MIXTURE MODELS

Gaussian Mixture Model, Expectation Maximization, K-means

Lê Hồng Phương

<*phuonglh@gmail.com*>
Vietnam National University of Hanoi
Hanoi University of Science

April 2015

Content

Latent Variable Models

- In many machine learning models, observed variables are correlated because they arise from a hidden common "cause".
- Models with hidden variables are also known as **latent variable models** or **LVMs**.
- Such models are more difficult to fit than models with no latent variables.

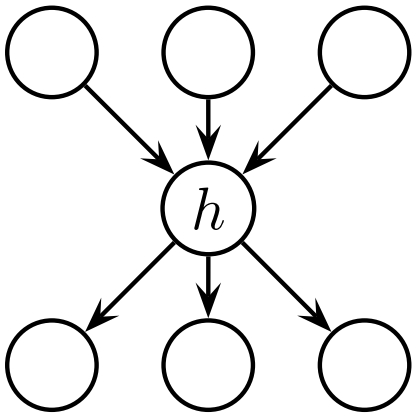
Latent Variable Models

However, LVMs have two main advantages:

- They have fewer parameters than models that directly represent correlation in the visible space.
- The hidden variables in an LVM can compute a compressed representation of the data. This forms the basis of unsupervised learning.

LVMs – Fewer Parameters

A directed graphical model (DGM) with hidden variables:

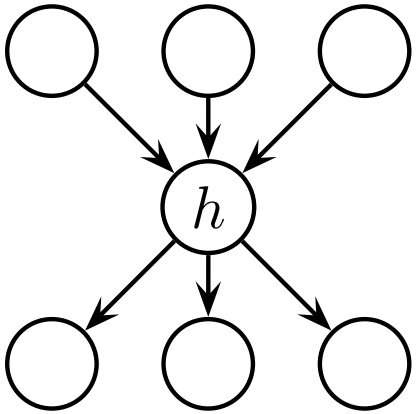


If all nodes are binary and all conditional probability distribution (CPDs) are tabular, the model has 17 free parameters.

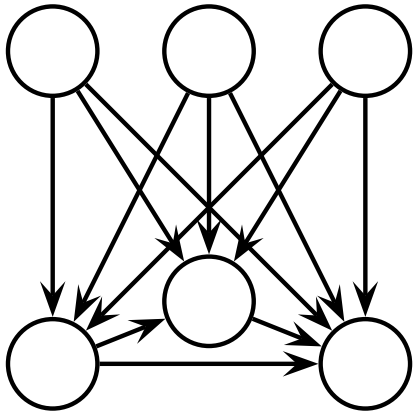
- The leaves represent medical symptoms.
- The roots represent primary causes: smoking, diet, exercises.
- The hidden variable can represent mediating factors, such as heart disease, which might not be directly visible.

LVMs – Fewer Parameters

A model without hidden variables has 59 parameters.



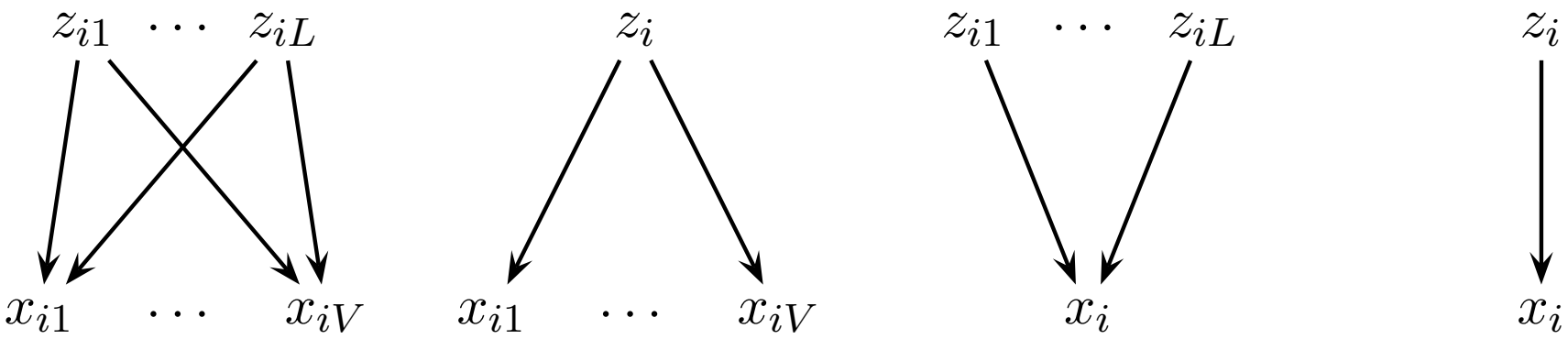
17 parameters



59 parameters

LVMs – Compact Representation

Some generic LVM structures:



- L latent variables $z_{i1}, \ldots, z_{iL}$ and V visible variables $x_{i1}, \ldots, x_{iV}$, usually $V \gg L$.
- By allowing $\mathbf{z}_i$ and $\mathbf{x}_i$ to be vector-valued, the one-to-one mapping representation can subsume all the others.

Latent Variable Models

Building a LVM:

- We imagine what types of hidden quantites might be used to describe the data.
- We encode that relationship in a *joint* probability distribution of hidden and observed random variables $p(\mathbf{z}, \mathbf{x})$.
- Then, given an observed data set, we uncover the particular hidden quantities that describe it through the posterior $p(\mathbf{z} \,|\, \mathbf{x})$.
- Furthermore, we use the posterior to form the *predictive distribution* $p(\mathbf{x}_{\text{new}} \,|\, \mathbf{z})$, the distribution over future data that the observations and the model imply.

Popular LVMs

Depending on the form of the likelihood $p(\mathbf{x}_i \,|\, \mathbf{z}_i)$ and the prior $p(\mathbf{z}_i)$, we can generate a variety of different LVMs:

$p(\mathbf{x}_i \,\|\, \mathbf{z}_i)$	$p(\mathbf{z}_i)$	**Name**
Multivariate Normal	Discrete	Mixture of Gaussians
Prod. Discretes	Discrete	Mixture of multinomials
Prod. Gaussians	Prod. Gaussians	Factor Analysis / Probabilistic PCA
Prod. Discrete	Prod. Gaussians	Multinomial PCA
Prod. Gaussians	Prod. Laplace	Probabilistic ICA / Sparse Coding
Prod. Discretes	Dirichlet	Latent Dirichlet Allocation

Mixture Models

The simplest form of LVM:

- $z_i \in \{1, 2, \ldots, K\}$, representing a discrete latent state.
- Discrete prior: $p(z_i) \sim \text{Cat}(\pi) \equiv \text{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$.
- The likelihood: $p(\mathbf{x}_i \,|z_i = k) = p_k(\mathbf{x}_i)$, where $p_k(\mathbf{x})$ is the k'th **base distribution** for the observations.

The mixture model:

$$p(\mathbf{x}_i \,|\theta) = \sum_{k=1}^{K} \pi_k p_k(\mathbf{x}_i \,|\theta).$$

That is, we mix together the K base distributions.

Mixture Models

$$p(\mathbf{x}_i \,|\theta) = \sum_{k=1}^{K} \pi_k p_k(\mathbf{x}_i \,|\theta)$$

This is a **convex combination** of the p_k's since we are taking the weighted sum, where the mixing weights π_k satisfy

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

Mixture Models

Intuition of mixture models:

- We assume that data are clustered and that each data point is drawn from a distribution associated with its assigned cluster.
- The hidden variables of the model are the cluster assignments and parameters to the per-cluster distributions.
- Given the observed data, the mixture model posterior is a conditional distribution over clusterings and parameters.
- This posterior distribution identifies a likely grouping of the data and the characteristics of each group.

Mixture of Gaussians

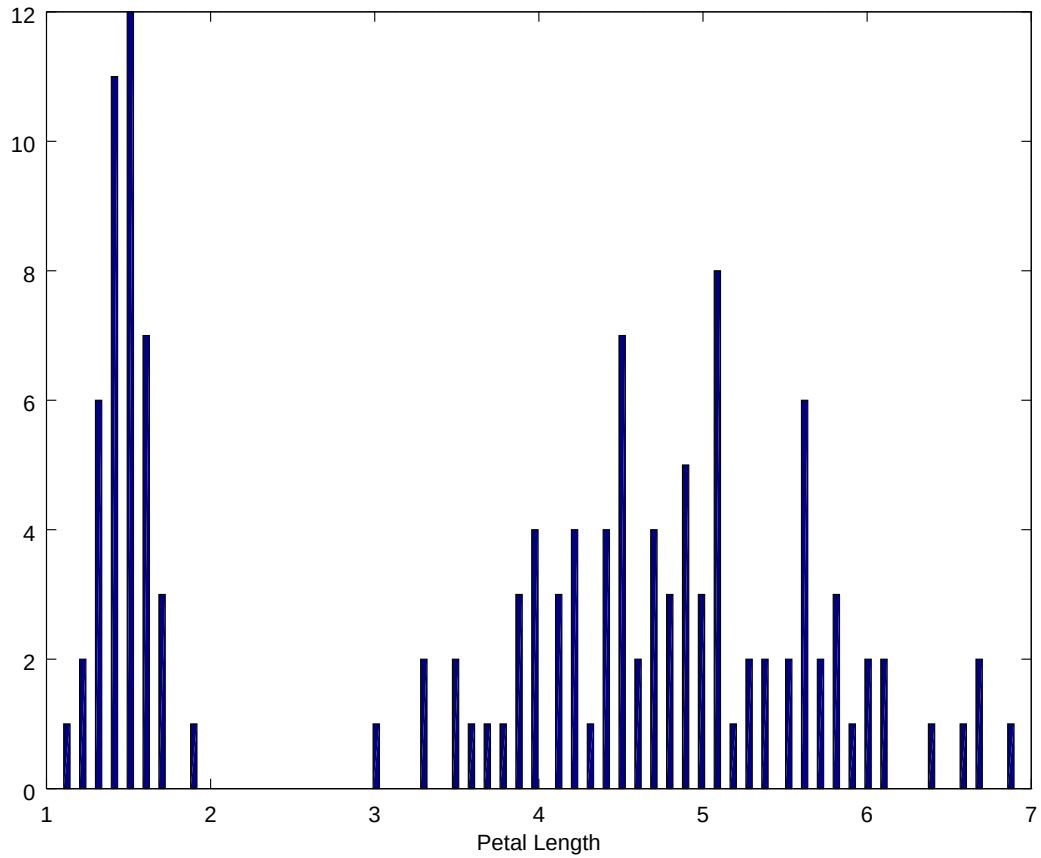
- The most widely used mixture model is the **mixture of Gaussians** (MOG), also called **Gaussian Mixture Model** (GMM).
- Each base distribution is a multivariate Gaussian with mean μ_k and covariance matrix Σ_k.
- The model has the form:

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

where $\theta = (\pi, \mu, \Sigma)$ are hidden quantities.

