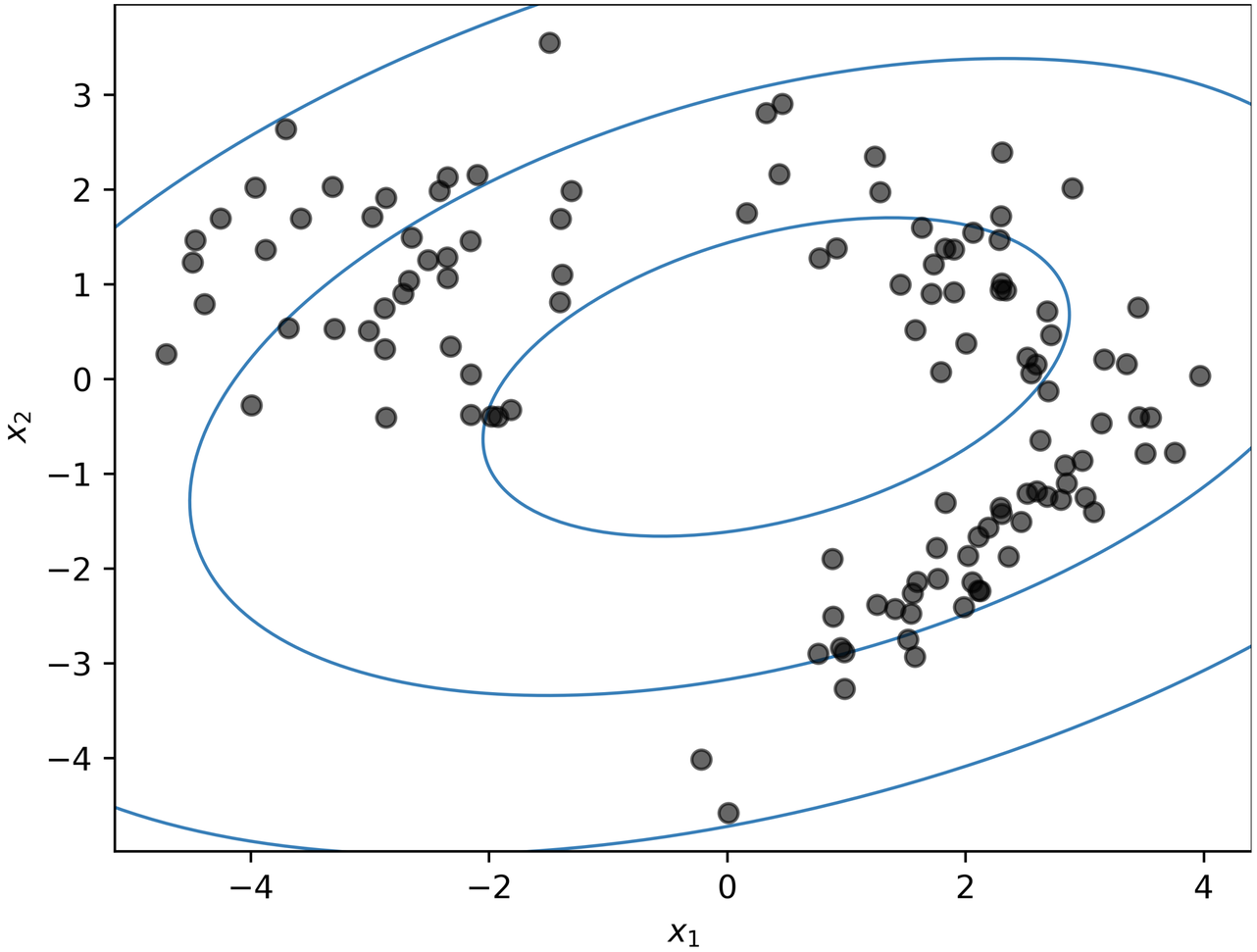
Gaussian Mixture Models

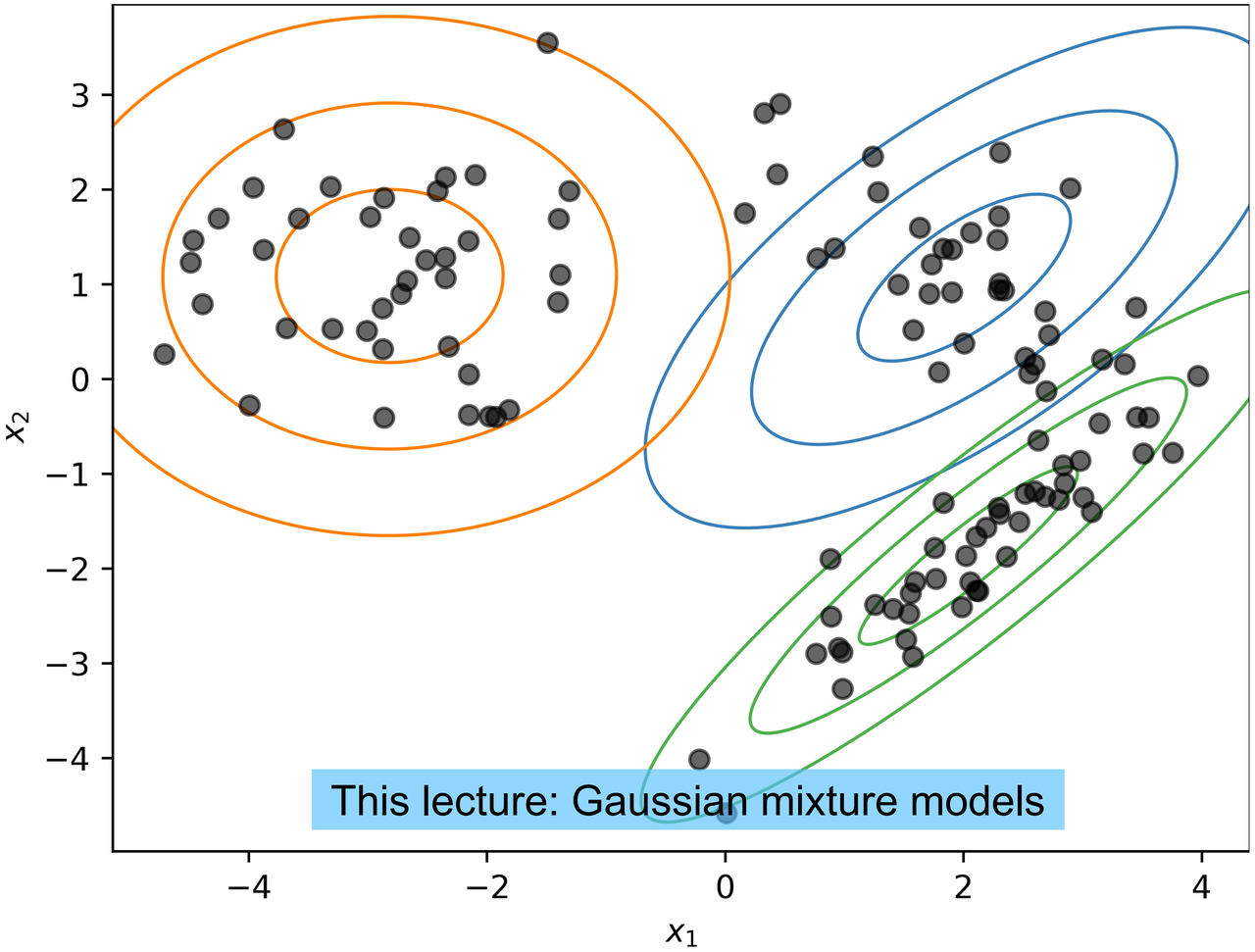
Rahul Shome
IML, S2 2024

Based on slides by Thang Bui

What is density estimation?

Goal: estimate the distribution your data come from, aka estimate $p(x)$





Applications: (1) exploratory analyses, (2) evaluation, (3) synthetic data generation

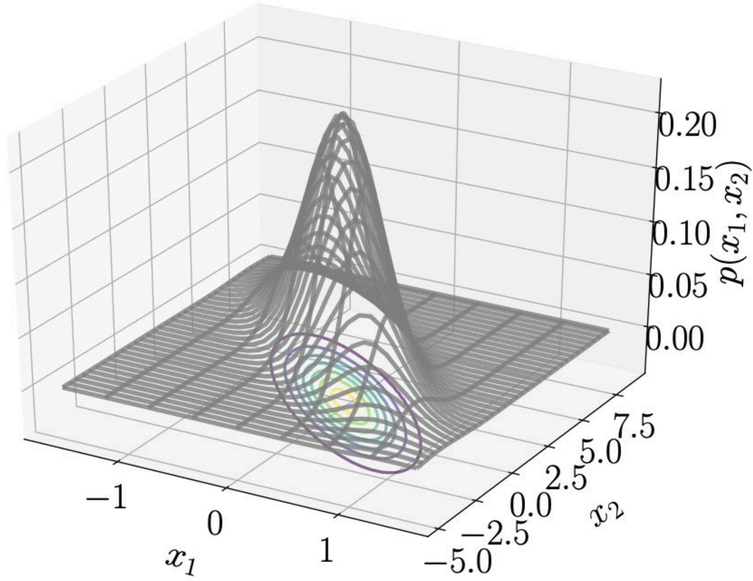
Are the data bimodal or skewed? Do datapoints come from the same density? Generate new samples that have the same characteristics

Recap: multivariate Gaussian distributions

A D-dimensional multivariate **Gaussian** or **normal** distribution is characterised by a *mean* vector μ and a *covariance matrix* Σ, with

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

We often write: $p(\mathbf{x}|\mu,\Sigma) = \mathcal{N}(\mathbf{x};\mu,\Sigma) or \mathbf{x} \sim \mathcal{N}(\mu,\Sigma)$



Standard multivariate **Gaussian** or **normal** distribution: $\mu = \mathbf{0}$ and $\Sigma = \mathbf{I}$

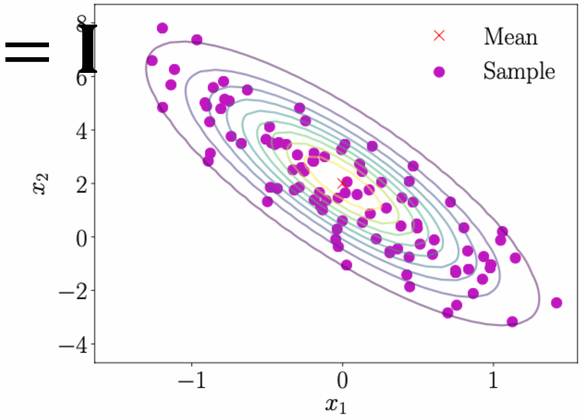
When Σ is a diagonal matrix, $p(\mathbf{x}|\mu,\Sigma) = \prod_{d=1}^{D} p(x_d|\mu_d,\Sigma_{d,d})$

Diagonal Gaussian, dimensions are independent

Alternative parameterisation, $\eta_1 = \Sigma^{-1}\mu \; and \; \eta_2 = \Sigma^{-1}$

[precision]

Easier for multiplication/division and identifying conditional independence



(b) Multivariate (two-dimensional) Gaussian, viewed from top. The red cross shows the mean and the colored lines show the contour lines of the density.

(Gaussian) mixture models: from one to multiple densities

