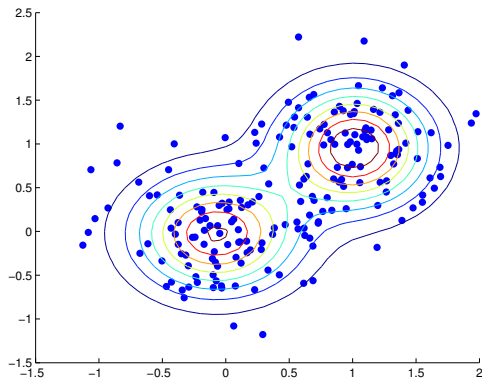
$$\hat{\sigma}_m^2 = \frac{\sum_n P(m|\mathbf{x}_n) \|\mathbf{x}_n - \hat{\mu}_m\|^2}{\sum_n P(m|\mathbf{x}_n)} = \frac{\sum_n P(m|\mathbf{x}_n) \|\mathbf{x}_n - \hat{\mu}_m\|^2}{N_m^*}$$

$$\hat{P}(m) = \frac{1}{n} \sum_n P(m|\mathbf{x}_n) = \frac{N_m^*}{n}$$

# Example 1 fit using a GMM

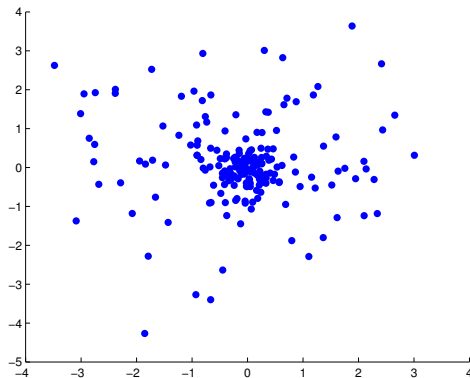


## Example 1 fit using a GMM



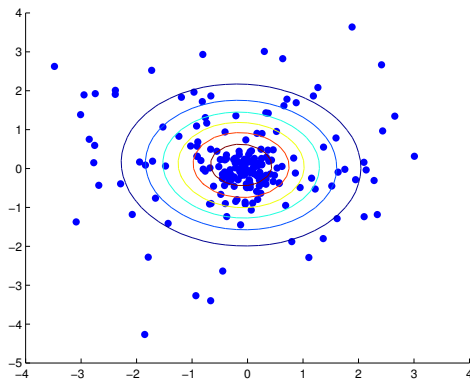
Fitted with a two component GMM using EM

## Peakily distributed data (Example 2)



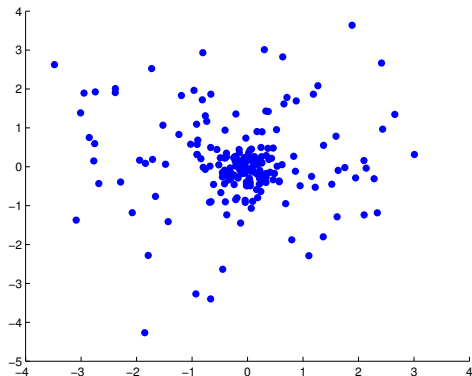
$$\mu_1 = \mu_2 = [0 \quad 0]^T \quad \Sigma_1 = 0.1I \quad \Sigma_2 = 2I$$

## Example 2 fit by a Gaussian

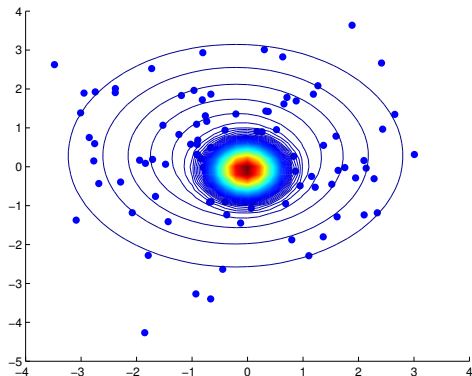


$$\mu_1 = \mu_2 = [0 \quad 0]^T \quad \Sigma_1 = 0.1I \quad \Sigma_2 = 2I$$

## Example 2 fit by a GMM



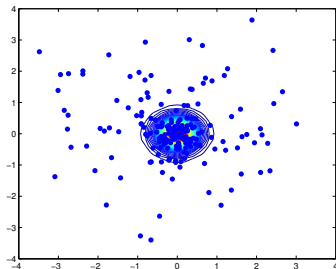
## Example 2 fit by a GMM



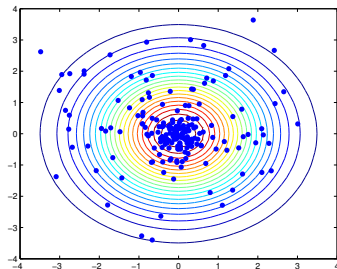
Fitted with a two component GMM using EM