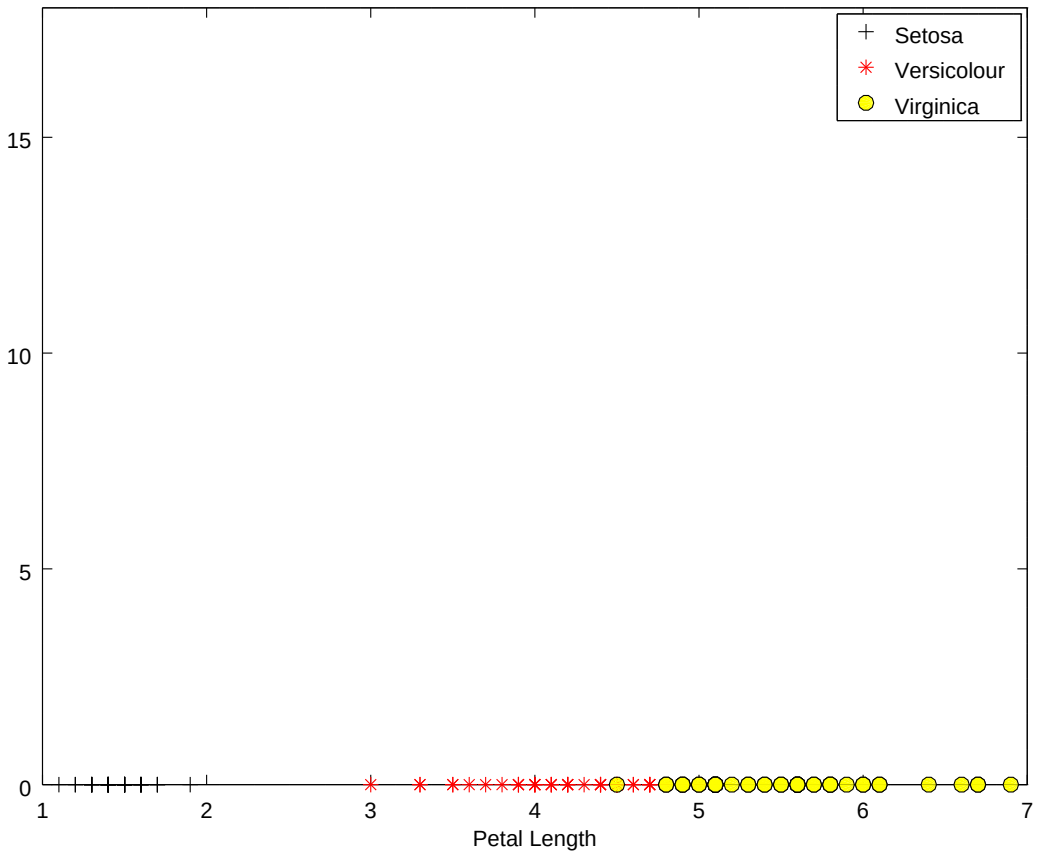
Mixture of Gaussians

Petal length of Iris flowers:



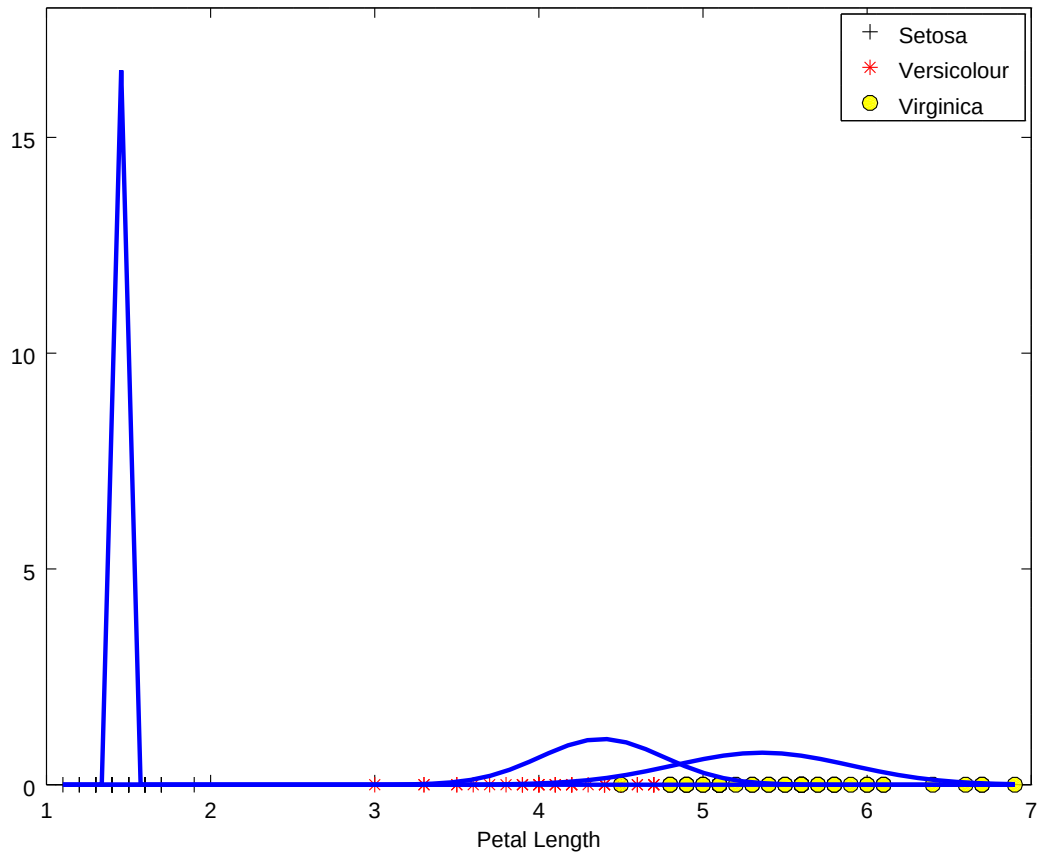
Mixture of Gaussians

Petal length of Iris flowers:



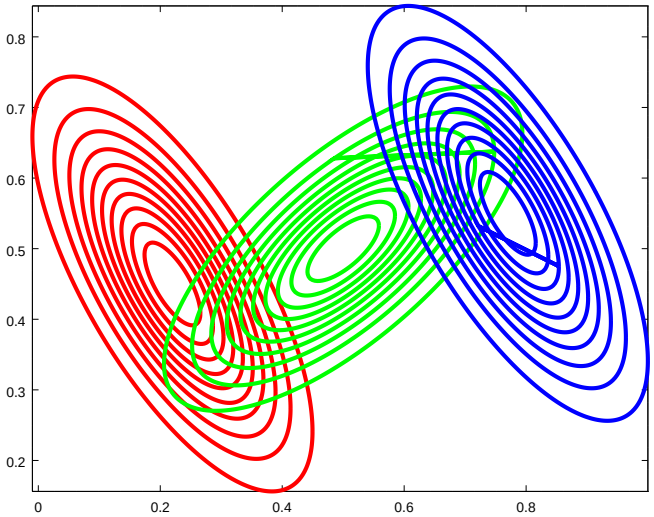
Mixture of Gaussians

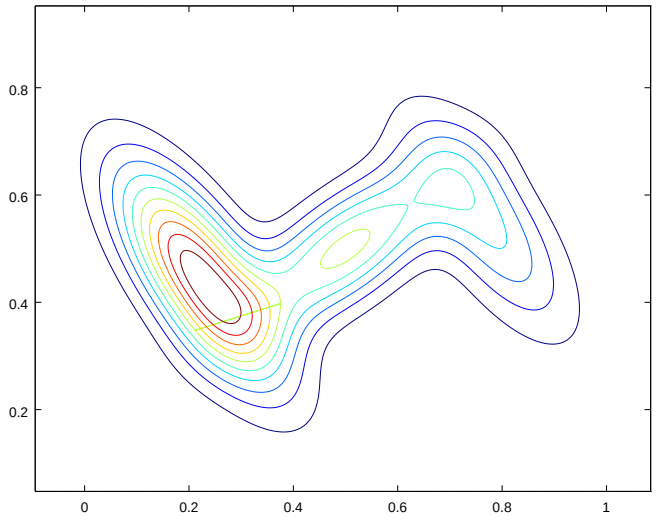
A mixture of 3 Gaussians in 1D:



Mixture of Gaussians

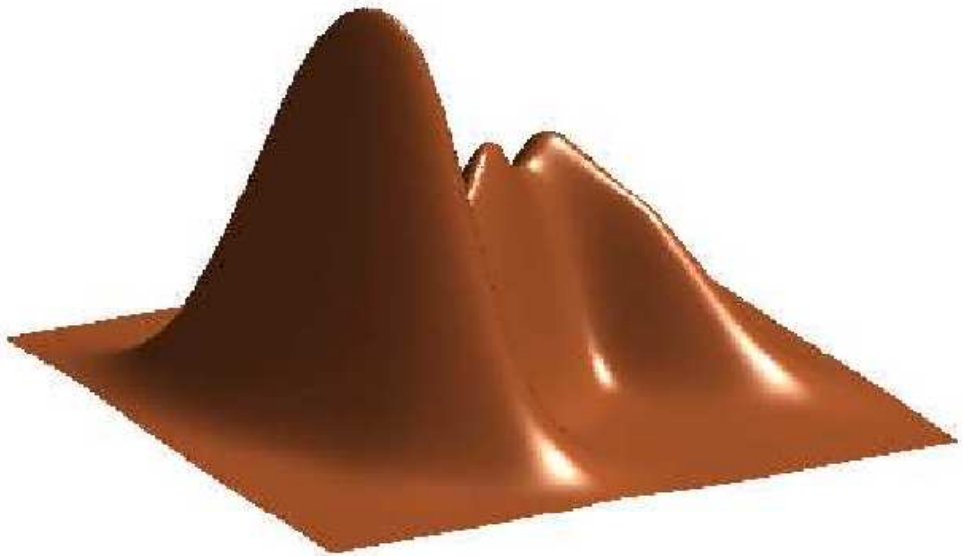
A mixture of 3 Gaussians in 2D:





Mixture of Gaussians

A mixture of Gaussians in 3D:



Non-convexity of ML/MAP Estimation

Consider the log-likelihood for an LVM on a dataset $\mathcal{D}$:

$$\begin{aligned}\ell(\theta) \equiv \log p(\mathcal{D}|\theta) &= \sum_{i=1}^{N} \log \left[p(\mathbf{x}_i \,|\theta)\right] \\ &= \sum_{i=1}^{N} \log \left[\sum_{\mathbf{z}_i} p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)\right].\end{aligned}$$

This objective is hard to optimize, since we cannot push the log inside the sum.

Non-convexity of ML/MAP Estimation

Suppose that the joint distribution $p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)$ is in the exponential family:

$$p(\mathbf{x}, \mathbf{z} \,|\theta) = \frac{1}{Z(\theta)} \exp[\theta^T \phi(\mathbf{x}, \mathbf{z})],$$

where $\phi(\mathbf{x}, \mathbf{z})$ are the sufficient statistics and $Z(\theta)$ is the normalization constant.
Then the **complete data log-likelihood** is:

$$\ell_c(\theta) \equiv \sum_{i=1}^{N} \log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta) = \underbrace{\theta^T \left(\sum_i \phi(\mathbf{x}_i, \mathbf{z}_i) \right)}_{\text{linear in } \theta} - N \underbrace{Z(\theta)}_{\text{convex}}.$$

Thus, $\ell_c(\theta)$ is concave.

Non-convexity of ML/MAP Estimation

- We have

$$\ell(\theta) = \sum_{i=1}^{N} \log \left[\sum_{\mathbf{z}_i} \exp\left(\theta^T \phi(\mathbf{x}_i, \mathbf{z}_i)\right) \right] - N \log Z(\theta).$$

  The log-sum-exp function is convex[1] and $Z(\theta)$ is convex.
- However, in general, the difference of two convex functions is not convex.
- So, the objective is neither convex nor concave, and has local optima.

[1] See (Boyd and Vandenberghe, 2004).

The EM Algorithm

- For many models in machine learning, computing the ML/MAP parameter estimate is easy if we observe all the values of all the relevant random variables.
- However, if we have missing data and/or latent variables, then computing ML/MAP estimate become hard.

The EM Algorithm

- One approach is to use a generic gradient-based optimizer to find a local maximum of the log-likelihood:

$$\ell(\theta) = \frac{1}{N} \log p(\mathcal{D}|\theta).$$

- However, we often have to enforce constraints:
  - The covariance matrices must be positive definite, or
  - Mixing weights must sum to one, ...
- Another approach is to use the **expectation maximization** algorithm (EM).

The EM Algorithm

The EM algorithm has some advantages:

- It is simple iterative algorithm, often with closed-form updates at each step.
- It automatically enforce the required constraints.

Basic Idea

- The goal is to maximize the log-likelihood of the observed data:

$$\ell(\theta) = \sum_{i=1}^{N} \log p(\mathbf{x}_i \,|\theta) = \sum_{i=1}^{N} \log \left[\sum_{\mathbf{z}_i} p(\mathbf{x}_i, \mathbf{z}_i \,|\theta) \right],$$

  where $\mathbf{x}_i$ are visible (observed) variables and $\mathbf{z}_i$ are hidden or missing variables.