Mixture models can be used to describe a distribution $p(x)$ by a *convex* combination of K simple (base) distributions

Why?

$$p(x) = \sum_{k=1}^{K} \pi_k p_k(x) 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

The components $p_k(x)$ are, typically, members of a family of basic distributions, e.g., Gaussians, Bernoullis, or Gammas, and π_k's are *mixture weights*.

When the base distributions are Gaussian distributions $p_k(x) = \mathcal{N}(x; \mu_k, \Sigma_k)$, we have a **Gaussian mixture model (GMM)**:

$$p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$$

$\theta = \{\pi_k, \mu_k, \Sigma_k fork = 1,2, \ldots, K\}$ are the parameters of the GMM.

Gaussian mixture models: check your understanding

Model: $p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k fork = 1,2, \ldots, K\}$ as the parameters of the GMM.

- Given a dataset generated by a mixture of 3 Gaussians, when we randomly sample a data point, it has the probability of 1/3 belonging to each Gaussian.
- A GMM is a linear combination of several Gaussian distributions.
- In GMMs, K (number of Gaussians) is a hyper-parameter.
- If a dataset is not generated by Gaussian distributions, it cannot be modelled by a GMM.

Gaussian mixture models: how to find θ

Model: $p_\theta(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k) 0 \leq \pi_k \leq 1, \sum_{k=1}^{K} \pi_k = 1$

where we defined $\theta = \{\pi_k, \mu_k, \Sigma_k fork = 1,2, \dots, K\}$ as the parameters of the GMM.

Data: $\mathbf{X} = \{x_1, x_2, \dots, x_N\}$, where x_n's are drawn i.i.d. from an unknown distribution $p(x)$.

