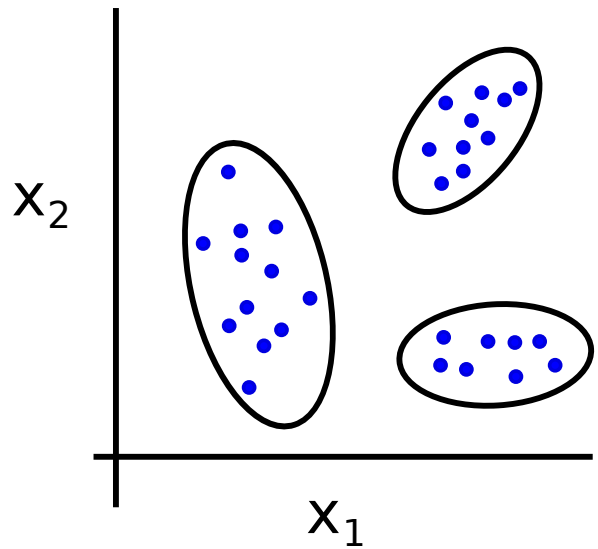
Clustering and the EM algorithm

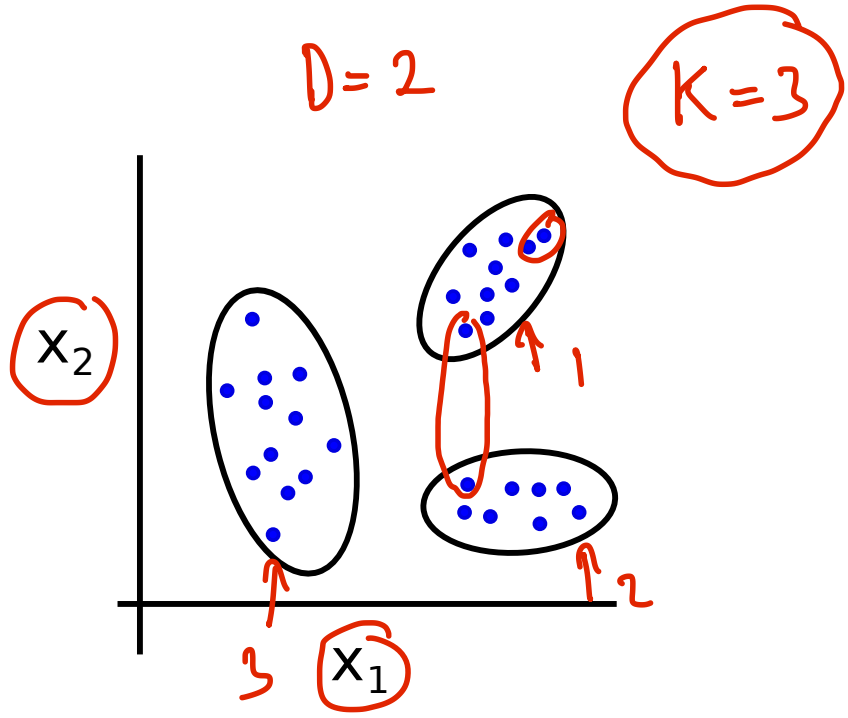
Rich Turner



What is clustering?

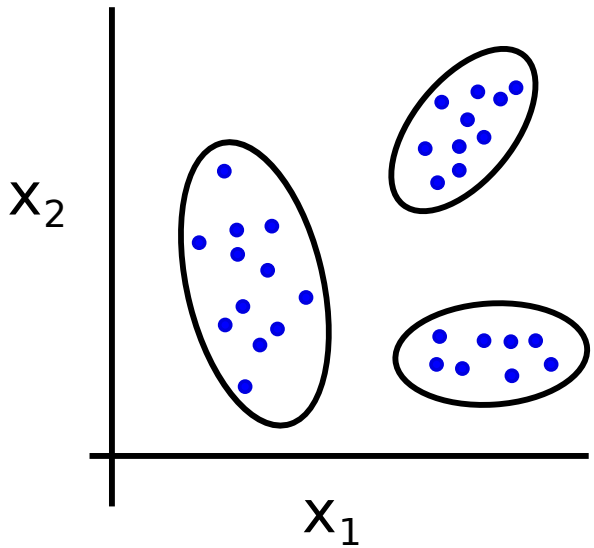
- **Roughly speaking**: two points belonging to the same cluster are generally more similar to each other or closer to each other than two points belonging to different clusters.

$\dim(\underline{x}) = D$



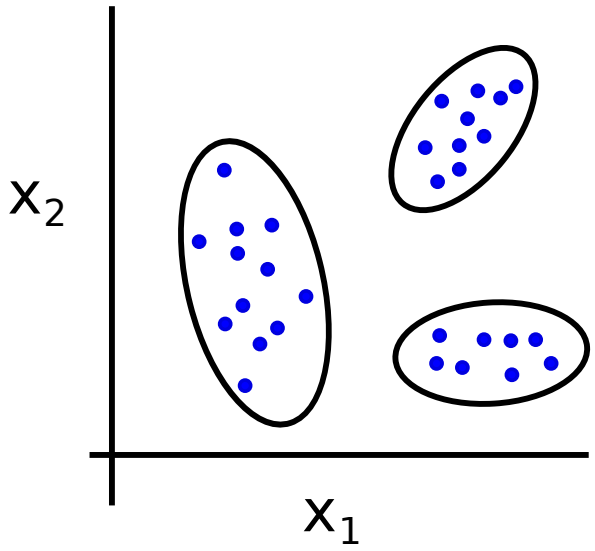
What is clustering?

- **Roughly speaking**: two points belonging to the same cluster are generally more similar to each other or closer to each other than two points belonging to different clusters.
- **Notation**: $\mathcal{D} = \{\mathrm{x}_1 \dots \mathrm{x}_N\} \rightarrow \boldsymbol{s} = \{s_1 \dots s_N\}$
  where $D = \dim(\mathrm{x}_n)$ and $s_n \in \{1 \dots K\}$

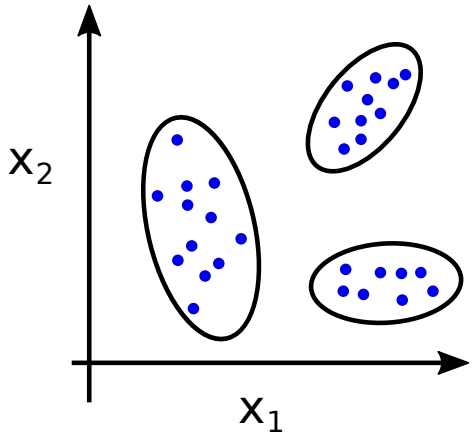


What is clustering?

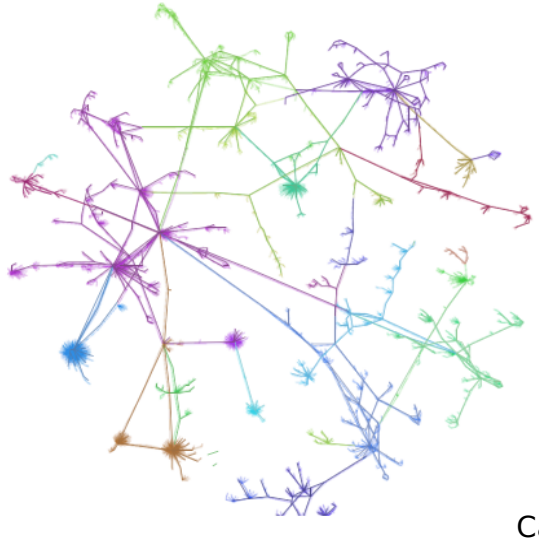
- **Roughly speaking**: two points belonging to the same cluster are generally more similar to each other or closer to each other than two points belonging to different clusters.
- **Notation**: $\mathcal{D} = \{\mathrm{x}_1 \dots \mathrm{x}_N\} \rightarrow \boldsymbol{s} = \{s_1 \dots s_N\}$
  where $D = \mathsf{dim}(\mathrm{x}_n)$ and $s_n \in \{1 \dots K\}$
- Unsupervised learning problem (no labels or rewards)



Where is clustering used?

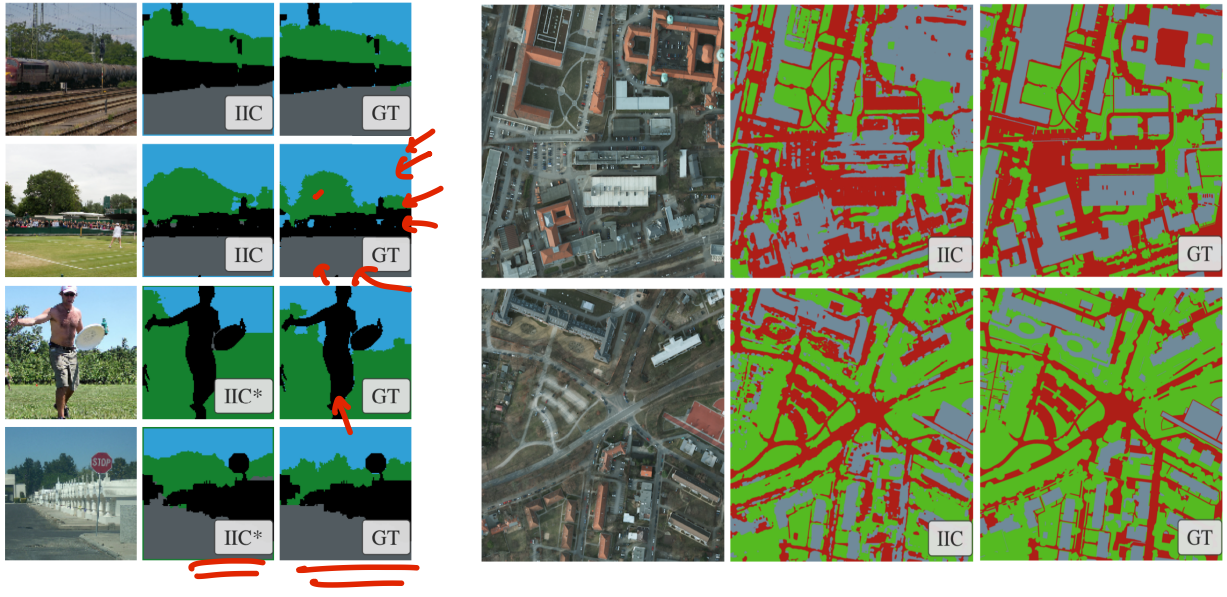


network community detection



Campbell et al
Social Network Analysis

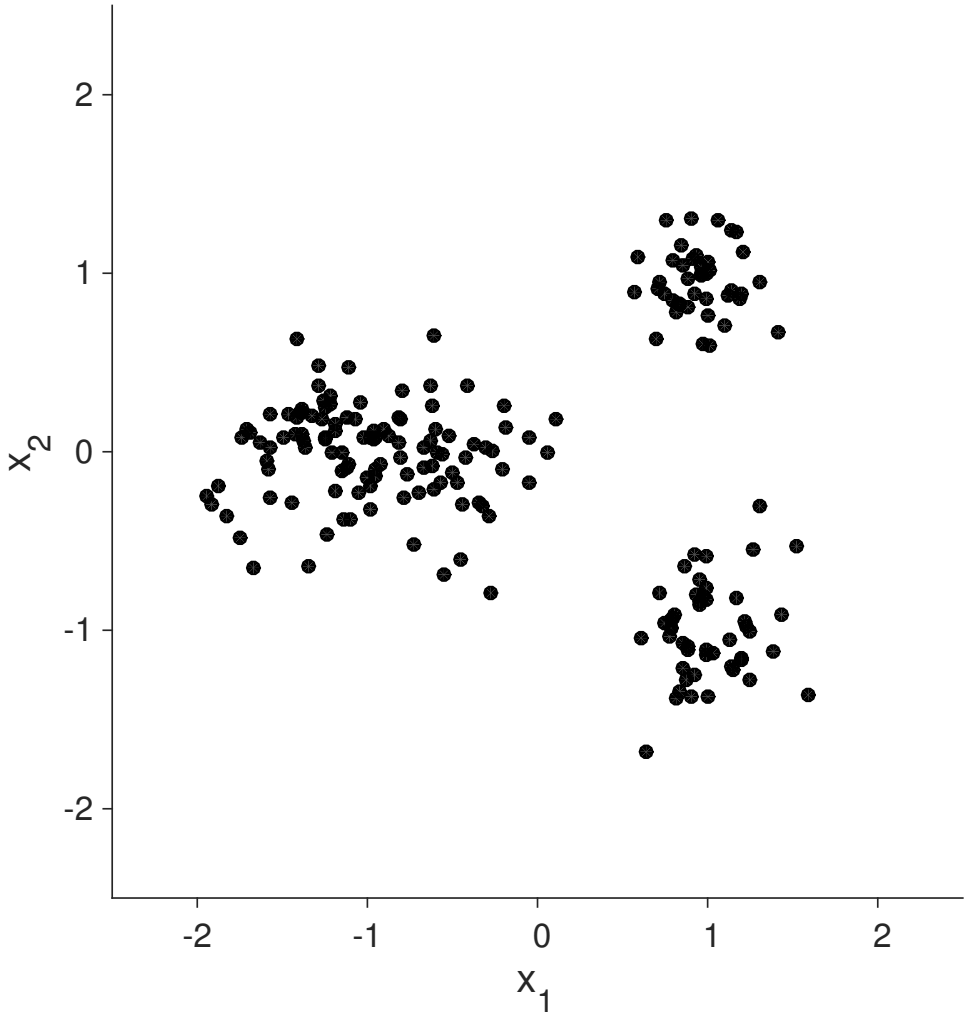
image segmentation



Ji et al Invariant Information Clustering

vector quantisation
genetic clustering
anomaly detection
crime analysis
...

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

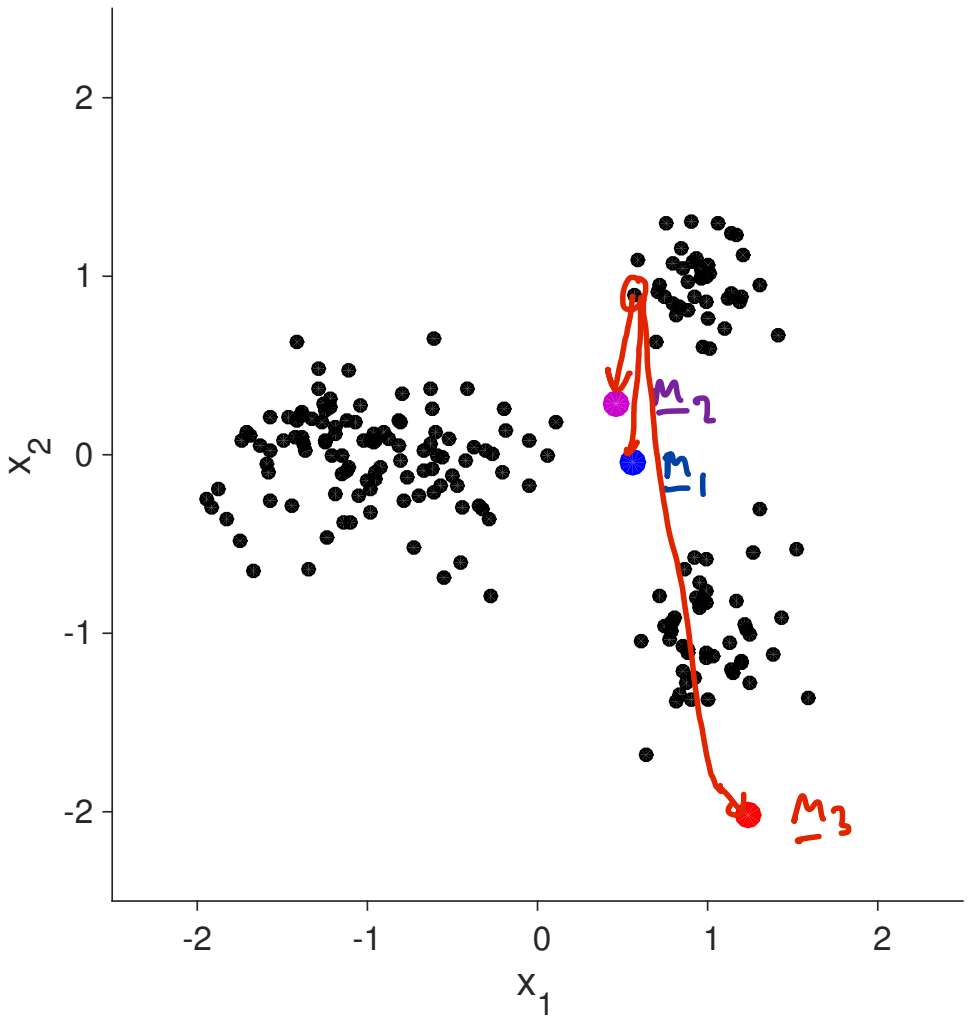
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$ assignment

$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$

endfor

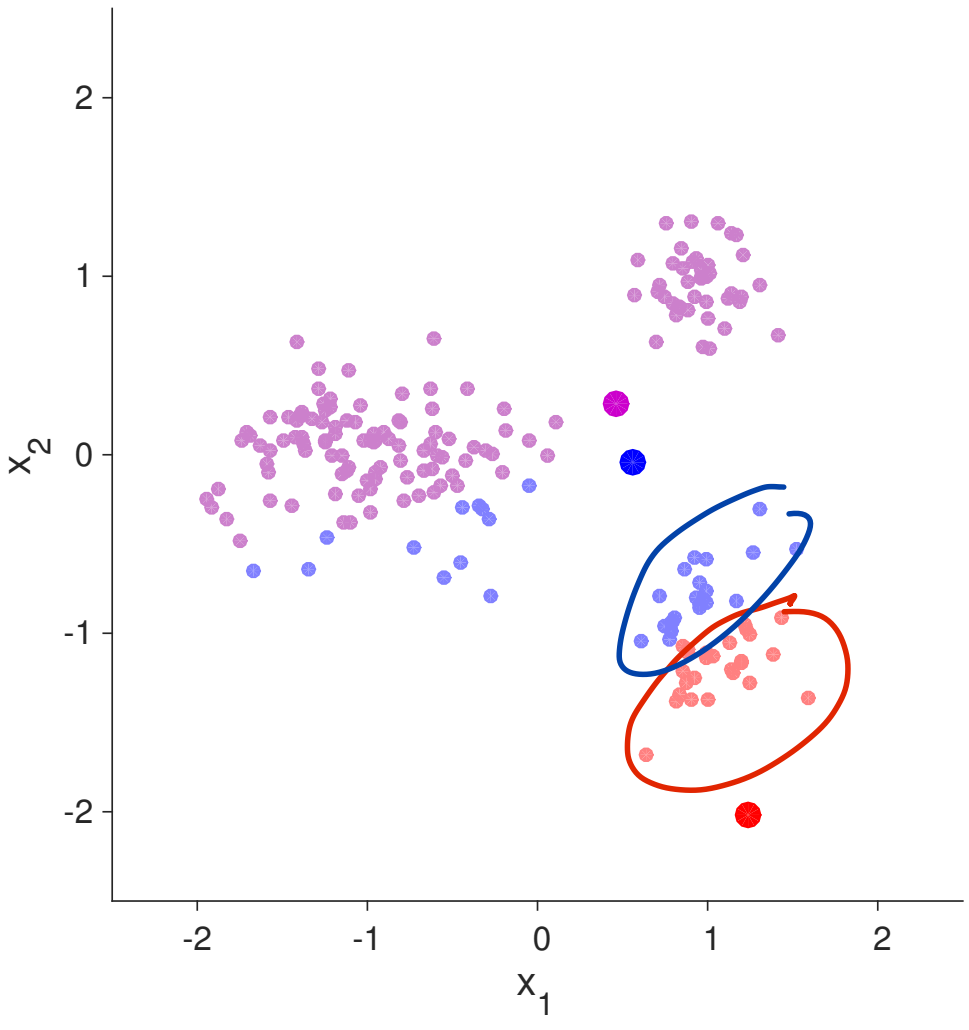
for $k = 1 \ldots K$ update

$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

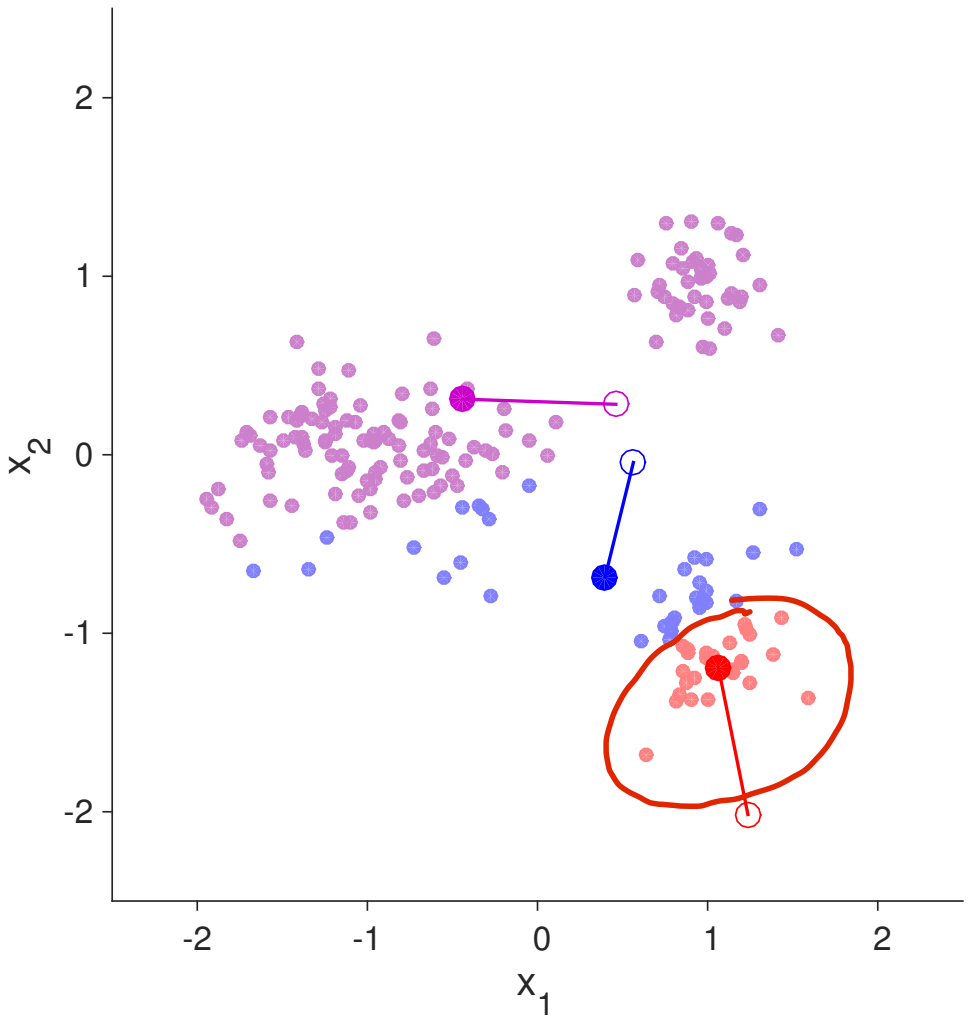
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

