# CSCI567 Machine Learning (Fall 2021)

Prof. Haipeng Luo

U of Southern California

Oct 28, 2021

# Administration

HW3: discuss solutions today

# Administration

HW3: discuss solutions today

HW4: to be released, due on Tue, 11/09

# Outline

1. Review of last lecture
2. Clustering
3. Gaussian mixture models

# Outline

# General training algorithm for decision trees

**DecisionTreeLearning**(Examples, Features)

- if Examples have the same class, return a leaf with this class
- else if Features is empty, return a leaf with the majority class
- else if Examples is empty, return a leaf with majority class of parent
- else

  find the best feature $A$ to split (e.g. based on conditional entropy)

  **Tree** $\leftarrow$ a root with test on $A$

  For each value $a$ of $A$:

    **Child** $\leftarrow$ **DecisionTreeLearning**(Examples with $A = a$, Features$\setminus\{A\}$)
    add **Child** to **Tree** as a new branch

- return **Tree**

# The AdaBoost Algorithm

Given a training set $S$ and a base algorithm $\mathcal{A}$, initialize $D_1$ to be uniform

For $t = 1, \ldots, T$

- obtain a weak classifier $h_t \leftarrow \mathcal{A}(S, D_t)$
- calculate the importance of $h_t$ as

$$\beta_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right) \qquad (\beta_t > 0 \Leftrightarrow \epsilon_t < 0.5)$$

  where $\epsilon_t = \sum_{n: h_t(\boldsymbol{x}_n) \neq y_n} D_t(n)$ is the weighted error of $h_t$.

- update distributions

$$D_{t+1}(n) \propto D_t(n) e^{-\beta_t y_n h_t(\boldsymbol{x}_n)} = \begin{cases} D_t(n) e^{-\beta_t} & \text{if } h_t(x_n) = y_n \\ D_t(n) e^{\beta_t} & \text{else} \end{cases}$$

Output the final classifier $H(\boldsymbol{x}) = \text{sgn}\left( \sum_{t=1}^{T} \beta_t h_t(\boldsymbol{x}) \right)$

# Outline

# Supervised learning v.s unsupervised learning

Recall there are different types of machine learning problems

# Supervised learning v.s unsupervised learning

Recall there are different types of machine learning problems

- **supervised learning** (what we have discussed so far)
  Aim to predict, e.g. classification and regression

# Supervised learning v.s unsupervised learning

Recall there are different types of machine learning problems

- **supervised learning** (what we have discussed so far)
  Aim to predict, e.g. classification and regression
- **unsupervised learning** (main focus from now on)
  Aim to discover hidden/latent patterns and explore data

# Supervised learning v.s unsupervised learning

Recall there are different types of machine learning problems

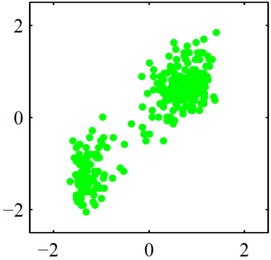
- **supervised learning** (what we have discussed so far)
  Aim to predict, e.g. classification and regression

- **unsupervised learning** (main focus from now on)
  Aim to discover hidden/latent patterns and explore data

Today's focus: **clustering**, an important unsupervised learning problem

# Clustering: informal definition

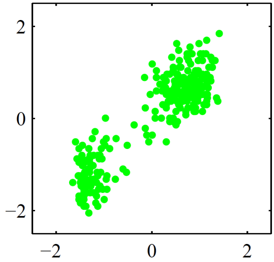
**Given**: a set of data points (feature vectors), *without labels*



# Clustering: informal definition

**Given**: a set of data points (feature vectors), *without labels*

**Output**: group the data into some clusters,

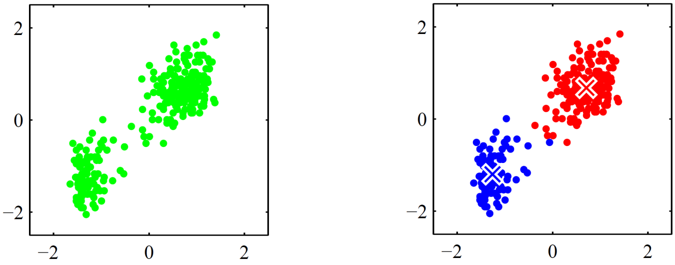


# Clustering: informal definition

**Given**: a set of data points (feature vectors), *without labels*

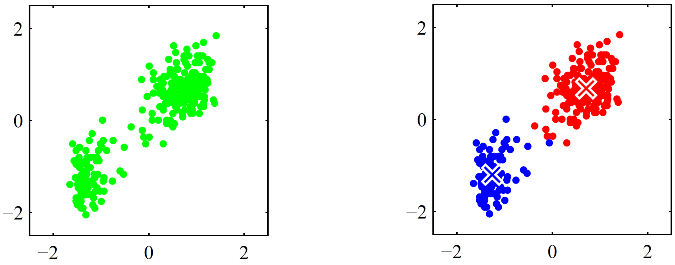
**Output**: group the data into some clusters, which means

- assign each point to a specific cluster
- find the center (representative/prototype/...) of each cluster



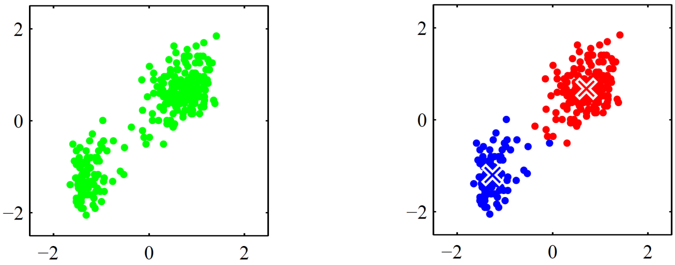
# Clustering: formal definition

**Given**: data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^\mathsf{D}$



# Clustering: formal definition

**Given**: data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^{\mathsf{D}}$ and #clusters $K$ we want

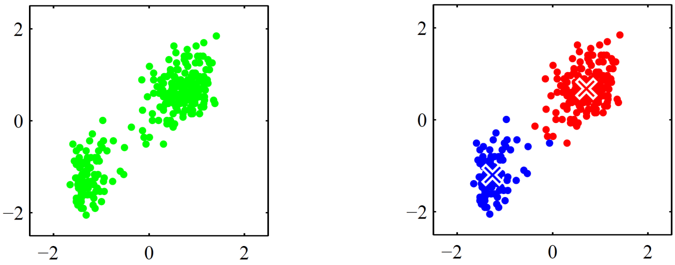


# Clustering: formal definition

**Given**: data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^{\mathsf{D}}$ and #clusters $K$ we want

**Output**: group the data into $K$ clusters, which means

- find assignment $\gamma_{nk} \in \{0, 1\}$ for each data point $n \in [N]$ and $k \in [K]$ s.t. $\sum_{k \in [K]} \gamma_{nk} = 1$ for any fixed $n$

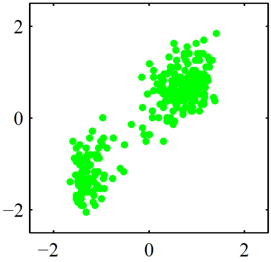


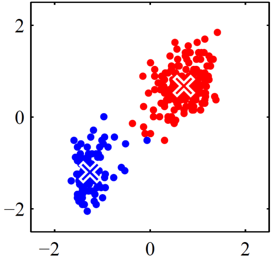
# Clustering: formal definition

**Given**: data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^{\mathsf{D}}$ and #clusters $K$ we want

**Output**: group the data into $K$ clusters, which means

- find assignment $\gamma_{nk} \in \{0, 1\}$ for each data point $n \in [N]$ and $k \in [K]$ s.t. $\sum_{k \in [K]} \gamma_{nk} = 1$ for any fixed $n$
- find the cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^{\mathsf{D}}$



# Many applications

- recognize communities in a social network
- group similar customers in market research
- image segmentation
- accelerate other algorithms (e.g. NNC as in programing projects)
- ...

# One example

**image compression**:

- each pixel is a point
- perform clustering over these points
- replace each point by the center of the cluster it belongs to




Original image  Large $K \longrightarrow$ Small $K$

# Formal Objective

Key difference from supervised learning problems: no labels given, which means *no ground-truth to even measure the quality of your answer!*

# Formal Objective

Key difference from supervised learning problems: no labels given, which means *no ground-truth to even measure the quality of your answer!*

Still, we can turn it into an optimization problem, e.g. through the popular **"K-means" objective**: find $\gamma_{nk}$ and $\boldsymbol{\mu}_k$ to minimize

$$F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

i.e. the **sum of squared distances of each point to its center**.

# Formal Objective

Key difference from supervised learning problems: no labels given, which means *no ground-truth to even measure the quality of your answer!*

Still, we can turn it into an optimization problem, e.g. through the popular **"K-means" objective**: find $\gamma_{nk}$ and $\boldsymbol{\mu}_k$ to minimize

$$F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

i.e. the **sum of squared distances of each point to its center**.