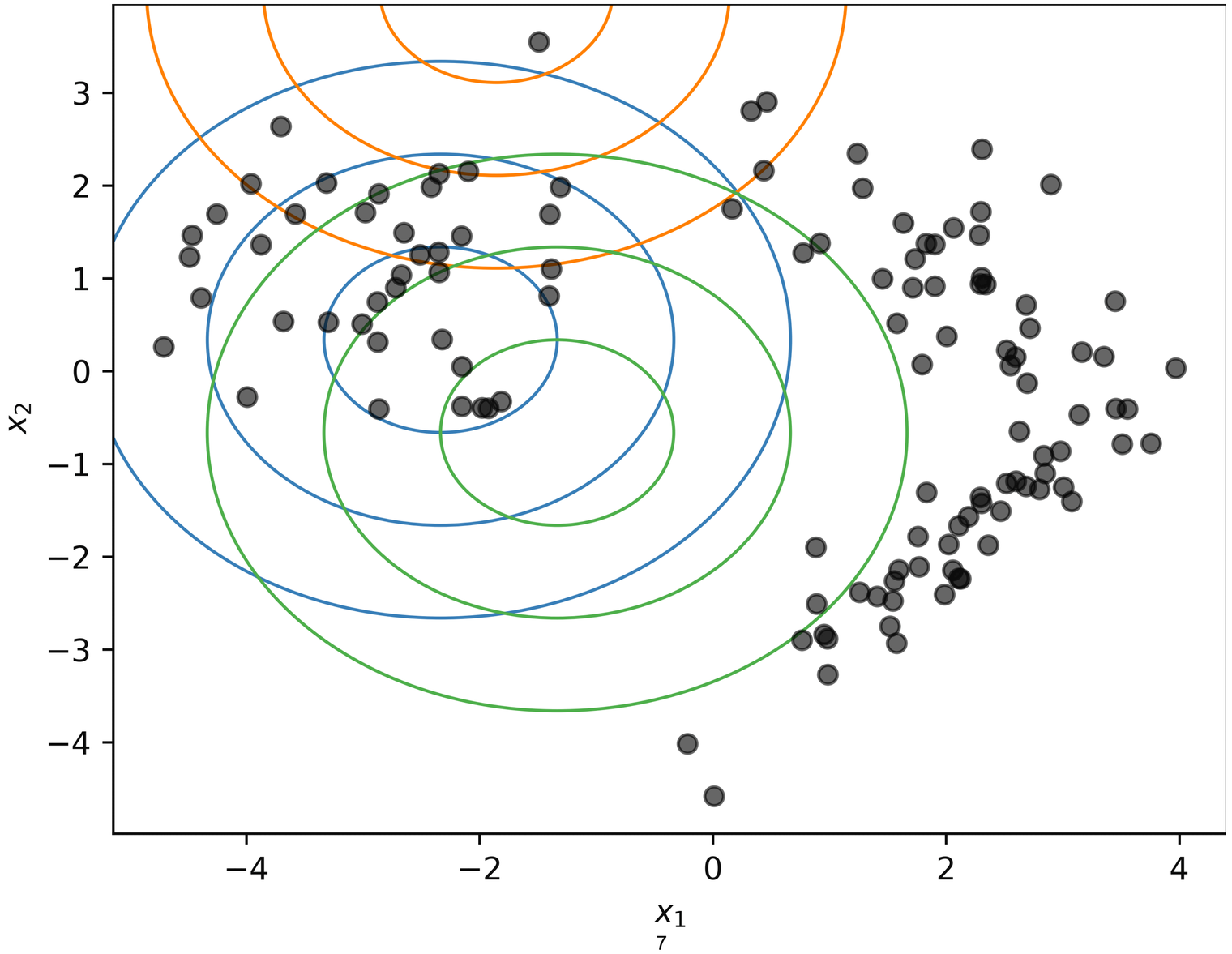
Objective: find a *good* approximation/representation of this unknown density $p(x)$ by means of a GMM with K components, that is finding θ.

$$P(\theta|\mathcal{D}) = \frac{P(\theta,\mathcal{D})}{P(\mathcal{D})} = \frac{P(\theta)P(\mathcal{D}|\theta)}{P(\mathcal{D})}$$

Weeks 6 and 7

- Maximum likelihood (ML/MLE), $\text{argmax}_\theta p(\mathcal{D}|\theta)$
- Maximum a-posteriori (MAP), $\text{argmax}_\theta p(\mathcal{D}|\theta)p(\theta)$
- Exact/approximate Bayesian inference

Maximum likelihood - Sneak peek



GMMs - likelihood

The likelihood, i.e., the predictive distribution of the training data given the parameters:

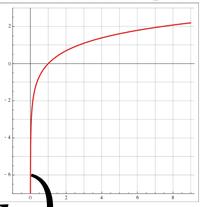
$$p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$$

Factorise due to iid assumption

Each individual likelihood term is a Gaussian mixture density

Goal: find maximise likelihood, $\text{argmax}_\theta p(X|\theta)$, which is equivalent to $\text{argmax}_\theta \log p(X|\theta)$

as log is monotonically increasing



Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Strategy: find the gradients of $L(\theta)$ and try to set them to zero.

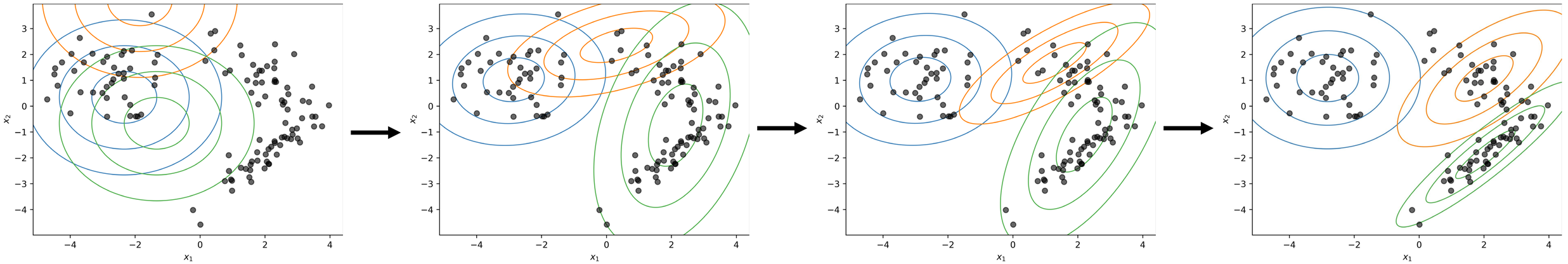
GMMs - log-likelihood gradients

Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Strategy: find the gradients of $L(\theta)$ and try to set them to zero.

$$\frac{\partial}{\partial \mu_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \mu_k} = \mathbf{0}^\top \quad \frac{\partial}{\partial \Sigma_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \Sigma_k} = \mathbf{0} \quad \frac{\partial}{\partial \pi_k} L = \sum_{n=1}^{N} \frac{\partial \log p(x_n|\theta)}{\partial \pi_k} = 0$$

Unlike linear regression, the optimal parameters are not analytically tractable. We resort to an iterative optimisation procedure.