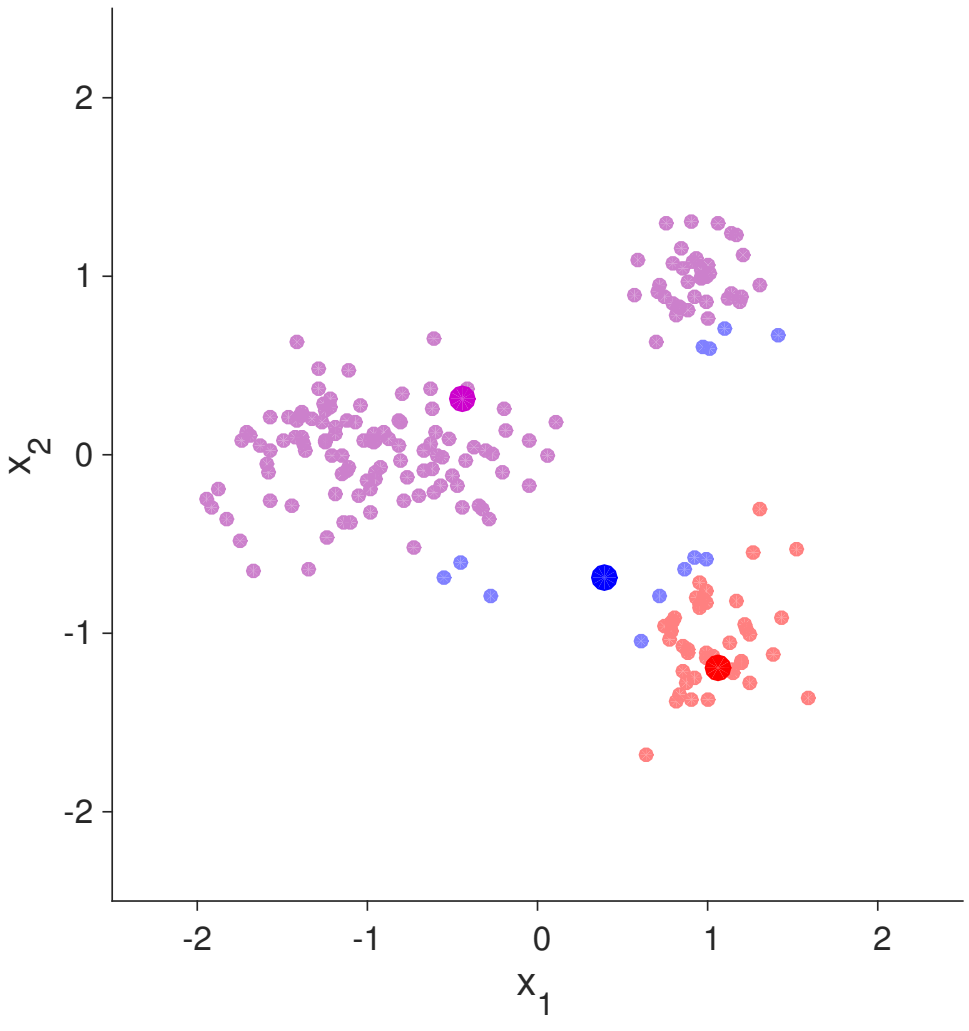
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

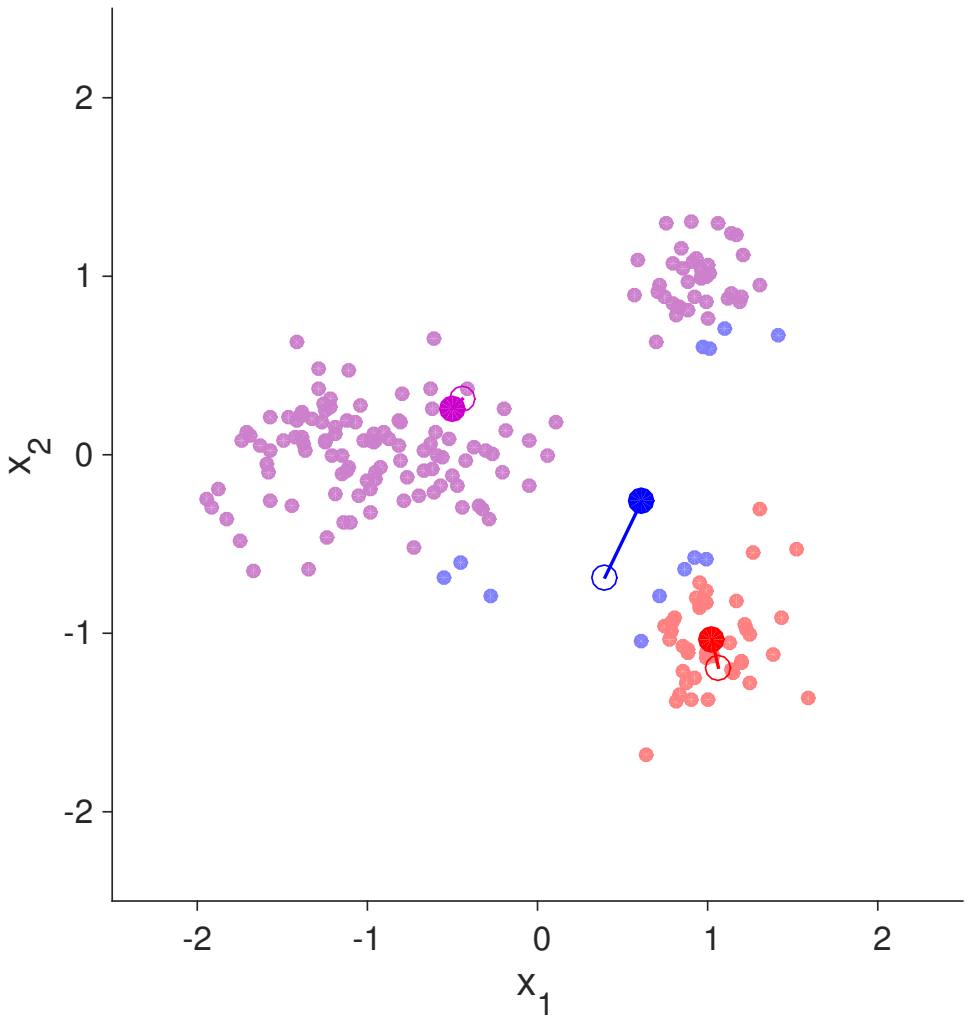
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

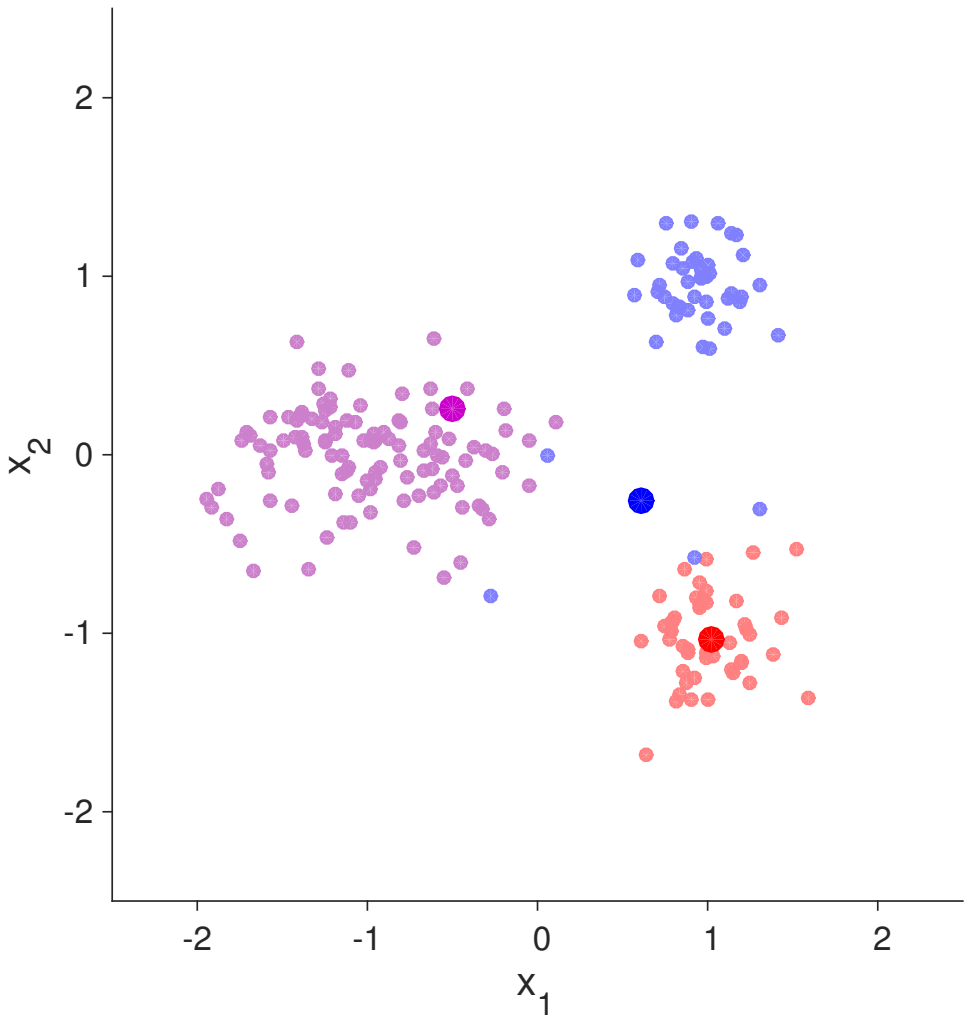
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

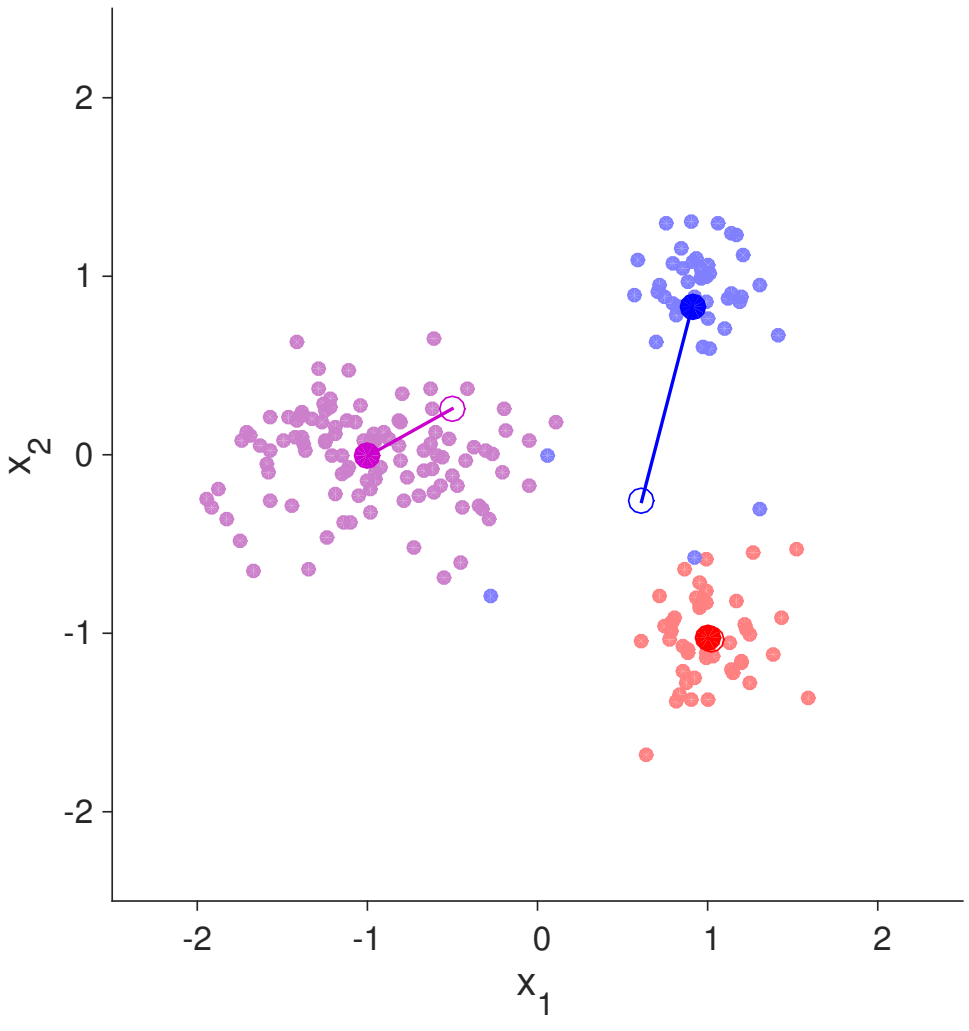
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

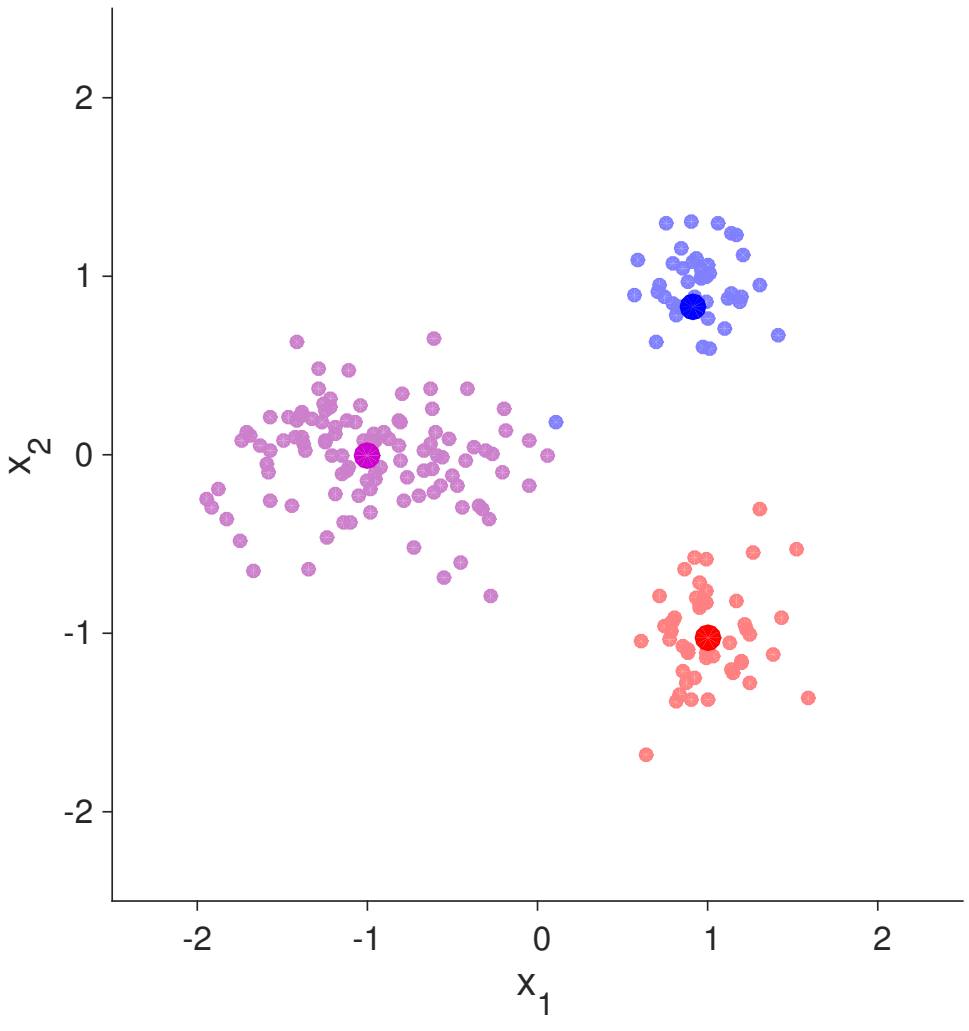
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

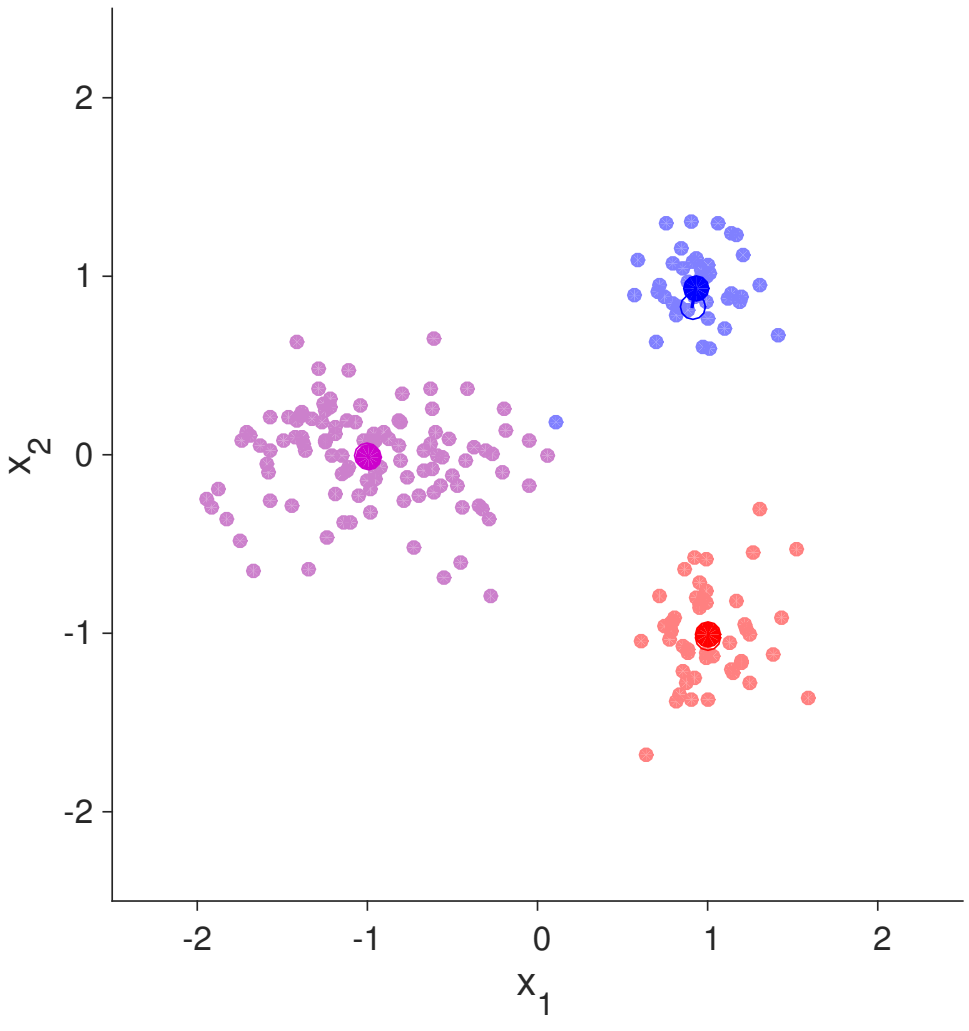
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

