Probabilistic Machine Learning:
6. EM for Gaussian Mixture Model
Introduction to Probabilistic Graphical Models

Tomasz Kajdanowicz, Piotr Bielak, Maciej Falkiewicz, Kacper Kania, Piotr Zieliński

Department of Computational Intelligence
Wroclaw University of Science and Technology

The presentation has been inspired and in some parts totally based on

1. Expectation–maximization algorithm article on Wikipedia
2. Machine Learning A Probabilistic Perspective, Kevin Murphy [1] Chapters 10. Directed graphical models (Bayes nets) and 11. Mixture models and the EM algorithm
3. Pattern Recognition and Machine Learning, Christopher M. Bishop [2] Chapter 9. Mixture Models and EM
4. Bayesian Reasoning and Machine Learning, Barber, D. [3] Chapters 2 Basic Graph Concepts and 3 Belief Networks

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Pre-reading

There was nothing to read.

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Table of Contents

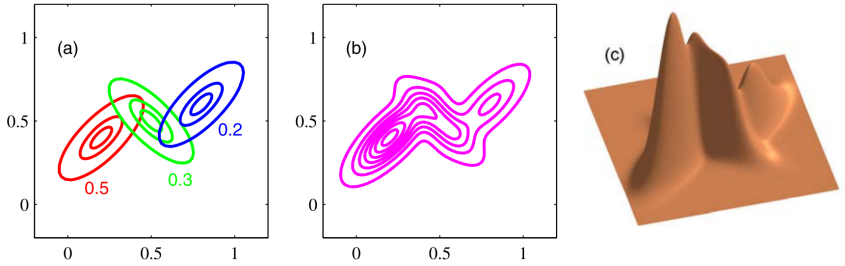
Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Gaussian Mixture Model

Recap from Lecture 2

- each Gaussian density $\mathcal{N}(x|\mu_k, \Sigma_k)$ is called *component* of the mixture
- each has own mean μ_k and covariance Σ_k
- α_k are mixing coefficients
- $\Sigma_{k=1}^{K} \alpha_k = 1$
- $0 \leq \alpha_k \leq 1$



Latent Variable Model

Soon you will get to know *Probabilistic Graphical Models* where the high-dimensional joint probability distributions of variables is represented by a graph.

Latent Variable Model

Soon you will get to know *Probabilistic Graphical Models* where the high-dimensional joint probability distributions of variables is represented by a graph.

An alternative approach is to assume that the *observed variables are correlated* because they arise from a *hidden common "cause"*. Model with hidden variables are also known as *latent variable models* or *LVM*s.

Mixture model

The simplest LVM

$z_i \in \{1, \ldots, K\}$ represent the discrete *latent state* with some prior, $\mathrm{P}(z_i) = \mathrm{Cat}(\boldsymbol{\pi})$, where $\boldsymbol{\pi} = \{\pi_i, \ldots, \pi_K\}$ is a vector of *mixing weights* satisfying $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$.

Mixture model

The simplest LVM

$z_i \in \{1, \ldots, K\}$ represent the discrete *latent state* with some prior, $\mathrm{P}(z_i) = \mathrm{Cat}(\boldsymbol{\pi})$, where $\boldsymbol{\pi} = \{\pi_i, \ldots, \pi_K\}$ is a vector of *mixing weights* satisfying $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$.
For the likelihood, we use $\mathrm{P}(x_i | z_i = k) = \mathrm{P}_k(x_i)$, where P_k is the k'th *base distribution* for the observations of any type.

$z_i \in \{1, \ldots, K\}$ represent the discrete *latent state* with some prior, $\mathrm{P}(z_i) = \mathrm{Cat}(\boldsymbol{\pi})$, where $\boldsymbol{\pi} = \{\pi_i, \ldots, \pi_K\}$ is a vector of *mixing weights* satisfying $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$.
For the likelihood, we use $\mathrm{P}(x_i|z_i = k) = \mathrm{P}_k(x_i)$, where P_k is the k'th *base distribution* for the observations of any type.
The overall model is known as a *mixture model*, since we are mixing together the K base distributions as follows:

$$\mathrm{P}(x_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathrm{P}_k(x_i|\boldsymbol{\theta}) \tag{1}$$

When P_k's are all Gaussians then it becomes a *Gaussian Mixture Model*.

Some popular LVMs

$p(\mathbf{x}_i \mid \mathbf{z}_i)$	$p(\mathbf{z}_i)$	Name	Section
MVN	Discrete	Mixture of Gaussians	11.2.1
Prod. Discrete	Discrete	Mixture of multinomials	11.2.2
Prod. Gaussian	Prod. Gaussian	Factor analysis/ probabilistic PCA	12.1.5
Prod. Gaussian	Prod. Laplace	Probabilistic ICA/ sparse coding	12.6
Prod. Discrete	Prod. Gaussian	Multinomial PCA	27.2.3
Prod. Discrete	Dirichlet	Latent Dirichlet allocation	27.3
Prod. Noisy-OR	Prod. Bernoulli	BN20/ QMR	10.2.3
Prod. Bernoulli	Prod. Bernoulli	Sigmoid belief net	27.7

Table 11.1 Summary of some popular directed latent variable models. Here "Prod" means product, so "Prod. Discrete" in the likelihood means a factored distribution of the form $\prod_j \text{Cat}(x_{ij}|\mathbf{z}_i)$, and "Prod. Gaussian" means a factored distribution of the form $\prod_j \mathcal{N}(x_{ij}|\mathbf{z}_i)$. "PCA" stands for "principal components analysis". "ICA" stands for "indepedendent components analysis".

Credit: *Machine Learning A Probabilistic Perspective, Kevin Murphy p. 339*

Parameter estimation for mixture models

For a second imagine z_i's are not latent, but observed.

Parameter estimation for mixture models

For a second imagine z_i's are not latent, but observed. Then the posterior for the parameters would be:

$$\mathsf{P}(\boldsymbol{\theta}|\mathcal{D}) = \mathrm{Dir}(\boldsymbol{\pi}|\mathcal{D}) \prod_{k=1}^{K} \mathrm{NIW}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\mathcal{D}) \tag{2}$$

Parameter estimation for mixture models

For a second imagine z_i's are not latent, but observed. Then the posterior for the parameters would be:

$$\mathsf{P}(\boldsymbol{\theta}|\mathcal{D}) = \mathrm{Dir}(\boldsymbol{\pi}|\mathcal{D}) \prod_{k=1}^{K} \mathrm{NIW}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|\mathcal{D}) \tag{2}$$

where NIW is the *Normal-inverse-wishart* distribution which is the posterior distribution for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, the parameters of Multivariate Normal Distribution. For details check [1] Section 4.6 Inferring the parameters of an MVN p.127-147.