Initialisation

After 50 iterations

GMMs - iterative procedure

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

$$\pi_k = \frac{N_k}{N}$$

GMMs - responsibility definition

We define **responsibility** r_{nk} for every data point x_n:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

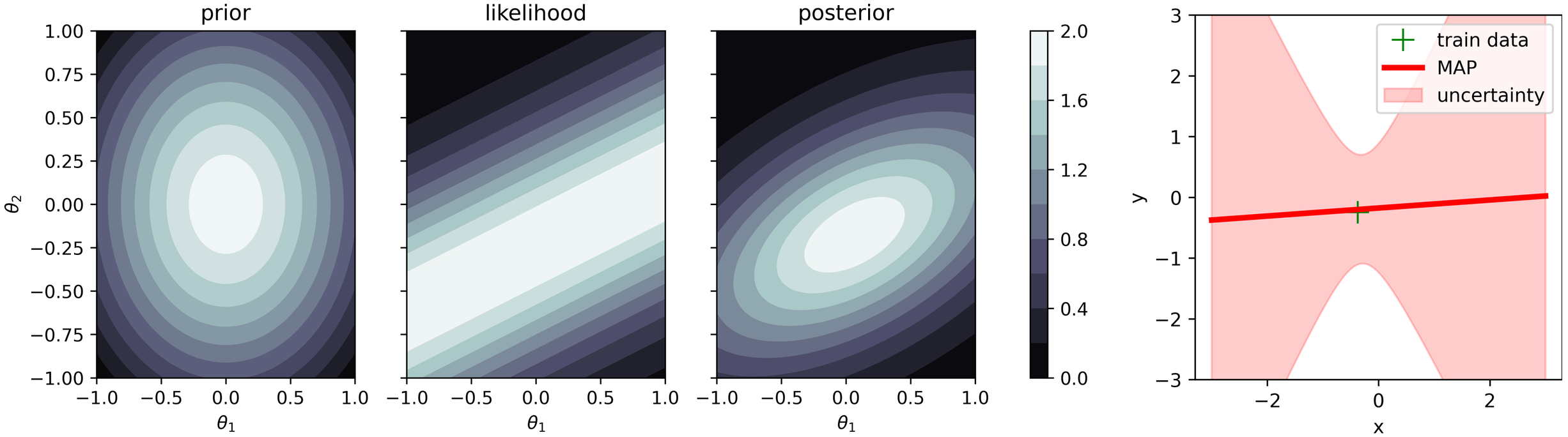
proportional to the likelihood of the k-th mixture component given the data point x_n

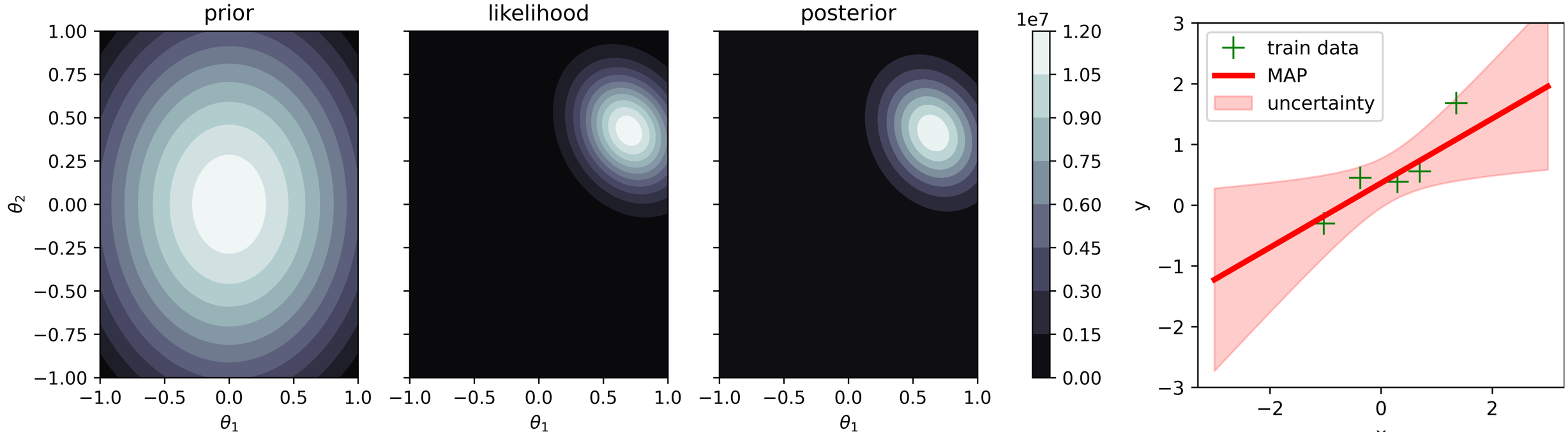
Note that $\mathbf{r}_n = [r_{n1}, r_{n2}, \ldots, r_{nK}]^\intercal \in \mathbb{R}^D$ is a probability vector, i.e., $\sum_k r_{nk} = 1 \, with$ $r_{nk} \geq 0$

This probability vector distributes probability mass among the K mixture components, and we can think of $\mathbf{r}_n$ as a "soft assignment" of x_n to the K mixture components.