- This is a hard to optimize, since the complete data log-likelihood cannot be computed when $\mathbf{z}_i$ is unknown:

$$\ell_c(\theta) = \sum_{i=1}^{N} \log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta).$$

Basic Idea

Define the **expected complete data log-likelihood** as follow:

$$Q(\theta, \theta^{(t-1)}) = \mathbb{E}[\ell_c(\theta)|\mathcal{D}, \theta^{(t-1)}],$$

where

- t is the current iteration number;
- Q is called the **auxiliary function**;
- The expectation is taken w.r.t. the old parameters $\theta^{(t-1)}$ and the observed data $\mathcal{D}$.

Basic Idea

Basic idea of the EM algorithm: Alternate between two steps:

E-step: Compute $Q(\theta, \theta^{(t-1)})$;

M-step: Optimize $\theta^{(t)} = \arg\max_\theta Q(\theta, \theta^{(t-1)})$;

In MAP estimation, the M-step is modified as follows:

$$\theta^{(t)} = \arg\max_\theta Q(\theta, \theta^{(t-1)}) + \log p(\theta).$$

EM for GMMs – Auxiliary Function

The expected complete data log-likelihood is given by:

$$
\begin{aligned}
Q(\theta, \theta^{(t-1)}) &= \mathbb{E}\left[\sum_i \log p(\mathbf{x}_i, z_i|\theta)\right] \\
&= \mathbb{E}\left[\sum_i \log\left(\prod_{k=1}^{K} (\pi_k p(\mathbf{x}_i\,|\theta_k))^{\delta(z_i=k)}\right)\right] \\
&= \sum_i \sum_k \mathbb{E}[\delta(z_i = k)] \log[\pi_k p(\mathbf{x}_i\,|\theta_k)] \\
&= \sum_i \sum_k p(z_i = k|\,\mathbf{x}_i, \theta^{(t-1)}) \log[\pi_k p(\mathbf{x}_i\,|\theta_k)] \\
&= \sum_i \sum_k r_{ik} \log \pi_k + \sum_i \sum_k r_{ik} \log p(\mathbf{x}_i\,|\theta_k),
\end{aligned}
$$

where $r_{ik} = p(z_i = k|\,\mathbf{x}_i, \theta^{(t-1)})$.

EM for GMMs – E-step

- r_{ik} is the **responsibility** that cluster k takes for the data point i.
- The E-step has the following simple form:

$$r_{ik} = \frac{\pi_k p(\mathbf{x}_i \,|\theta_k^{(t-1)})}{\sum_s \pi_s p(\mathbf{x}_i \,|\theta_s^{(t-1)})}.$$

- The E-step is the same for any mixture model.

EM for GMMs – M-step

In the M-step, we optimize Q w.r.t. π and the θ_k. For π we have

$$\pi_k = \frac{1}{N} \sum_i r_{ik} = \frac{r_k}{N},$$

where $r_k = \sum_i r_{ik}$ is the weighted number of points assigned to cluster k.

EM for GMMs – M-step

To compute (μ_k, Σ_k) parameters, we consider the parts of Q that depend on these variables:

$$\begin{aligned}\ell(\mu_k, \Sigma_k) &= \sum_i \sum_k r_{ik} \log p(\mathbf{x}_i \,|\theta_k) \\ &= -\frac{1}{2} \sum_i r_{ik} \left[\log |\Sigma_k| + (\mathbf{x}_i - \mu_k)^T \Sigma_k^{-1} (\mathbf{x}_i - \mu_k)\right]\end{aligned}$$

This is just the weighted version of the standard problem of computing the MLEs of an multivariate normal:

$$\mu_k = \frac{\sum_i r_{ik}\, \mathbf{x}_i}{r_k}$$

$$\Sigma_k = \frac{\sum_i r_{ik} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T}{r_k} = \frac{\sum_i r_{ik}\, \mathbf{x}_i\, \mathbf{x}_i^T}{r_k} - \mu_k \mu_k^T.$$

Iris Dataset

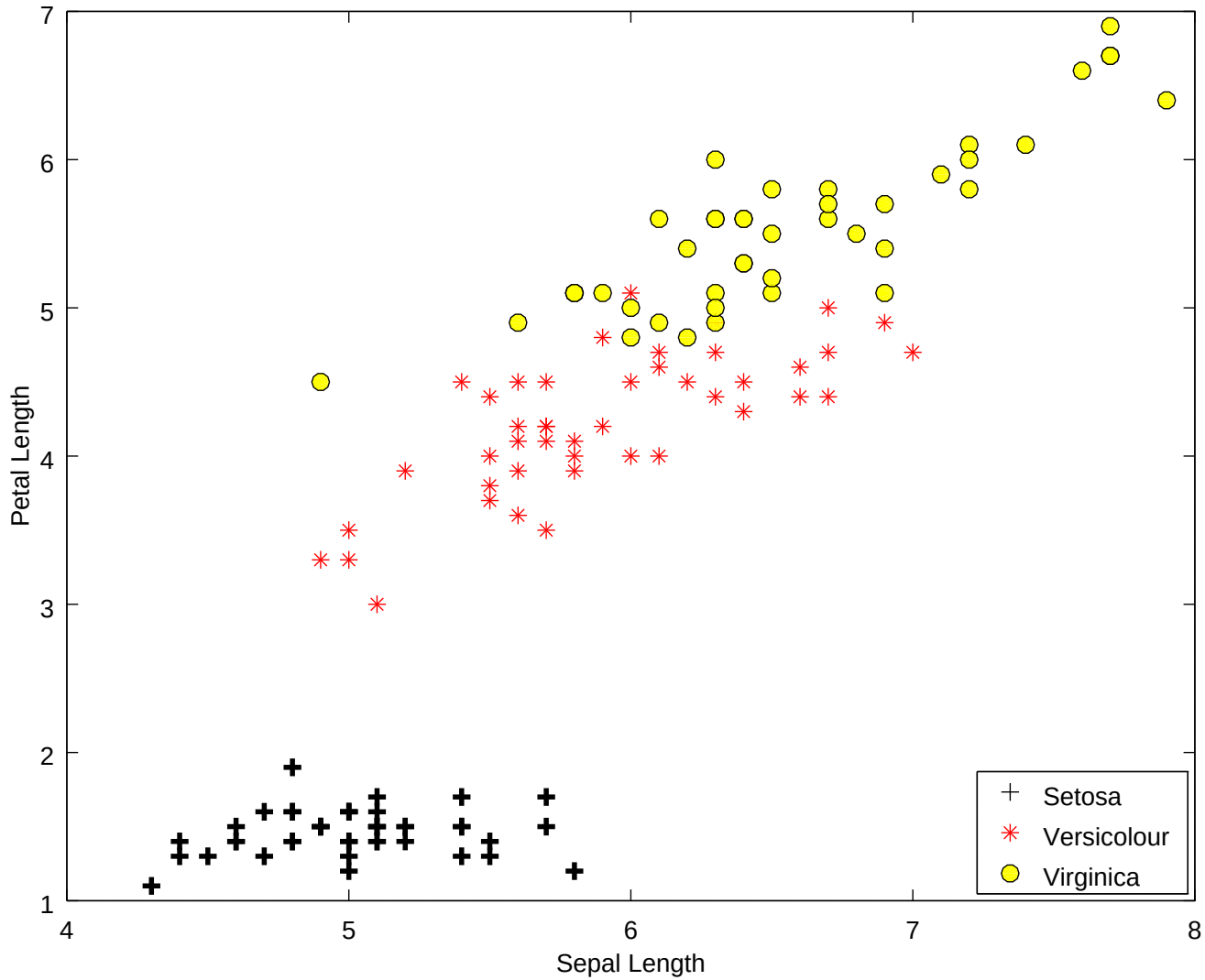
- The well-known Iris dataset:

  `http://archive.ics.uci.edu/ml/datasets/Iris`

- First appeared in a paper of Ronald Fisher in 1936, is still frequently in use today.
- Training set: 130 examples; test set: 20 examples

Feature	Label
sepal length	`Setosa`
sepal width	`Versicolour`
petal length	`Virginica`
petal width	

Iris Dataset – Scatter Plot using 2 Features



Iris Dataset – EM Algorithm

Initialization:

$$\pi = (1/3, 1/3, 1/3)$$

$$\mu \equiv (\mu_1, \mu_2, \mu_3) = \begin{pmatrix} 5.0140 & 5.9023 & 6.5605 \\ 1.4628 & 4.2295 & 5.5326 \end{pmatrix}$$

$$\Sigma_k = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \forall k = 1, 2, 3.$$

Iris Dataset – EM Algorithm

After 30 iterations:

$$\pi = (0.33077, 0.39265, 0.27658)$$

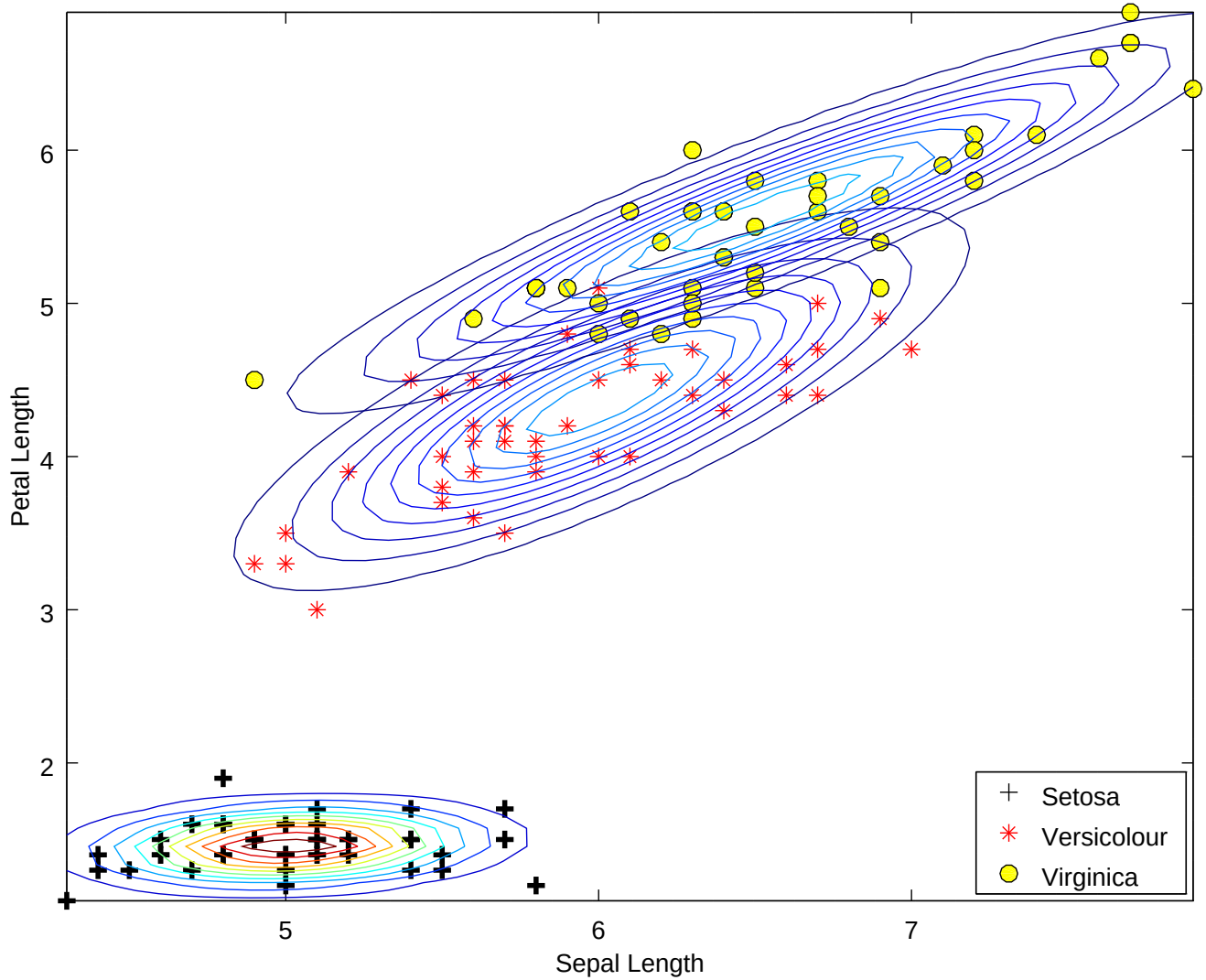
$$\mu \equiv (\mu_1, \mu_2, \mu_3) = \begin{pmatrix} 5.0140 & 6.0090 & 6.5379 \\ 1.4628 & 4.3715 & 5.5864 \end{pmatrix}$$

$$\Sigma_1 = \begin{pmatrix} 0.12306 & 0.00819 \\ 0.00819 & 0.02279 \end{pmatrix}$$

$$\Sigma_2 = \begin{pmatrix} 0.28735 & 0.24421 \\ 0.24421 & 0.32315 \end{pmatrix}$$