We can easily find the globally optimal MAP estimate and hence globally optimal MLE.

Unidentifiability

But z_i's are not observed. For each configuration of latent variables we get a single MLE/MAP estimate. We say the parameters are not *identifiable*.

Unidentifiability

But z_i's are not observed. For each configuration of latent variables we get a single MLE/MAP estimate. We say the parameters are not *identifiable*.

There are $K!$ possible labelings, but some of the peaks might get merged. Nevertheless, there can be an exponential number, since finding the optimal MLE for a GMM is NP-hard.

Unidentifiability

But z_i's are not observed. For each configuration of latent variables we get a single MLE/MAP estimate. We say the parameters are not *identifiable*.

There are $K!$ possible labelings, but some of the peaks might get merged. Nevertheless, there can be an exponential number, since finding the optimal MLE for a GMM is NP-hard.

For example, suppose we draw some Monte Carlo samples from the posterior, $\boldsymbol{\Theta}^{(s)} \sim \mathrm{P}(\boldsymbol{\Theta}|\mathcal{D})$ and try to approximate the posterior mean, $\bar{\boldsymbol{\Theta}} = \frac{1}{S}\sum_{i=1}^{S} \boldsymbol{\Theta}^{(s)}$. Such mean would be meaningless, because the samples can come from different "modes" of the posterior.

GMM log-likelihood I

Let's take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right) \tag{3}$$

GMM log-likelihood I

Let's take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right) \tag{3}$$

Let's consider a non-trivial GGM, that is $K > 1$ and restrict to *spherical covariance matrix* (aka isotropic), that is $\boldsymbol{\Sigma}_k = \sigma_k^2 \mathrm{I}$.

GMM log-likelihood I

Let's take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right) \tag{3}$$

Let's consider a non-trivial GGM, that is $K > 1$ and restrict to *spherical covariance matrix* (aka isotropic), that is $\boldsymbol{\Sigma}_k = \sigma_k^2 \mathrm{I}$.

Suppose that one of the components of the mixture model, has its mean $\boldsymbol{\mu}_j$ exactly equal to one of the data points so that $\boldsymbol{\mu}_j = x_n$.

GMM log-likelihood I

Let's take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right) \tag{3}$$

Let's consider a non-trivial GGM, that is $K > 1$ and restrict to *spherical covariance matrix* (aka isotropic), that is $\boldsymbol{\Sigma}_k = \sigma_k^2 \mathrm{I}$.

Suppose that one of the components of the mixture model, has its mean $\boldsymbol{\mu}_j$ exactly equal to one of the data points so that $\boldsymbol{\mu}_j = x_n$.
This data point will then contribute a term in the likelihood function of the form

$$\mathcal{N}(x_n|x_n, \sigma_j^2 \mathrm{I}) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_j} \tag{4}$$

GMM log-likelihood II

$$\mathcal{N}(x_n|x_n, \sigma_j^2 \mathrm{I}) = \frac{1}{\sqrt{(2\pi)}} \frac{1}{\sigma_j}$$

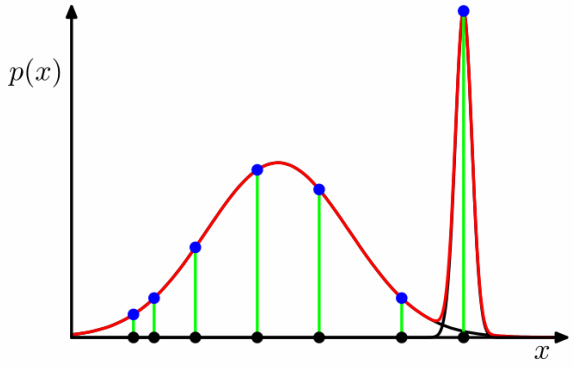
Now consider the limit $\sigma_j \to 0$. We see that this term goes to infinity and so the log-likelihood function will also go to infinity.

GMM log-likelihood II

$$\mathcal{N}(x_n|x_n, \sigma_j^2 \mathrm{I}) = \frac{1}{\sqrt{(2\pi)}} \frac{1}{\sigma_j}$$

Now consider the limit $\sigma_j \to 0$. We see that this term goes to infinity and so the log-likelihood function will also go to infinity.

Illustration of how singularities in the likelihood function arise with mixtures of Gaussians. This should be compared with the case of a single Gaussian shown in Figure 1.14 for which no singularities arise.



Credit: *Pattern Recognition and Machine Learning*, Christopher M. Bishop p. 434

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Variational Inference for GMM

Homework in L05.

Variational Inference for GMM

Homework in L05.

Anybody remembers the pre-reading for Lecture 3 - David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877 [4] ?

Variational Inference for GMM

Homework in L05.

Anybody remembers the pre-reading for Lecture 3 - David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877 [4] ?

The *Coordinate Ascent Mean-field variational inference* (CAVI) algorithm for univariate Gaussian Mixture is presented there. CAVI iteratively optimizes each factor of the mean-field variational density, while holding the others fixed. It climbs the ELBO to a local optimum.

Variational Inference for GMM

Homework in L05.

Anybody remembers the pre-reading for Lecture 3 - David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877 [4] ?

The *Coordinate Ascent Mean-field variational inference* (CAVI) algorithm for univariate Gaussian Mixture is presented there. CAVI iteratively optimizes each factor of the mean-field variational density, while holding the others fixed. It climbs the ELBO to a local optimum.

Pyro supports iterative optimization in SVI with `TraceEnum_ELBO()` and `infer={'enumerate': 'sequential' or 'parallel'}` property for `pyro.sample` which can be configure in a convenient wat using the `config_enumerate()` decorator.

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

K-means Clustering I

As well as providing a framework for building more complex probability distributions, mixture models can also be used to cluster data.