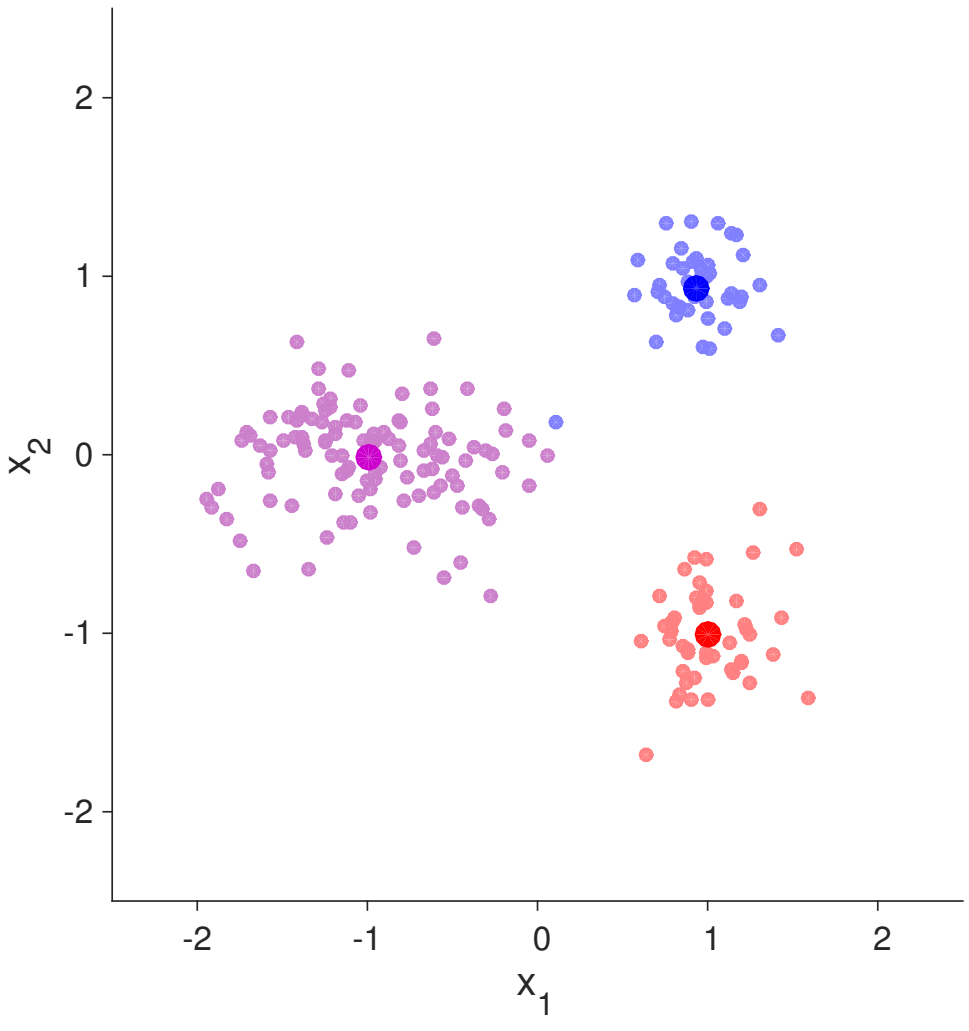
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

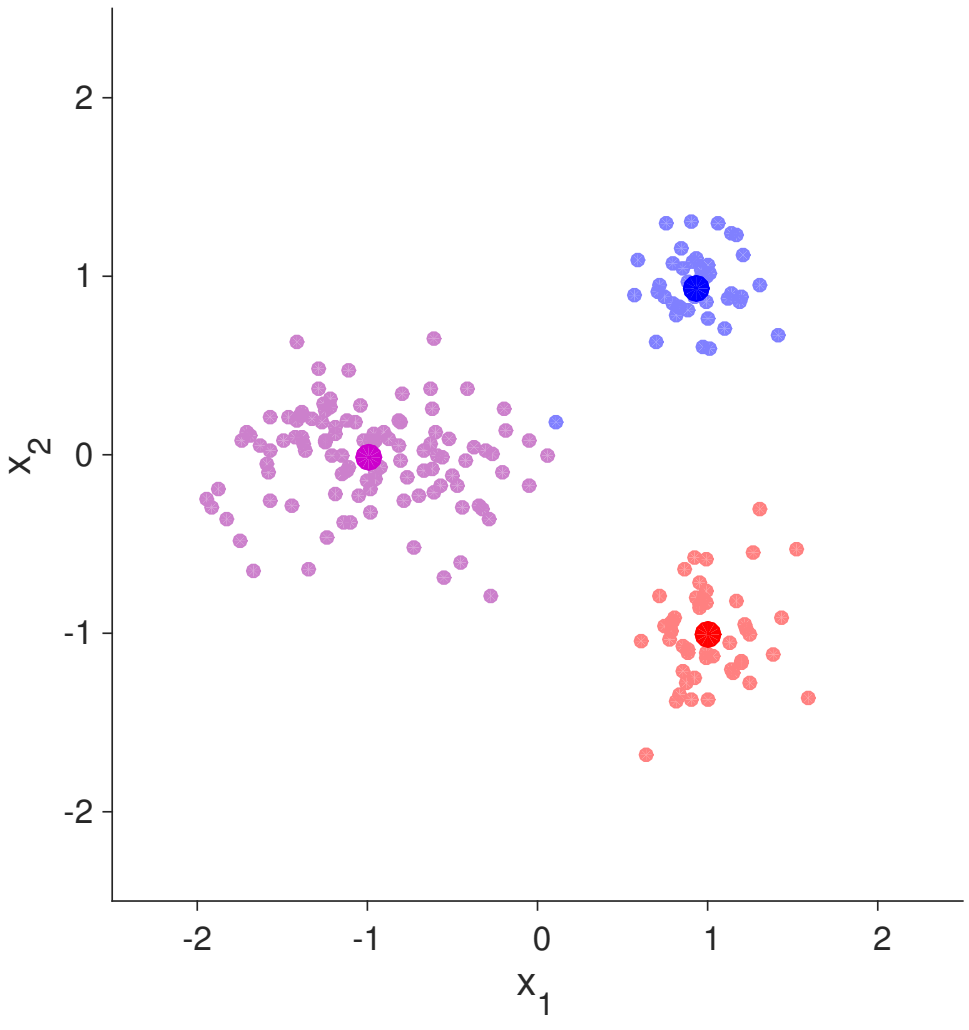
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

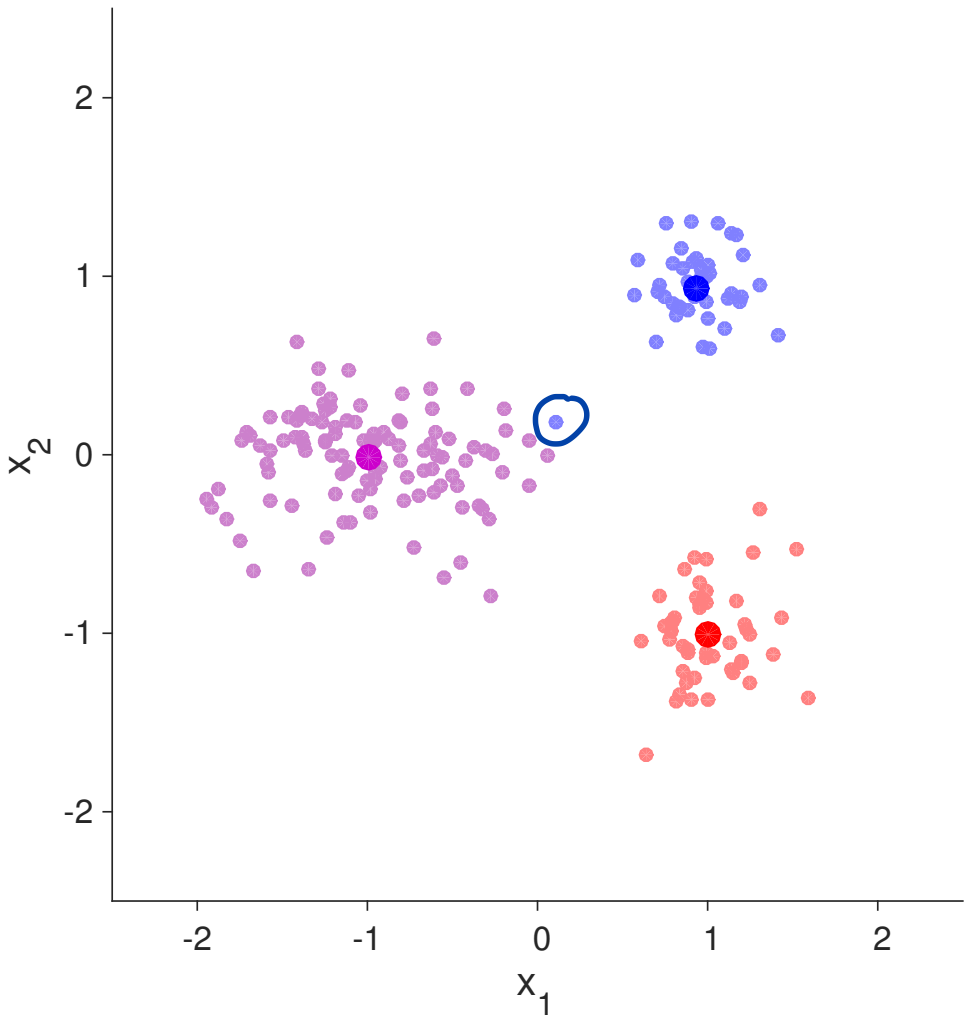
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

for $k = 1 \ldots K$

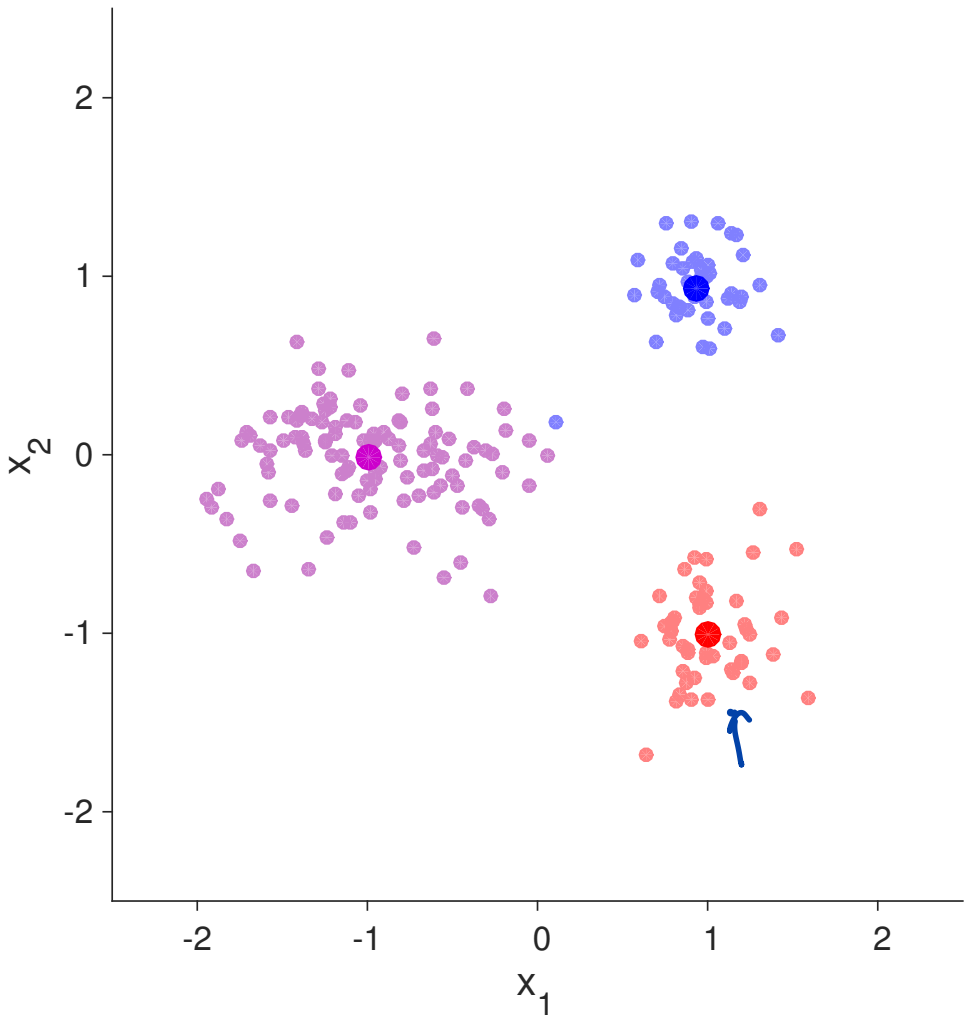
$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Question: is K-means guaranteed to converge for any dataset $\mathcal{D}$?

could one or more of the cluster centres diverge or oscillate?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

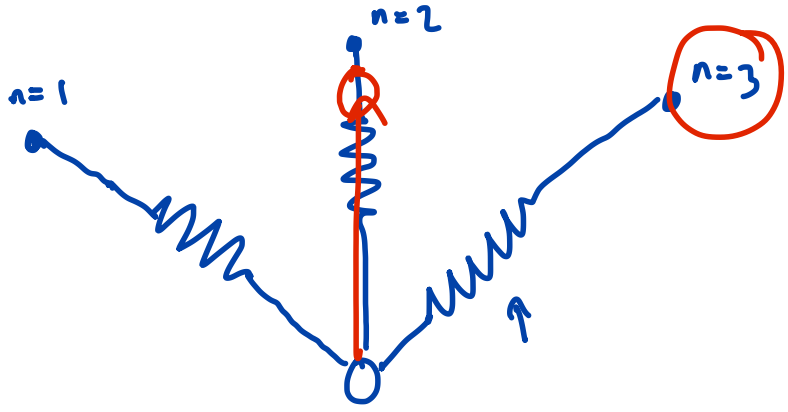
endfor

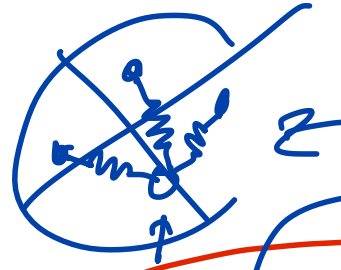
for $k = 1 \ldots K$

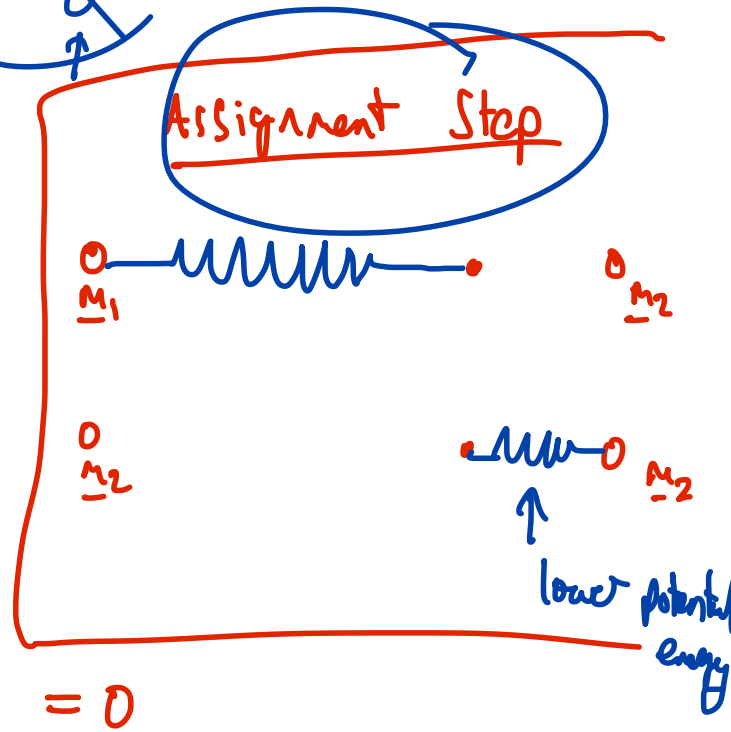
$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)







Update

Spring constant

$$\frac{d}{d\underline{m}_1} \text{P.E.} = \frac{1}{2}\rho \sum_n \| \underline{m}_1 - \underline{x}_n \|^2$$

$$\frac{d}{dm_{1,e}} \frac{1}{2}\rho \sum_{n=1}^{N} \sum_{d=1}^{D} (m_{1,d} - x_{n,d})^2 = 0$$

$$\frac{1}{2} \times 2 \times \rho \sum_n (m_{1,e} - x_{n,e}) = 0$$

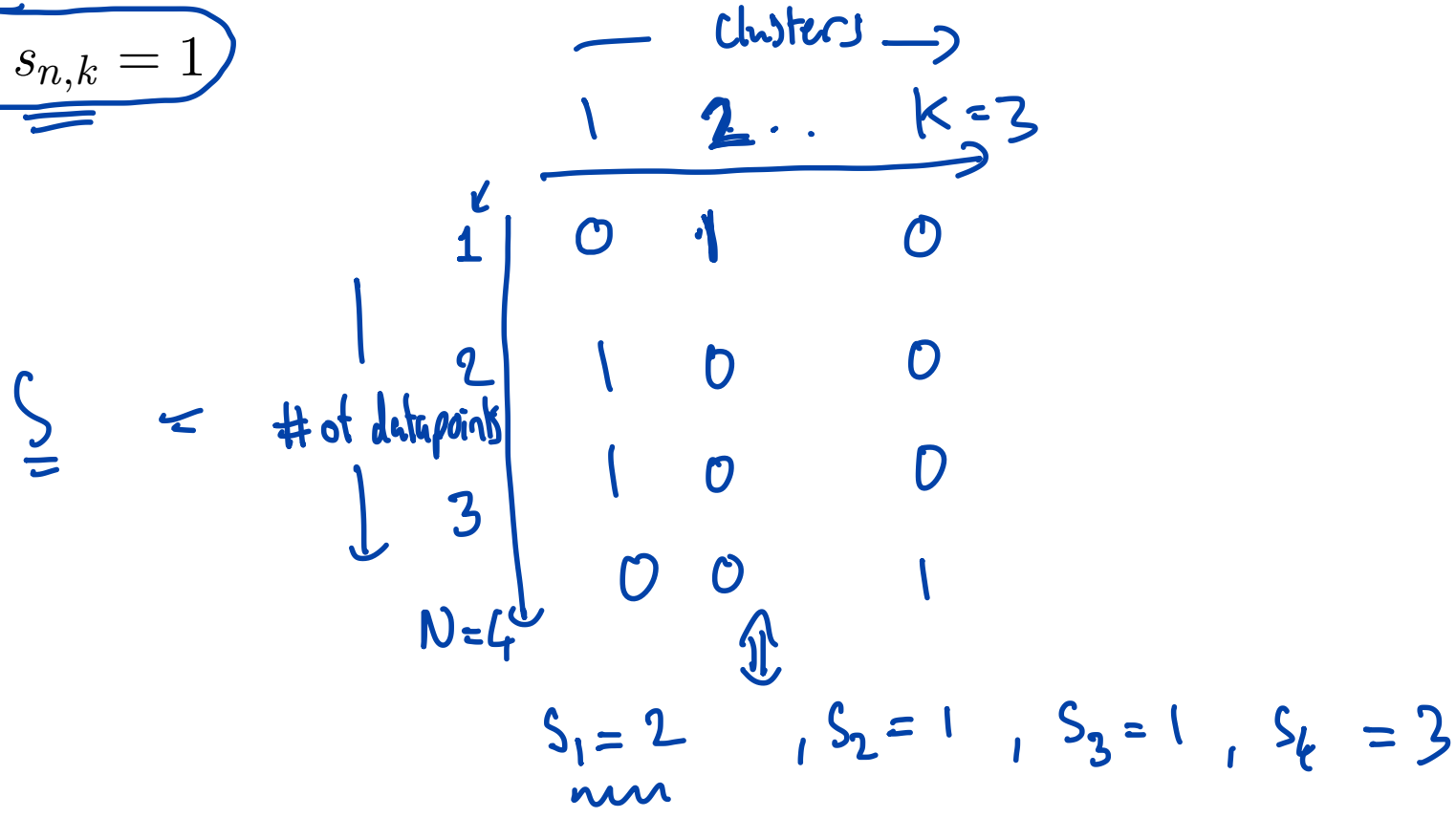
$$m_{1,e} = \frac{1}{N_{cluster}} \sum_{n=1}^{N_{cluster}} x_{n,e} \quad \Rightarrow \quad \underline{m}_1 = \frac{1}{N_{cluster}} \sum_{n=1}^{N_{cluster}} \underline{x}_n$$

K-means as optimisation

$s_n \in \{1 \dots K\}$

Let $s_{n,k} = 1$ if data point n is assigned to cluster k and zero otherwise

Note: $\sum_{k=1}^{K} s_{n,k} = 1$



K-means as optimisation

Let $s_{n,k} = 1$ if data point n is assigned to cluster k and zero otherwise

Note: $\sum_{k=1}^{K} s_{n,k} = 1$

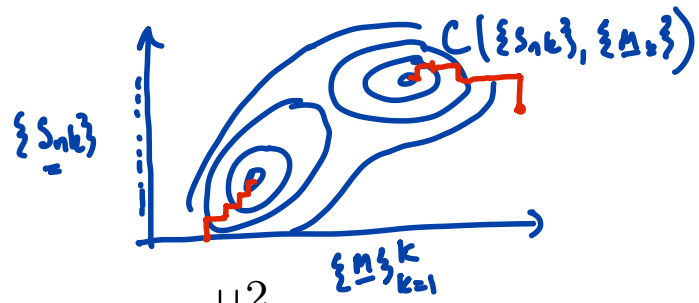
Cost:

$$\mathcal{C}(\{s_{n,k}\}, \{\boldsymbol{m}_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} s_{n,k} ||\boldsymbol{x}_n - \boldsymbol{m}_k||^2$$

K-means as optimisation

Let $s_{n,k} = 1$ if data point n is assigned to cluster k and zero otherwise

Note: $\sum_{k=1}^{K} s_{n,k} = 1$



Cost:

$$\mathcal{C}(\{s_{n,k}\}, \{\boldsymbol{m}_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} s_{n,k} ||\boldsymbol{x}_n - \boldsymbol{m}_k||^2$$

K-means tries to minimise the cost function $\mathcal{C}$ with respect to $\{s_{n,k}\}$ and $\{\boldsymbol{m}_k\}$, subject to $\sum_k s_{n,k} = 1$ and $s_{n,k} \in \{0, 1\}$

K-means sequentially:

- minimises $\mathcal{C}$ with respect to $\{s_{n,k}\}$, holding $\{\boldsymbol{m}_k\}$ fixed. assignment
- minimises $\mathcal{C}$ with respect to $\{\boldsymbol{m}_k\}$, holding $\{s_{n,k}\}$ fixed. update