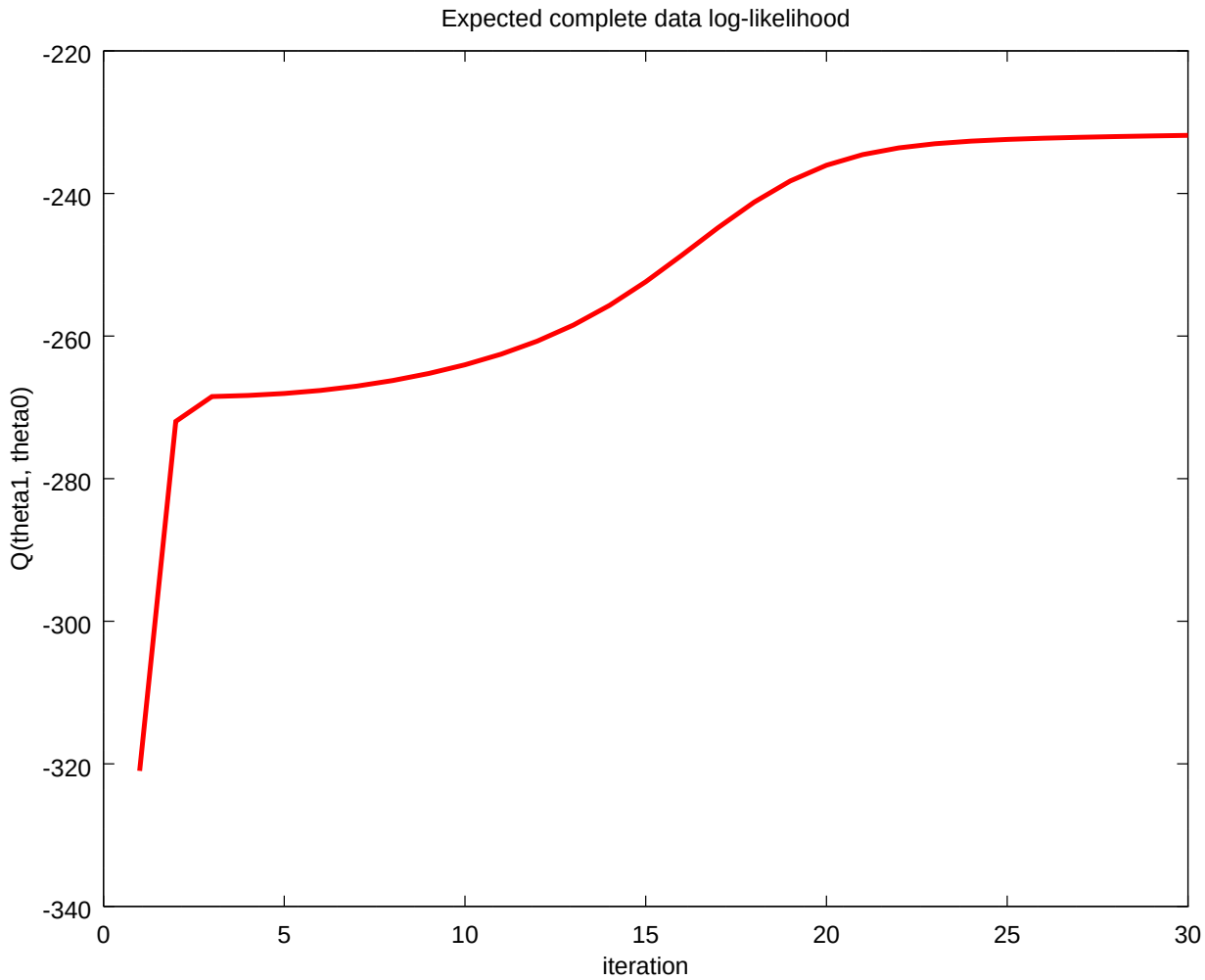
$$\Sigma_3 = \begin{pmatrix} 0.49077 & 0.38449 \\ 0.38449 & 0.35657 \end{pmatrix}$$

Iris Dataset – Clusters



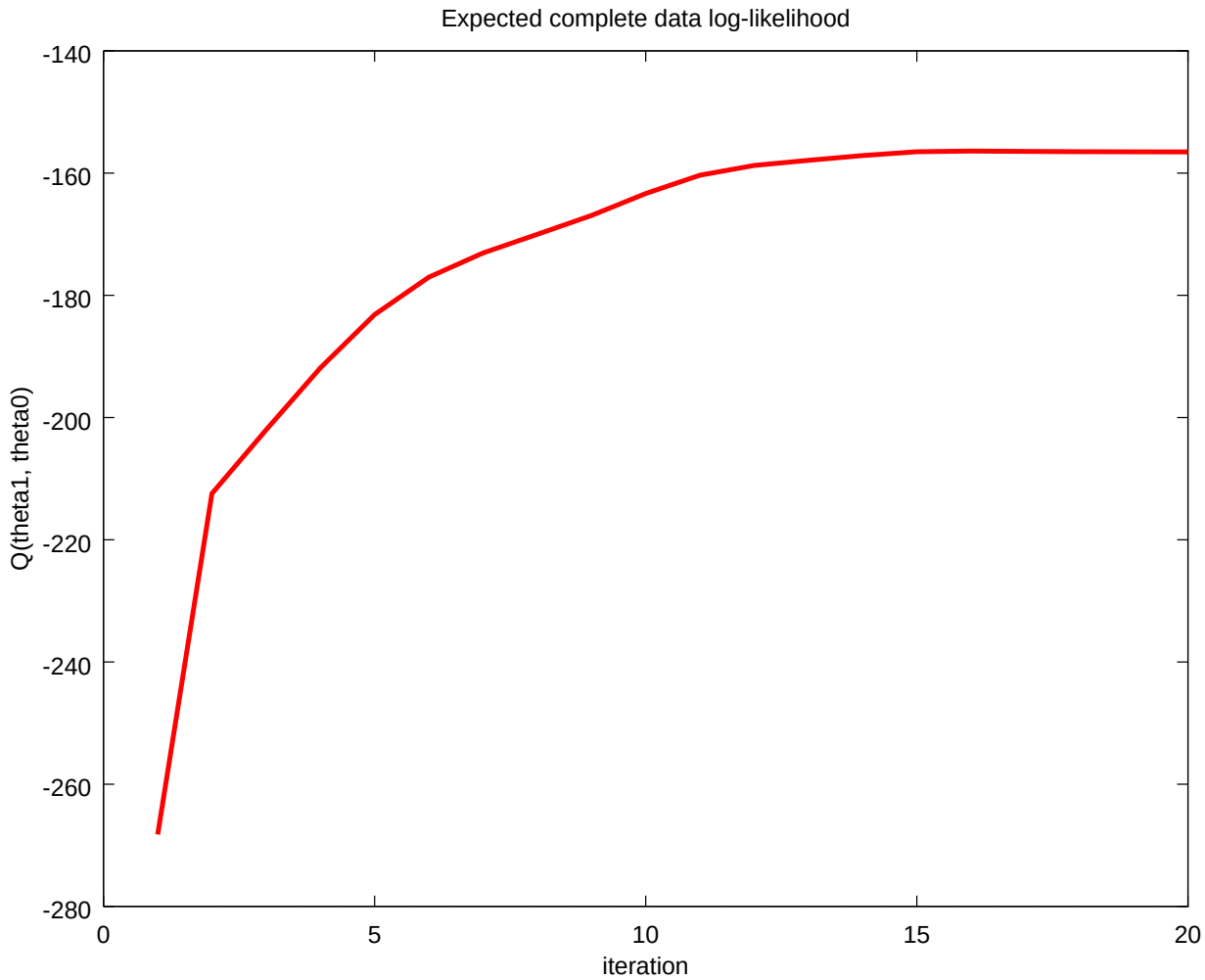
Iris Dataset – $Q(\theta^{(t)}, \theta^{(t-1)})$

2 Features



Iris Dataset – $Q(\theta^{(t)}, \theta^{(t-1)})$

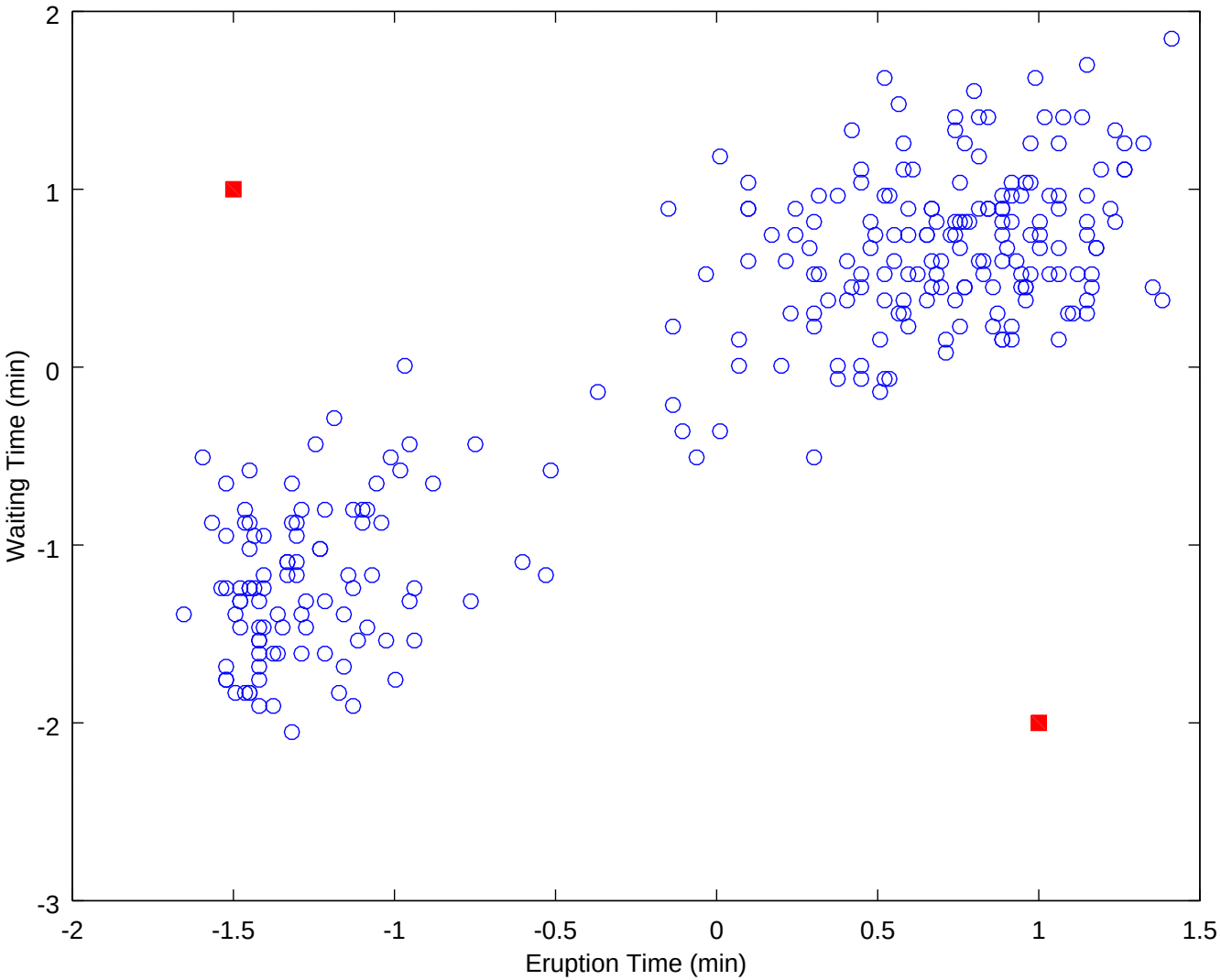
4 Features



Old Faithful Dataset

- Waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.
- 272 observations on 2 variables.
  - eruptions: Eruption time in mins
  - waiting: Waiting time to next eruption
- We should normalize the data for ease of processing.