Unfortunately, finding the exact minimizer is *NP-hard!*

# Alternating minimization

Instead, use a heuristic that alternatingly minimizes over $\{\gamma_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$:

# Alternating minimization

Instead, use a heuristic that alternatingly minimizes over $\{\gamma_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$:

Initialize $\{\boldsymbol{\mu}_k^{(1)}\}$

# Alternating minimization

Instead, use a heuristic that alternatingly minimizes over $\{\gamma_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$:

Initialize $\{\boldsymbol{\mu}_k^{(1)}\}$

For $t = 1, 2, \ldots$

- find

$$\{\gamma_{nk}^{(t+1)}\} = \operatorname*{argmin}_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k^{(t)}\}\right)$$

# Alternating minimization

Instead, use a heuristic that alternatingly minimizes over $\{\gamma_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$:

Initialize $\{\boldsymbol{\mu}_k^{(1)}\}$

For $t = 1, 2, \ldots$

- find
$$\{\gamma_{nk}^{(t+1)}\} = \operatorname*{argmin}_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k^{(t)}\}\right)$$

- find
$$\{\boldsymbol{\mu}_k^{(t+1)}\} = \operatorname*{argmin}_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}^{(t+1)}\}, \{\boldsymbol{\mu}_k\}\right)$$

# A closer look

The first step

$$\min_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \min_{\{\gamma_{nk}\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

# A closer look

The first step

$$\min_{\{\gamma_{nk}\}} F(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}) = \min_{\{\gamma_{nk}\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$
$$= \sum_n \min_{\{\gamma_{nk}\}} \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

# A closer look

The first step

$$\min_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \min_{\{\gamma_{nk}\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$
$$= \sum_n \min_{\{\gamma_{nk}\}} \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

is simply to **assign each $x_n$ to the closest $\boldsymbol{\mu}_k$**, i.e.

$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

for all $k \in [K]$ and $n \in [N]$.

# A closer look

The second step

$$\min_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \min_{\{\boldsymbol{\mu}_k\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

# A closer look

The second step

$$\min_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \min_{\{\boldsymbol{\mu}_k\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$
$$= \sum_k \min_{\boldsymbol{\mu}_k} \sum_{n:\gamma_{nk}=1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

# A closer look

The second step

$$\min_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \min_{\{\boldsymbol{\mu}_k\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$
$$= \sum_k \min_{\boldsymbol{\mu}_k} \sum_{n:\gamma_{nk}=1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

is simply **to average the points of each cluster** (hence the name)

$$\boldsymbol{\mu}_k = \frac{\sum_{n:\gamma_{nk}=1} \boldsymbol{x}_n}{|\{n : \gamma_{nk} = 1\}|} = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

for each $k \in [K]$.

# The K-means algorithm

**Step 0** Initialize $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

# The K-means algorithm

**Step 0** Initialize $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

**Step 1** Fix the centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$, assign each point to the closest center:

$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

# The K-means algorithm

**Step 0** Initialize $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

**Step 1** Fix the centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$, assign each point to the closest center:

$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

**Step 2** Fix the assignment $\{\gamma_{nk}\}$, update the centers

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

# The K-means algorithm

**Step 0** Initialize $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

**Step 1** Fix the centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$, assign each point to the closest center:

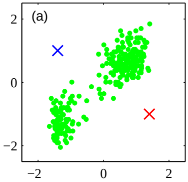
$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

**Step 2** Fix the assignment $\{\gamma_{nk}\}$, update the centers

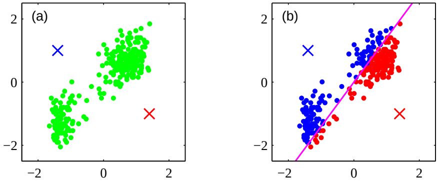
$$\boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk}\boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

**Step 3** Return to Step 1 if not converged

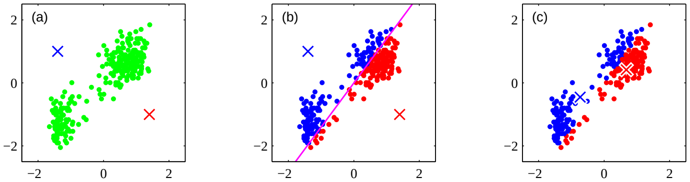
# An example



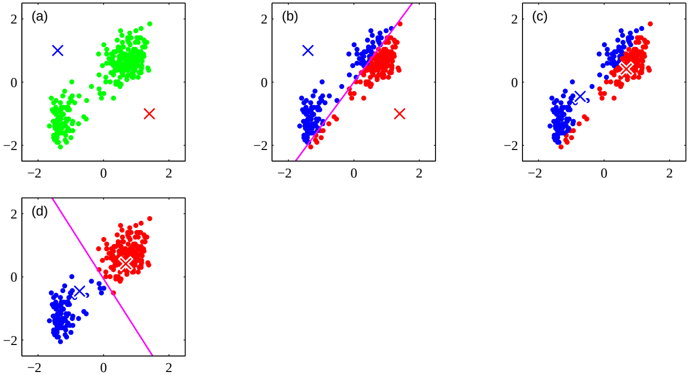
# An example



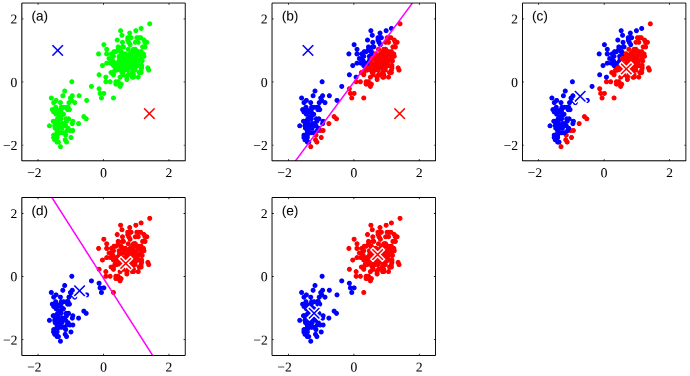
# An example



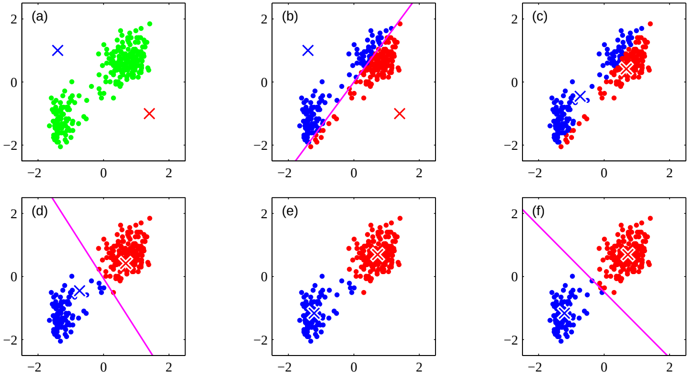
# An example



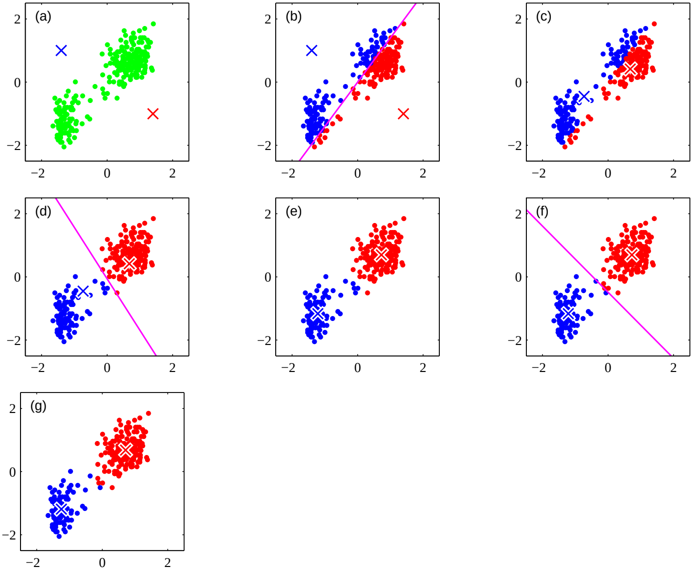
# An example



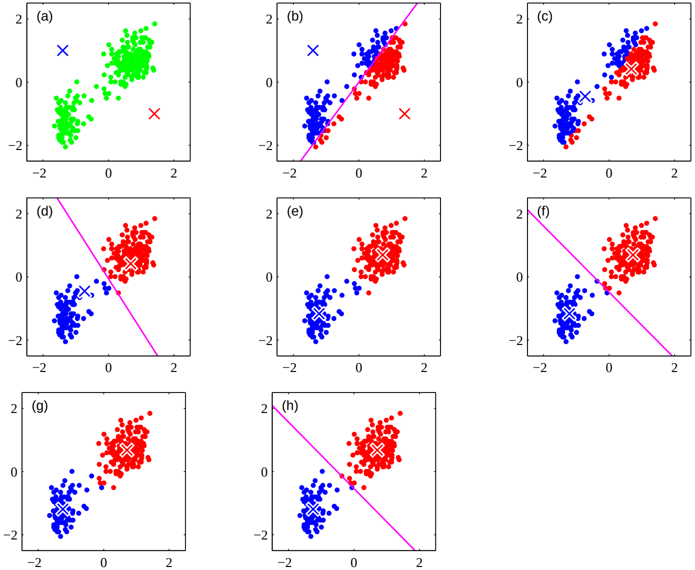
# An example



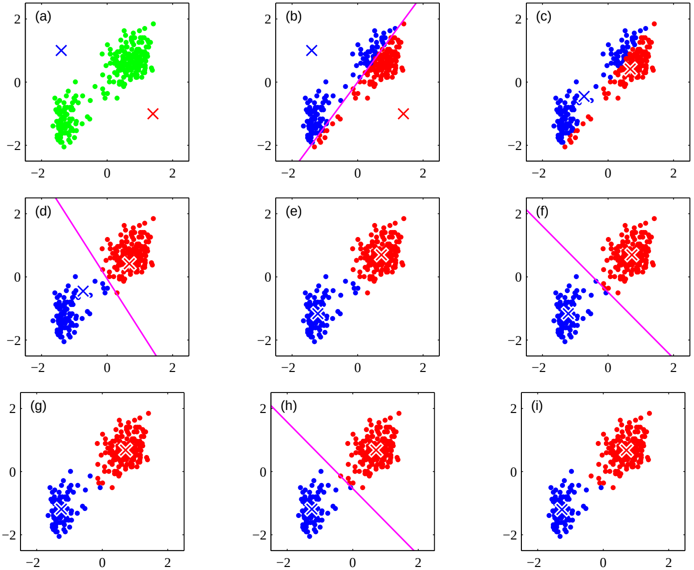
# An example



# An example



# An example



# How to initialize?

There are different ways to initialize:

# How to initialize?

There are different ways to initialize:

- randomly pick $K$ points as initial centers

# How to initialize?

There are different ways to initialize:

- randomly pick $K$ points as initial centers
- or randomly assign each point to a cluster, then average

# How to initialize?

There are different ways to initialize:

- randomly pick $K$ points as initial centers
- or randomly assign each point to a cluster, then average
- or more sophisticated approaches (e.g. K-means++)

# How to initialize?

There are different ways to initialize:

- randomly pick $K$ points as initial centers
- or randomly assign each point to a cluster, then average
- or more sophisticated approaches (e.g. K-means++)

Initialization matters for **convergence**.