K-means as optimisation

Let $s_{n,k} = 1$ if data point n is assigned to cluster k and zero otherwise

Note: $\sum_{k=1}^{K} s_{n,k} = 1$

Cost:

$$\mathcal{C}(\{s_{n,k}\}, \{\boldsymbol{m}_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} s_{n,k} ||\boldsymbol{x}_n - \boldsymbol{m}_k||^2$$

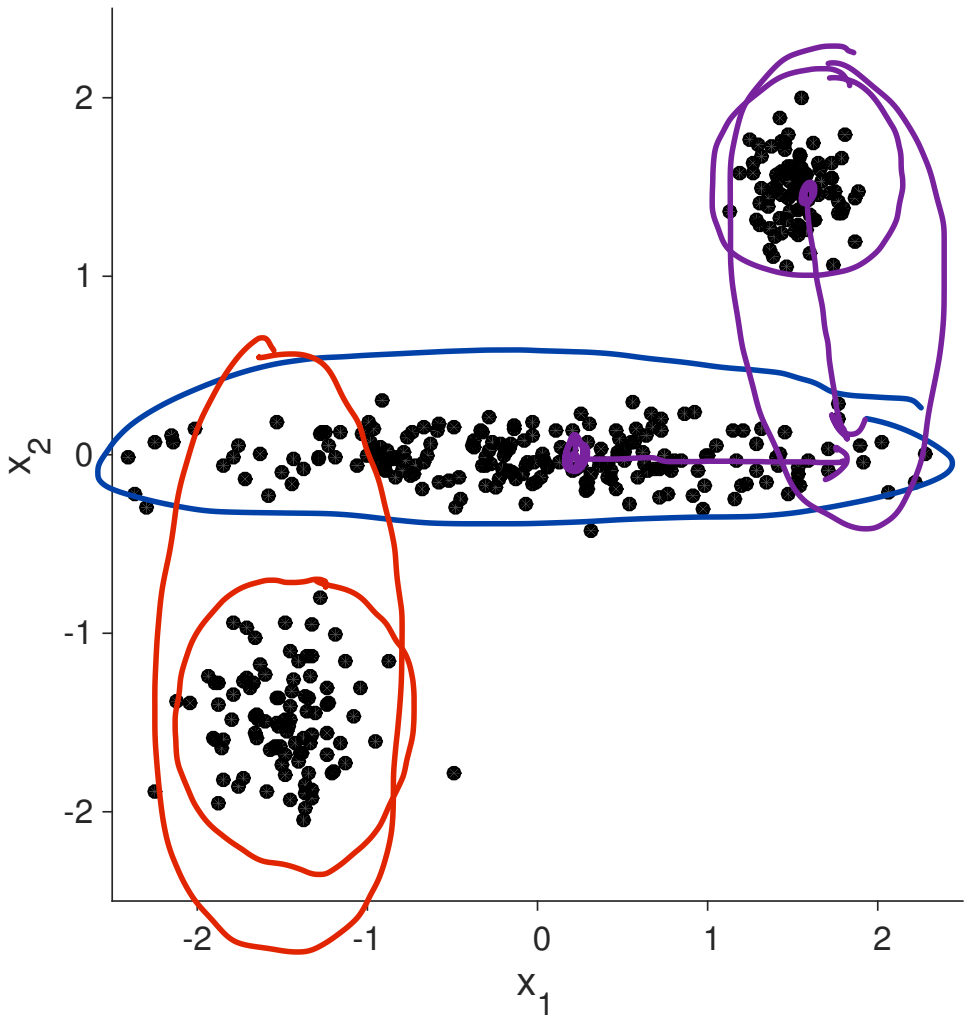
K-means tries to minimise the cost function $\mathcal{C}$ with respect to $\{s_{n,k}\}$ and $\{\boldsymbol{m}_k\}$, subject to $\sum_k s_{n,k} = 1$ and $s_{n,k} \in \{0, 1\}$

K-means sequentially:

- minimises $\mathcal{C}$ with respect to $\{s_{n,k}\}$, holding $\{\boldsymbol{m}_k\}$ fixed.
- minimises $\mathcal{C}$ with respect to $\{\boldsymbol{m}_k\}$, holding $\{s_{n,k}\}$ fixed.

$\mathcal{C}$ is a Lyupanov function $\implies$ K-means converges, but...
Finding the global optimum of $\mathcal{C}$ is a hard problem.

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

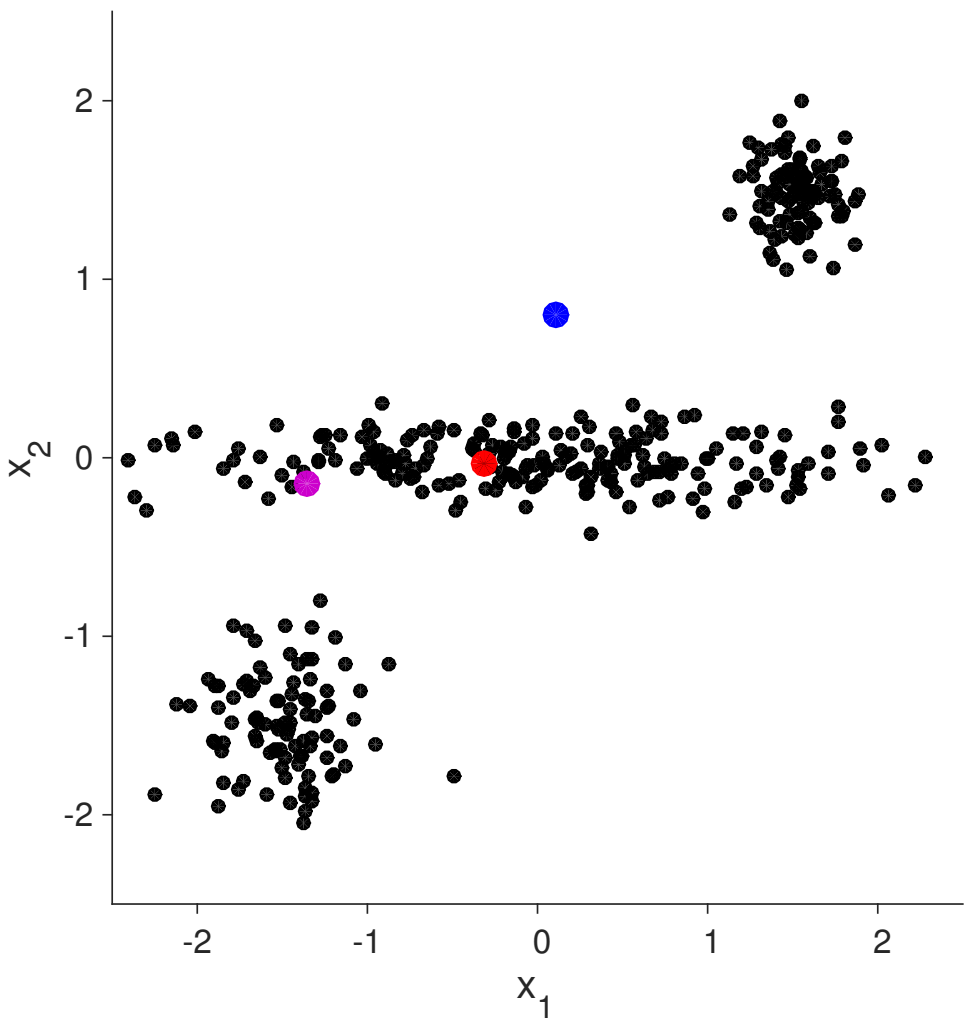
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

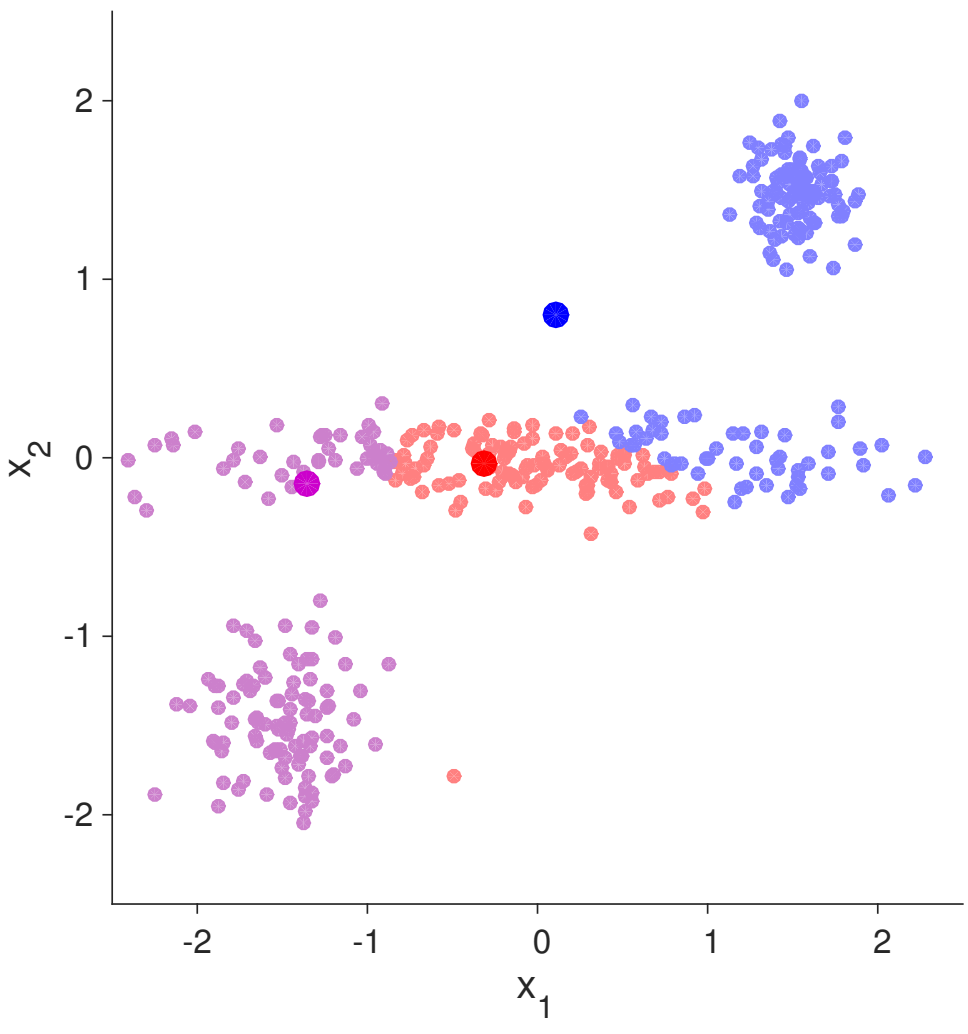
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

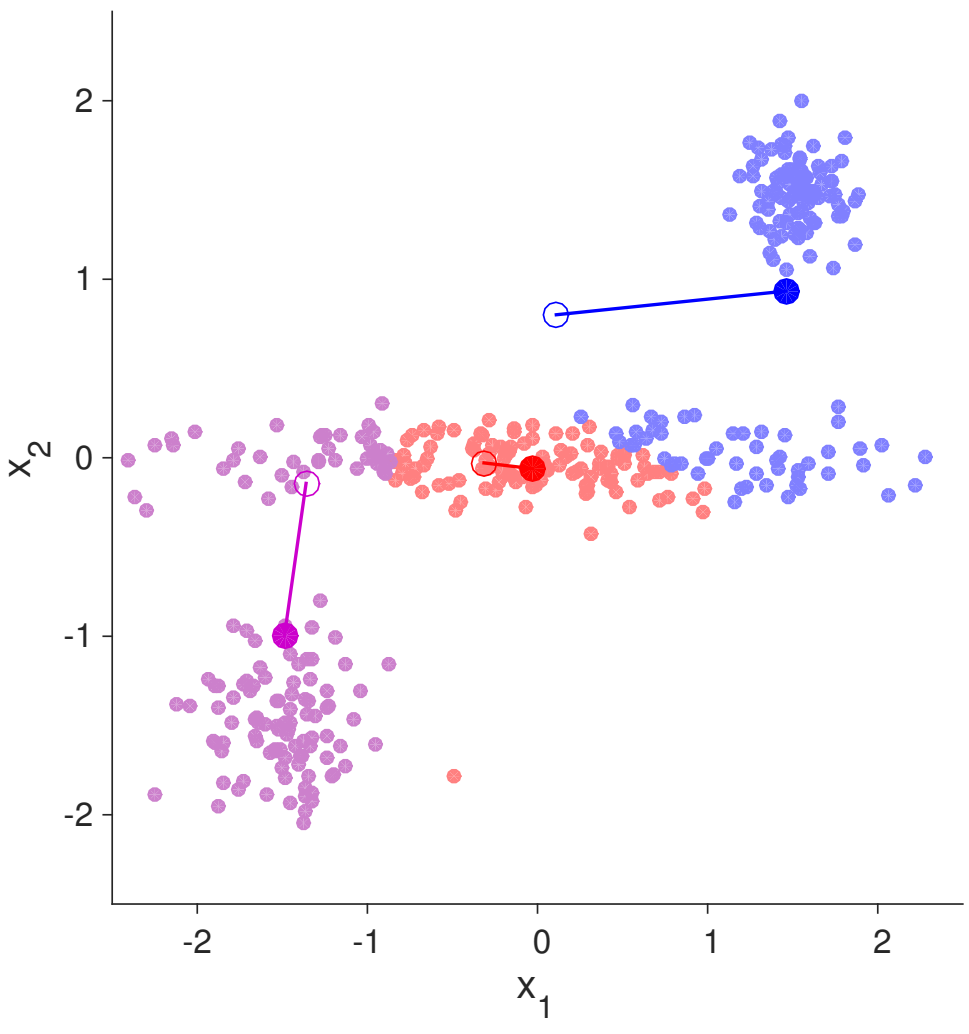
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

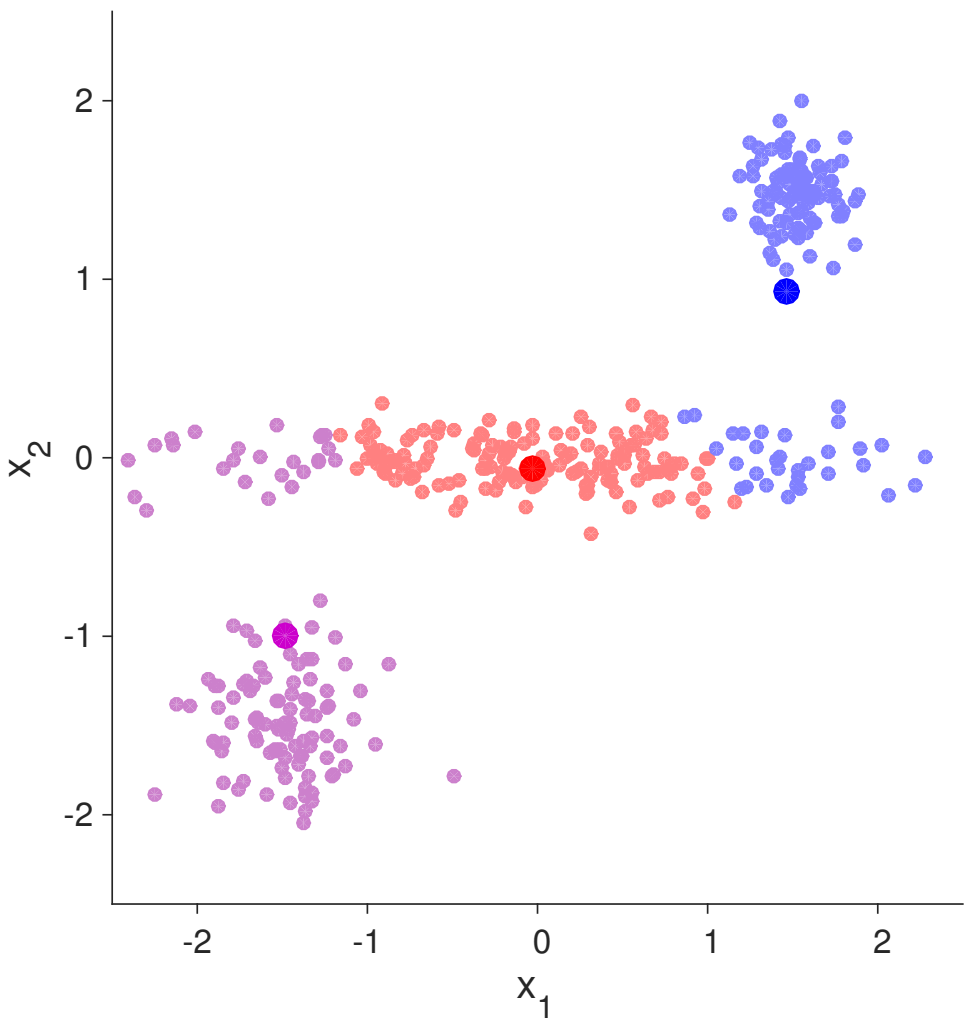
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

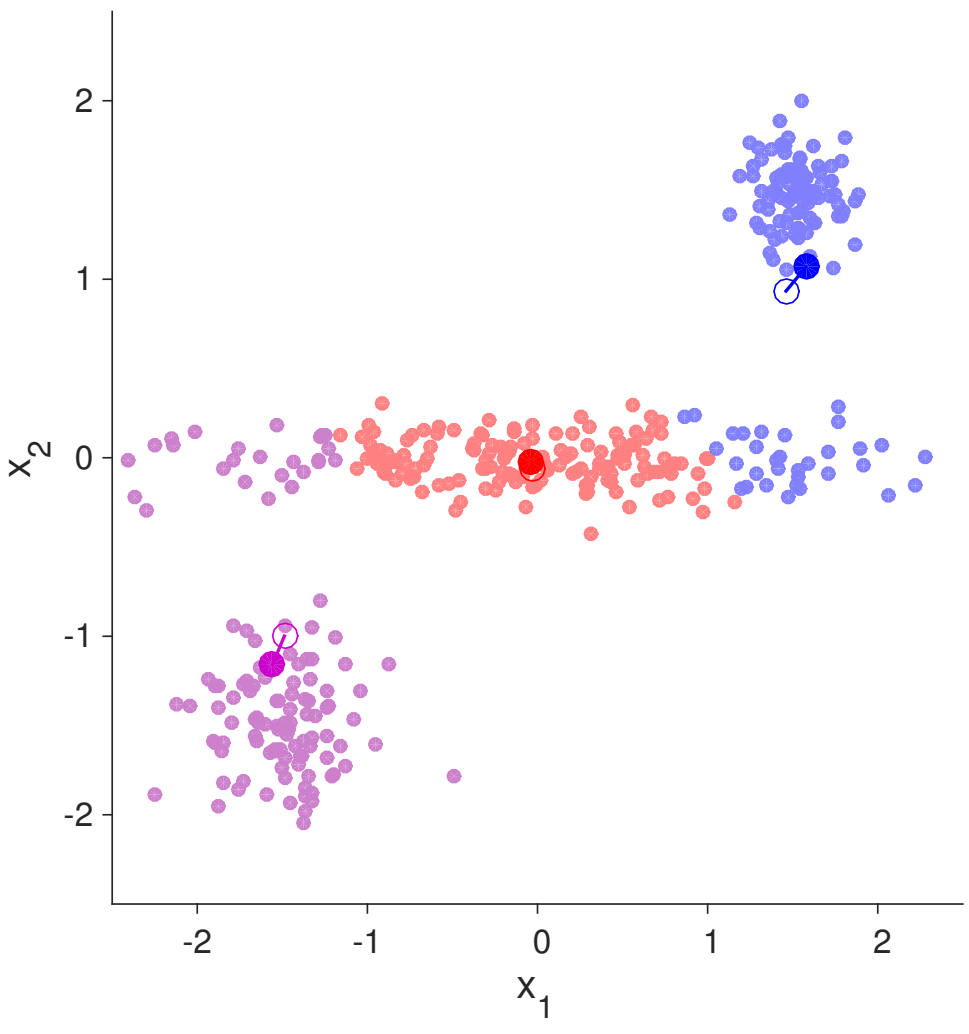
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