Old Faithful Dataset – Scatter Plot



Old Faithful Dataset – EM Algorithm

Initialization:

$$\pi = (1/2, 1/2)$$

$$\mu \equiv (\mu_1, \mu_2) = \begin{pmatrix} -1.5 & 1.0 \\ 1.0 & -2.0 \end{pmatrix}$$

$$\Sigma_k = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \forall k = 1, 2.$$

Old Faithful Dataset – EM Algorithm

After 30 iterations:

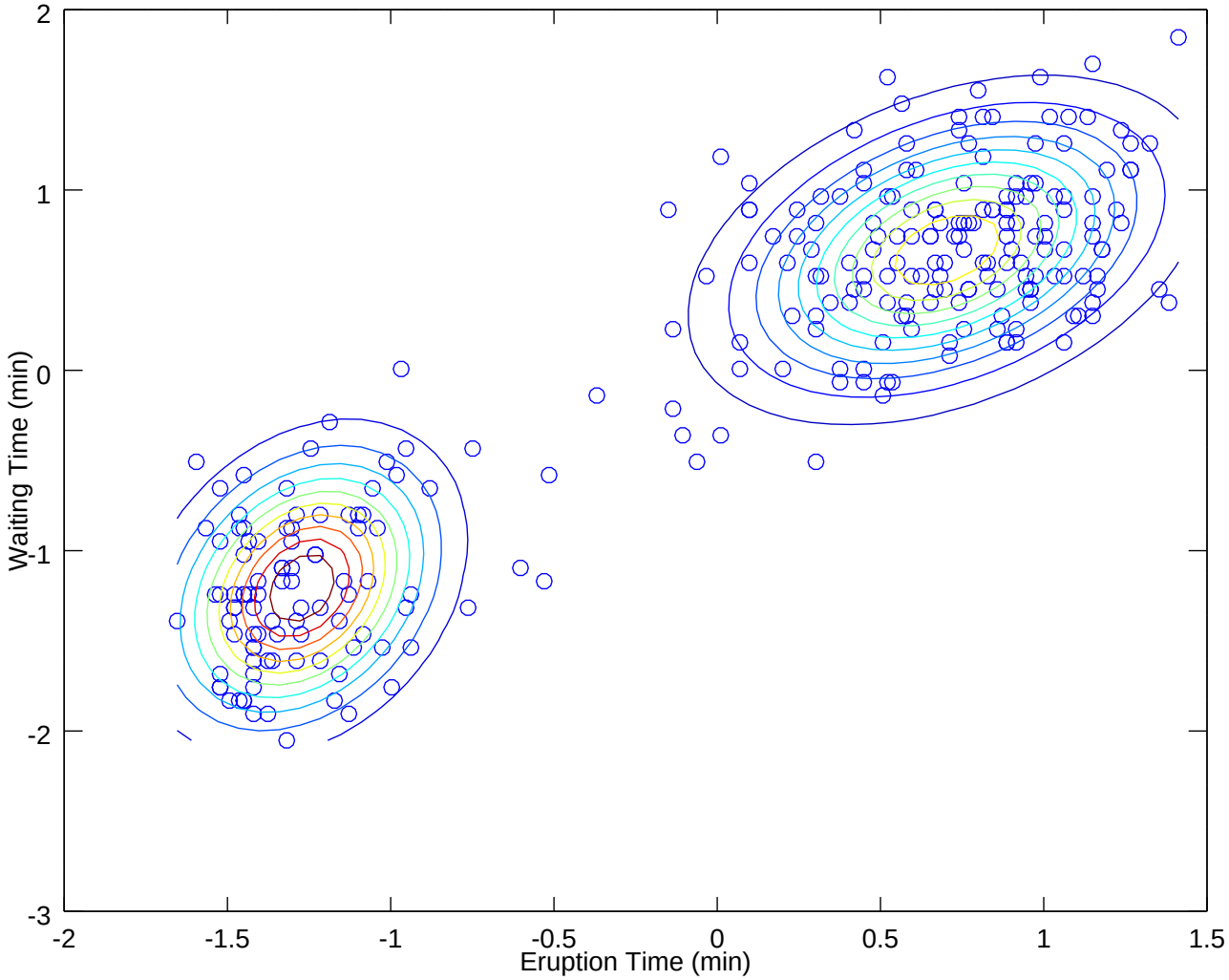
$$\pi = (0.64410, 0.35590)$$

$$\mu \equiv (\mu_1, \mu_2) = \begin{pmatrix} 0.70261 & -1.27156 \\ 0.66729 & -1.20764 \end{pmatrix}$$

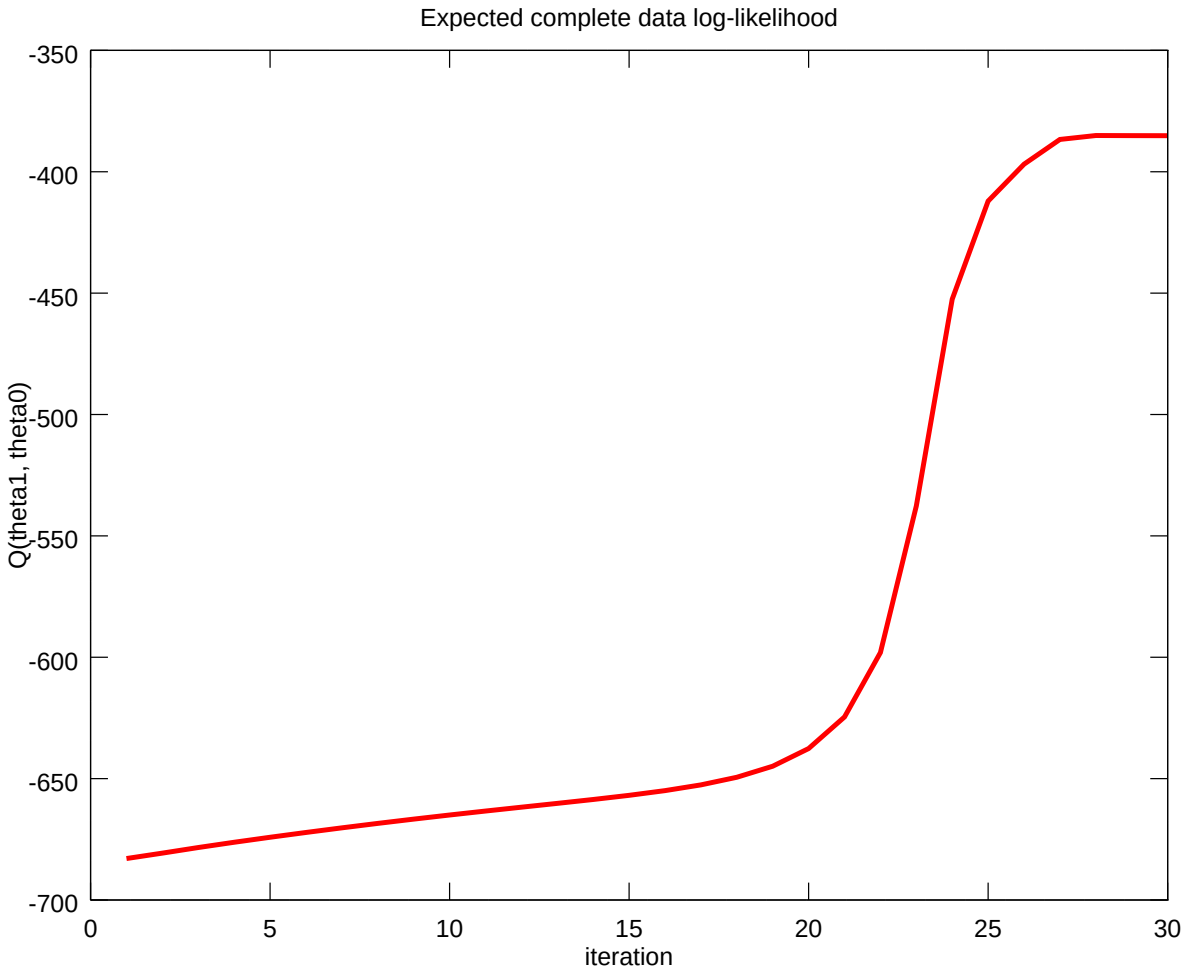
$$\Sigma_1 = \begin{pmatrix} 0.130411 & 0.060554 \\ 0.060554 & 0.194970 \end{pmatrix}$$

$$\Sigma_2 = \begin{pmatrix} 0.053137 & 0.028082 \\ 0.028082 & 0.182343 \end{pmatrix}$$

Old Faithful Dataset – Clusters



Old Faithful Dataset – $Q(\theta^{(t)}, \theta^{(t-1)})$



K-means Algorithm

- **K-means algorithm** is a popular variant of the EM algorithm for GMMs.
- Consider a GMM in which we make the following assumptions:
  1. $\Sigma_k = \sigma^2 \mathbb{I}_D$ is fixed;
  2. $\pi_k = 1/K$ is fixed.

  So only the cluster centers $\mu_k \in \mathbb{R}^D$ have to be estimated.
- Consider the delta function approximation to the posterior computed during the E step:

$$p(z_i = k | \mathbf{x}_i, \theta) \approx \mathbb{I}(k = z^*),$$

  where $z^* = \arg\max_k p(z_i = k | \mathbf{x}_i, \theta)$.
- This is sometimes called **hard EM** since we make a hard assignment of points to clusters.