# Convergence

K-means will converge in a finite number of iterations, why?

# Convergence

K-means will converge in a finite number of iterations, why?

- objective decreases at each step

# Convergence

K-means will converge in a finite number of iterations, why?

- objective decreases at each step
- objective is lower bounded by 0

# Convergence

K-means will converge in a finite number of iterations, why?

- objective decreases at each step
- objective is lower bounded by 0
- #possible_assignments is finite ($K^N$, exponentially large though)

# Convergence

K-means will converge in a finite number of iterations, why?

- objective decreases at each step
- objective is lower bounded by 0
- #possible_assignments is finite ($K^N$, exponentially large though)

However

- it could take *exponentially many iterations* to converge

# Convergence

K-means will converge in a finite number of iterations, why?

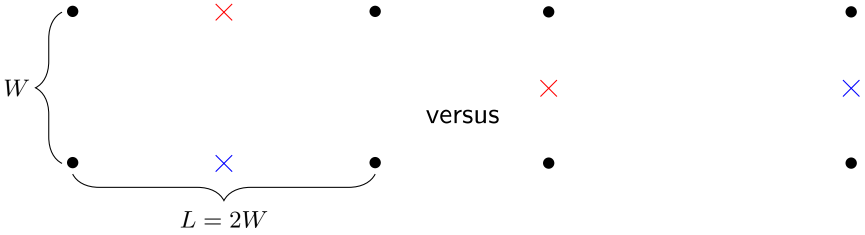
- objective decreases at each step
- objective is lower bounded by 0
- #possible_assignments is finite ($K^N$, exponentially large though)

However

- it could take *exponentially many iterations* to converge
- and it *might not converge to the global minimum* of the K-means objective

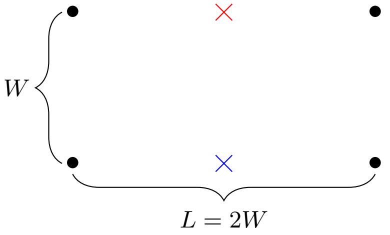
# Local minimum v.s global minimum

**Simple example**: 4 data points, 2 clusters, 2 different initializations



# Local minimum v.s global minimum

**Simple example**: 4 data points, 2 clusters, 2 different initializations

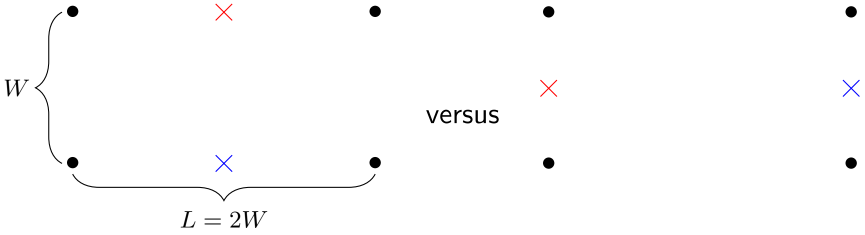



versus

K-means converges immediately in both cases,

# Local minimum v.s global minimum

**Simple example**: 4 data points, 2 clusters, 2 different initializations

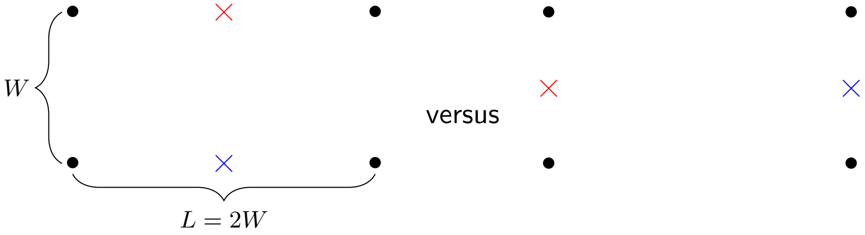



K-means converges immediately in both cases, but

- left has K-means objective $L^2 = 4W^2$

# Local minimum v.s global minimum

**Simple example**: 4 data points, 2 clusters, 2 different initializations

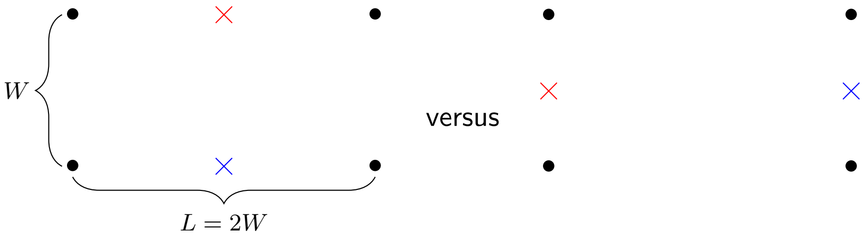



K-means converges immediately in both cases, but

- left has K-means objective $L^2 = 4W^2$
- right has K-means objective $W^2$, *4 times better than left!*

# Local minimum v.s global minimum

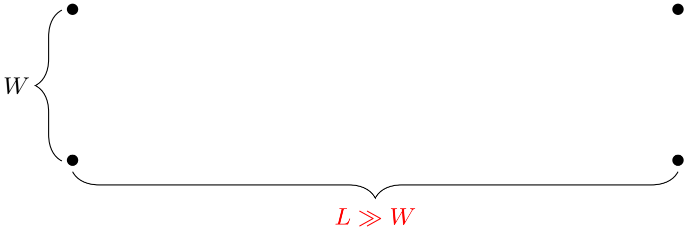
**Simple example**: 4 data points, 2 clusters, 2 different initializations



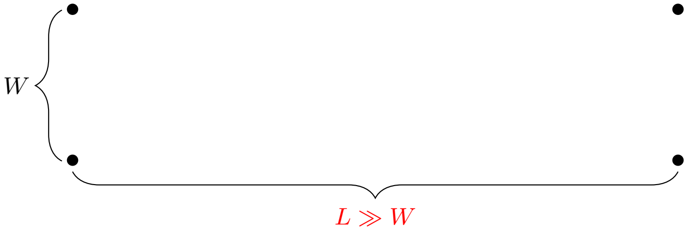

K-means converges immediately in both cases, but

- left has K-means objective $L^2 = 4W^2$
- right has K-means objective $W^2$, *4 times better than left!*
- in fact, left is **local minimum**, and right is **global minimum**.

# Local minimum v.s global minimum

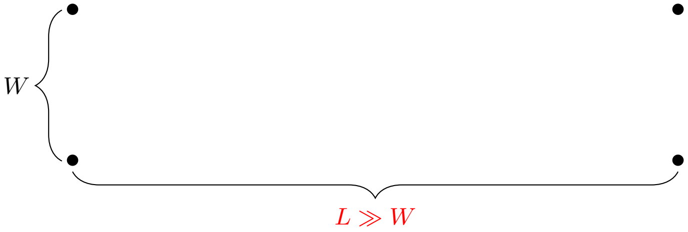


# Local minimum v.s global minimum



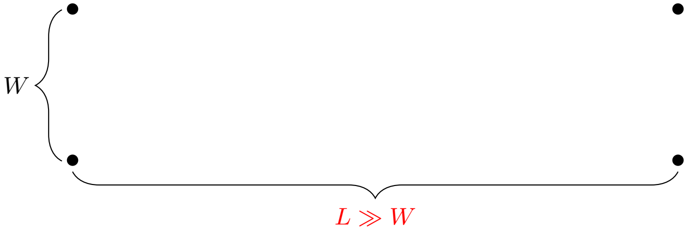

- moreover, local minimum can be *arbitrarily worse* if we increase $L$

# Local minimum v.s global minimum



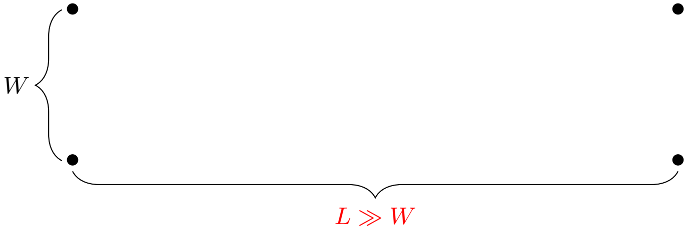

- moreover, local minimum can be *arbitrarily worse* if we increase $L$
- so *initialization matters a lot* for K-means

# How common initialization methods perform?



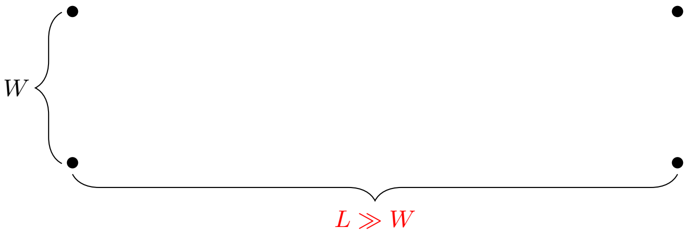

- randomly pick $K$ points as initial centers
- or randomly assign each point to a cluster, then average
- or more sophisticated approaches