## Example 2: component Gaussians



$$P(x|m=1)$$



$$P(x|m=2)$$

- GMMs trained using the EM algorithm are able to self organise to fit a data set
- Individual components take responsibility for parts of the data set (probabilistically)
- Soft assignment to components not hard assignment — “soft clustering”
- GMMs scale very well, e.g.: large speech recognition systems can have 30,000 GMMs, each with 32 components: sometimes 1 million Gaussian components!! And the parameters all estimated from (a lot of) data by EM

# HMMs with Gaussian observation probabilities

We can use a Gaussian distribution to model the observation probability:

$$b_j(x) = \mathcal{N}(x; \mu_j, \Sigma_j)$$

We need to find estimate parameters  $\hat{\mu}_j$ ,  $\hat{\Sigma}_j$  for each state  $j$ . Use the EM algorithm to weight each sample  $x_t$  by the occupation probability  $\gamma_j(t)$ :

$$\hat{\mu}_j = \frac{\sum_{t=1}^T \gamma_j(t) x_t}{\sum_{t=1}^T \gamma_j(t)}$$

And likewise for the covariance matrices:

$$\hat{\Sigma}_j = \frac{\sum_{t=1}^T \gamma_j(t) (x_t - \hat{\mu}_j)(x_t - \hat{\mu}_j)^T}{\sum_{t=1}^T \gamma_j(t)}$$

# Extension to Gaussian mixture model (GMM)

- The assumption of a Gaussian distribution at each state is very strong; in practice the acoustic feature vectors associated with a state may be strongly non-Gaussian
- In this case an  $M$ -component Gaussian mixture model is an appropriate density function:

$$b_j(x) = p(x|q=j) = \sum_{m=1}^M c_{jm} \mathcal{N}(x; \mu_{jm}, \Sigma_{jm})$$

Given enough components, this family of functions can model any distribution.

- Train using the EM algorithm again, in which the component occupation probabilities are estimated along with the state occupation probabilities in the E-step

# EM training of HMM/GMM

- Rather than estimating the state-time alignment, we estimate the component/state-time alignment, and component-state occupation probabilities  $\gamma_{jm}(t)$ : the probability of occupying mixture component  $m$  of state  $j$  at time  $t$ .

( $\xi_{tm}(j)$  in Jurafsky and Martin's SLP)

- Re-estimate the parameters of component  $m$  of state  $j$  as follows

$$\hat{\mu}_{jm} = \frac{\sum_{t=1}^T \gamma_{jm}(t) \mathbf{x}_t}{\sum_{t=1}^T \gamma_{jm}(t)}$$
$$\hat{\Sigma}_{jm} = \frac{\sum_{t=1}^T \gamma_{jm}(t) (\mathbf{x}_t - \hat{\mu}_{jm})(\mathbf{x}_t - \hat{\mu}_{jm})^T}{\sum_{t=1}^T \gamma_{jm}(t)}$$

- The mixture coefficients are re-estimated in a similar way to transition probabilities:

$$\hat{c}_{jm} = \frac{\sum_{t=1}^T \gamma_{jm}(t)}{\sum_{m'=1}^M \sum_{t=1}^T \gamma_{jm'}(t)}$$

# Doing the computation

- The forward, backward and Viterbi recursions result in a long sequence of probabilities being multiplied
- This can cause floating point *underflow* problems
- In practice computations are performed in the log domain (in which multiplies become adds)
- Working in the log domain also avoids needing to perform the exponentiation when computing Gaussians

- Gales and Young (2007). “The Application of Hidden Markov Models in Speech Recognition”, *Foundations and Trends in Signal Processing*, **1** (3), 195–304: section 2.2.
- Jurafsky and Martin (2008). *Speech and Language Processing* (2nd ed.): sections 6.1–6.5; 9.2; 9.4. (Errata at <http://www.cs.colorado.edu/~martin/SLP/Errata/SLP2-PIEV-Errata.html>)
- Rabiner and Juang (1989). “An introduction to hidden Markov models”, *IEEE ASSP Magazine*, **3** (1), 4–16.
- Renals and Hain (2010). “Speech Recognition”, *Computational Linguistics and Natural Language Processing Handbook*, Clark, Fox and Lappin (eds.), Blackwells.