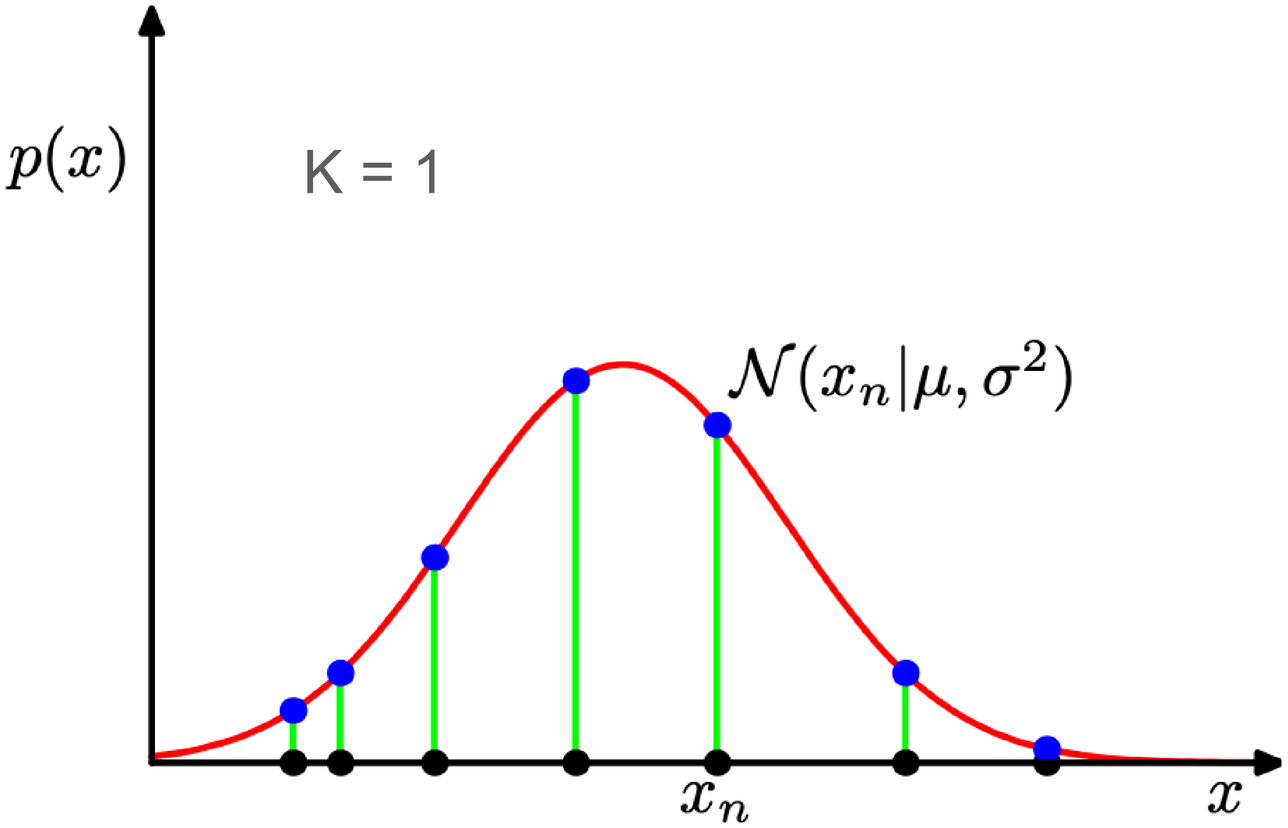
Let's now derive the parameter updates in step 2!