# How common initialization methods perform?



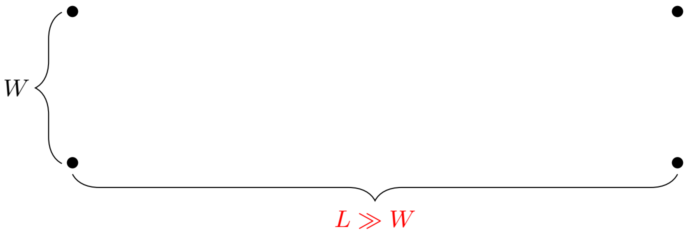

- randomly pick $K$ points as initial centers: fails with $1/3$ probability
- or randomly assign each point to a cluster, then average
- or more sophisticated approaches

# How common initialization methods perform?




- randomly pick $K$ points as initial centers: fails with $1/3$ probability
- or randomly assign each point to a cluster, then average: similarly fail with a constant probability
- or more sophisticated approaches

# How common initialization methods perform?




- randomly pick $K$ points as initial centers: fails with $1/3$ probability
- or randomly assign each point to a cluster, then average: similarly fail with a constant probability
- or more sophisticated approaches: **K-means++** *guarantees* to find a solution that in expectation is at most $O(\log K)$ times of the optimal

# K-means++

**K-means++** is K-means with a better initialization procedure:

# K-means++

**K-means++** is K-means with a better initialization procedure:

Start with a random data point as the first center $\boldsymbol{\mu}_1$

# K-means++

**K-means++** is K-means with a better initialization procedure:

Start with a random data point as the first center $\boldsymbol{\mu}_1$

For $k = 2, \ldots, K$

- randomly pick the $k$-th center $\boldsymbol{\mu}_k$ such that

$$\Pr[\boldsymbol{\mu}_k = \boldsymbol{x}_n] \propto \min_{j=1,\ldots,k-1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2$$

# K-means++

**K-means++** is K-means with a better initialization procedure:

Start with a random data point as the first center $\boldsymbol{\mu}_1$

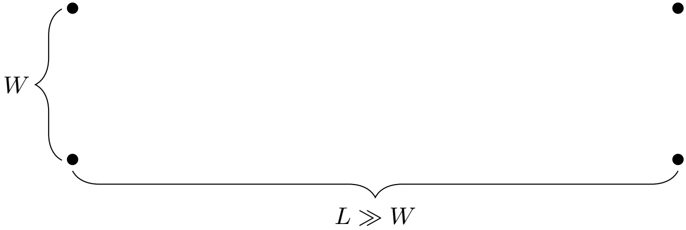
For $k = 2, \ldots, K$

- randomly pick the $k$-th center $\boldsymbol{\mu}_k$ such that

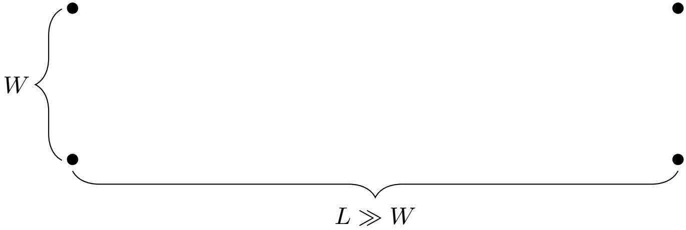
$$\Pr[\boldsymbol{\mu}_k = \boldsymbol{x}_n] \propto \min_{j=1,\ldots,k-1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2$$

Intuitively this *spreads out the initial centers*.

# K-means++ on the same example

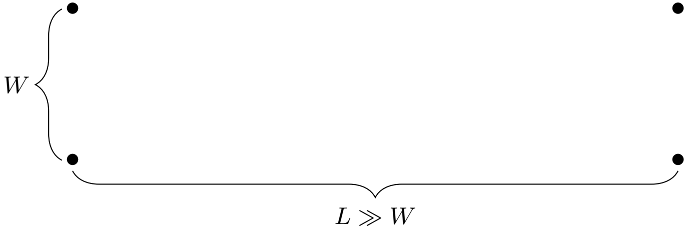


# K-means++ on the same example




Suppose we pick top left as $\boldsymbol{\mu}_1$, then

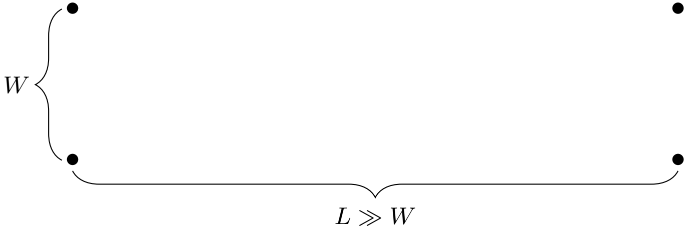
# K-means++ on the same example




Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$,

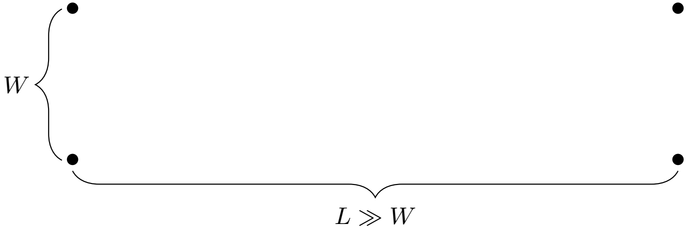
# K-means++ on the same example




Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\quad\Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$

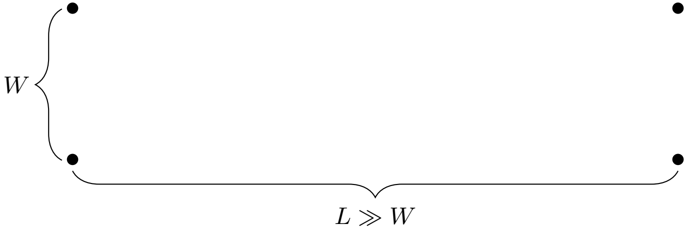
# K-means++ on the same example




Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$
- $\Pr[\boldsymbol{\mu}_2 = \text{bottom right}] \propto W^2 + L^2$

# K-means++ on the same example



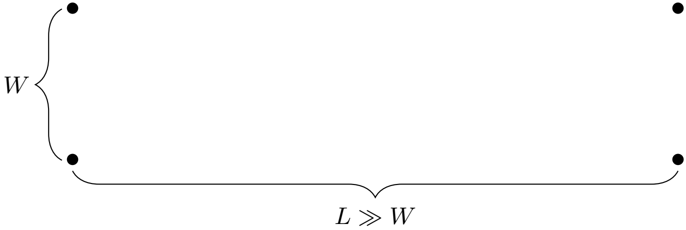

Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\qquad \Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$
- $\Pr[\boldsymbol{\mu}_2 = \text{bottom right}] \propto W^2 + L^2$

So the **expected K-means objective** is

$$\frac{W^2}{2(W^2+L^2)} \cdot L^2 + \left(\frac{L^2}{2(W^2+L^2)} + \frac{1}{2}\right) \cdot W^2$$

# K-means++ on the same example



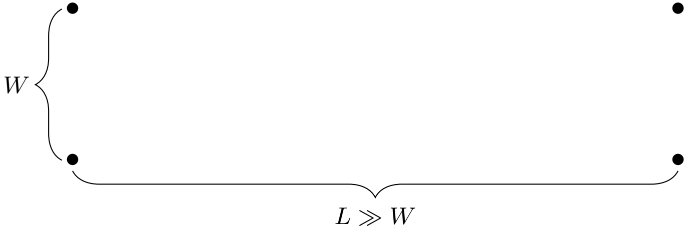

Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\quad \Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$
- $\Pr[\boldsymbol{\mu}_2 = \text{bottom right}] \propto W^2 + L^2$

So the **expected K-means objective** is

$$\frac{W^2}{2(W^2+L^2)} \cdot L^2 + \left(\frac{L^2}{2(W^2+L^2)} + \frac{1}{2}\right) \cdot W^2 \le \frac{3}{2}W^2,$$

# K-means++ on the same example




Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\qquad \Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$
- $\Pr[\boldsymbol{\mu}_2 = \text{bottom right}] \propto W^2 + L^2$

So the **expected K-means objective** is

$$\frac{W^2}{2(W^2+L^2)} \cdot L^2 + \left(\frac{L^2}{2(W^2+L^2)} + \frac{1}{2}\right) \cdot W^2 \le \frac{3}{2}W^2,$$

that is, *at most 1.5 times of the optimal*.

# Summary for K-means

K-means is alternating minimization for the K-means objective.

The initialization matters a lot for the convergence.

K-means++ uses a theoretically (and often empirically) better initialization.

# Outline

# Gaussian mixture models

Gaussian mixture models (GMM) is a probabilistic approach for clustering

# Gaussian mixture models

Gaussian mixture models (GMM) is a probabilistic approach for clustering

- more explanatory than minimizing the K-means objective
- can be seen as a soft version of K-means

# Gaussian mixture models

Gaussian mixture models (GMM) is a probabilistic approach for clustering

- more explanatory than minimizing the K-means objective
- can be seen as a soft version of K-means

To solve GMM, we will introduce a powerful method for learning probabilistic model: **Expectation–Maximization (EM) algorithm**

# A generative model

For classification, we discussed the sigmoid model to "explain" how the labels are generated.

# A generative model

For classification, we discussed the sigmoid model to "explain" how the labels are generated.

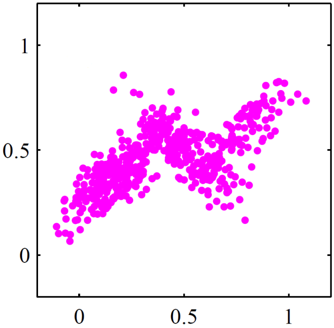
Similarly, for clustering, we want to come up with a probabilistic model $p$ to **"explain" how the data is generated**.

# A generative model

For classification, we discussed the sigmoid model to "explain" how the labels are generated.

Similarly, for clustering, we want to come up with a probabilistic model $p$ to **"explain" how the data is generated**.

That is, each point is an independent sample of $\boldsymbol{x} \sim p$.