K-means Clustering I

As well as providing a framework for building more complex probability distributions, mixture models can also be used to cluster data.

Let's first consider the problem of finding clusters in a set of data points, which we approach first using a nonprobabilistic technique called the *K-means algorithm*.

K-means Clustering I

As well as providing a framework for building more complex probability distributions, mixture models can also be used to cluster data.

Let's first consider the problem of finding clusters in a set of data points, which we approach first using a nonprobabilistic technique called the *K-means algorithm*.

Suppose we have a data set $\{x_1, \ldots, x_N\}$ consisting of N observations of a random D-dimensional Euclidean variable x.

K-means Clustering I

As well as providing a framework for building more complex probability distributions, mixture models can also be used to cluster data.

Let's first consider the problem of finding clusters in a set of data points, which we approach first using a nonprobabilistic technique called the *K-means algorithm*.

Suppose we have a data set $\{x_1, \ldots, x_N\}$ consisting of N observations of a random D-dimensional Euclidean variable x.
Our goal is to partition the data set into some number K of clusters, where we shall suppose for the moment that the value of K is given.

K-means Clustering I

As well as providing a framework for building more complex probability distributions, mixture models can also be used to cluster data.

Let's first consider the problem of finding clusters in a set of data points, which we approach first using a nonprobabilistic technique called the *K-means algorithm*.

Suppose we have a data set $\{x_1, \ldots, x_N\}$ consisting of N observations of a random D-dimensional Euclidean variable x.
Our goal is to partition the data set into some number K of clusters, where we shall suppose for the moment that the value of K is given.
Intuitively, we might think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster.

K-means Clustering II

We can formalize this notion by first introducing a set of D-dimensional vectors $\boldsymbol{\mu}_k$, where $k = 1, \ldots, K$, in which $\boldsymbol{\mu}_k$ is a *prototype* associated with the k th cluster. As we shall see shortly, we can think of the $\boldsymbol{\mu}_k$ as representing the *centres of the clusters*.

K-means Clustering II

We can formalize this notion by first introducing a set of D-dimensional vectors $\boldsymbol{\mu}_k$, where $k = 1, \ldots, K$, in which $\boldsymbol{\mu}_k$ is a *prototype* associated with the k th cluster. As we shall see shortly, we can think of the $\boldsymbol{\mu}_k$ as representing the *centres of the clusters*.
Our goal is then to find

- an assignment of data points to clusters,
- as well as a set of vectors $\{\boldsymbol{\mu}_k\}$,

such that *the sum of the squares of the distances of each data point to its closest vector* $\boldsymbol{\mu}_k$, is a minimum.

K-means Clustering III

We can define the following objective function, sometimes called a *distortion measure*:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2, \tag{5}$$

where r_{nk}'s are entries of onehot vectors denoting cluster assignment.

K-means Clustering III

We can define the following objective function, sometimes called a *distortion measure*:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2, \tag{5}$$

where r_{nk}'s are entries of onehot vectors denoting cluster assignment.

We minimize J in an *interative procedure* in which each iteration involves *two successive steps* corresponding to

1. optimizations with respect to r_{nk} keeping $\boldsymbol{\mu}_k$'s fixed
2. optimizations with respect to $\boldsymbol{\mu}_k$ keeping r_{nk}'s fixed

K-means Clustering III

We can define the following objective function, sometimes called a *distortion measure*:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2, \tag{5}$$

where r_{nk}'s are entries of onehot vectors denoting cluster assignment.

We minimize J in an *interative procedure* in which each iteration involves *two successive steps* corresponding to

1. optimizations with respect to r_{nk} keeping $\boldsymbol{\mu}_k$'s fixed
2. optimizations with respect to $\boldsymbol{\mu}_k$ keeping r_{nk}'s fixed

To see how perform this steps see [2] chapter 9.1. *K*-means Clustering p. 424-425.

K-means Clustering – aplication

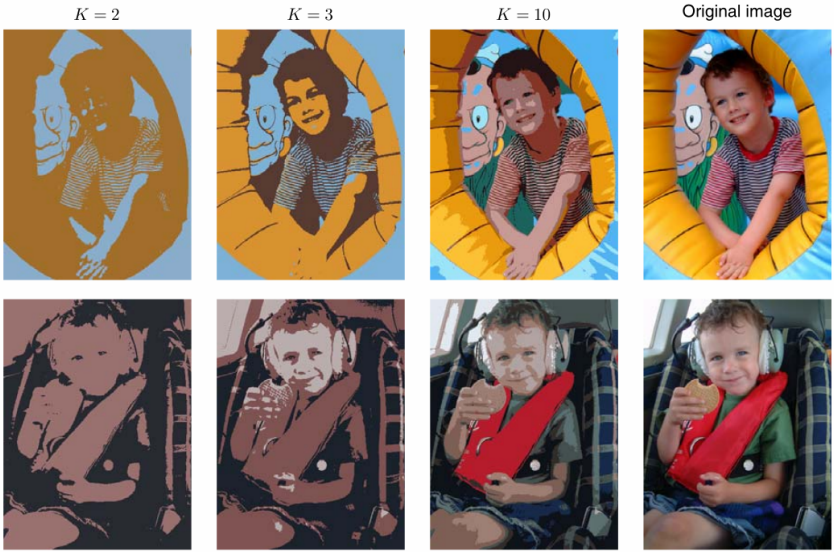


Figure 9.3 Two examples of the application of the K-means clustering algorithm to image segmentation showing the initial images together with their K-means segmentations obtained using various values of K. This also illustrates of the use of vector quantization for data compression, in which smaller values of K give higher compression at the expense of poorer image quality.

EM for GMM I

Again take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right)$$

EM for GMM I

Again take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right)$$

At the ML estimate the derivatives of log-likelihood with respect to the means $\boldsymbol{\mu}_k$ must satisfy:

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k(\mathbf{x}_n - \boldsymbol{\mu}_k) \qquad (9.16)$$

Credit: *Pattern Recognition and Machine Learning*, Christopher M. Bishop p. 435

EM for GMM I

Again take a look at the log-likelihood of GMM given data $\mathcal{D}$

$$\log \mathrm{P}(\mathcal{D}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right)$$

At the ML estimate the derivatives of log-likelihood with respect to the means $\boldsymbol{\mu}_k$ must satisfy:

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k(\mathbf{x}_n - \boldsymbol{\mu}_k) \qquad (9.16)$$

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 435*

$\gamma(z_{nk})$ can be seen as *responsibility* that component k takes for "explaining" the observation x_n.