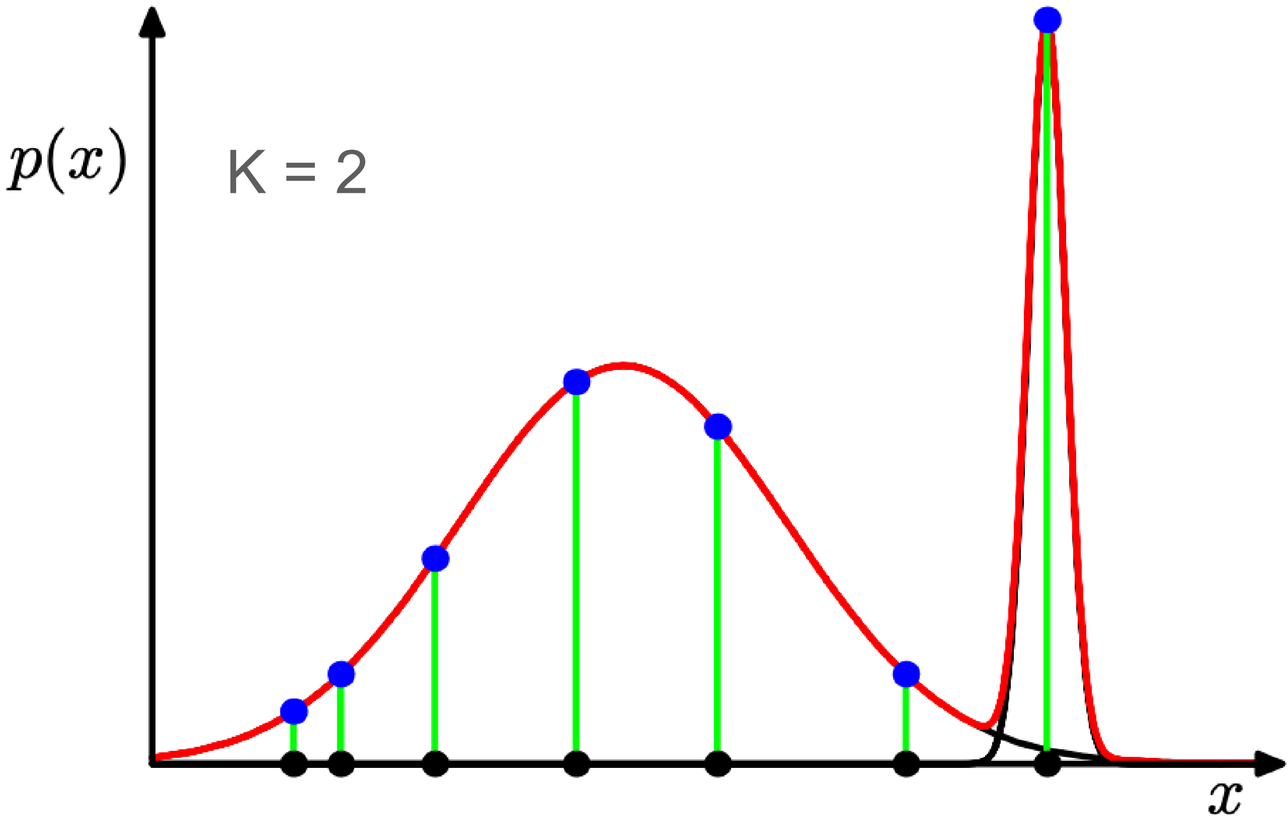
More on Bayesian linear regression





Maximum likelihood - overfitting



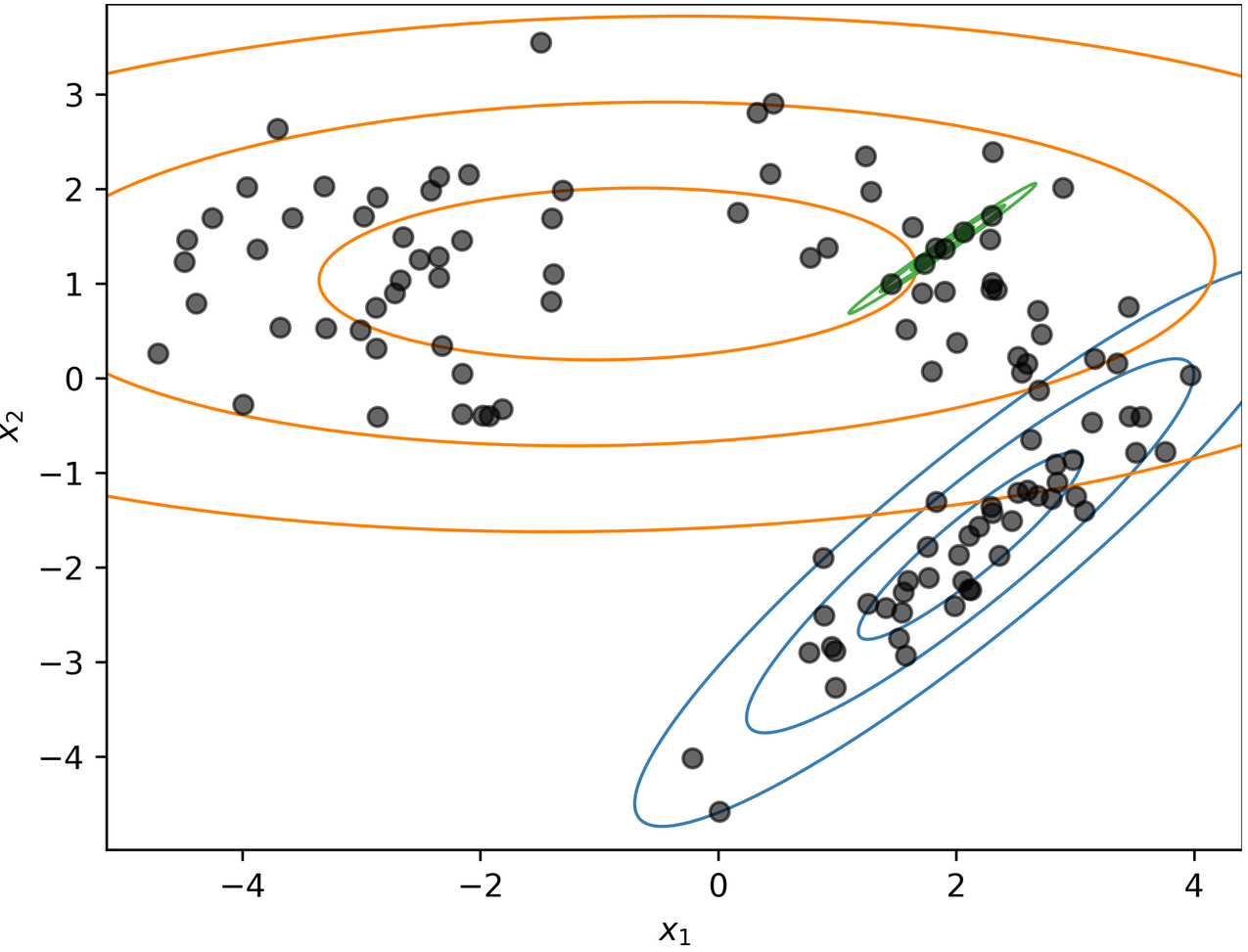


Hot-fix: reset the mean to a randomly chosen value, and the covariance to some large value and hope we don't have to reset again

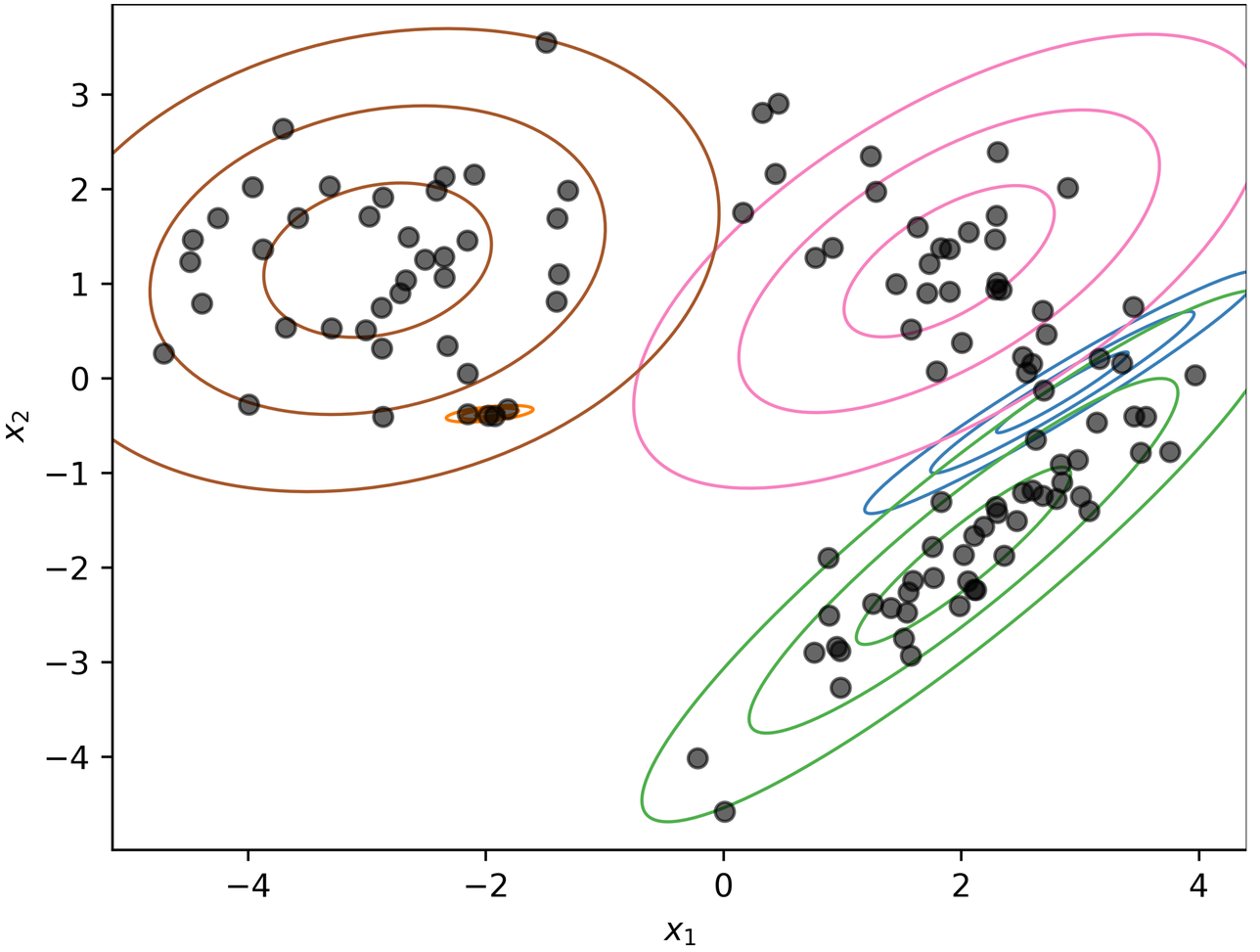
$$p_2(x_7|\mu_2, \sigma_2^2) = \mathcal{N}(x_7; x_7, \sigma_2^2) = \frac{1}{\sqrt{2\pi}}\frac{1}{\sigma_2}$$

One component collapses to a data point!