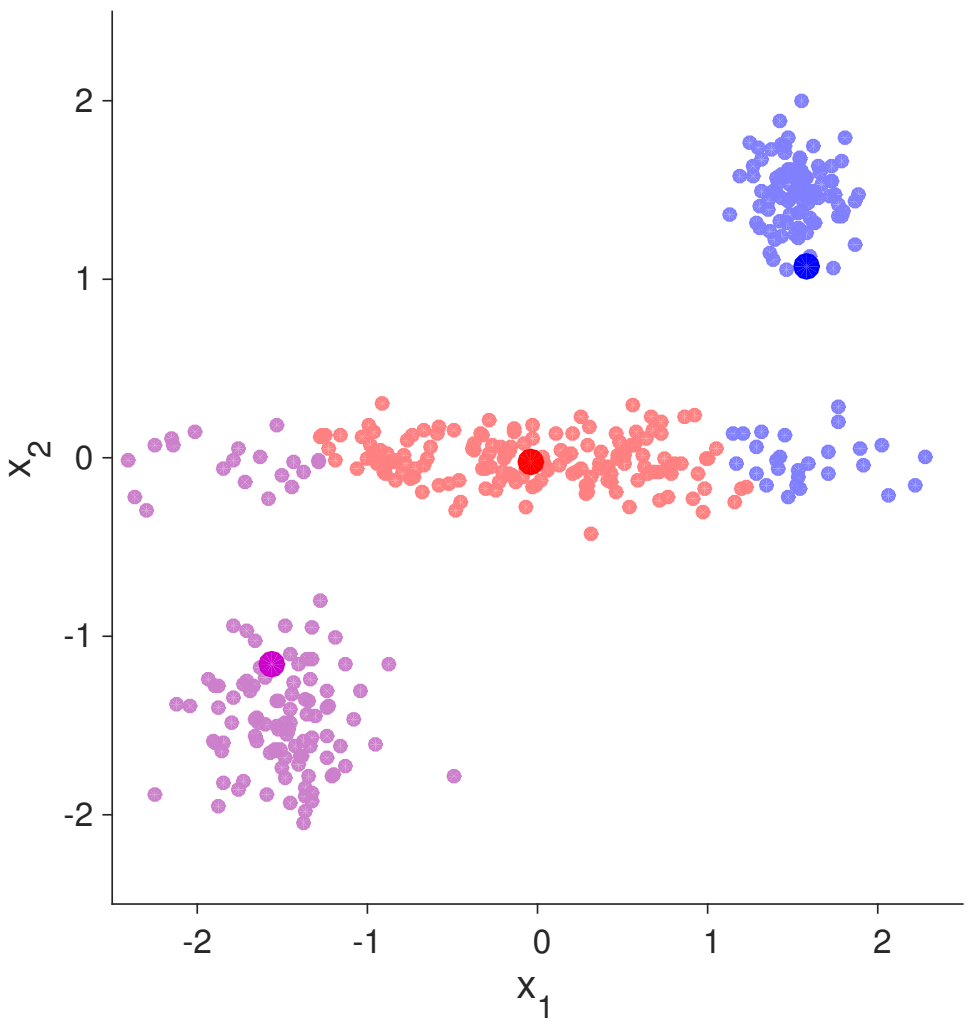
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

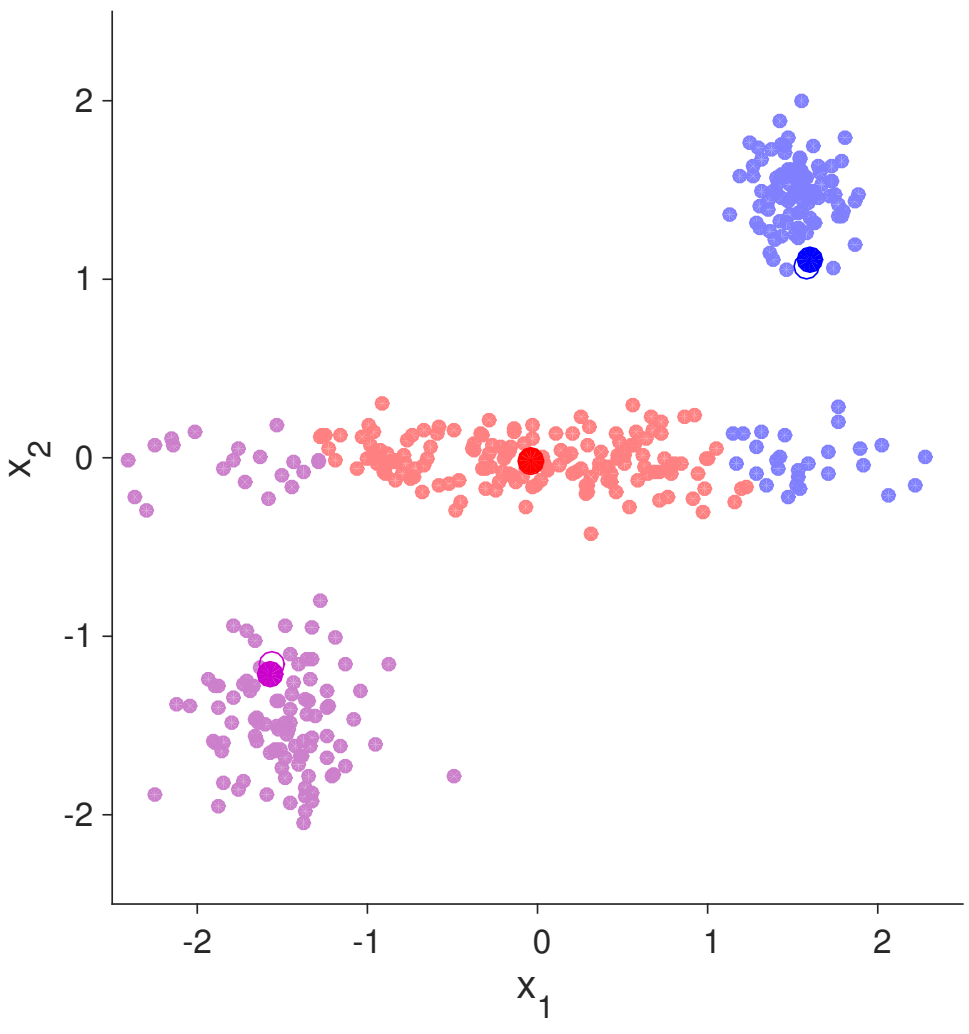
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

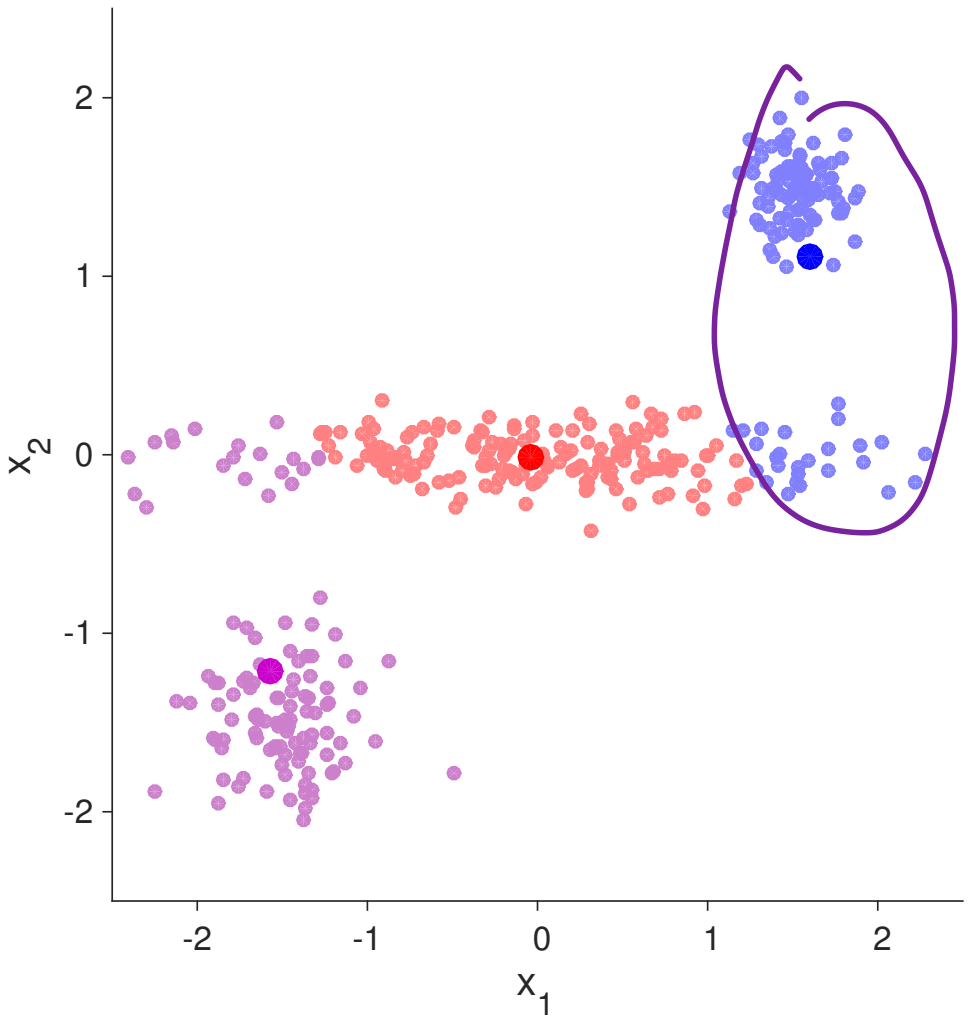
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

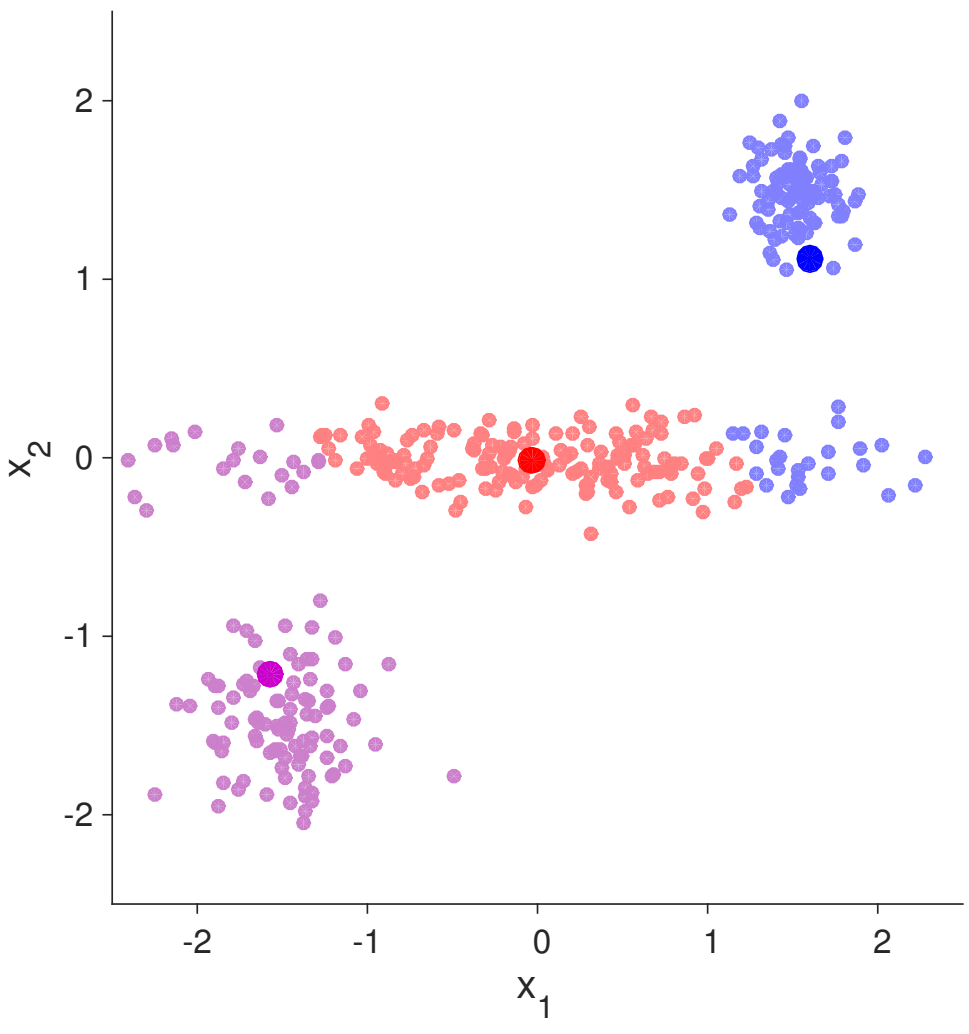
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

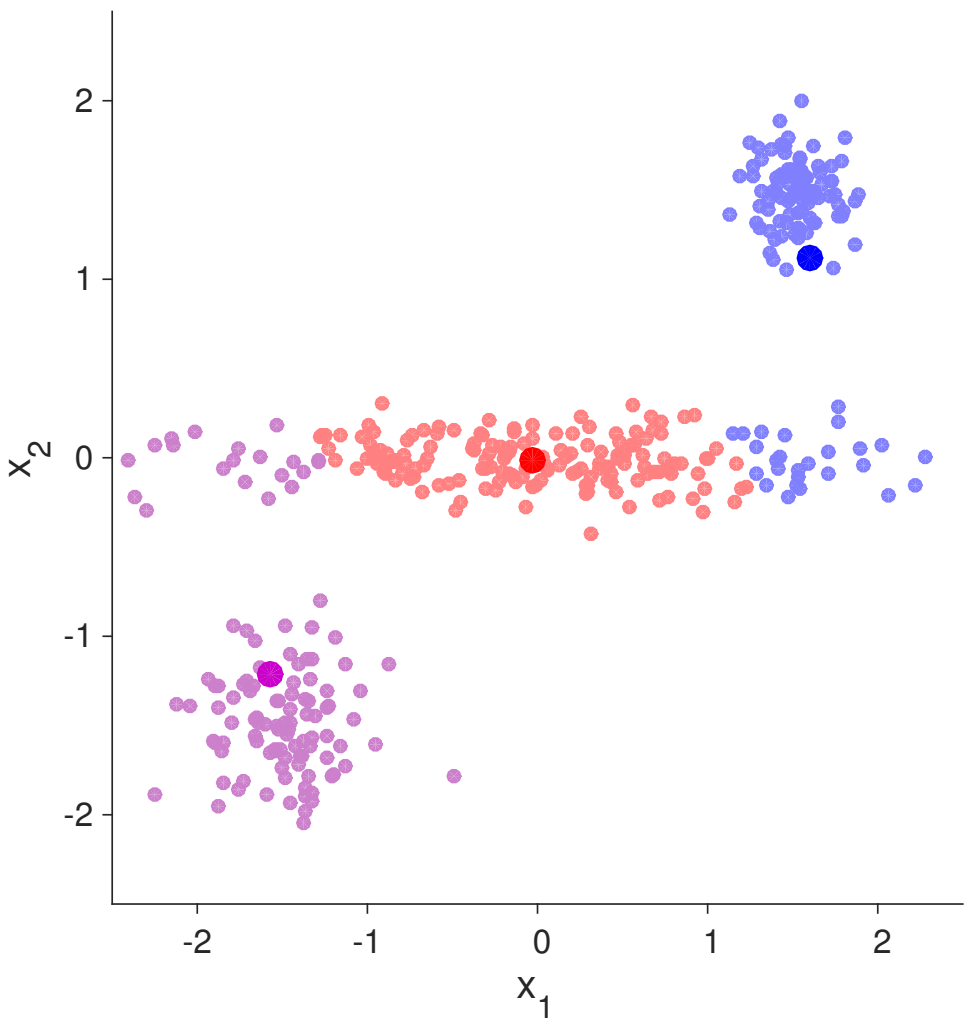
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

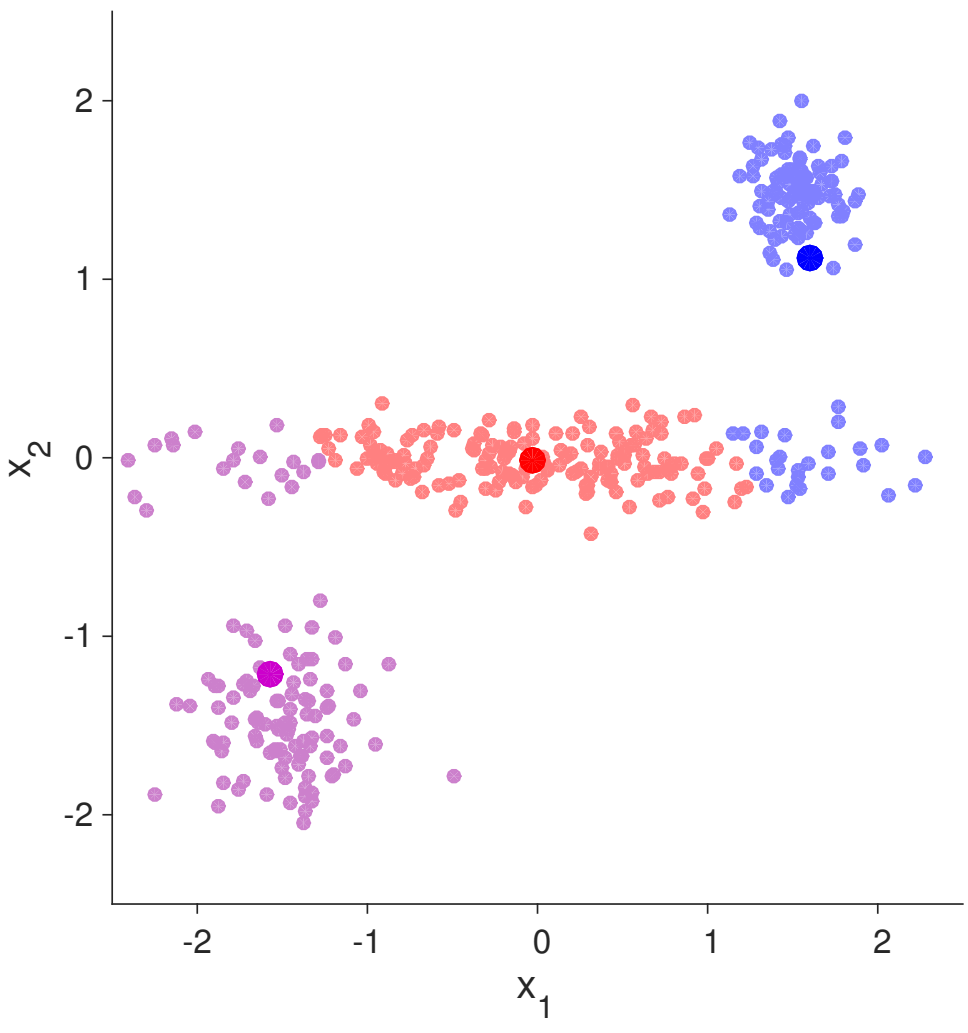
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

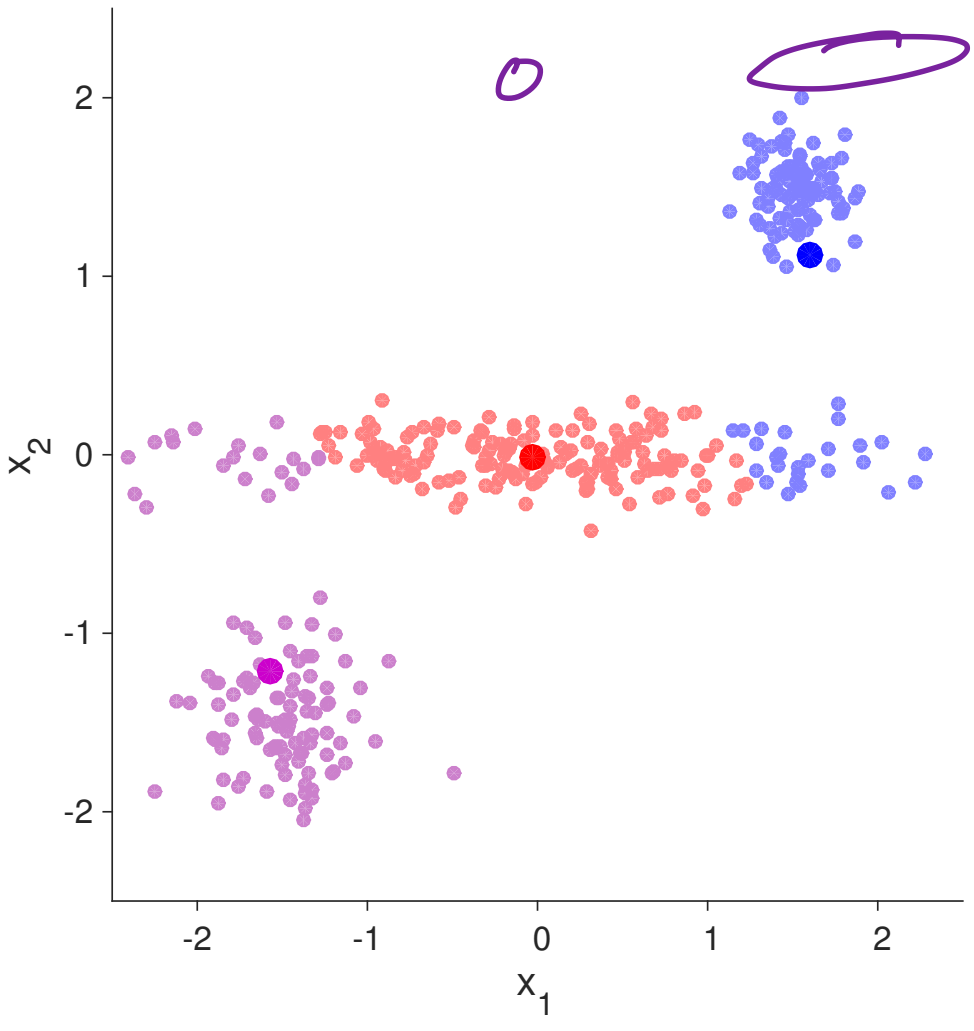
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

until convergence (s_n fixed)

Where will K-means converge to when run on these data?