EM for GMM II

By multyplpying by $\boldsymbol{\Sigma}_k^{-1}$ (which we assume to be nonsingular) and rearranging we can obtain

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n, \tag{6}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{7}$$

EM for GMM II

By multyplpying by $\mathbf{\Sigma}_k^{-1}$ (which we assume to be nonsingular) and rearranging we can obtain

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n, \tag{6}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}). \tag{7}$$

N_k can be interpreted as the effective number of points associated to cluster k.

EM for GMM III

Similar computations can be performed when taking derivatives with respect to $\boldsymbol{\Sigma}_k$.

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \tag{9.19}$$

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 436*

EM for GMM III

And finaly we can maximize log-likelihood with respect to the mixing coefficients π_k, taking into account the constraints. This can be achieved using a Lagrange multiplier and maximizing the following quantity

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right) \tag{9.20}$$

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 436*

EM for GMM IV

which gives

$$0 = \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \tag{9.21}$$

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 436*

EM for GMM IV

which gives

$$0 = \sum_{n=1}^{N} \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \tag{9.21}$$

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 436*

If we now multiply both sides by π_k and sum over k making use of the constraint, we find $\lambda = -N$. Using this to eliminate λ and rearranging we obtain

$$\pi_k = \frac{N}{N_k}, \tag{8}$$

so that the mixing coefficient for the kth component is given by the average responsibility which that component takes for explaining the data points.

EM for GMM V

So you just gave closed-form solution for all the parameters. This is the end?

EM for GMM V

So you just gave closed-form solution for all the parameters. This is the end?

These equations do not constitute a closed-form solution for the parameters of the mixture model because the responsibilities $\gamma(z_{nk})$ depend on those parameters in a complex way.

EM for GMM V

So you just gave closed-form solution for all the parameters. This is the end?

These equations do not constitute a closed-form solution for the parameters of the mixture model because the responsibilities $\gamma(z_{nk})$ depend on those parameters in a complex way.

However, these results do suggest a simple iterative scheme for finding a solution to the maximum likelihood problem, which turns out to be an instance of the EM algorithm for the particular case of the Gaussian mixture model.

EM for GMM V

The algorithm

We first choose some initial values for the means, covariances, and mixing coefficients.

EM for GMM V

The algorithm

We first choose some initial values for the means, covariances, and mixing coefficients.

Then we alternate between the following two updates:

1. *The E step* - Evaluate the *responsibilities* $\gamma(z_{nk})$

We first choose some initial values for the means, covariances, and mixing coefficients.

Then we alternate between the following two updates:

1. *The E step* - Evaluate the *responsibilities* $\gamma(z_{nk})$
2. *The M step* - Reestimate the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$, and mixing coefficients π_k using the results shown above.

We first choose some initial values for the means, covariances, and mixing coefficients.

Then we alternate between the following two updates:

1. *The E step* - Evaluate the *responsibilities* $\gamma(z_{nk})$
2. *The M step* - Reestimate the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$, and mixing coefficients π_k using the results shown above.

It can be show that each update to the parameters resulting from an E step followed by an M step is guaranteed to increase the log-likelihood function.

We first choose some initial values for the means, covariances, and mixing coefficients.

Then we alternate between the following two updates:

1. *The E step* - Evaluate the *responsibilities* $\gamma(z_{nk})$
2. *The M step* - Reestimate the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$, and mixing coefficients π_k using the results shown above.

It can be show that each update to the parameters resulting from an E step followed by an M step is guaranteed to increase the log-likelihood function.

In practice, the algorithm is deemed to have converged when the change in the log-likelihood function, or alternatively in the parameters, falls below some threshold.

EM for GMM VI

Example

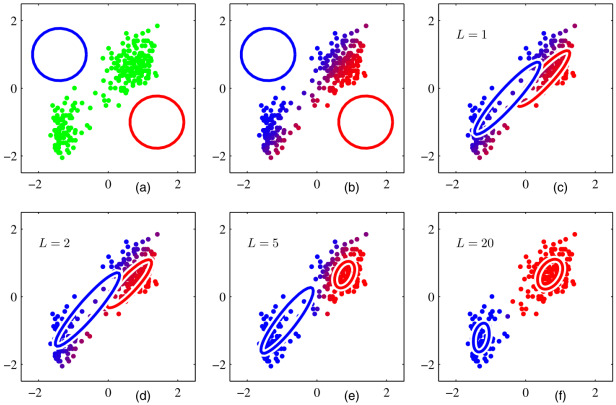


Illustration of the EM algorithm using the Old Faithful set. *Plot (a)* shows the data points in green (unasigned), together with the configuration of the mixture model as the one standard-deviation contours.

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 437*

EM for GMM VI

Example

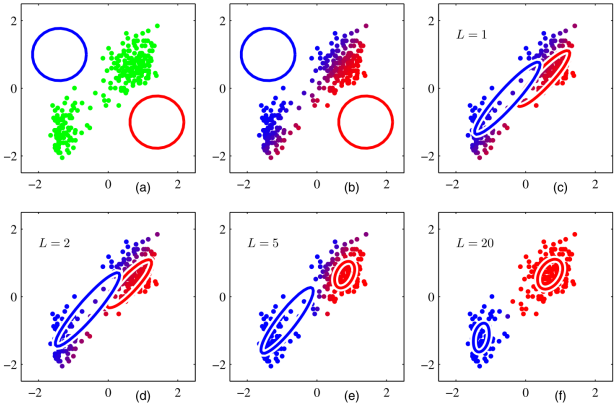


Illustration of the EM algorithm using the Old Faithful set. *Plot (a)* shows the data points in green (unasigned), together with the configuration of the mixture model as the one standard-deviation contours. *Plot (b)* shows situation after the first E step, where data points are coloured according to γ.