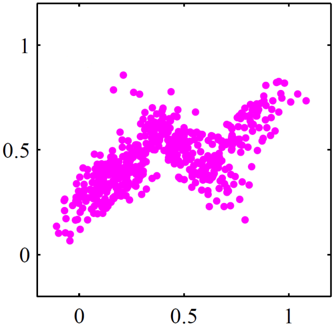
# A generative model

For classification, we discussed the sigmoid model to “explain” how the labels are generated.

Similarly, for clustering, we want to come up with a probabilistic model $p$ to **“explain” how the data is generated**.

That is, each point is an independent sample of $\boldsymbol{x} \sim p$.

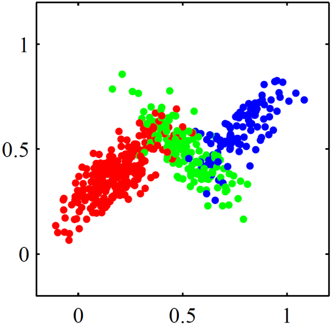
*What probabilistic model generates data like this?*



# GMM: intuition

GMM is a natural model to explain such data

Assume there are 3 ground-truth Gaussian models.

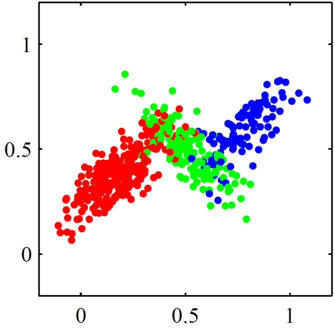


# GMM: intuition

GMM is a natural model to explain such data

Assume there are 3 ground-truth Gaussian models. To generate a point, we

- first **randomly pick one of the Gaussian models**,

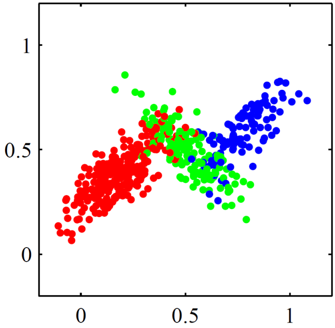


# GMM: intuition

GMM is a natural model to explain such data

Assume there are 3 ground-truth Gaussian models. To generate a point, we

- first **randomly pick one of the Gaussian models**,
- then **draw a point according this Gaussian**.

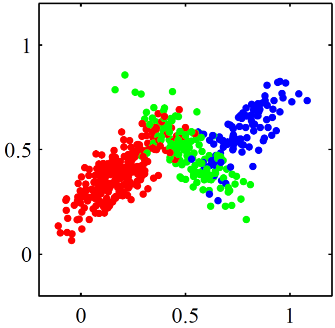


# GMM: intuition

GMM is a natural model to explain such data

Assume there are 3 ground-truth Gaussian models. To generate a point, we

- first **randomly pick one of the Gaussian models**,
- then **draw a point according this Gaussian**.




Hence the name "**Gaussian mixture model**".

# GMM: formal definition

A GMM has the following density function:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

# GMM: formal definition

A GMM has the following density function:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where

- $K$: the number of Gaussian components (same as #clusters we want)

# GMM: formal definition

A GMM has the following density function:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where

- $K$: the number of Gaussian components (same as #clusters we want)
- $\omega_1, \ldots, \omega_K$: mixture weights, a distribution over $K$ components

# GMM: formal definition

A GMM has the following density function:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where

- $K$: the number of Gaussian components (same as #clusters we want)
- $\omega_1, \ldots, \omega_K$: mixture weights, a distribution over $K$ components
- $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$: mean and covariance matrix of the $k$-th Gaussian

# GMM: formal definition

A GMM has the following density function:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where

- $K$: the number of Gaussian components (same as #clusters we want)
- $\omega_1, \ldots, \omega_K$: mixture weights, a distribution over $K$ components
- $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$: mean and covariance matrix of the $k$-th Gaussian
- $N$: the density function for a Gaussian

# Another view

By introducing a **latent variable** $z \in [K]$, which indicates cluster membership, we can see $p$ as a **marginal distribution**

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}, z = k)$$

# Another view

By introducing a **latent variable** $z \in [K]$, which indicates cluster membership, we can see $p$ as a **marginal distribution**

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}, z = k) = \sum_{k=1}^{K} p(z = k) p(\boldsymbol{x} | z = k)$$

# Another view

By introducing a **latent variable** $z \in [K]$, which indicates cluster membership, we can see $p$ as a **marginal distribution**

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}, z = k) = \sum_{k=1}^{K} p(z = k) p(\boldsymbol{x} | z = k) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

# Another view

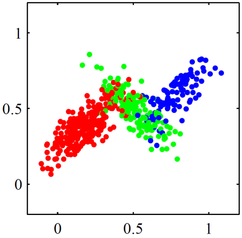
By introducing a **latent variable** $z \in [K]$, which indicates cluster membership, we can see $p$ as a **marginal distribution**

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}, z = k) = \sum_{k=1}^{K} p(z = k) p(\boldsymbol{x} | z = k) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$\boldsymbol{x}$ and $z$ are both random variables drawn from the model

- $\boldsymbol{x}$ is observed
- $z$ is unobserved/latent

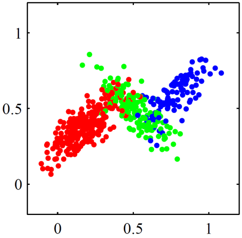
# An example




The conditional distributions are

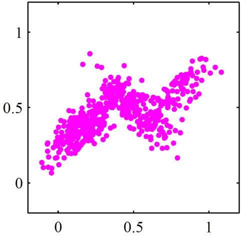
$$p(\boldsymbol{x} \mid z = \text{red}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$
$$p(\boldsymbol{x} \mid z = \text{blue}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$
$$p(\boldsymbol{x} \mid z = \text{green}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$

# An example




The conditional distributions are

$$p(\boldsymbol{x} \mid z = \text{red}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$
$$p(\boldsymbol{x} \mid z = \text{blue}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$
$$p(\boldsymbol{x} \mid z = \text{green}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$




The marginal distribution is

$$p(\boldsymbol{x}) = p(\text{red})N(\boldsymbol{x} \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + p(\text{blue})N(\boldsymbol{x} \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) + p(\text{green})N(\boldsymbol{x} \mid \boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

In the process, we will learn the latent variable $z_n$ as well:

$$p(z_n = k \mid \boldsymbol{x}_n)$$

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

In the process, we will learn the latent variable $z_n$ as well:

$$p(z_n = k \mid \boldsymbol{x}_n) \triangleq \gamma_{nk} \in [0, 1]$$

i.e. "soft assignment" of each point to each cluster, as opposed to "hard assignment" by K-means.

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.

In the process, we will learn the latent variable $z_n$ as well:

$$p(z_n = k \mid \boldsymbol{x}_n) \triangleq \gamma_{nk} \in [0, 1]$$

i.e. "soft assignment" of each point to each cluster, as opposed to "hard assignment" by K-means.

GMM is more explanatory than K-means

- both learn the cluster centers $\boldsymbol{\mu}_k$'s

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$.

In the process, we will learn the latent variable $z_n$ as well:

$$p(z_n = k \mid \boldsymbol{x}_n) \triangleq \gamma_{nk} \in [0, 1]$$

i.e. "soft assignment" of each point to each cluster, as opposed to "hard assignment" by K-means.

GMM is more explanatory than K-means

- both learn the cluster centers $\boldsymbol{\mu}_k$'s
- in addition, GMM learns cluster weight $\omega_k$ and covariance $\boldsymbol{\Sigma}_k$,

# Learning GMMs

Learning a GMM means finding all the parameters $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$.

In the process, we will learn the latent variable $z_n$ as well:

$$p(z_n = k \mid \boldsymbol{x}_n) \triangleq \gamma_{nk} \in [0, 1]$$

i.e. "soft assignment" of each point to each cluster, as opposed to "hard assignment" by K-means.

GMM is more explanatory than K-means

- both learn the cluster centers $\boldsymbol{\mu}_k$'s
- in addition, GMM learns cluster weight $\omega_k$ and covariance $\boldsymbol{\Sigma}_k$, thus
  - we can *predict probability of seeing a new point*
  - we can *generate synthetic data*

# How to learn these parameters?

An obvious attempt is **maximum-likelihood estimation (MLE)**: find

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \ \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) \triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} P(\boldsymbol{\theta})$$

# How to learn these parameters?

An obvious attempt is **maximum-likelihood estimation (MLE)**: find

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \ \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) \triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} P(\boldsymbol{\theta})$$

This is called incomplete log-likelihood (since $z_n$'s are unobserved), and is *intractable in general* (non-concave problem).

# How to learn these parameters?

An obvious attempt is **maximum-likelihood estimation (MLE)**: find

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \ \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) \triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} P(\boldsymbol{\theta})$$

This is called incomplete log-likelihood (since $z_n$'s are unobserved), and is *intractable in general* (non-concave problem).

One solution is to still apply GD/SGD, but a much more effective approach is the **Expectation–Maximization (EM) algorithm**.

# Preview of EM for learning GMMs

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

# Preview of EM for learning GMMs

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

# Preview of EM for learning GMMs

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

**Step 2 (M-Step)** **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

# Preview of EM for learning GMMs

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

**Step 2 (M-Step)** **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk}\boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

**Step 3** return to Step 1 if not converged

# Preview of EM for learning GMMs

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

**Step 2 (M-Step)** **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

**Step 3** return to Step 1 if not converged

We will see how this is a special case of EM.