K-means Algorithm

- The most probable cluster for a data point $\mathbf{x}_i$ can be computed by finding the nearest cluster:

$$z_i = \arg\min_k \| \mathbf{x}_i - \mu_k \|^2.$$

- Hence, in each step, we must find the Euclidean distance between N data points and K cluster centers, which take $O(NKD)$ time.
- Given the hard cluster assignments, the M step updates each cluster centers by computing the mean of the points assigned to it:

$$\mu_k = \frac{1}{N_k} \sum_{i:z_i=k} \mathbf{x}_i \, .$$

K-means Algorithm

Algorithm 1: K-means Algorithm

Data: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N, K$

Result: $\mu_1, \mu_2, \ldots, \mu_K$

Initialize $\mu_k, k = 1, 2, \ldots, K$;

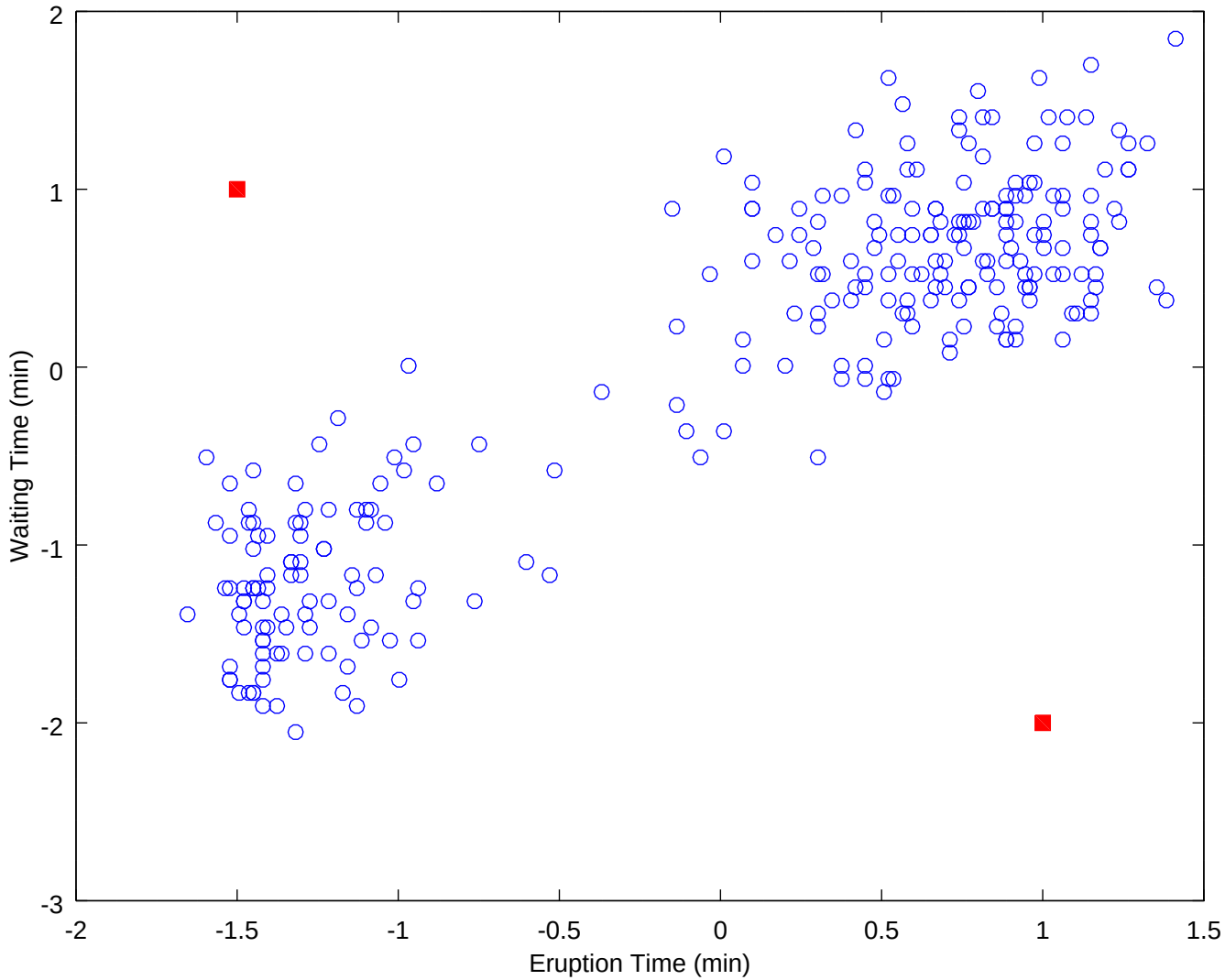
repeat

  $z_i \leftarrow \arg\min_k \| \mathbf{x}_i - \mu_k \|^2$;

  $\mu_k \leftarrow \frac{1}{N_k} \sum_{i:z_i=k} \mathbf{x}_i$;

until *(converged)*;

Old Faithful Dataset – Scatter Plot



Old Faithful Dataset – K-means Algorithm

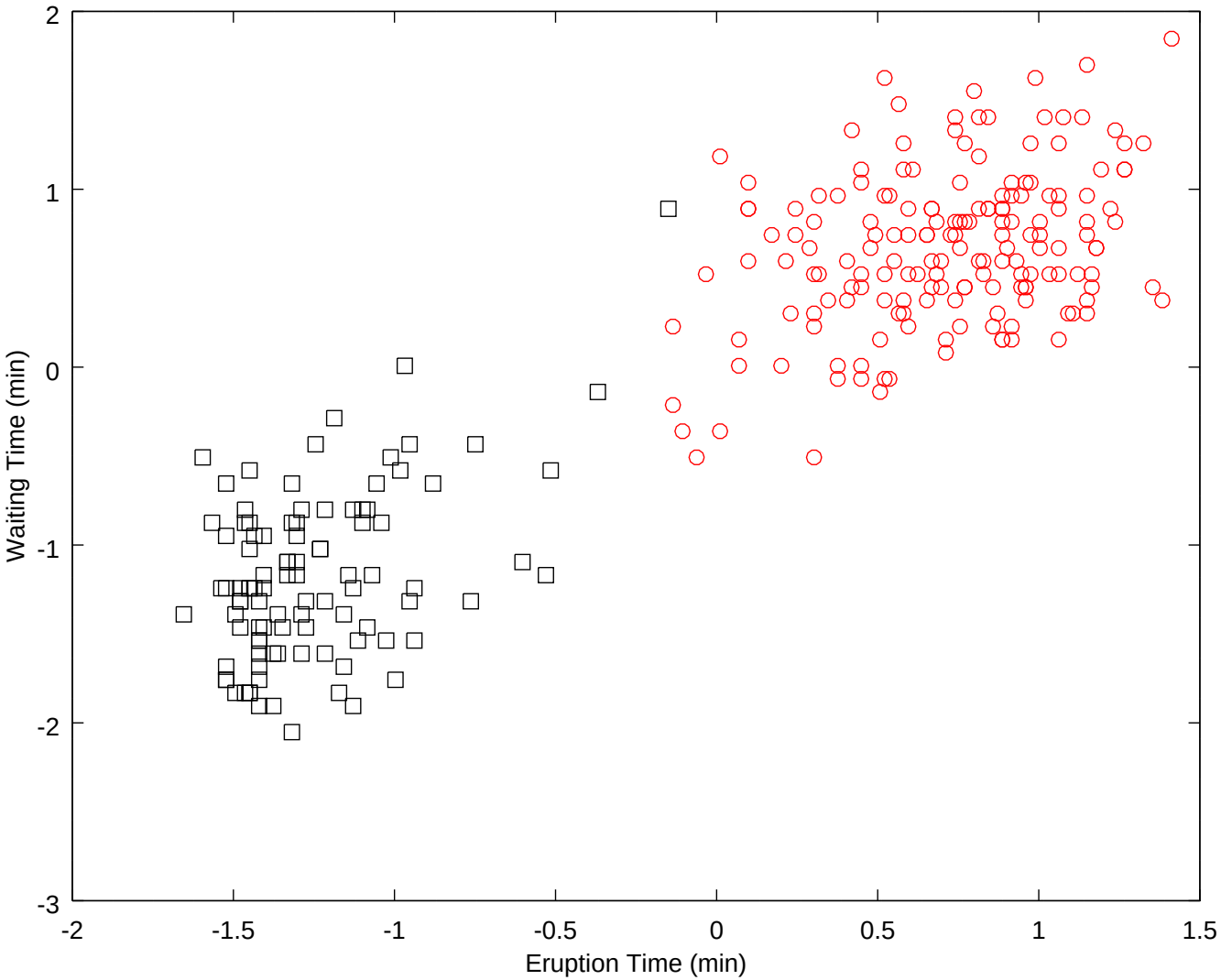
Initialization: Random cluster assignments with centroids:

$$\mu \equiv (\mu_1, \mu_2) = \begin{pmatrix} -1.5 & 1.0 \\ 1.0 & -2.0 \end{pmatrix}$$

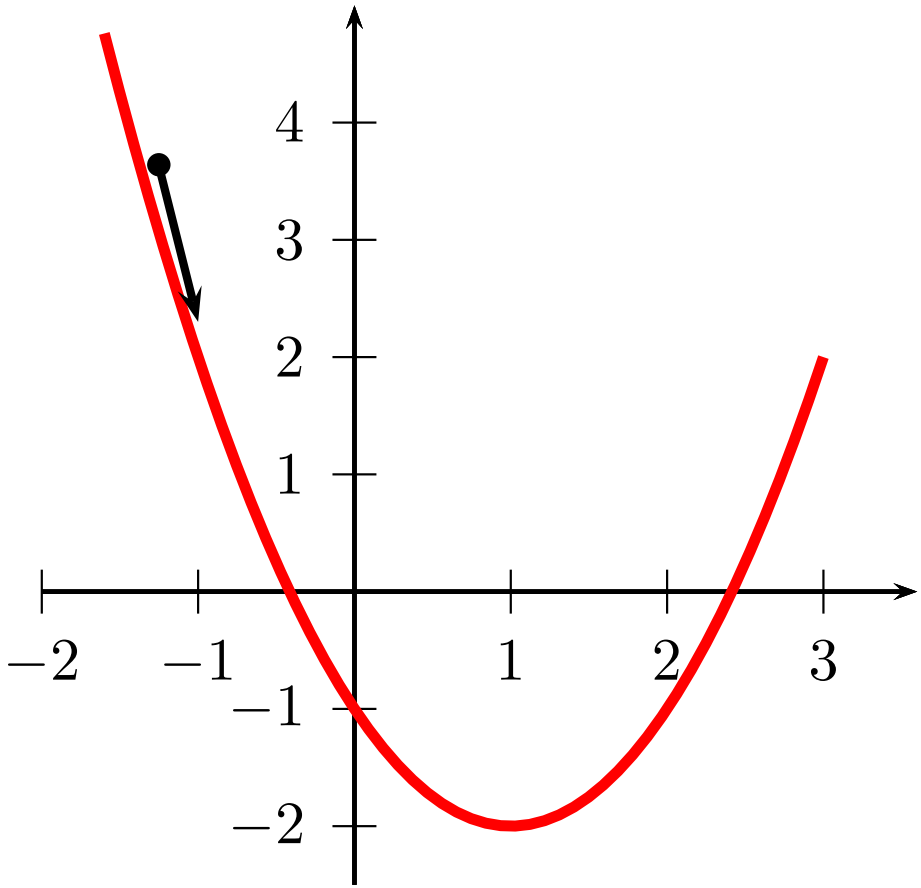
After 2 iterations:

$$\mu \equiv (\mu_1, \mu_2) = \begin{pmatrix} 0.098318 & -0.739366 \\ 0.596025 & -0.621445 \end{pmatrix}$$