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 437*

EM for GMM VI

Example

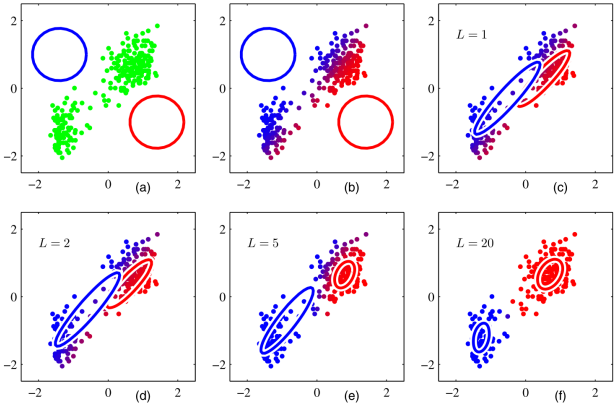


Illustration of the EM algorithm using the Old Faithful set. *Plot (a)* shows the data points in green (unasigned), together with the configuration of the mixture model as the one standard-deviation contours. *Plot (b)* shows situation after the first E step, where data points are coloured according to γ. *Plot (c)* shows distribution update after M step.

Credit: *Pattern Recognition and Machine Learning*, *Christopher M. Bishop p. 437*

EM in general I

Given the statistical model which generates a set $\mathcal{D}$ of observed data, a set of unobserved latent data or missing values Z, and a vector of unknown parameters θ, along with a likelihood function $L(\theta; \mathcal{D}, Z) = \mathrm{P}(\mathcal{D}, Z|\theta)$, the maximum likelihood estimate (MLE) of the unknown parameters is determined by maximizing the marginal likelihood of the observed data

$$L(\theta; \mathcal{D}) = \mathrm{P}(\mathcal{D}|\theta) = \int \mathrm{P}(\mathcal{D}; Z|\theta)\mathrm{d}Z \tag{9}$$

EM in general I

Given the statistical model which generates a set $\mathcal{D}$ of observed data, a set of unobserved latent data or missing values Z, and a vector of unknown parameters θ, along with a likelihood function $L(\theta; \mathcal{D}, Z) = \mathrm{P}(\mathcal{D}, Z|\theta)$, the maximum likelihood estimate (MLE) of the unknown parameters is determined by maximizing the marginal likelihood of the observed data

$$L(\theta; \mathcal{D}) = \mathrm{P}(\mathcal{D}|\theta) = \int \mathrm{P}(\mathcal{D}; Z|\theta)\mathrm{d}Z \tag{9}$$

The EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying these two steps:

1. *Expectation step (E step)* Define $Q(\theta|\theta^{t-1}) = \mathbb{E}_{Z|X,\theta^{t-1}}[\log L(\theta; \mathcal{D}, Z)]$, that is the expected value of the log likelihood function of θ with respect to the current conditional distribution of Z given $\mathcal{D}$ and the current estimates of the parameters.

EM in general I

Given the statistical model which generates a set $\mathcal{D}$ of observed data, a set of unobserved latent data or missing values Z, and a vector of unknown parameters θ, along with a likelihood function $L(\theta; \mathcal{D}, Z) = \mathrm{P}(\mathcal{D}, Z|\theta)$, the maximum likelihood estimate (MLE) of the unknown parameters is determined by maximizing the marginal likelihood of the observed data

$$L(\theta; \mathcal{D}) = \mathrm{P}(\mathcal{D}|\theta) = \int \mathrm{P}(\mathcal{D}; Z|\theta)\mathrm{d}Z \tag{9}$$

The EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying these two steps:

1. *Expectation step (E step)* Define $Q(\theta|\theta^{t-1}) = \mathbb{E}_{Z|X,\theta^{t-1}}[\log L(\theta; \mathcal{D}, Z)]$, that is the expected value of the log likelihood function of θ with respect to the current conditional distribution of Z given $\mathcal{D}$ and the current estimates of the parameters.
2. *Maximization steo (M step)* Find parameters thet maximize Q:

$$\theta^t = \arg\max Q(\theta|\theta^{t-1}) \tag{10}$$

EM in general II

For a complete description, proves, extensions and aplication to other models check [1] chapter 11.4 The EM algorithm p.348-369.

EM in general II

For a complete description, proves, extensions and aplication to other models check [1] chapter 11.4 The EM algorithm p.348-369.

For an intuitive introduction of the general EM from the perspective of GMM and even more intuitive examples check [2] chapter 9.3. An Alternative View of EM p.439-441

Why not both?

Anybody wondering why not benefit from the EM framework in Variational Inference?

Why not both?

Anybody wondering why not benefit from the EM framework in Variational Inference?

It hase been already done – Nikolaos Nasios and Adrian G. Bors. “Variational learning for Gaussian mixture models”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 36.4 (2006), pp. 849–862 [5]

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Graphs

Graph

A graph G consists of nodes (also called vertices) and edges (also called links) between the nodes. Edges may be directed (they have an arrow in a single direction) or undirected. Edges can also have associated weights. A graph with all edges directed is called a directed graph, and one with all edges undirected is called an undirected graph

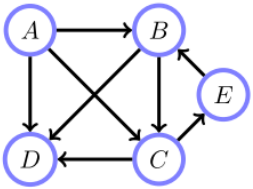


Figure: An directed graph G consists of directed edges between nodes

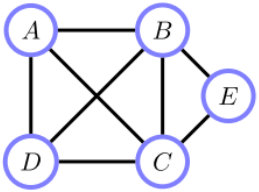


Figure: An undirected graph G consists of undirected edges between nodes

What do we need graphs for?

Two variables will be independent if they are not linked by a path on the graph.

Path, ancestors, descendants

Path

A *path* $A \to B$ from node A to node B is a sequence of nodes that connects A to B. That is, a path is of the form $A_0, A_1, \ldots, A_{n-1}, A_n$, with $A_0 = A$ and $A_n = B$ and each edge (A_{k-1}, A_k), $k = 1, \ldots, n$ being in the graph. A directed path is a sequence of nodes which when we follow the direction of the arrows leads us from A to B.

Ancestor

In directed graphs, the nodes A such that $A \to B$ and $B \not\to A$ are the *ancestors* of B.

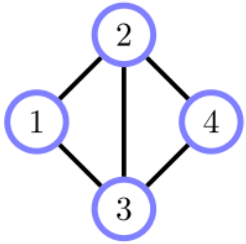
Descendant

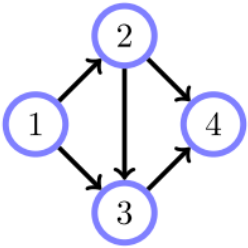
The nodes B such that $A \to B$ and $B \not\to A$ are the *descendants* of A.