input: $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_n \in \mathbb{R}^D$

initialise: $\boldsymbol{m}_k \in \mathbb{R}^D$ for $k = 1 \ldots K$

repeat

for $n = 1 \ldots N$

$$s_n = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{m}_k||$$

endfor

$|x_{d,n} - m_{dk}|$

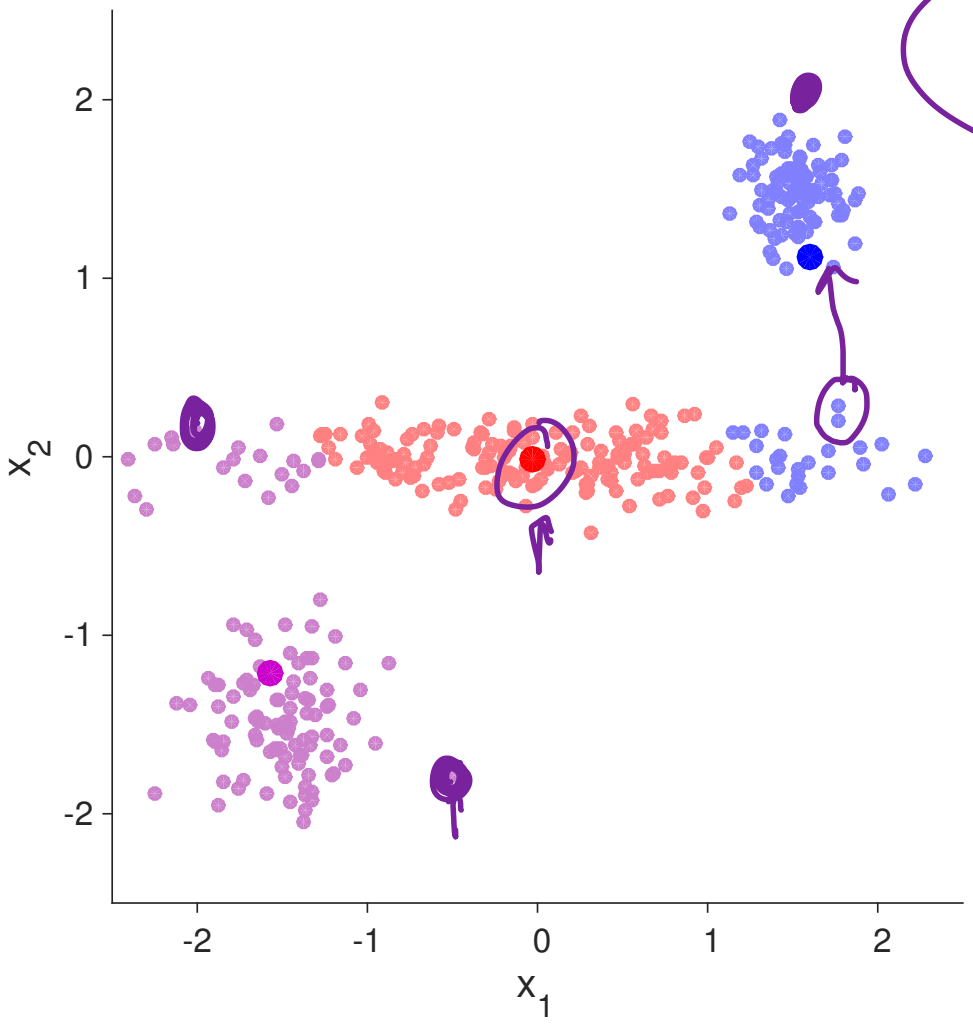
for $k = 1 \ldots K$

$$\boldsymbol{m}_k = \text{mean}(\boldsymbol{x}_n : s_n = k)$$

endfor

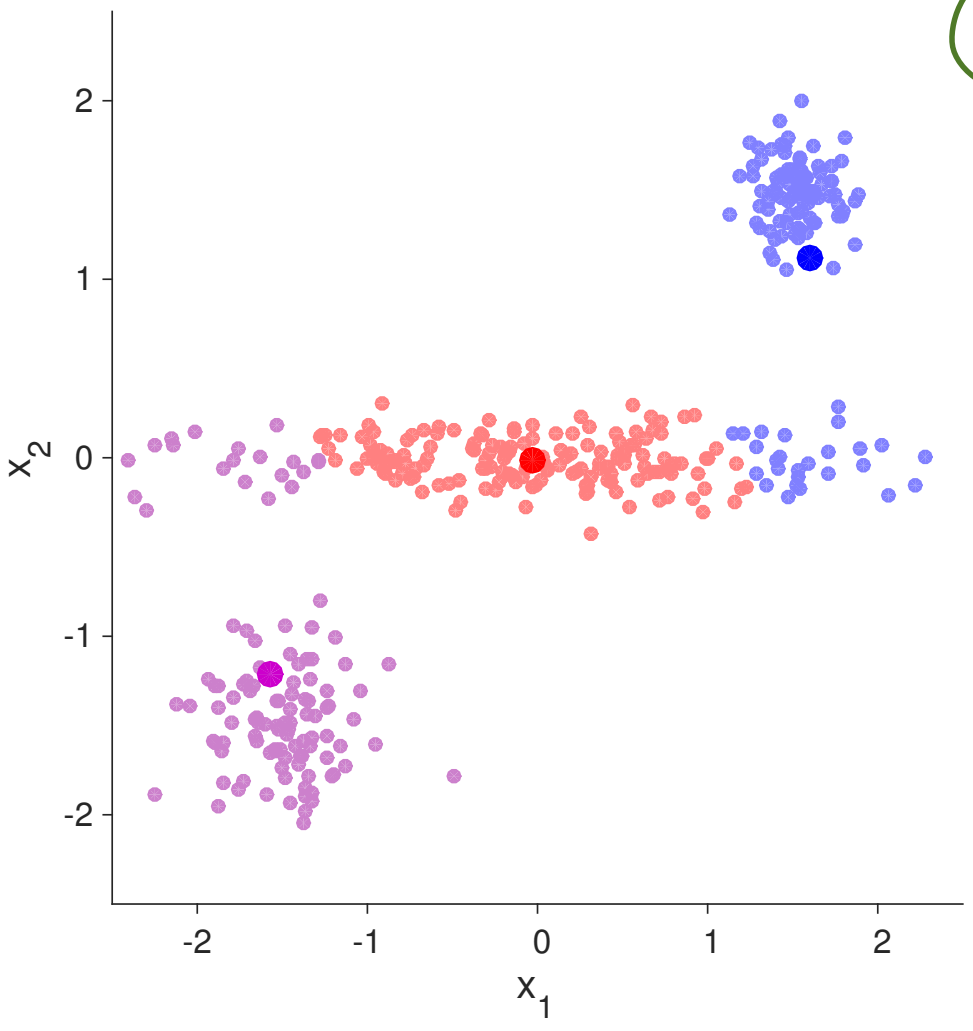
until convergence (s_n fixed)

Pathologies of K-means



- strange results when clusters have very different shapes
- returns hard assignments
  - soft assignments more sensible
- initialisation delicate
  - k++ means algorithm: spreads out initial centres

Pathologies of K-means

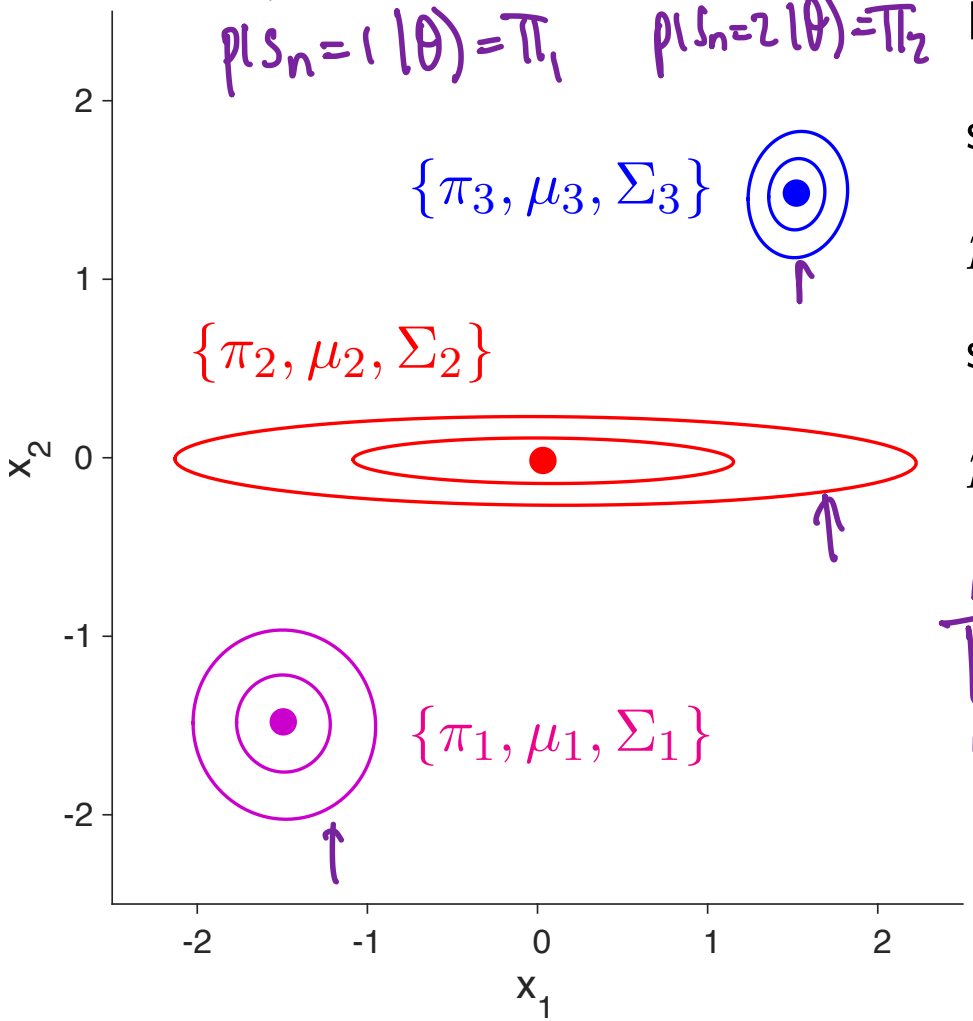


- strange results when clusters have very different shapes
- returns hard assignments
  - soft assignments more sensible
- initialisation delicate
  - k++ means algorithm: spreads out initial centres

Mixture of Gaussians: Generative Model

$D=2, K=3$

$p(s_n=1|\theta)=\pi_1$ $\quad p(s_n=2|\theta)=\pi_2$



For each data point $1 \dots N$

sample cluster membership:

$p(s_n = k|\theta) = \pi_k$ note $\sum_{k=1}^{K} \pi_k = 1$

sample data-value given cluster mem.:

$p(\boldsymbol{x}_n|s_n = k, \theta) = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{m}_k, \Sigma_k)$

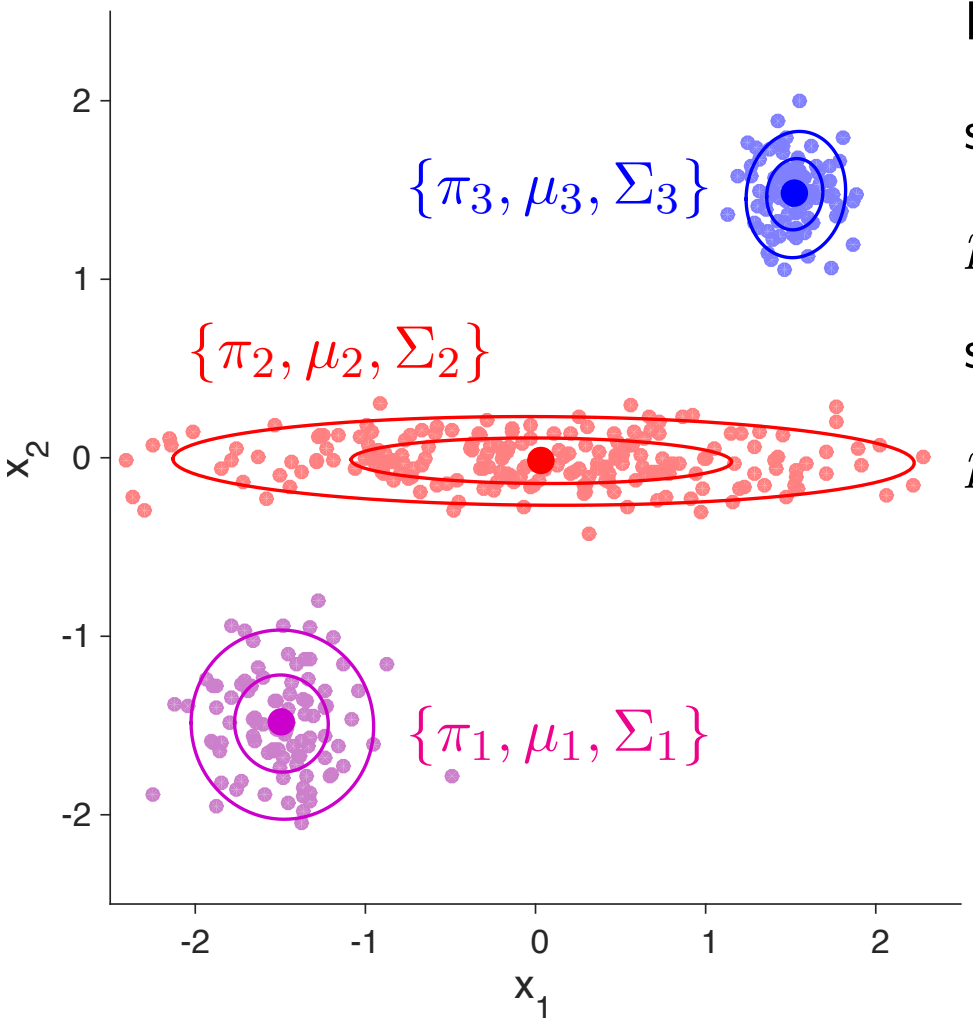
$$\prod_{n=1}^{N} p(\underline{x}_n, s_n|\theta) = \prod_{n=1}^{N} p(s_n|\theta)\, p(\underline{x}_n|s_n, \theta)$$

$$\Downarrow$$

$$\prod_{n} p(\underline{x}_n|\theta)$$

$$\theta = \{\pi_k, \underline{m}_k, \underline{\Sigma}_k\}_{k=1}^{K}$$

Mixture of Gaussians: Generative Model



For each data point $1 \ldots N$

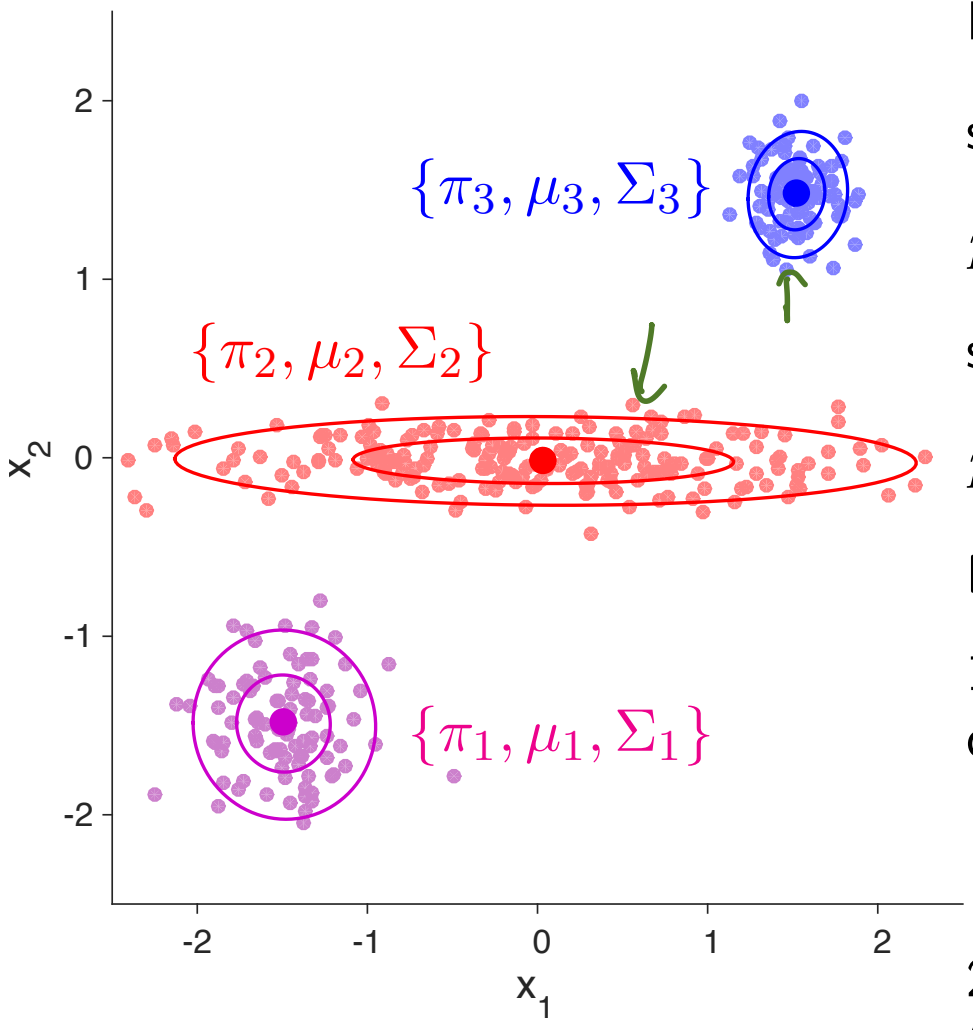
sample cluster membership:

$p(s_n = k|\theta) = \pi_k$ note $\sum_{k=1}^{K} \pi_k = 1$

sample data-value given cluster mem.:

$p(\boldsymbol{x}_n|s_n = k, \theta) = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{m}_k, \Sigma_k)$

Mixture of Gaussians: Generative Model



For each data point $1 \ldots N$

sample cluster membership:

$p(s_n = k|\theta) = \pi_k$ note $\sum_{k=1}^{K} \pi_k = 1$

sample data-value given cluster mem.:

$p(\boldsymbol{x}_n|s_n = k, \theta) = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{m}_k, \Sigma_k)$

Perform clustering by:

1. Learning the parameters from data using maximum-likelihood

$$\theta_{\text{ML}} = \arg\max_{\theta} \log p(\{\boldsymbol{x}_n\}_{n=1}^{N}|\theta)$$

2. Inferring the cluster membership from the observed data

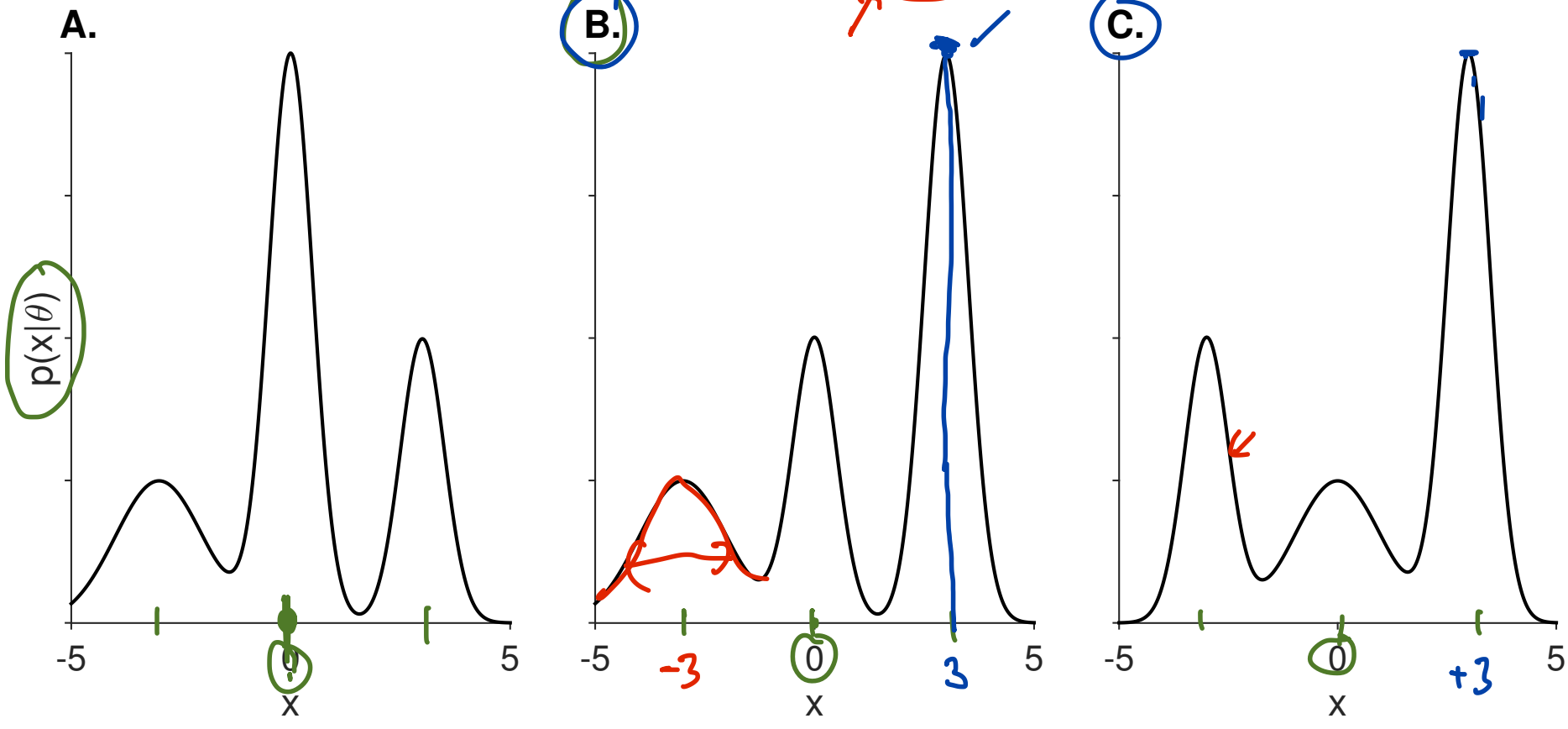
$$p(s_n = k|\mathbf{x}_n, \theta_{\text{ML}})$$

Which plot shows the density $p(x|\theta)$ of this 1D Mixture of Gaussians?

mixing proportions: $\pi_1 = \frac{1}{4}, \pi_2 = \frac{1}{2}$ and $\pi_3 = \frac{1}{4}$

means: $\mu_1 = 0, \mu_2 = 3$ and $\mu_3 = -3$

standard deviations: $\sigma_1 = \frac{1}{2}, \sigma_2 = \frac{1}{2}$ and $\sigma_3 = 1$