# Demo

Generate 50 data points from a mixture of 2 Gaussians with

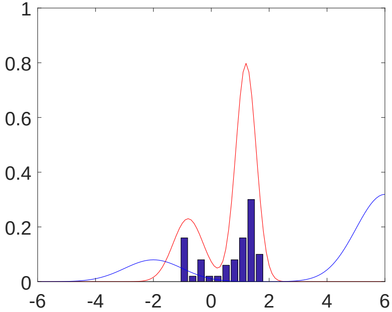
- $\omega_1 = 0.3, \mu_1 = -0.8, \Sigma_1 = 0.52$
- $\omega_2 = 0.7, \mu_2 = 1.2, \Sigma_2 = 0.35$

# Demo

Generate 50 data points from a mixture of 2 Gaussians with

- $\omega_1 = 0.3, \mu_1 = -0.8, \Sigma_1 = 0.52$
- $\omega_2 = 0.7, \mu_2 = 1.2, \Sigma_2 = 0.35$

**histogram represents the data**



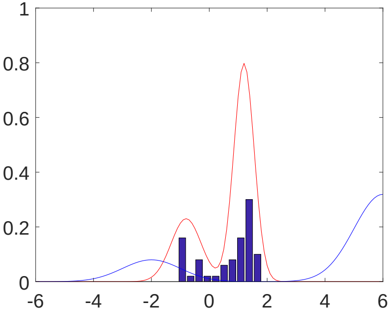
# Demo

Generate 50 data points from a mixture of 2 Gaussians with

- $\omega_1 = 0.3, \mu_1 = -0.8, \Sigma_1 = 0.52$
- $\omega_2 = 0.7, \mu_2 = 1.2, \Sigma_2 = 0.35$

**histogram represents the data**

red curve represents the ground-truth density $p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$



# Demo

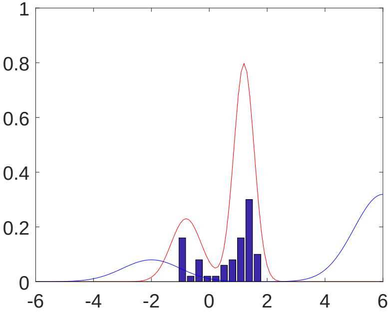
Generate 50 data points from a mixture of 2 Gaussians with

- $\omega_1 = 0.3, \mu_1 = -0.8, \Sigma_1 = 0.52$
- $\omega_2 = 0.7, \mu_2 = 1.2, \Sigma_2 = 0.35$

**histogram represents the data**

red curve represents the ground-truth density $p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

blue curve represents the learned density for a specific round



# Demo

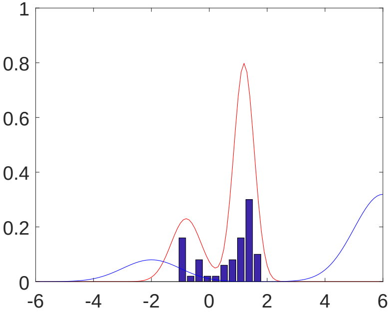
Generate 50 data points from a mixture of 2 Gaussians with

- $\omega_1 = 0.3, \mu_1 = -0.8, \Sigma_1 = 0.52$
- $\omega_2 = 0.7, \mu_2 = 1.2, \Sigma_2 = 0.35$

**histogram represents the data**

red curve represents the ground-truth density $p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

blue curve represents the learned density for a specific round




EM_demo.pdf shows how the blue curve moves towards red curve quickly via EM

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \, ; \boldsymbol{\theta})$$

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int_{z_n} p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}) dz_n$$

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int_{z_n} p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}) dz_n$$

- $\boldsymbol{\theta}$ is the parameters for a general probabilistic model

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int_{z_n} p(\boldsymbol{x}_n, z_n \ ; \boldsymbol{\theta}) dz_n$$

- $\boldsymbol{\theta}$ is the parameters for a general probabilistic model
- $\boldsymbol{x}_n$'s are observed random variables

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int_{z_n} p(\boldsymbol{x}_n, z_n \ ; \boldsymbol{\theta}) dz_n$$

- $\boldsymbol{\theta}$ is the parameters for a general probabilistic model
- $\boldsymbol{x}_n$'s are observed random variables
- $z_n$'s are latent variables

# EM algorithm

In general EM is **a heuristic to solve MLE with latent variables** (not just GMM), i.e. find the maximizer of

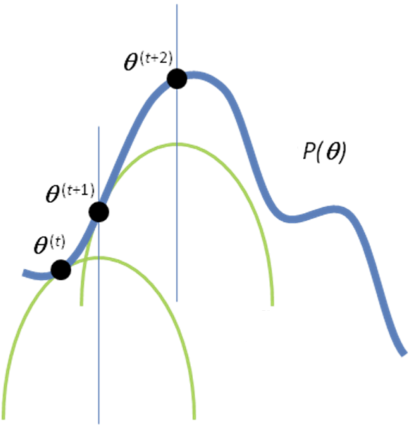
$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \ ; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int_{z_n} p(\boldsymbol{x}_n, z_n \ ; \boldsymbol{\theta}) dz_n$$

- $\boldsymbol{\theta}$ is the parameters for a general probabilistic model
- $\boldsymbol{x}_n$'s are observed random variables
- $z_n$'s are latent variables

Again, directly solving the objective is intractable.

# High level idea

Keep maximizing **a lower bound of $P$ that is more manageable**



# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\ ;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\ ;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\ ;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\ ;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\ ;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\ ;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln \frac{p(\boldsymbol{x}, z\ ;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\ ;\boldsymbol{\theta})}\right] \qquad \text{(true for any dist. } q\text{)}$$

# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\,;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}\right] \qquad \text{(true for any dist. } q\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] - \mathbb{E}_{z\sim q}\left[\ln q(z)\right] - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\,;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}\right] \qquad \text{(true for any dist. } q\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] - \mathbb{E}_{z\sim q}\left[\ln q(z)\right] - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q) - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right] \qquad (H \text{ is entropy})$$

# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\,;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}\right] \qquad \text{(true for any dist. } q\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] - \mathbb{E}_{z\sim q}\left[\ln q(z)\right] - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q) - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right] \qquad (H \text{ is entropy})$$

$$\geq \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q) - \ln \mathbb{E}_{z\sim q}\left[\frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

(Jensen's inequality)

# Derivation of EM

**Finding the lower bound of $P$:**

$$\ln p(\boldsymbol{x}\,;\boldsymbol{\theta}) = \ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})} \qquad \text{(true for any } z\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln \frac{p(\boldsymbol{x}, z\,;\boldsymbol{\theta})}{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}\right] \qquad \text{(true for any dist. } q\text{)}$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] - \mathbb{E}_{z\sim q}\left[\ln q(z)\right] - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q) - \mathbb{E}_{z\sim q}\left[\ln \frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right] \qquad (H \text{ is entropy})$$

$$\geq \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q) - \ln \mathbb{E}_{z\sim q}\left[\frac{p(z|\boldsymbol{x}\,;\boldsymbol{\theta})}{q(z)}\right]$$

(Jensen's inequality)

$$= \mathbb{E}_{z\sim q}\left[\ln p(\boldsymbol{x}, z\,;\boldsymbol{\theta})\right] + H(q)$$

# Alternatively maximize the lower bound

Therefore, we obtain a lower bound for the log-likelihood function

$$
\begin{aligned}
P(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) \\
&\geq \sum_{n=1}^{N} \left(\mathbb{E}_{z_n \sim q_n}\left[\ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta})\right] + H(q_n)\right) = F(\boldsymbol{\theta}, \{q_n\})
\end{aligned}
$$

# Alternatively maximize the lower bound

Therefore, we obtain a lower bound for the log-likelihood function

$$P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta})$$
$$\geq \sum_{n=1}^{N} \left( \mathbb{E}_{z_n \sim q_n} \left[ \ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}) \right] + H(q_n) \right) = F(\boldsymbol{\theta}, \{q_n\})$$

This holds for *any* $\{q_n\}$, so how do we choose?

# Alternatively maximize the lower bound

Therefore, we obtain a lower bound for the log-likelihood function

$$
\begin{aligned}
P(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \, ; \boldsymbol{\theta}) \\
&\geq \sum_{n=1}^{N} \left( \mathbb{E}_{z_n \sim q_n} \left[ \ln p(\boldsymbol{x}_n, z_n \, ; \boldsymbol{\theta}) \right] + H(q_n) \right) = F(\boldsymbol{\theta}, \{q_n\})
\end{aligned}
$$

This holds for *any* $\{q_n\}$, so how do we choose? Naturally, *the one that maximizes the lower bound* (i.e. the tightest lower bound)!

# Alternatively maximize the lower bound

Therefore, we obtain a lower bound for the log-likelihood function

$$\begin{aligned} P(\boldsymbol{\theta}) &= \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta}) \\ &\geq \sum_{n=1}^{N} \left( \mathbb{E}_{z_n \sim q_n} \left[ \ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}) \right] + H(q_n) \right) = F(\boldsymbol{\theta}, \{q_n\}) \end{aligned}$$

This holds for *any* $\{q_n\}$, so how do we choose? Naturally, *the one that maximizes the lower bound* (i.e. the tightest lower bound)!

Equivalently, this is the same as alternatingly maximizing $F$ over $\{q_n\}$ and $\boldsymbol{\theta}$ (similar to K-means).