Cycle

Cycle

A cycle is a directed path that starts and returns to the same node $a \to b \to \ldots \to z \to a$

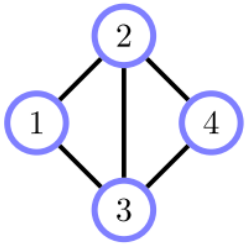


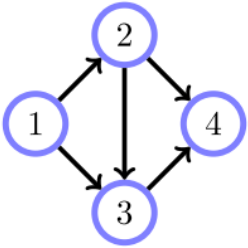


Loop

Loop

A loop is a path containing more than two nodes, irrespective of edge direction, that starts and returns to the same node.

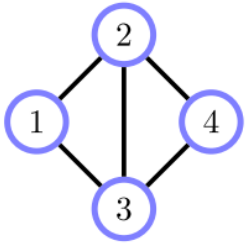


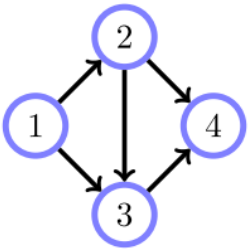


Chord

Chord

A chord is an edge that connects two non-adjacent nodes in a loop.





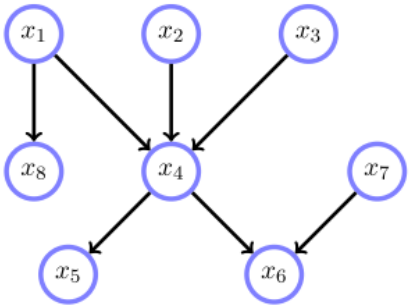
Directed Acyclic Graph (DAG)

DAG

A DAG is a graph G with directed edges (arrows on each link) between the nodes such that by following a path of nodes from one node to another along the direction of each edge no path will revisit a node.

In a DAG the ancestors of B are those nodes who have a directed path ending at B. Conversely, the descendants of A are those nodes who have a directed path starting at A.

Relationships in a DAG



parents, children, family

- the parents of x_4 are $pa(x_4) = \{x_1, x_2, x_3\}$
- the children of x_4 are $ch(x_4) = \{x_5, x_6\}$
- the family of a node is itself and its parents

Markov blanket

The Markov blanket of a node is its parents, children and the parents of its children, excluding itself. (Markov blanket of x_4 is $x_1, x_2, x_3, x_5, x_6, x_7$

Neighbour, Clique

Neighbour

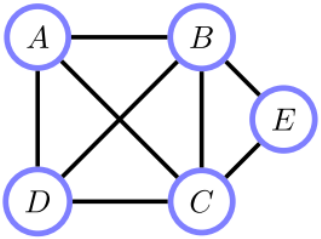
For an undirected graph G the neighbours of x, $ne(x)$ are those nodes directly connected to x.

Clique

Given an undirected graph, a clique is a fully connected subset of nodes.

- all the members of the clique are neighbours
- for a maximal clique there is no larger clique that contains the clique

Clique example



- the graph has two maximal cliques, $C1 = \{A, B, C, D\}$ and $C2 = \{B, C, E\}$
- A, B, C are fully connected, but this is a non-maximal clique
- there is a larger fully connected set: A, B, C, D

Cliques play a role in:

- modelling - describe variables that are all dependent on each other
- inference - describe sets of variables with no simpler structure that makes the relationship between them and for which no simpler efficient inference procedure is likely to exist

Connected graph

Connected graph

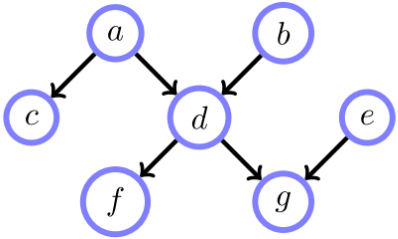
An undirected graph is connected if there is a path between every pair of nodes (i.e. there are no isolated islands). For a graph which is not connected, the connected components are those subgraphs which are connected.

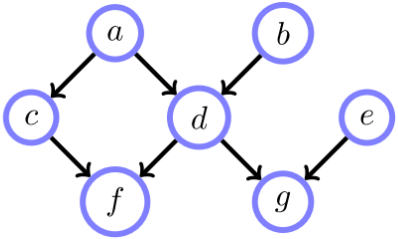
Singly Connected Graph

A graph is *singly connected* if there is only one path from any node A to any other node B. Otherwise the graph is multiply connected (for directed and undirected graphs)

An alternative name for a singly connected graph is a tree. A multiply-connected graph is also called loopy.

Singly, multiply connected

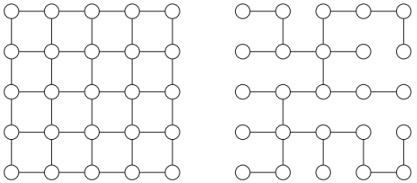




Spanning Tree

Spanning Tree

A spanning tree of an undirected graph G is a singly-connected subset of the existing edges such that the resulting singly-connected graph covers all nodes of G



A maximum weight spanning tree is a spanning tree such that the sum of all weights on the edges of the tree is at least as large as any other spanning tree of G.

Finding a maximal weight spanning tree

find a spanning tree with maximal weight()

- pick the next candidate edge which has the largest weight and add this to the edge set
- if this results in an edge set with cycles, then reject the candidate edge and propose the next largest edge weight
- pick the edge with the largest weight and add this to the edge set

There may be more than one maximal weight spanning tree.

Table of Contents

Estimation in Gaussian Mixture Model
- Introduction
- Variational Inference
- Expectation-Maximization (EM) algorithm

Introduction to Probabilistic Graphical Models
- Graphs
- Probabilistic Graphical Models – a starter

Chain rule

By the *chain rule* of probability, we can always represent a *joint distribution* as follows, using any ordering of the variables:

$$\mathrm{P}(x_{1:V}) = \mathrm{P}(x_1)\mathrm{P}(x_2|x_1)\mathrm{P}(x_3|x_2,x_1)\mathrm{P}(x_4|x_1,x_2,x_3)\ldots\mathrm{P}(x_V|x_{1:V-1}) \tag{11}$$

, where V is the number of variables, and where we have dropped the conditioning on the fixed parameters θ for brevity.