Local minima



K = 3



K = 5

Practical initialisation: k-means, mean/cov = sample mean/cov, mixture weights = proportions

Relation to k-means

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

Assume $\Sigma_1 = \Sigma_2 = \cdots = \Sigma_K = \epsilon \mathrm{I}$

Numerical example: $x_n = 5$, K = 3

μ_k	π_k	$\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$	r_{nk}
8	0.3	0.02	0.27
-2	0.4	0.00016	0.01
4	0.3	0.0528	0.72

Soft assignment

$\epsilon = 4$

μ_k	π_k	$\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$	r_{nk}
8	0.3	0.44×10^{-195}	0
-2	0.4	1.5×10^{-1064}	0
4	0.3	0.23×10^{-22}	1

Hard assignment

$\epsilon = 0.01$

GMMs - iterative procedure = *EM* algorithm

Initialise $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Repeat until convergence:

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ using the current responsibilities r_{nk}:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} x_n, N_k = \sum_{n=1}^N r_{nk}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

$$\pi_k = \frac{N_k}{N}$$

M step

Expectation maximisation (EM) algorithm*

Log-likelihood: $L(\theta) = \log p(X|\theta) = \sum_{n=1}^{N} \log p(x_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)$

Goal: maximum likelihood, $\text{argmax}_\theta p(X|\theta)$, which is equivalent to $\text{argmax}_\theta \log p(X|\theta)$

Assume: **latent/unknown/hidden variables** $\{z_n\}_{n=1}^{N}$ representing *component assignment*

$$L(\theta) = \log p(X|\theta) = \log \int p(X, z|\theta) \mathrm{d}z = \log \int q(z) \frac{p(X, z|\theta)}{q(z)}$$

$$= \mathcal{F}(q(z), \theta)$$

An arbitrary $q(z)$

Instead of maximising $L(\theta)$ directly, we will maximise the lower bound $\mathcal{F}(q(z), \theta)$, alternating between $q(z) and \theta$ while keeping the other fixed.

EM for GMMs

Maximising lower bound wrt $q(z)$ whilst keeping θ fixed

$$q(z)^{(t+1)} \leftarrow \text{argmax}_{q(z)} \mathcal{F}(q(z), \theta^{(t)}) = p(z|X, \theta^{(t)})$$

fill in values for the hidden variables according to their posterior probabilities

1. **Evaluate responsibilities** r_{nk} for every data point x_n using current parameters:

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n; \mu_j, \Sigma_j)}$$

E step

Maximising lower bound wrt θ whilst keeping $q(z)$ fixed

$$\theta^{(t+1)} \leftarrow \text{argmax}_{\theta} \mathcal{F}(q(z)^{t+1}, \theta)$$

learn model as if hidden variables were not hidden

2. **Re-estimate parameters** $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ using the current responsibilities r_{nk}:

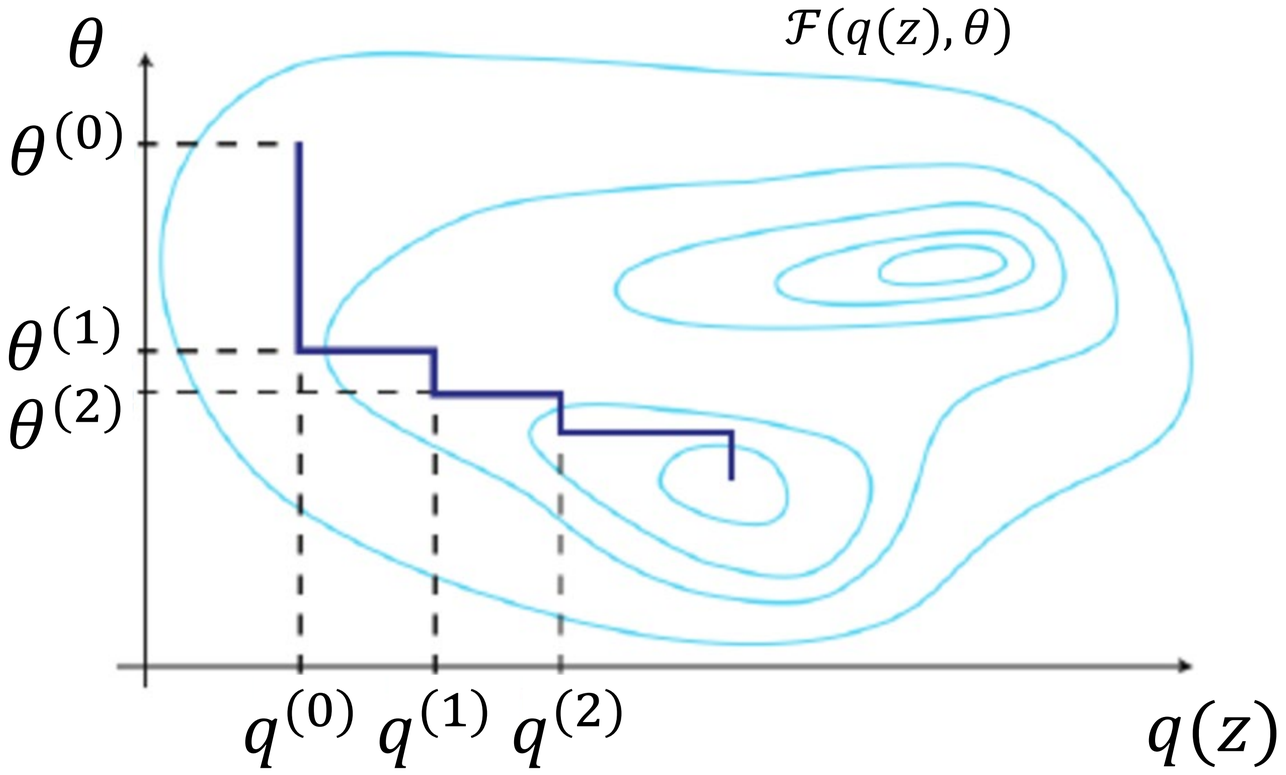
$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n, N_k = \sum_{n=1}^{N} r_{nk}$$