# Maximizing over $\{q_n\}$

Fix $\boldsymbol{\theta}^{(t)}$, the solution to

$$\operatorname*{argmax}_{q_n} \mathbb{E}_{z_n \sim q_n}\left[\ln p(\boldsymbol{x}_n, z_n\ ; \boldsymbol{\theta}^{(t)})\right] + H(q_n)$$

is $q_n^{(t)}$ s.t.

$$q_n^{(t)}(z_n) = p(z_n \mid \boldsymbol{x}_n\ ; \boldsymbol{\theta}^{(t)})$$

i.e., the *posterior distribution of $z_n$* given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}^{(t)}$. (Verified in HW4)

# Maximizing over $\{q_n\}$

Fix $\boldsymbol{\theta}^{(t)}$, the solution to

$$\underset{q_n}{\operatorname{argmax}} \, \mathbb{E}_{z_n \sim q_n}\left[\ln p(\boldsymbol{x}_n, z_n\,; \boldsymbol{\theta}^{(t)})\right] + H(q_n)$$

is $q_n^{(t)}$ s.t.

$$q_n^{(t)}(z_n) = p(z_n \mid \boldsymbol{x}_n\,; \boldsymbol{\theta}^{(t)}) \propto p(\boldsymbol{x}_n, z_n\,; \boldsymbol{\theta}^{(t)})$$

i.e., the *posterior distribution of $z_n$* given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}^{(t)}$. (Verified in HW4)

# Maximizing over $\{q_n\}$

Fix $\boldsymbol{\theta}^{(t)}$, the solution to

$$\underset{q_n}{\operatorname{argmax}} \, \mathbb{E}_{z_n \sim q_n}\left[\ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}^{(t)})\right] + H(q_n)$$

is $q_n^{(t)}$ s.t.

$$q_n^{(t)}(z_n) = p(z_n \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)}) \propto \; p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}^{(t)})$$

i.e., the *posterior distribution of $z_n$* given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}^{(t)}$. (Verified in HW4)

So at $\boldsymbol{\theta}^{(t)}$, we found the tightest lower bound $F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$:

# Maximizing over $\{q_n\}$

Fix $\boldsymbol{\theta}^{(t)}$, the solution to

$$\underset{q_n}{\operatorname{argmax}}\, \mathbb{E}_{z_n \sim q_n}\left[\ln p(\boldsymbol{x}_n, z_n\,; \boldsymbol{\theta}^{(t)})\right] + H(q_n)$$

is $q_n^{(t)}$ s.t.

$$q_n^{(t)}(z_n) = p(z_n \mid \boldsymbol{x}_n\,; \boldsymbol{\theta}^{(t)}) \propto p(\boldsymbol{x}_n, z_n\,; \boldsymbol{\theta}^{(t)})$$

i.e., the *posterior distribution of $z_n$* given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}^{(t)}$. (Verified in HW4)

So at $\boldsymbol{\theta}^{(t)}$, we found the tightest lower bound $F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$:

- $F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right) \leq P(\boldsymbol{\theta})$ for all $\boldsymbol{\theta}$.

# Maximizing over $\{q_n\}$

Fix $\boldsymbol{\theta}^{(t)}$, the solution to

$$\underset{q_n}{\operatorname{argmax}} \, \mathbb{E}_{z_n \sim q_n} \left[ \ln p(\boldsymbol{x}_n, z_n \, ; \boldsymbol{\theta}^{(t)}) \right] + H(q_n)$$

is $q_n^{(t)}$ s.t.

$$q_n^{(t)}(z_n) = p(z_n \mid \boldsymbol{x}_n \, ; \boldsymbol{\theta}^{(t)}) \propto \; p(\boldsymbol{x}_n, z_n \, ; \boldsymbol{\theta}^{(t)})$$

i.e., the *posterior distribution of $z_n$* given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}^{(t)}$. (Verified in HW4)

So at $\boldsymbol{\theta}^{(t)}$, we found the tightest lower bound $F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$:

- $F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right) \leq P(\boldsymbol{\theta})$ for all $\boldsymbol{\theta}$.
- $F\left(\boldsymbol{\theta}^{(t)}, \{q_n^{(t)}\}\right) = P(\boldsymbol{\theta}^{(t)})$ (verify yourself by going through Slide 40)

# Maximizing over $\boldsymbol{\theta}$

Fix $\{q_n^{(t)}\}$, maximize over $\boldsymbol{\theta}$:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$$

# Maximizing over $\boldsymbol{\theta}$

Fix $\{q_n^{(t)}\}$, maximize over $\boldsymbol{\theta}$:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \; ; \boldsymbol{\theta})\right] \quad (H(q_n^{(t)}) \text{ is independent of } \boldsymbol{\theta})$$

# Maximizing over $\boldsymbol{\theta}$

Fix $\{q_n^{(t)}\}$, maximize over $\boldsymbol{\theta}$:

$$
\begin{aligned}
&\operatorname*{argmax}_{\boldsymbol{\theta}} F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right) \\
&= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n ; \boldsymbol{\theta})\right] \qquad (H(q_n^{(t)}) \text{ is independent of } \boldsymbol{\theta}) \\
&\triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta} ; \boldsymbol{\theta}^{(t)}) \qquad (\{q_n^{(t)}\} \text{ are computed via } \boldsymbol{\theta}^{(t)})
\end{aligned}
$$

# Maximizing over $\boldsymbol{\theta}$

Fix $\{q_n^{(t)}\}$, maximize over $\boldsymbol{\theta}$:

$$\begin{aligned}
&\operatorname*{argmax}_{\boldsymbol{\theta}} F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right) \\
&= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta})\right] \qquad (H(q_n^{(t)}) \text{ is independent of } \boldsymbol{\theta}) \\
&\triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta} \,; \boldsymbol{\theta}^{(t)}) \qquad (\{q_n^{(t)}\} \text{ are computed via } \boldsymbol{\theta}^{(t)})
\end{aligned}$$

$Q$ is the (expected) **complete likelihood** and is usually more tractable.

# Maximizing over $\boldsymbol{\theta}$

Fix $\{q_n^{(t)}\}$, maximize over $\boldsymbol{\theta}$:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} F\left(\boldsymbol{\theta}, \{q_n^{(t)}\}\right)$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta})\right] \quad (H(q_n^{(t)}) \text{ is independent of } \boldsymbol{\theta})$$

$$\triangleq \operatorname*{argmax}_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta} \,; \boldsymbol{\theta}^{(t)}) \qquad (\{q_n^{(t)}\} \text{ are computed via } \boldsymbol{\theta}^{(t)})$$

$Q$ is the (expected) **complete likelihood** and is usually more tractable.

- versus the incomplete likelihood: $P(\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n \,; \boldsymbol{\theta})$

# General EM algorithm

**Step 0** Initialize $\boldsymbol{\theta}^{(1)}$, $t = 1$

# General EM algorithm

**Step 0** Initialize $\boldsymbol{\theta}^{(1)}$, $t = 1$

**Step 1 (E-Step)** **update the posterior of latent variables**

$$q_n^{(t)}(\cdot) = p(\cdot \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)})$$

# General EM algorithm

**Step 0** Initialize $\boldsymbol{\theta}^{(1)}$, $t = 1$

**Step 1 (E-Step)** **update the posterior of latent variables**

$$q_n^{(t)}(\cdot) = p(\cdot \mid \boldsymbol{x}_n ; \boldsymbol{\theta}^{(t)})$$

and obtain **Expectation** of complete likelihood

$$Q(\boldsymbol{\theta} ; \boldsymbol{\theta}^{(t)}) = \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n ; \boldsymbol{\theta})\right]$$

# General EM algorithm

**Step 0** Initialize $\boldsymbol{\theta}^{(1)}$, $t = 1$

**Step 1 (E-Step)** **update the posterior of latent variables**

$$q_n^{(t)}(\cdot) = p(\cdot \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)})$$

and obtain **Expectation** of complete likelihood

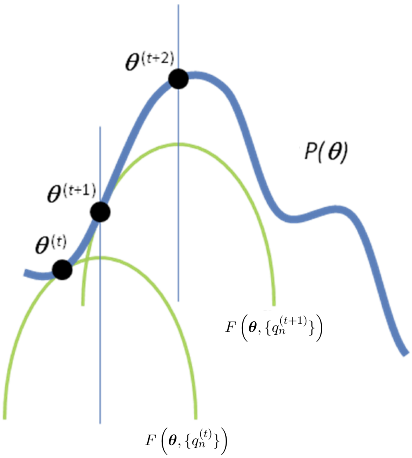
$$Q(\boldsymbol{\theta} \,; \boldsymbol{\theta}^{(t)}) = \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[ \ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta}) \right]$$

**Step 2 (M-Step)** **update the model parameter** via **Maximization**

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \,; \boldsymbol{\theta}^{(t)})$$

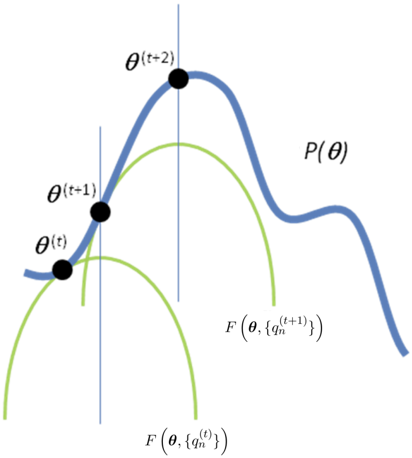
**Step 3** $t \leftarrow t + 1$ and return to Step 1 if not converged

# Pictorial explanation




$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

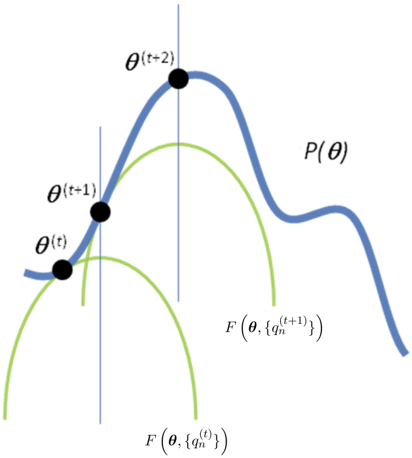
# Pictorial explanation




$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

$$P(\boldsymbol{\theta}^{(t+1)}) \geq F\left(\boldsymbol{\theta}^{(t+1)}\,; \{q_n^{(t)}\}\right)$$