Chain rule

By the *chain rule* of probability, we can always represent a *joint distribution* as follows, using any ordering of the variables:

$$\mathrm{P}(x_{1:V}) = \mathrm{P}(x_1)\mathrm{P}(x_2|x_1)\mathrm{P}(x_3|x_2,x_1)\mathrm{P}(x_4|x_1,x_2,x_3)\ldots\mathrm{P}(x_V|x_{1:V-1}) \tag{11}$$

, where V is the number of variables, and where we have dropped the conditioning on the fixed parameters θ for brevity.

The problem with this expression is that it becomes more and more complicated to represent the conditional distributions $\mathrm{P}(x_t|x_{1:t-1})$ as t gets large.

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Recall that X and Y are conditionally independent given Z, denoted $X \perp\!\!\!\perp Y|Z$, if and only if (iff) the conditional joint can be written as a product of conditional marginals,

$$X \perp\!\!\!\perp Y|Z \Leftrightarrow \mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z) \tag{12}$$

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Recall that X and Y are conditionally independent given Z, denoted $X \perp\!\!\!\perp Y|Z$, if and only if (iff) the conditional joint can be written as a product of conditional marginals,

$$X \perp\!\!\!\perp Y|Z \Leftrightarrow \mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z) \tag{12}$$

How can this be helpful?

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Recall that X and Y are conditionally independent given Z, denoted $X \perp\!\!\!\perp Y|Z$, if and only if (iff) the conditional joint can be written as a product of conditional marginals,

$$X \perp\!\!\!\perp Y|Z \Leftrightarrow \mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z) \tag{12}$$

How can this be helpful? Let's assume that *"the future is independent of the past given the present"*.

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Recall that X and Y are conditionally independent given Z, denoted $X \perp\!\!\!\perp Y|Z$, if and only if (iff) the conditional joint can be written as a product of conditional marginals,

$$X \perp\!\!\!\perp Y|Z \Leftrightarrow P(X, Y|Z) = P(X|Z)P(Y|Z) \tag{12}$$

How can this be helpful? Let's assume that *"the future is independent of the past given the present"*.
This is called the (first order) *Markov assumption*. Using this assumption, plus the chain rule, we can write the joint distribution as follows

$$P(x_{1:V}) = P(x_1) \prod_{t=1}^{V} P(x_t|x_{t-1}) \tag{13}$$

Conditional Independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence.

Recall that X and Y are conditionally independent given Z, denoted $X \perp\!\!\!\perp Y|Z$, if and only if (iff) the conditional joint can be written as a product of conditional marginals,

$$X \perp\!\!\!\perp Y|Z \Leftrightarrow \mathrm{P}(X, Y|Z) = \mathrm{P}(X|Z)\mathrm{P}(Y|Z) \tag{12}$$

How can this be helpful? Let's assume that *"the future is independent of the past given the present"*.
This is called the (first order) *Markov assumption*. Using this assumption, plus the chain rule, we can write the joint distribution as follows

$$\mathrm{P}(x_{1:V}) = \mathrm{P}(x_1) \prod_{t=1}^{V} \mathrm{P}(x_t|x_{t-1}) \tag{13}$$

This is called a (first-order) *Markov chain*. They can be characterized by an initial distribution over states, $\mathrm{P}(x_1 = i)$, plus a state transition matrix $\mathrm{P}(x_t = j|x_{t-1} = i)$.

How to determine the Conditional Independece?

d-separation

Conditional Independence

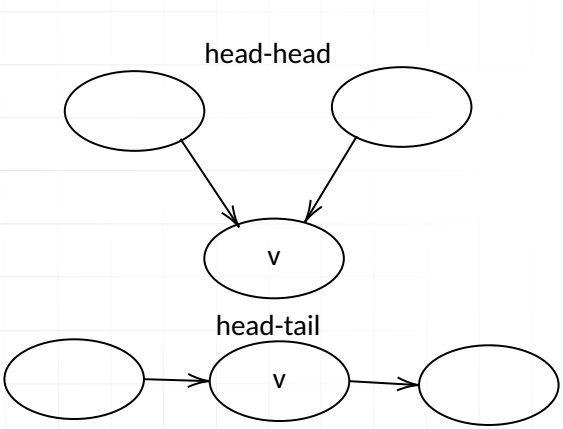
For random variables X, Y, Z if X and Y are ***d-separated*** by Z then $(X \perp Y|Z)$

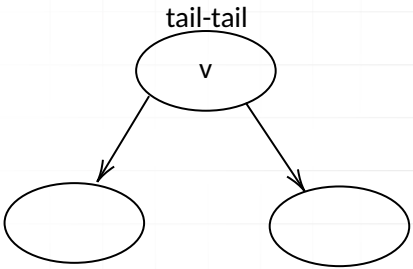
d-separation

X and Y are d-separated by Z if all paths from a vertex of X to a vertex of Y are **blocked** with respect to v.

Types of connection on the path

with respect to Z





Blocked path

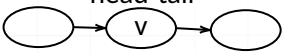
Blocked path

A path (not necessary directed) between two vertices is **blocked** with respect to C (even set of vertices) if it passes through a vertex v s.t. either:

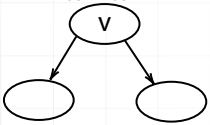
- the connections are head-tail or tail-tail and $v \in C$
- the connections are head-head and $v \notin C$ and none of descendants of v are in C