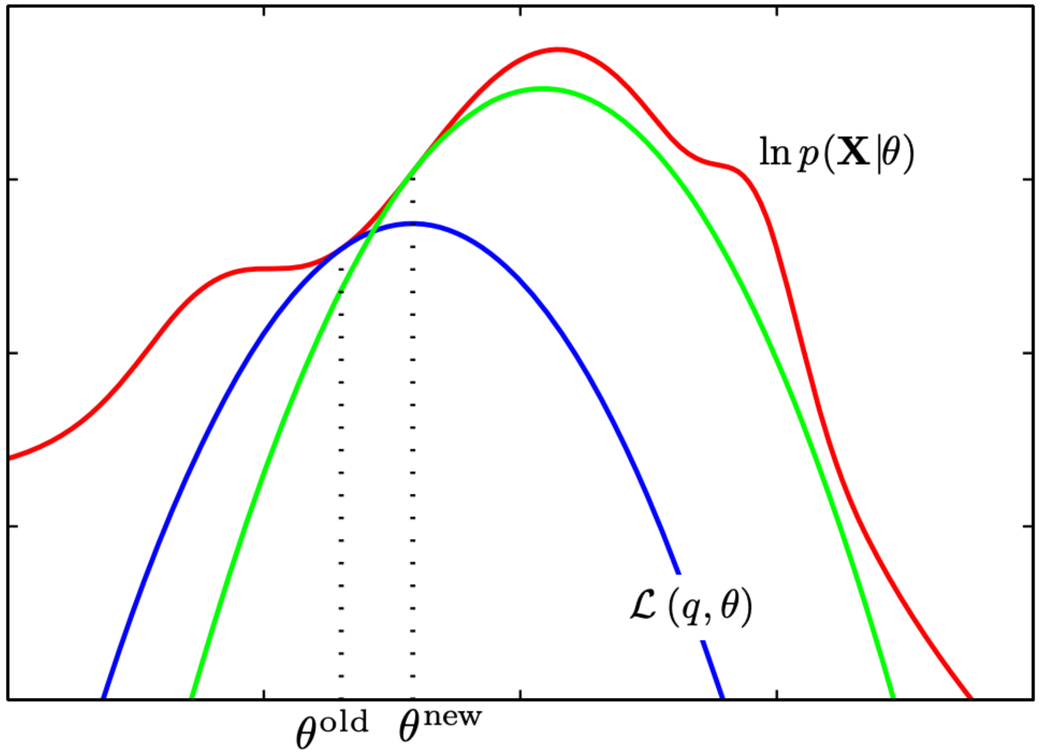
$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)(x_n - \mu_k)^\intercal$$

$$\pi_k = \frac{N_k}{N}$$

M step

EM algorithm - visualisation





EM algorithm - more visualisation

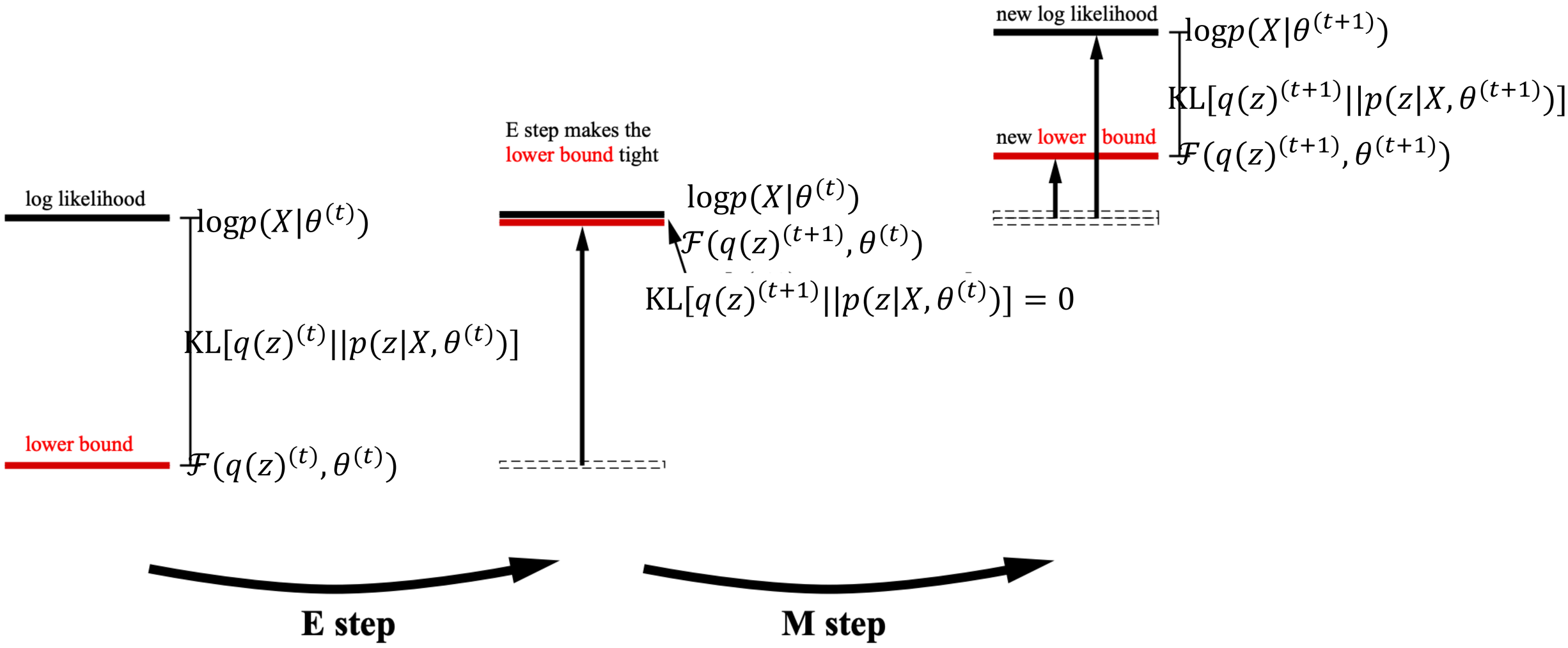


Figure credit: Beal (2003), Variational Algorithms for Approximate Bayesian Inference