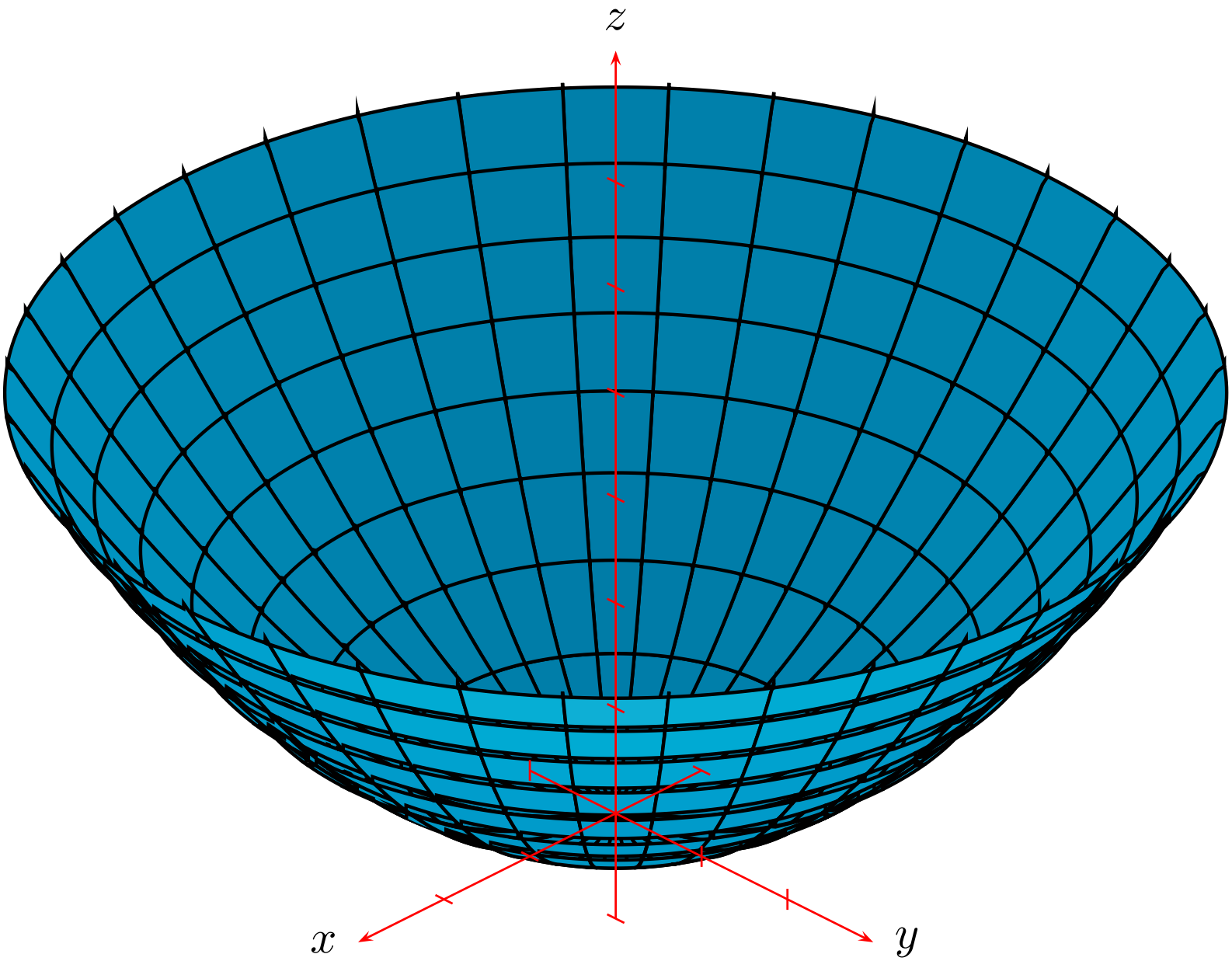
Old Faithful Dataset – K-means Algorithm



Convex Function

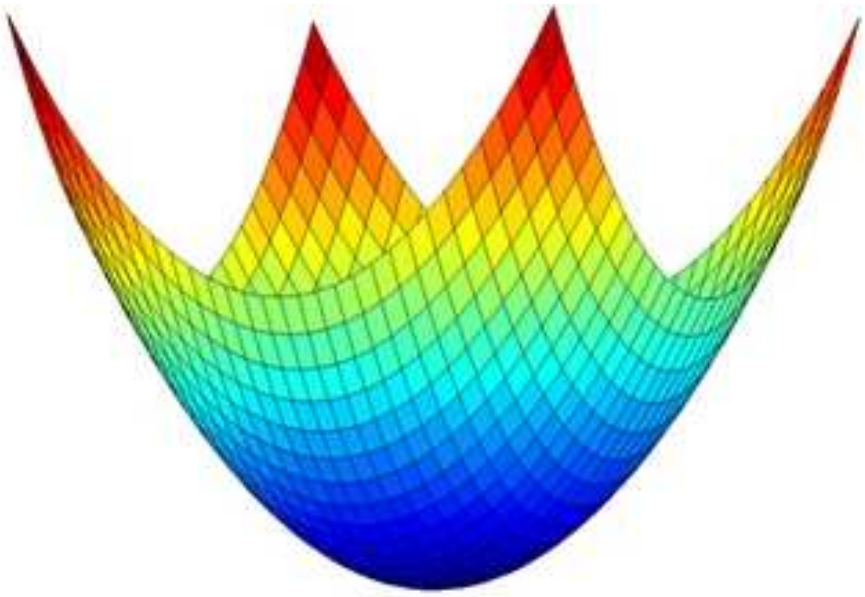


Convex Function



Convex Function

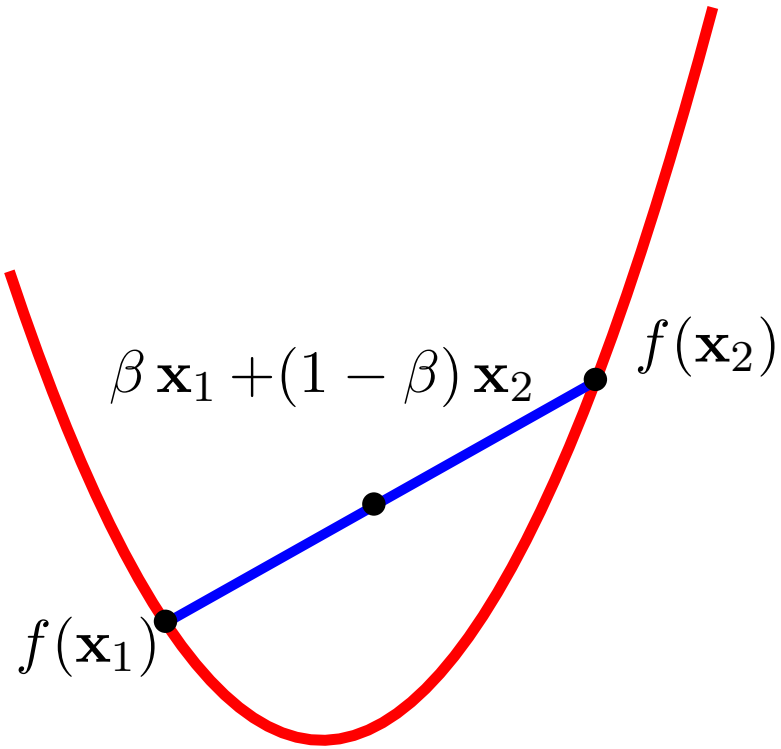
The graph of a convex function always leans to the minimum:



Convex Function

A function $f : \mathbb{R}^n \to \mathbb{R}$ is called a **convex function** if $\forall\, \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n, \forall \beta \in [0, 1]$:

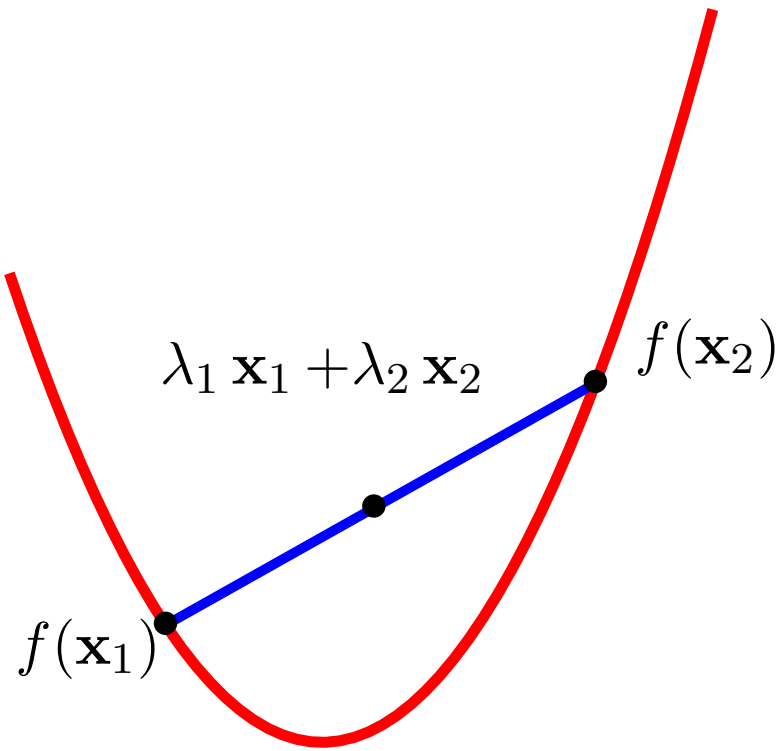
$$f(\beta\, \mathbf{x}_1 + (1 - \beta)\, \mathbf{x}_2) \leq \beta f(\mathbf{x}_1) + (1 - \beta) f(\mathbf{x}_2).$$



Convex Function

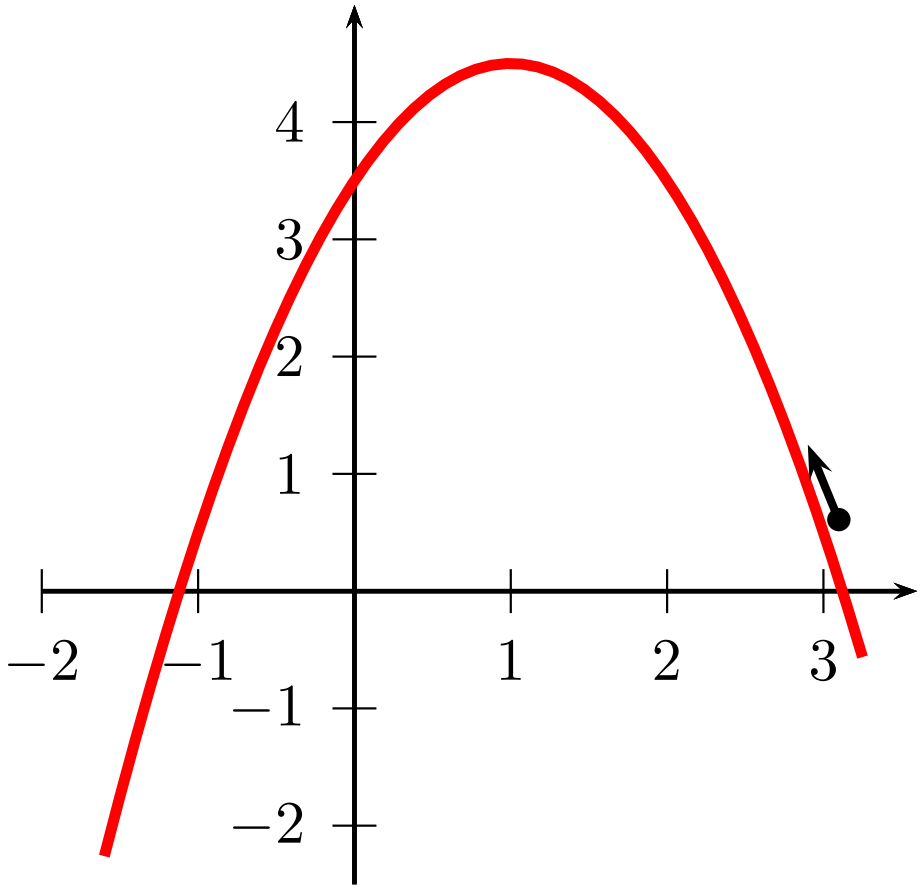
A different formulation: $f : \mathbb{R}^n \to \mathbb{R}$ is a **convex function** if $\forall\, \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n, \forall \lambda_1, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1$:

$$f(\lambda_1\, \mathbf{x}_1 + \lambda_2\, \mathbf{x}_2) \leq \lambda_1 f(\mathbf{x}_1) + \lambda_2 f(\mathbf{x}_2).$$



Concave Function

A function f is called a **concave function** if $-f$ is a convex function.



Jensen's Inequality

Theorem (Jensen's inequality)

For any convex function f, we have that:

$$f\left(\sum_{i=1}^{n} \lambda_i \mathbf{x}_i\right) \leq \sum_{i=1}^{n} \lambda_i f(\mathbf{x}_i),$$

where $\lambda_i \geq 0$ and $\sum_{i=1}^{n} \lambda_i = 1$.

This is clearly true for $n = 2$, by the definition of convexity, and can be proved by induction for $n > 2$.

Entropy

Definition (Entropy)

Entropy of a probability distribution p of a discrete random variable X on a set $\mathcal{X}$ is defined as

$$\mathbb{H}(p) = -\sum_{x \in \mathcal{X}} p(x) \log p(x).$$

If X is a continuous random variable on $\mathbb{R}$ with density function p, then entropy of p is defined as

$$\mathbb{H}(p) = -\int_{-\infty}^{+\infty} p(x) \log p(x)\, dx.$$

Entropy

- Entropy of a random variable is a measure of its **uncertainty**.
- We usually use log base 2, in which case, the units are called **bits**.
- The discrete distribution with maximum entropy is the uniform distribution.
  - Hence, for a K-ary random variable, the entropy is maximized if $p(x = k) = 1/K$.
  - In this case, $\mathbb{H}(p) = \log_2 K$.
- The distribution with minimum entropy (zero) is any delta function that put all its mass into one state. Such distribution has no uncertainty.