$v \in C$

head-tail

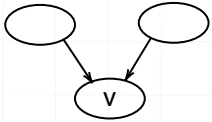


tail-tail

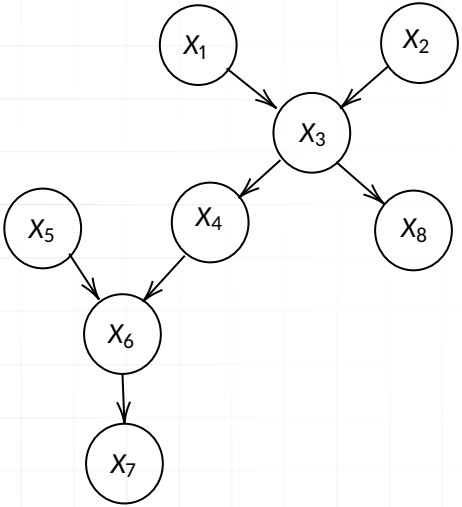


$v \notin C$ and none of descendants of v are in C

head-head

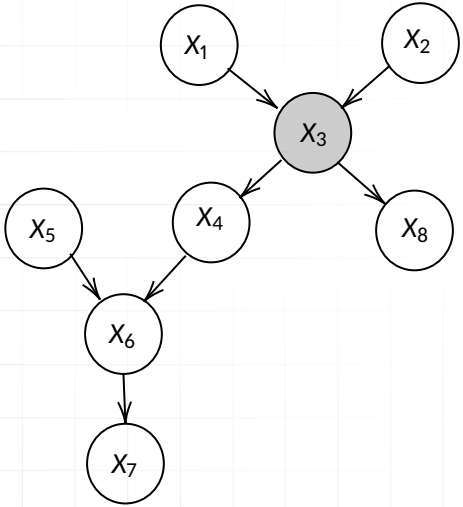


Example



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

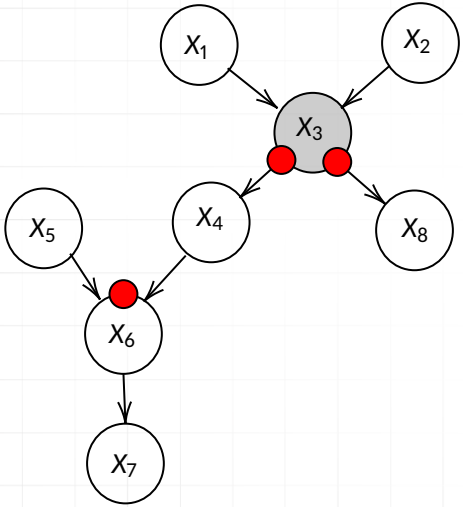
Example



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated
1	4	yes
1	2	no
4	5	yes

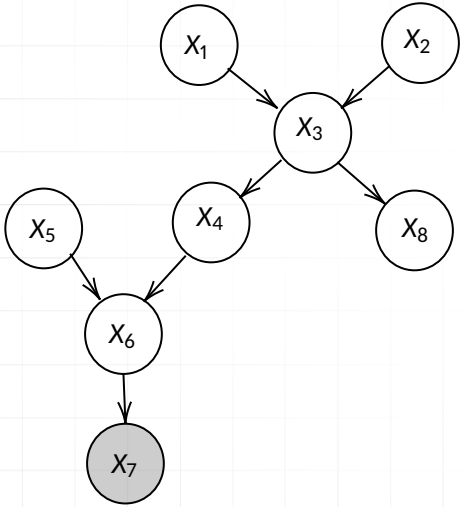
Example



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated
1	4	yes
1	2	no
4	5	yes
4	7	no
4	8	yes
4	6	no
2	7	yes
2	5	yes
5	8	yes

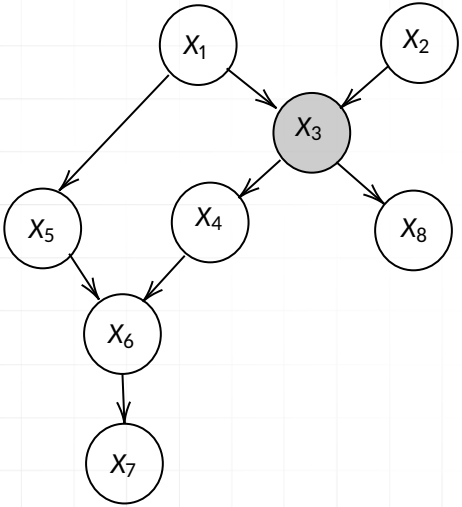
Example 2



- taking e.g.:
  - $C = \{X_7\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated

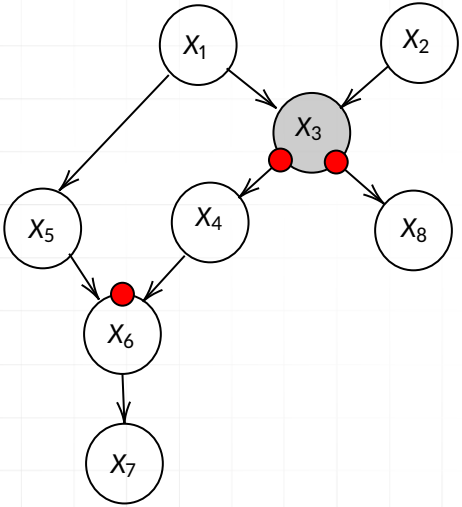
Example 3



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated

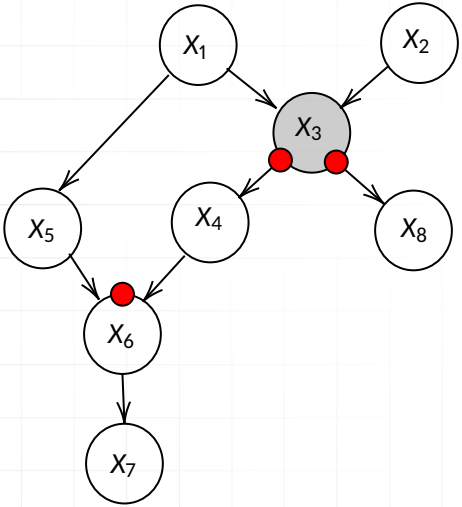
Example 3



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated

Example 3



- taking e.g.:
  - $C = \{X_3\}$
- $(X_i \perp X_j)|X_C$???

i	j	d-separated
2	5	no
2	7	no
2	4	yes
7	8	yes

To be continued...

Bibliography I

[1] Kevin P. Murphy. *Machine learning : a probabilistic perspective*. Cambridge, Mass. [u.a.]: MIT Press, 2013.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] D. Barber. *Bayesian Reasoning and Machine Learning*. 04-2011. In press. Cambridge University Press, 2011.

[4] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877.

[5] Nikolaos Nasios and Adrian G. Bors. "Variational learning for Gaussian mixture models". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 36.4 (2006), pp. 849–862.