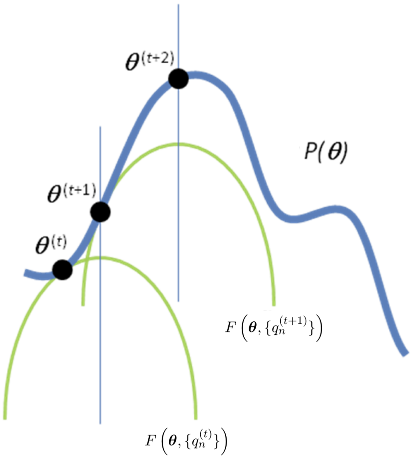
# Pictorial explanation




$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

$$
\begin{aligned}
P(\boldsymbol{\theta}^{(t+1)}) &\geq F\left(\boldsymbol{\theta}^{(t+1)}; \{q_n^{(t)}\}\right) \\
&\geq F\left(\boldsymbol{\theta}^{(t)}; \{q_n^{(t)}\}\right)
\end{aligned}
$$

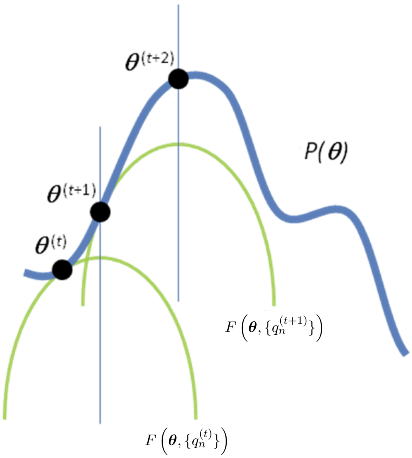
# Pictorial explanation




$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

$$
\begin{aligned}
P(\boldsymbol{\theta}^{(t+1)}) &\geq F\left(\boldsymbol{\theta}^{(t+1)}; \{q_n^{(t)}\}\right) \\
&\geq F\left(\boldsymbol{\theta}^{(t)}; \{q_n^{(t)}\}\right) \\
&= P(\boldsymbol{\theta}^{(t)})
\end{aligned}
$$

# Pictorial explanation




$P(\boldsymbol{\theta})$ is non-concave, but $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(t)})$ often is concave and easy to maximize.

$$
\begin{aligned}
P(\boldsymbol{\theta}^{(t+1)}) &\geq F\left(\boldsymbol{\theta}^{(t+1)}; \{q_n^{(t)}\}\right) \\
&\geq F\left(\boldsymbol{\theta}^{(t)}; \{q_n^{(t)}\}\right) \\
&= P(\boldsymbol{\theta}^{(t)})
\end{aligned}
$$

So EM always increases the objective value and will converge to some local maximum (similar to K-means).

# Apply EM to learn GMMs

**E-Step**:

$$
\begin{aligned}
q_n^{(t)}(z_n = k) &= p\left(z_n = k \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)}\right) \\
&\propto p\left(\boldsymbol{x}_n, z_n = k \,; \boldsymbol{\theta}^{(t)}\right)
\end{aligned}
$$

# Apply EM to learn GMMs

**E-Step**:

$$
\begin{aligned}
q_n^{(t)}(z_n = k) &= p\left(z_n = k \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)}\right) \\
&\propto p\left(\boldsymbol{x}_n, z_n = k \,; \boldsymbol{\theta}^{(t)}\right) \\
&= p\left(z_n = k \,; \boldsymbol{\theta}^{(t)}\right) p(\boldsymbol{x}_n \mid z_n = k \,; \boldsymbol{\theta}^{(t)}\right)
\end{aligned}
$$

# Apply EM to learn GMMs

**E-Step**:

$$
\begin{aligned}
q_n^{(t)}(z_n = k) &= p\left(z_n = k \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)}\right) \\
&\propto p\left(\boldsymbol{x}_n, z_n = k \,; \boldsymbol{\theta}^{(t)}\right) \\
&= p\left(z_n = k \,; \boldsymbol{\theta}^{(t)}\right) p(\boldsymbol{x}_n \mid z_n = k \,; \boldsymbol{\theta}^{(t)}\Big) \\
&= \omega_k^{(t)} N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}\right)
\end{aligned}
$$

# Apply EM to learn GMMs

**E-Step**:

$$
\begin{aligned}
q_n^{(t)}(z_n = k) &= p\left(z_n = k \mid \boldsymbol{x}_n \,; \boldsymbol{\theta}^{(t)}\right) \\
&\propto p\left(\boldsymbol{x}_n, z_n = k \,; \boldsymbol{\theta}^{(t)}\right) \\
&= p\left(z_n = k \,; \boldsymbol{\theta}^{(t)}\right) p(\boldsymbol{x}_n \mid z_n = k \,; \boldsymbol{\theta}^{(t)}\Big) \\
&= \omega_k^{(t)} N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}\right)
\end{aligned}
$$

This computes the "soft assignment" $\gamma_{nk} = q_n^{(t)}(z_n = k)$, i.e. conditional probability of $\boldsymbol{x}_n$ belonging to cluster $k$.

# Apply EM to learn GMMs

**M-Step**:

$$\operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} [\ln p(\boldsymbol{x}_n, z_n \; ; \boldsymbol{\theta})]$$

# Apply EM to learn GMMs

**M-Step**:

$$\begin{aligned}\operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \;; \boldsymbol{\theta})\right] \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(z_n \;; \boldsymbol{\theta}) + \ln p(\boldsymbol{x}_n | z_n \;; \boldsymbol{\theta})\right]\end{aligned}$$

# Apply EM to learn GMMs

**M-Step**:

$$\begin{aligned}
\operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \,; \boldsymbol{\theta})\right] \\
&= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(z_n \,; \boldsymbol{\theta}) + \ln p(\boldsymbol{x}_n | z_n \,; \boldsymbol{\theta})\right] \\
&= \operatorname*{argmax}_{\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left(\ln \omega_k + \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)
\end{aligned}$$

# Apply EM to learn GMMs

**M-Step**:

$$\begin{aligned}
\underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(\boldsymbol{x}_n, z_n \, ; \boldsymbol{\theta})\right] \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[\ln p(z_n \, ; \boldsymbol{\theta}) + \ln p(\boldsymbol{x}_n | z_n \, ; \boldsymbol{\theta})\right] \\
&= \underset{\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}}{\operatorname{argmax}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left(\ln \omega_k + \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)
\end{aligned}$$

To find $\omega_1, \ldots, \omega_K$, solve

$$\underset{\boldsymbol{\omega}}{\operatorname{argmax}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \ln \omega_k$$

# Apply EM to learn GMMs

**M-Step**:

$$
\begin{aligned}
\operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[ \ln p(\boldsymbol{x}_n, z_n \, ; \boldsymbol{\theta}) \right] \\
&= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{n=1}^{N} \mathbb{E}_{z_n \sim q_n^{(t)}} \left[ \ln p(z_n \, ; \boldsymbol{\theta}) + \ln p(\boldsymbol{x}_n | z_n \, ; \boldsymbol{\theta}) \right] \\
&= \operatorname*{argmax}_{\{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \left( \ln \omega_k + \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)
\end{aligned}
$$

To find $\omega_1, \ldots, \omega_K$, solve

$$
\operatorname*{argmax}_{\boldsymbol{\omega}} \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \ln \omega_k
$$

To find each $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, solve

$$
\operatorname*{argmax}_{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k} \sum_{n=1}^{N} \gamma_{nk} \ln N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)
$$

# M-Step (continued)

Solutions to previous two problems are very natural, for each $k$

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N}$$

i.e. (weighted) fraction of examples belonging to cluster $k$

# M-Step (continued)

Solutions to previous two problems are very natural, for each $k$

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N}$$

i.e. (weighted) fraction of examples belonging to cluster $k$

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

i.e. (weighted) average of examples belonging to cluster $k$

# M-Step (continued)

Solutions to previous two problems are very natural, for each $k$

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N}$$

i.e. (weighted) fraction of examples belonging to cluster $k$

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

i.e. (weighted) average of examples belonging to cluster $k$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

i.e (weighted) covariance of examples belonging to cluster $k$

# M-Step (continued)

Solutions to previous two problems are very natural, for each $k$

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N}$$

i.e. (weighted) fraction of examples belonging to cluster $k$

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

i.e. (weighted) average of examples belonging to cluster $k$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

i.e (weighted) covariance of examples belonging to cluster $k$

You will verify some of these in HW4.

# Putting it together

EM for learning GMMs:

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

# Putting it together

EM for learning GMMs:

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step) update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

# Putting it together

EM for learning GMMs:

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

**Step 2 (M-Step)** **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

# Putting it together

EM for learning GMMs:

**Step 0** Initialize $\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ for each $k \in [K]$

**Step 1 (E-Step)** **update the "soft assignment"** (fixing parameters)

$$\gamma_{nk} = p(z_n = k \mid \boldsymbol{x}_n) \propto \omega_k N\left(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

**Step 2 (M-Step)** **update the model parameter** (fixing assignments)

$$\omega_k = \frac{\sum_n \gamma_{nk}}{N} \qquad \boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk}\boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

**Step 3** return to Step 1 if not converged

# Connection to K-means

K-means is in fact a special case of EM for (a simplified) GMM:

# Connection to K-means

K-means is in fact a special case of EM for (a simplified) GMM:

- assume $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}$ for some fixed $\sigma$ so only $\omega_k$ and $\boldsymbol{\mu}_k$ are parameters

# Connection to K-means

K-means is in fact a special case of EM for (a simplified) GMM:

- assume $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}$ for some fixed $\sigma$ so only $\omega_k$ and $\boldsymbol{\mu}_k$ are parameters
- when $\sigma \rightarrow 0$, EM becomes K-means

# Connection to K-means

K-means is in fact a special case of EM for (a simplified) GMM:

- assume $\boldsymbol{\Sigma}_k = \sigma^2 \boldsymbol{I}$ for some fixed $\sigma$ so only $\omega_k$ and $\boldsymbol{\mu}_k$ are parameters
- when $\sigma \rightarrow 0$, EM becomes K-means

GMM is a soft version of K-means and it provides a probabilistic interpretation of the data, which means we can predict and generate data after learning.