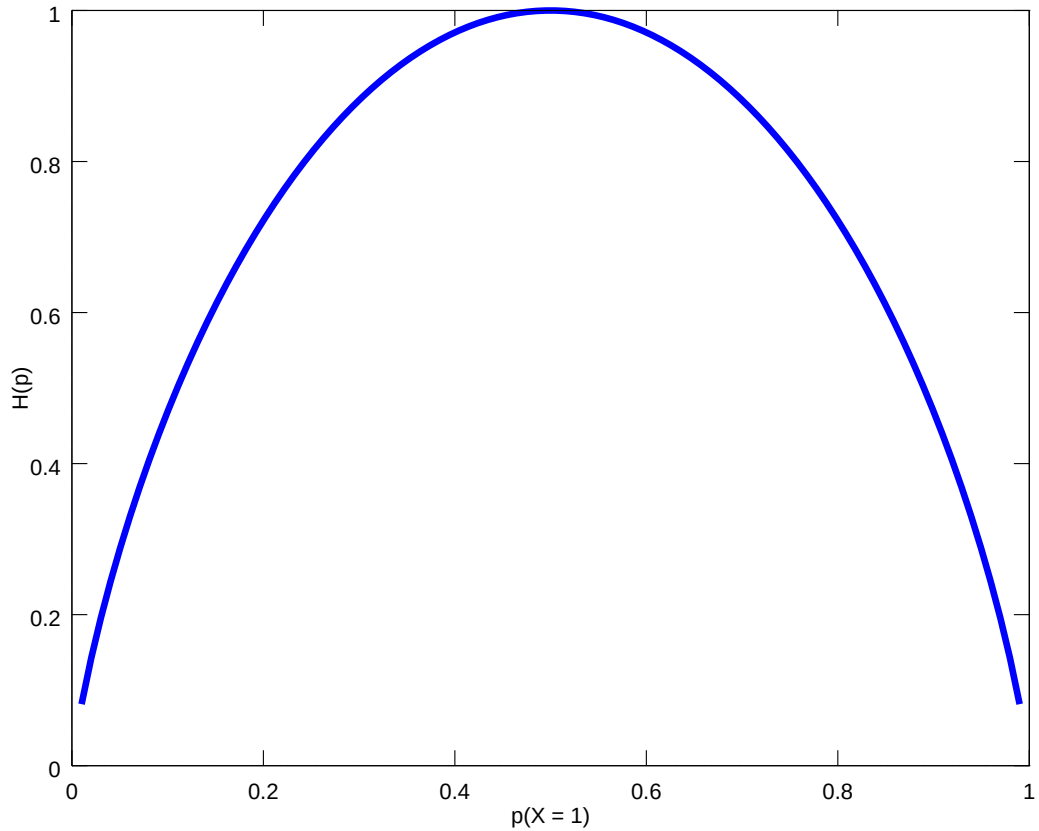
Entropy

For the special case of Bernoulli random variable, $X \in \{0, 1\}$, we can write $p(X = 1) = \theta$ and $P(X = 0) = 1 - \theta$. We have

$$\begin{aligned}\mathbb{H}(p) &= -[p(X = 1) \log_2 p(X = 1) + p(X = 0) \log_2 p(X = 0)] \\ &= -[\theta \log_2 \theta + (1 - \theta) \log_2 (1 - \theta)].\end{aligned}$$

This is called the **binary entropy function**.

Binary Entropy Function



The maximum value of 1 occurs when the distribution is uniform, $\theta = 0.5$.

Kullback-Liebler Divergence

Definition (Relative entropy, Kullback–Liebler Divergence)

The relative entropy $\mathbb{KL}$ between two discrete probability distributions p and q defined on $\mathcal{X}$ is:

$$\mathbb{KL}(p, q) = \sum_{x \in \mathcal{X}} p(x) \frac{\log p(x)}{\log q(x)}.$$

We can rewrite this as

$$\begin{aligned} \mathbb{KL}(p, q) &= \sum_{x \in \mathcal{X}} p(x) \log p(x) - \sum_{x \in \mathcal{X}} p(x) \log q(x) \\ &= -\mathbb{H}(p) + \mathbb{H}(p, q), \end{aligned}$$

where $\mathbb{H}(p, q)$ is called the **cross-entropy**:

$$\mathbb{H}(p, q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x).$$

Kullback-Liebler Divergence

- One can show that the cross entropy is the average number of bits needed to encode data coming from a source with distribution p when we use model q to define our codebook.
- Hence, the regular entropy $\mathbb{H}(p) \equiv \mathbb{H}(p, p)$ is the expected number of bits if we use the true model.
- So, the $\mathbb{KL}$ divergence is the average number of *extra* bits needed to encode data if we use distribution q to encode data instead of the true distribution p.

Kullback-Liebler Divergence

The "extra number of bits" interpretation should make it clear that:

Lemma

For all distributions p, q, we have:

- $\mathbb{KL}(p, q) \geq 0$;
- $\mathbb{KL}(p, q) = 0 \Leftrightarrow p \equiv q$.

Kullback-Liebler Divergence

Proof. Let $A = \{x : p(x) > 0\}$ be the support of $p(x)$. Then

$$
\begin{aligned}
-\mathbb{KL}(p, q) &= -\sum_{x \in A} p(x) \log \frac{p(x)}{q(x)} = \sum_{x \in A} p(x) \log \frac{q(x)}{p(x)} \\
&\leq \log \left(\sum_{x \in A} p(x) \frac{q(x)}{p(x)} \right) = \log \left(\sum_{x \in A} q(x) \right) \\
&\leq \sum_{x \in \mathcal{X}} q(x) = \log 1 = 0.
\end{aligned}
$$

- The first inequality follows from Jensen's, since $\log(x)$ is strictly concave function.
- We have equality if $p(x) = cq(x)$ for some constant c and $\sum_{x \in A} q(x) = \sum_{x \in \mathcal{X}} q(x) = 1$, which implies $c = 1$.
- Hence, $\mathbb{KL}(p, q) = 0$ iff $p(x) \equiv q(x), \forall x$.

Kullback-Liebler Divergence

A consequence of this result is that the *discrete distribution with the maximum entropy is the uniform distribution.*

- Let p is the distribution of a random variable X defined on a set $\mathcal{X}$. Let $u(x) = \frac{1}{|\mathcal{X}|}$ is the uniform distribution.
- We have

$$\begin{aligned} 0 \leq \mathbb{KL}(p, u) &= \sum_x p(x) \log \frac{p(x)}{u(x)} \\ &\leq \sum_x p(x) \log p(x) - \sum_x p(x) \log u(x) \\ &= -\mathbb{H}(p) + \log |\mathcal{X}|. \end{aligned}$$

Hence, $\mathbb{H}(p) \leq \log |\mathcal{X}|$.

Theoretical Basis for EM

We show that EM monotonically increases the observed data log likelihood until it reaches a local maximum. We will show that:

$$\boxed{\ell(\theta^{(t+1)}) \geq Q(\theta^{(t+1)}, \theta^{(t)}) \geq Q(\theta^{(t)}, \theta^{(t)}) = \ell(\theta^{(t)})}$$

We have:

$$\begin{aligned}
\ell(\theta) &= \sum_{i=1}^{N} \log\left[p(\mathbf{x}_i \,|\theta)\right] = \sum_{i=1}^{N} \log\left[\sum_{\mathbf{z}_i} p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)\right], \\
&= \sum_{i=1}^{N} \log\left[\sum_{\mathbf{z}_i} q(\mathbf{z}_i) \frac{p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)}{q(\mathbf{z}_i)}\right],
\end{aligned}$$

where $q(\mathbf{z}_i)$ is an arbitrary distribution over the hidden variables.

Theoretical Basis for EM

Since $\log(u)$ is a concave function, from the Jensen's inequality, we have the following *lower bound*:

$$\begin{aligned}
\ell(\theta) &\geq \sum_i \sum_{\mathbf{z}_i} q(\mathbf{z}_i) \log \frac{p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)}{q(\mathbf{z}_i)} \\
&= \sum_i \left[\sum_{\mathbf{z}_i} q(\mathbf{z}_i) \log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta) - \sum_{\mathbf{z}_i} q(\mathbf{z}_i) \log q(\mathbf{z}_i) \right] \\
&= \underbrace{\sum_i \mathbb{E}_{q_i}[\log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)]}_{\text{expected complete data log likelihood}} + \underbrace{\sum_i \mathbb{H}(q_i)}_{\text{constant wrt. } \theta} \\
&\equiv Q(\theta, q).
\end{aligned}$$

Theoretical Basis for EM

Denote

$$
\begin{aligned}
L(\theta, q_i) &\equiv \sum_{\mathbf{z}_i} q_i(\mathbf{z}_i) \log \frac{p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)}{q_i(\mathbf{z}_i)} \\
&= \sum_{\mathbf{z}_i} q_i(\mathbf{z}_i) \log \frac{p(\mathbf{z}_i \,|\, \mathbf{x}_i, \theta) p(\mathbf{x}_i \,|\theta)}{q_i(\mathbf{z}_i)} \\
&= \sum_{\mathbf{z}_i} q_i(\mathbf{z}_i) \log \frac{p(\mathbf{z}_i \,|\, \mathbf{x}_i, \theta)}{q_i(\mathbf{z}_i)} + \sum_{\mathbf{z}_i} q_i(\mathbf{z}_i) \log p(\mathbf{x}_i \,|\theta) \\
&= -\,\mathbb{KL}(q_i(\mathbf{z}_i), p(\mathbf{z}_i \,|\, \mathbf{x}_i, \theta) + \log p(\mathbf{x}_i \,|\theta).
\end{aligned}
$$

The $p(\mathbf{x}_i \,|\theta)$ term is independent of q_i, so we can maximize the lower bound by setting $q_i(\mathbf{z}_i) \equiv p(\mathbf{z}_i \,|\, \mathbf{x}_i, \theta)$.

Theoretical Basis for EM

Since θ is unknown, we instead use $q_i^{(t)}(\mathbf{z}_i) = p(\mathbf{z}_i \,|\, \mathbf{x}_i, \theta^{(t)})$. We get the lower bound:

$$Q(\theta, q^{(t)}) \equiv \underbrace{\sum_i \mathbb{E}_{q_i^{(t)}} [\log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)]}_{Q(\theta, \theta^{(t)})} + \sum_i \mathbb{H}(q_i^{(t)}).$$

The M step becomes:

$$\begin{aligned} \theta^{(t+1)} &= \arg\max_\theta Q(\theta, \theta^{(t)}) \\ &\equiv \arg\max_\theta \sum_i \mathbb{E}_{q_i^{(t)}} [\log p(\mathbf{x}_i, \mathbf{z}_i \,|\theta)]. \end{aligned}$$

Theoretical Basis for EM

Since the $\mathbb{KL}$ is zero, we have:

$$L(\theta^{(t)}, q_i) = 0 + \log p(\mathbf{x}_i \,|\theta^{(t)}),$$

and hence

$$Q(\theta^{(t)}, \theta^{(t)}) = \sum_i p(\mathbf{x}_i \,|\theta^{(t)}) = \ell(\theta^{(t)}).$$

We see that the lower bound is tight after the M step. In summary, we have:

$$\ell(\theta^{(t+1)}) \geq Q(\theta^{(t+1)}, \theta^{(t)}) \geq Q(\theta^{(t)}, \theta^{(t)}) = \ell(\theta^{(t)})$$

Exercises

1. Implement the EM algorithm for GMMs;
2. Implement the K-means algorithm;
3. Test your implementations on some datasets (Iris, Old Faithful,...).