Mixture of Gaussians: $p(s = k|\theta) = \pi_k \quad p(x|s = k, \theta) = \mathcal{N}(x; \mu_k, \sigma_k^2)$

$$p(\underline{x} \mid \theta) \overset{\text{sum rule}}{=} \sum_{k=1}^{K} p(s=k, \underline{x} \mid \theta) \quad \| \text{ product rule}$$

$$= \sum_{k=1}^{K} p(s=k \mid \theta)\, p(\underline{x} \mid s=k, \theta)$$

$$= \sum_{k=1}^{K} \pi_k \, N(\underline{x} ; \underline{\mu}_k, \underline{\underline{\Sigma}}_k)$$

π_k: mixing proportions

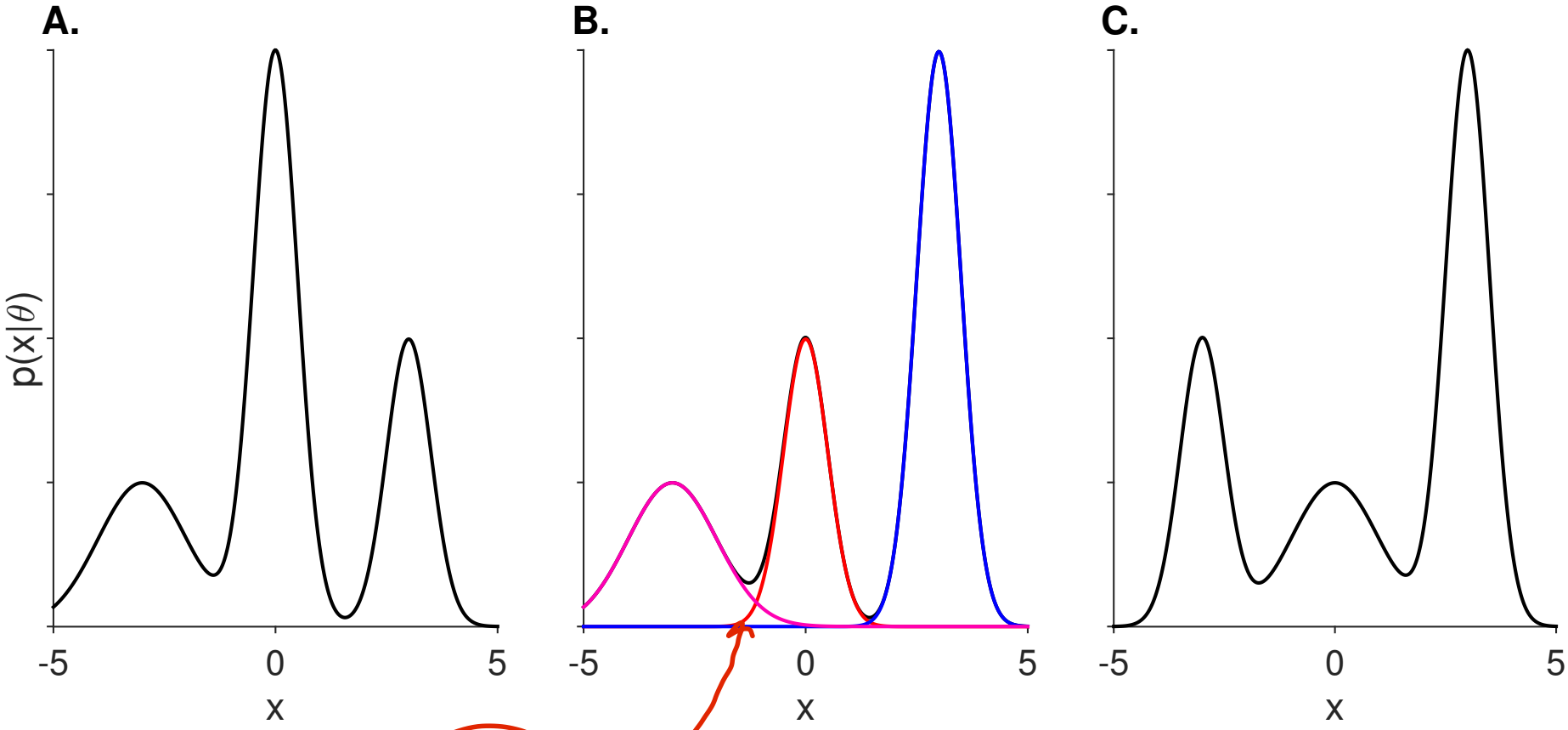
Mixture of Gaussians Model

Which plot shows the density $p(x|\theta)$ of this 1D Mixture of Gaussians?

mixing proportions: $\pi_1 = \frac{1}{4}$, $\pi_2 = \frac{1}{2}$ and $\pi_3 = \frac{1}{4}$

means: $\mu_1 = 0$, $\mu_2 = 3$ and $\mu_3 = -3$

standard deviations: $\sigma_1 = \frac{1}{2}$, $\sigma_2 = \frac{1}{2}$ and $\sigma_3 = 1$



$$p(x|\theta) = \frac{1}{4}\mathcal{N}(x; 0, \frac{1}{2}^2) + \frac{1}{2}\mathcal{N}(x; 3, \frac{1}{2}^2) + \frac{1}{4}\mathcal{N}(x; -3, 1)$$

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

$$= \log \prod_{n=1}^{N} p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{x}_n, s_n = k|\theta)$$

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

$$= \log \prod_{n=1}^{N} p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{x}_n, s_n = k|\theta)$$

$$= \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(s_n = k|\theta) p(\mathbf{x}_n|s_n = k, \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n; \mu_k, \sigma_k^2)$$

product rule

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

$$= \log \prod_{n=1}^{N} p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{x}_n, s_n = k|\theta)$$

$$= \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(s_n = k|\theta) p(\mathbf{x}_n|s_n = k, \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n; \mu_k, \sigma_k^2)$$

- Method 1: direct gradient based optimisation of the log-likelihood

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

$$= \log \prod_{n=1}^{N} p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{x}_n, s_n = k|\theta)$$

$$= \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(s_n = k|\theta) p(\mathbf{x}_n|s_n = k, \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n; \mu_k, \sigma_k^2)$$

- Method 1: direct gradient based optimisation of the log-likelihood
- Method 2: the **expectation-maximisation (EM) algorithm**

Different ways of maximising the likelihood

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta)$$

$$= \log \prod_{n=1}^N p(\mathbf{x}_n|\theta) = \sum_{n=1}^N \log p(\mathbf{x}_n|\theta) = \sum_{n=1}^N \log \sum_{k=1}^K p(\mathbf{x}_n, s_n = k|\theta)$$

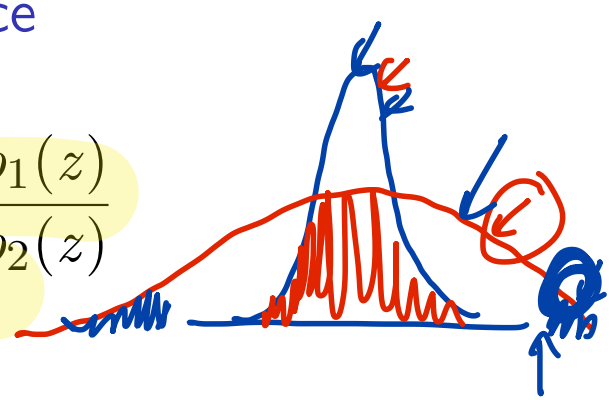
$$= \sum_{n=1}^N \log \sum_{k=1}^K p(s_n = k|\theta) p(\mathbf{x}_n|s_n = k, \theta) = \sum_{n=1}^N \log \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n; \mu_k, \sigma_k^2)$$

- Method 1: direct gradient based optimisation of the log-likelihood
- Method 2: the **expectation-maximisation (EM) algorithm**

Direct optimisation is generally faster, but the EM algorithm (i) is sometimes **simpler to implement**, (ii) is more **widely used**, and (iii) leads to **important generalisations**.

A brief introduction to the Kullback-Leibler divergence

$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z\in\mathcal{Z}} p_1(z)\log\frac{p_1(z)}{p_2(z)}$$



$K = 2$

$p(s=2) = \pi = \pi_1 \quad p(s=1) = 1-\pi = \pi_2 \quad \in \quad \pi_1 + \pi_2 = 1$

$q(s=2) = \rho \qquad q(s=1) = 1-\rho$

$KL(q(s)\,||\,p(s)) = (1-\rho)\log\frac{(1-\rho)}{1-\pi} + \rho\log\frac{\rho}{\pi}$

$KL(p(s)\,||\,q(s)) = (1-\pi)\log\frac{(1-\pi)}{1-\rho} + \pi\log\frac{\pi}{\rho}$

0.2 0.2 0.8 0.1

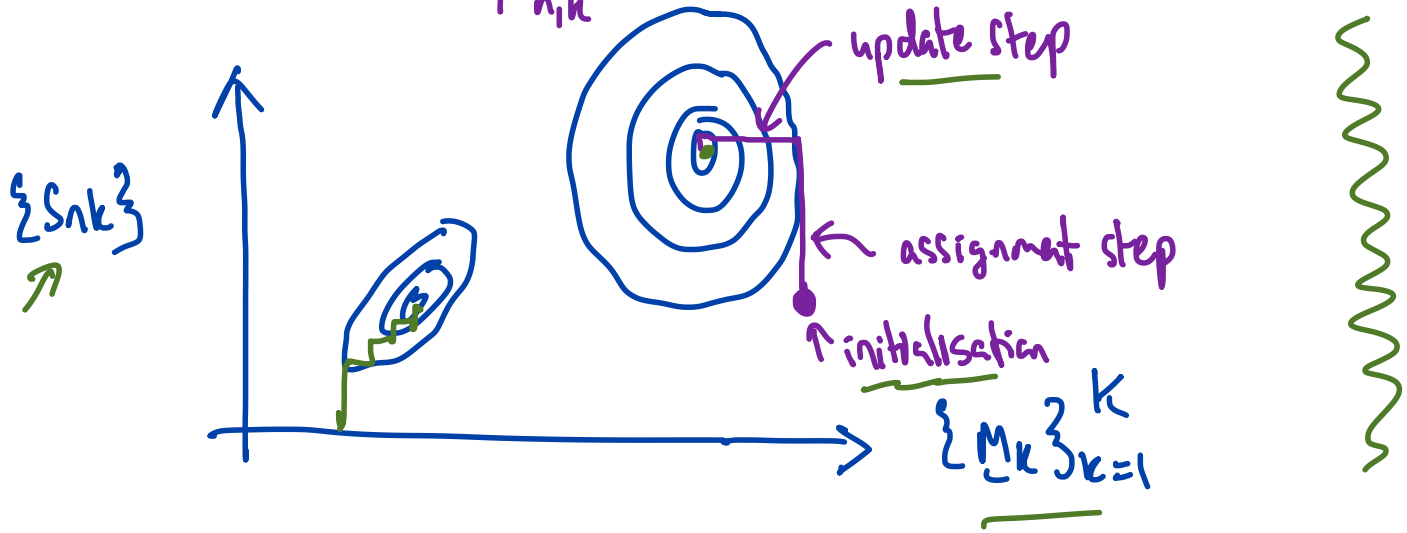
Summary K-means Algorithm

Algorithm pseudo-code

1. initialise cluster centres $\{\underline{m}_k\}_{k=1}^{K}$ ← $\underline{x}_n$
2. assign each point to "nearest" cluster centre
3. update centres to mean of assigned points

iterate

→ $E = \frac{1}{2} \sum_{n,k} s_{nk} \| \underline{x}_n - \underline{m}_k \|^2$



Mixture of Gaussians Model for Clustering : Summary

Mixture of Gaussians Model

$$p(s_n = k \mid \theta) = \pi_k$$

$$p(\underline{x}_n \mid s_n = k, \theta) = \mathcal{N}(\underline{x}_n ; \underline{\mu}_k, \underline{\Sigma}_k)$$

$\underline{\mu}_k$, $\underline{\Sigma}_k$

$$\theta = \{\pi_k, \underline{\mu}_k, \underline{\Sigma}_k\}_{k=1}^{K}$$

$p(x_1) p(x_2 \mid x_1) p(x_3 \mid x_2, x_1) \cdots$

[latent variable]

Clustering approach

① $\underline{\theta}_{ML} = \arg\max_{\theta} \log p(\{\underline{x}_n\}_{n=1}^{N} \mid \theta)$

$$= \arg\max_{\theta} \sum_{n=1}^{N} \log p(\underline{x}_n \mid \theta) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} p(s_n = k \mid \theta)\, p(\underline{x}_n \mid s_n = k, \theta)$$

($p(s_n = k \mid \theta) = \pi_k$, $p(\underline{x}_n \mid s_n = k, \theta) = \mathcal{N}(\underline{x}_n ; \underline{\mu}_k, \underline{\Sigma}_k)$)

② $p(s_n = k \mid \underline{x}_n, \theta_{ML}) = p(s_n = k \mid \theta_{ML})\, p(\underline{x}_n \mid s_n = k, \theta_{ML}) / p(\underline{x}_n \mid \theta_{ML})$

A Question about likelihood functions

$s_{nk} = [0/1]$

(1) From last lecture $p(\underline{x}_n|\theta)$ = likelihood of parameters

From notebooks $p(\underline{x}_n|\underline{s}_n, \theta) = \left(\prod_k \left[\pi_k \mathcal{N}(\underline{x}_n|\underline{\mu}_k, \underline{\underline{\Sigma}}_k)\right]\right)^{s_{nk}}$

= likelihood of latent variables s_n

What's going on??!

A brief introduction to the Kullback-Leibler divergence

$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z \in \mathcal{Z}} p_1(z) \log \frac{p_1(z)}{p_2(z)}$$

$p_2(z)$ $p_1(z)$

Important properties:

A brief introduction to the Kullback-Leibler divergence

$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z \in \mathcal{Z}} p_1(z) \log \frac{p_1(z)}{p_2(z)}$$

Important properties:

- Gibb's inequality: $\mathcal{KL}(p_1(z)||p_2(z)) \geq 0$, equality at $p_1(z) = p_2(z)$
  - proof via Jensen's inequality or differentiation (see MacKay pg. 35)

A brief introduction to the Kullback-Leibler divergence

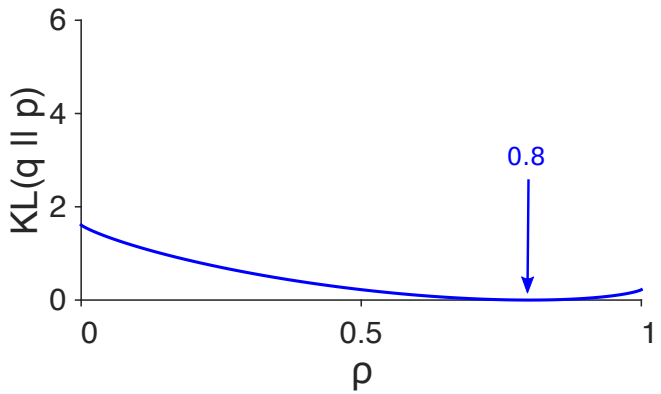
$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z \in \mathcal{Z}} p_1(z) \log \frac{p_1(z)}{p_2(z)}$$

Important properties:

- Gibb's inequality: $\mathcal{KL}(p_1(z)||p_2(z)) \geq 0$, equality at $p_1(z) = p_2(z)$
  - proof via Jensen's inequality or differentiation (see MacKay pg. 35)

Example:

- binary variables $z \in \{0, 1\}$
- $p(z = 1) = 0.8$ and $q(z = 1) = \rho$



A brief introduction to the Kullback-Leibler divergence

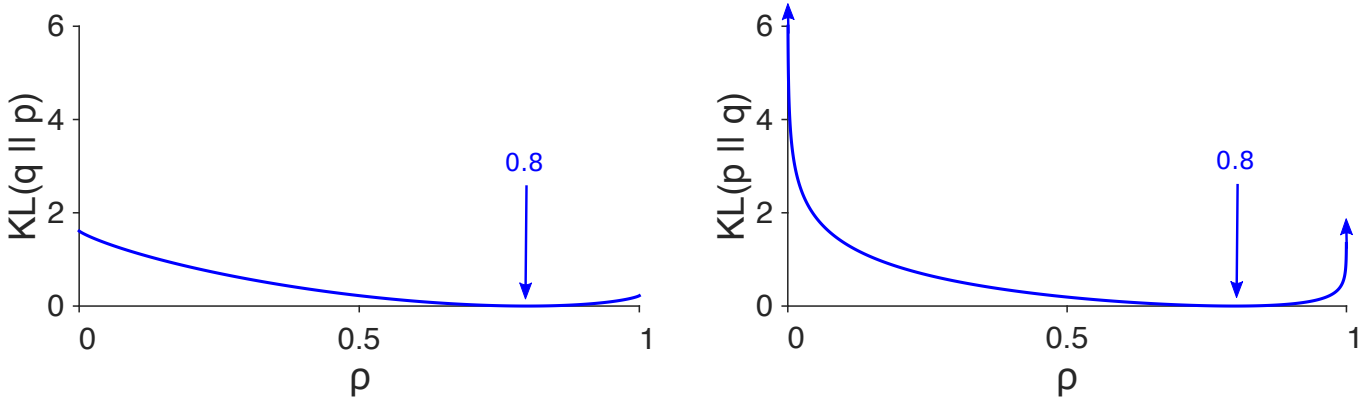
$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z \in \mathcal{Z}} p_1(z) \log \frac{p_1(z)}{p_2(z)}$$

Important properties:

- Gibb's inequality: $\mathcal{KL}(p_1(z)||p_2(z)) \geq 0$, equality at $p_1(z) = p_2(z)$
  - proof via Jensen's inequality or differentiation (see MacKay pg. 35)

Example:

- binary variables $z \in \{0, 1\}$
- $p(z = 1) = 0.8$ and $q(z = 1) = \rho$



A brief introduction to the Kullback-Leibler divergence

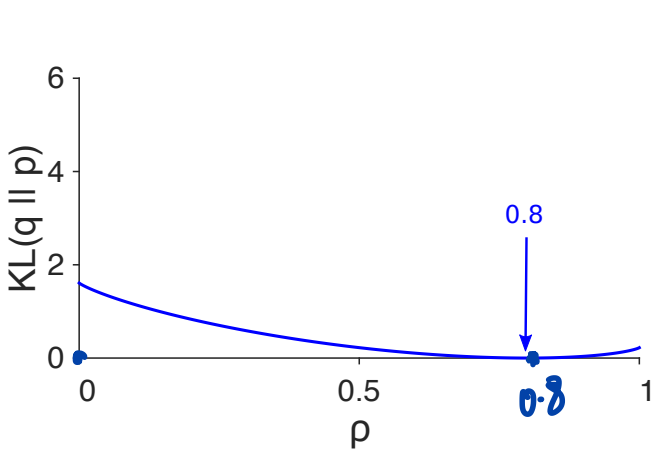
$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_{z\in\mathcal{Z}} p_1(z)\log\frac{p_1(z)}{p_2(z)}$$

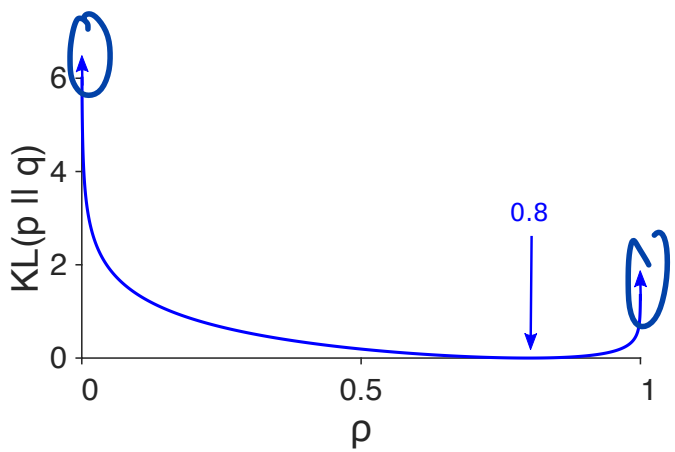
Important properties:

- Gibb's inequality: $\mathcal{KL}(p_1(z)||p_2(z)) \geq 0$, equality at $p_1(z) = p_2(z)$
  - proof via Jensen's inequality or differentiation (see MacKay pg. 35)
- Non-symmetric: $\mathcal{KL}(p_1(z)||p_2(z)) \neq \mathcal{KL}(p_2(z)||p_1(z))$
  - hence named *divergence* and not *distance*

Example:

- binary variables $z \in \{0, 1\}$
- $p(z = 1) = 0.8$ and $q(z = 1) = \rho$





The Expectation Maximisation Algorithm and the Variational Free-energy

- Build toward **Expectation Maximisation (EM) Algorithm**

$$KL(q(s) \,\|\, p(s)) = \sum_{k=1}^{K} q(s=k) \log \frac{q(s=k)}{p(s=k)}$$

$s \in \{1 \dots K\}$

$$\frac{d}{dq(s=l)} \left[KL(q(s) \| p(s)) - \lambda \left(\sum_{k=1}^{K} q(s=k) - 1 \right) \right] = 0$$

evaluates to 0 at solution

$$\log \frac{q(s=l)}{p(s=l)} + \frac{q(s=l)}{q(s=l)} - \lambda = 0$$

$$q(s=l) = p(s=l)\, e^{\lambda - 1}$$

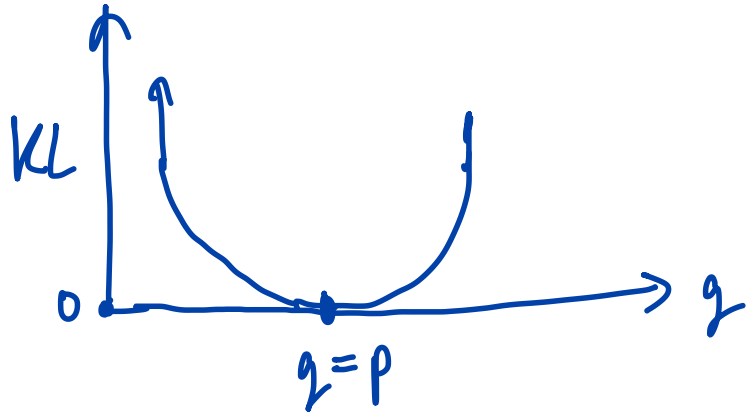
$$\sum_{l=1}^{K} q(s=l) = 1$$

$$\Rightarrow e^{\lambda-1} = 1 \Rightarrow q(s=l) = p(s=l)$$

$$\frac{d^2}{d\,q(s=l)^2}\,KL(q(s)\,\|\,p(s)) = \frac{d}{d\,q(s=l)}\left[\log\frac{q(s=l)}{p(s=l)} + 1\right]$$

$$= \frac{1}{q(s=l)} > 0$$

+ve



The Expectation Maximisation Algorithm and the Variational Free-energy

- Build toward **Expectation Maximisation (EM) Algorithm**
- EM Algorithm can be used for maximum-likelihood parameter learning of **many latent variable models**

The Expectation Maximisation Algorithm and the Variational Free-energy

- Build toward **Expectation Maximisation (EM) Algorithm**
- EM Algorithm can be used for maximum-likelihood parameter learning of **many latent variable models**
- Leads to a variety of extensions such as **approximate variational inference**

The Expectation Maximisation Algorithm and the Variational Free-energy

- Build toward **Expectation Maximisation (EM) Algorithm**
- EM Algorithm can be used for maximum-likelihood parameter learning of **many latent variable models**
- Leads to a variety of extensions such as **approximate variational inference**

For generality, we simplify notation: let $\mathbf{x} = \{\mathbf{x}_n\}_{n=1}^N$ and $\mathbf{s} = \{s_n\}_{n=1}^N$ so

The Expectation Maximisation Algorithm and the Variational Free-energy

- Build toward **Expectation Maximisation (EM) Algorithm**
- EM Algorithm can be used for maximum-likelihood parameter learning of **many latent variable models**
- Leads to a variety of extensions such as **approximate variational inference**

For generality, we simplify notation: let $\mathbf{x} = \{\mathbf{x}_n\}_{n=1}^N$ and $\mathbf{s} = \{s_n\}_{n=1}^N$ so

$$\log p(\{\mathbf{x}_n\}_{n=1}^N|\theta) = \log p(\mathbf{x}|\theta) = \log \sum_{\mathbf{s}} p(\mathbf{x}, \mathbf{s}|\theta)$$

A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$

$$\underbrace{\log p(\boldsymbol{x}|\theta)}_{\text{log-likelihood}}$$

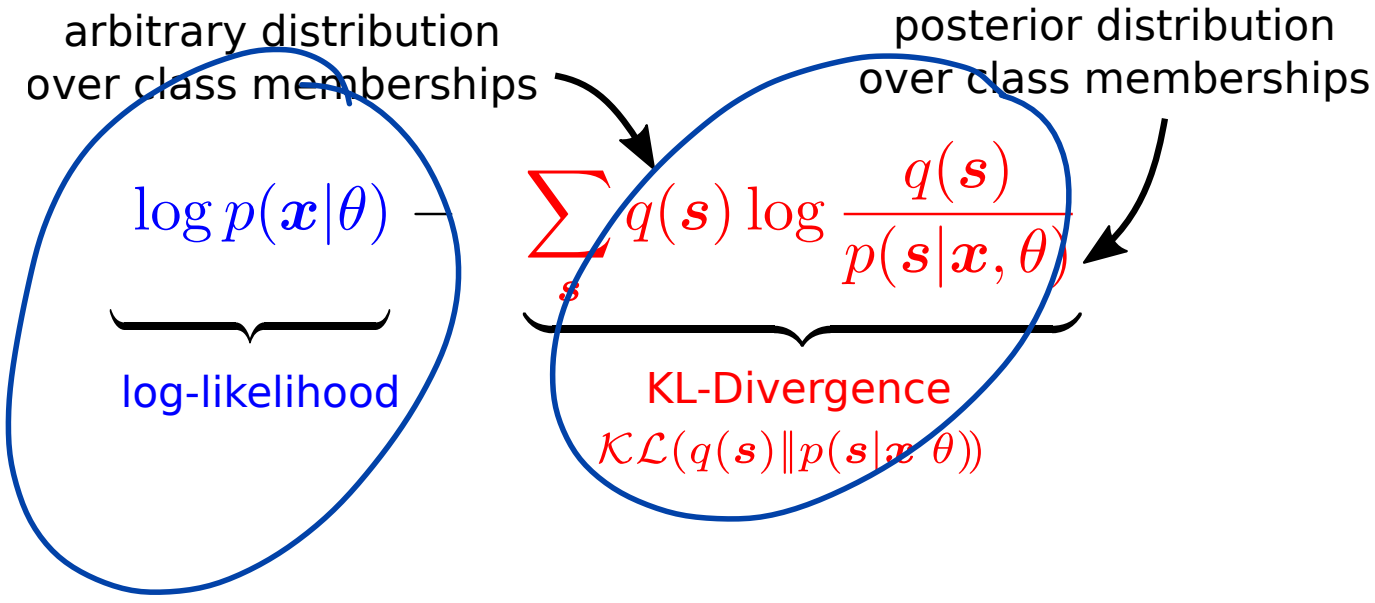
A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$

arbitrary distribution over class memberships

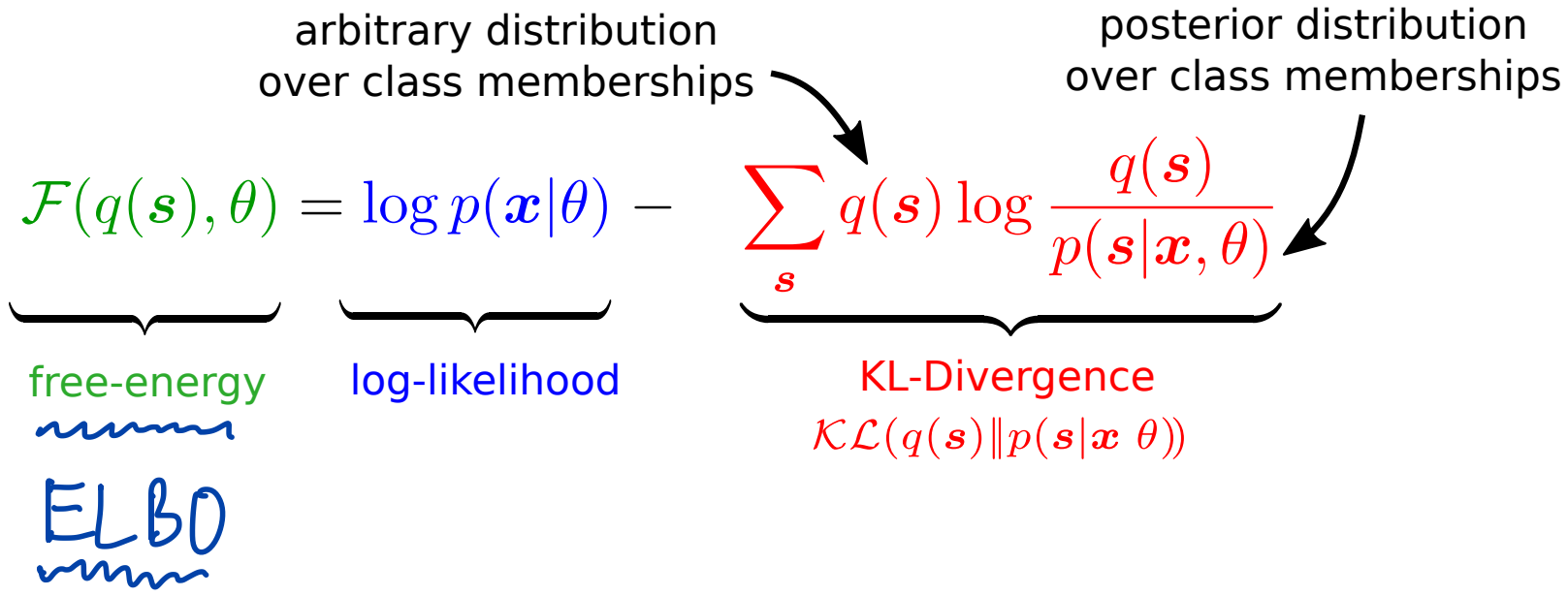
posterior distribution over class memberships

$$\underbrace{\log p(\boldsymbol{x}|\theta)}_{\text{log-likelihood}} - \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \frac{q(\boldsymbol{s})}{p(\boldsymbol{s}|\boldsymbol{x},\theta)}$$

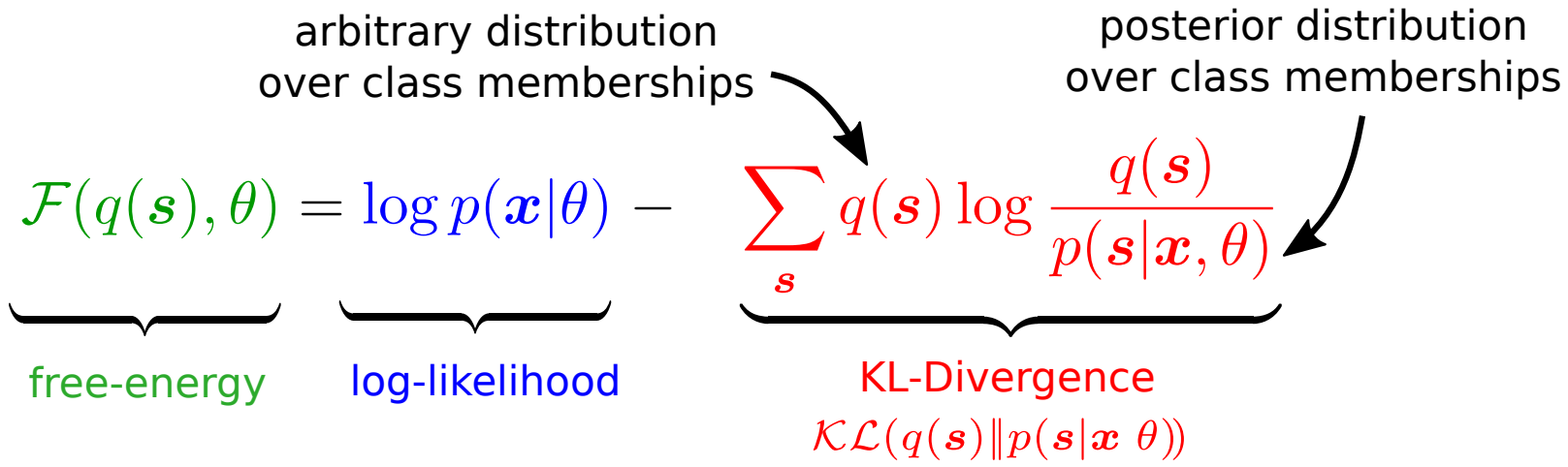
A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$



A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$



A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$



A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$

arbitrary distribution over class memberships

posterior distribution over class memberships

$$\underbrace{\mathcal{F}(q(\boldsymbol{s}),\theta)}_{\text{free-energy}} = \underbrace{\log p(\boldsymbol{x}|\theta)}_{\text{log-likelihood}} - \underbrace{\sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \frac{q(\boldsymbol{s})}{p(\boldsymbol{s}|\boldsymbol{x},\theta)}}_{\text{KL-Divergence } \mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x}\ \theta))}$$

Non-negativity of KL-Divergence

$$\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta)) \geq 0 \implies$$

Free-energy is lower bound on log-likelihood

$$\mathcal{F}(q(\boldsymbol{s}),\theta) \leq \log p(\boldsymbol{x}|\theta)$$

A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$

arbitrary distribution over class memberships

posterior distribution over class memberships

$$\underbrace{\mathcal{F}(q(\boldsymbol{s}),\theta)}_{\text{free-energy}} = \underbrace{\log p(\boldsymbol{x}|\theta)}_{\text{log-likelihood}} - \underbrace{\sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \frac{q(\boldsymbol{s})}{p(\boldsymbol{s}|\boldsymbol{x},\theta)}}_{\text{KL-Divergence } \mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x}\ \theta))}$$

Non-negativity of KL-Divergence

$$\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta)) \geq 0 \implies$$

Free-energy is lower bound on log-likelihood

$$\mathcal{F}(q(\boldsymbol{s}),\theta) \leq \log p(\boldsymbol{x}|\theta)$$

KL-Divergence equal to 0 when $q(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x},\theta)$

$$\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta)) = 0 \implies$$

Free-energy equal to log-likelihood when $q(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x},\theta)$

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \log p(\boldsymbol{x}|\theta)$$

A lower bound on the log-likelihood $\log p(\boldsymbol{x}|\theta)$

arbitrary distribution over class memberships

posterior distribution over class memberships

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \log p(\boldsymbol{x}|\theta) - \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \frac{q(\boldsymbol{s})}{p(\boldsymbol{s}|\boldsymbol{x},\theta)}$$

free-energy | log-likelihood | KL-Divergence $\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta))$

$$\Rightarrow \mathcal{F}(q(\boldsymbol{s}),\theta) = \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \frac{p(\boldsymbol{x}|\boldsymbol{s},\theta)p(\boldsymbol{s}|\theta)}{q(\boldsymbol{s})} \Rightarrow \text{simple to compute}$$

Non-negativity of KL-Divergence

$$\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta)) \geq 0 \implies$$

Free-energy is lower bound on log-likelihood

$$\mathcal{F}(q(\boldsymbol{s}),\theta) \leq \log p(\boldsymbol{x}|\theta)$$

KL-Divergence equal to 0 when $q(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x},\theta)$

$$\mathcal{KL}(q(\boldsymbol{s})\|p(\boldsymbol{s}|\boldsymbol{x},\theta)) = 0 \implies$$

Free-energy equal to log-likelihood when $q(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x},\theta)$

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \log p(\boldsymbol{x}|\theta)$$

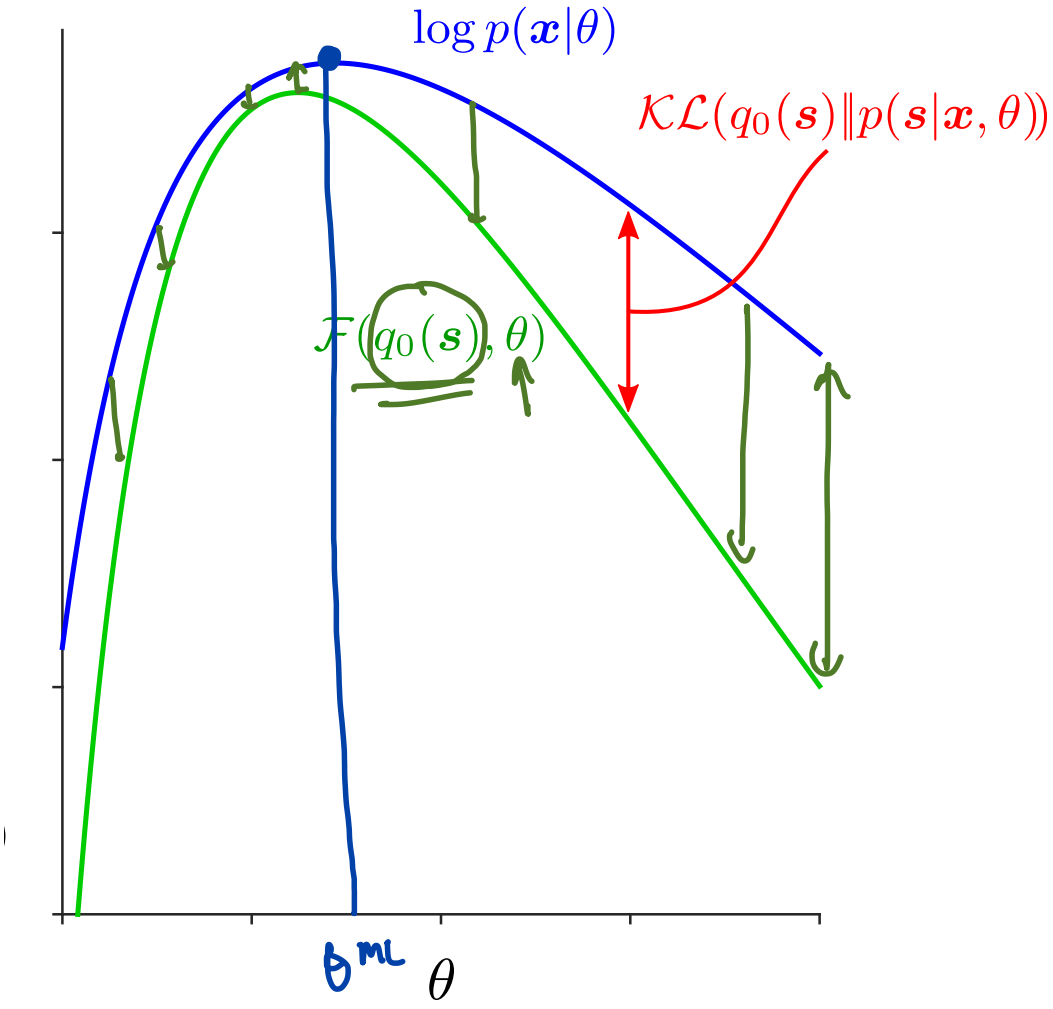
Visualising the free-energy lower bound

Visualising the free-energy lower bound

$$\mathcal{F}(q(\boldsymbol{s}), \theta) \; = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

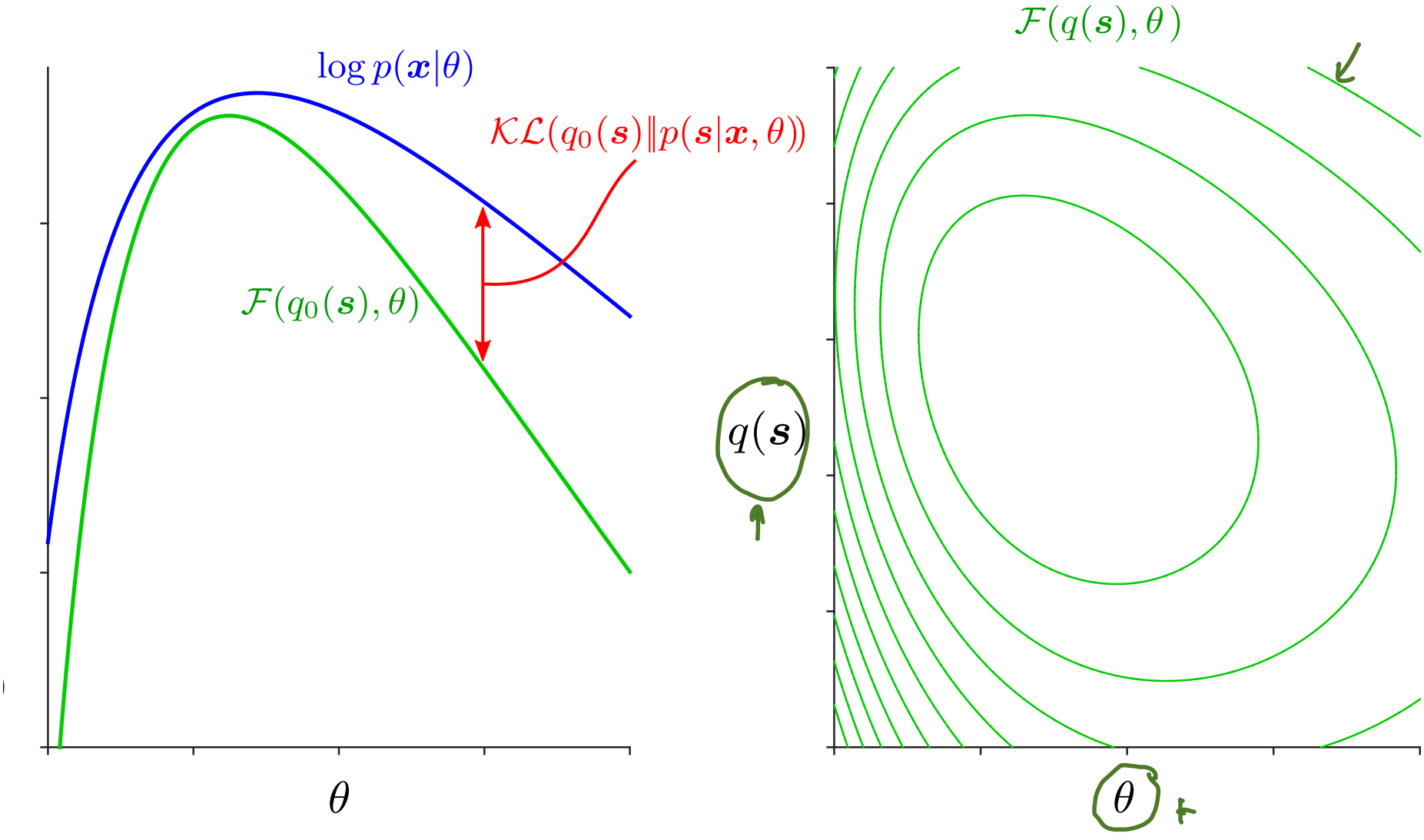
Visualising the free-energy lower bound

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$



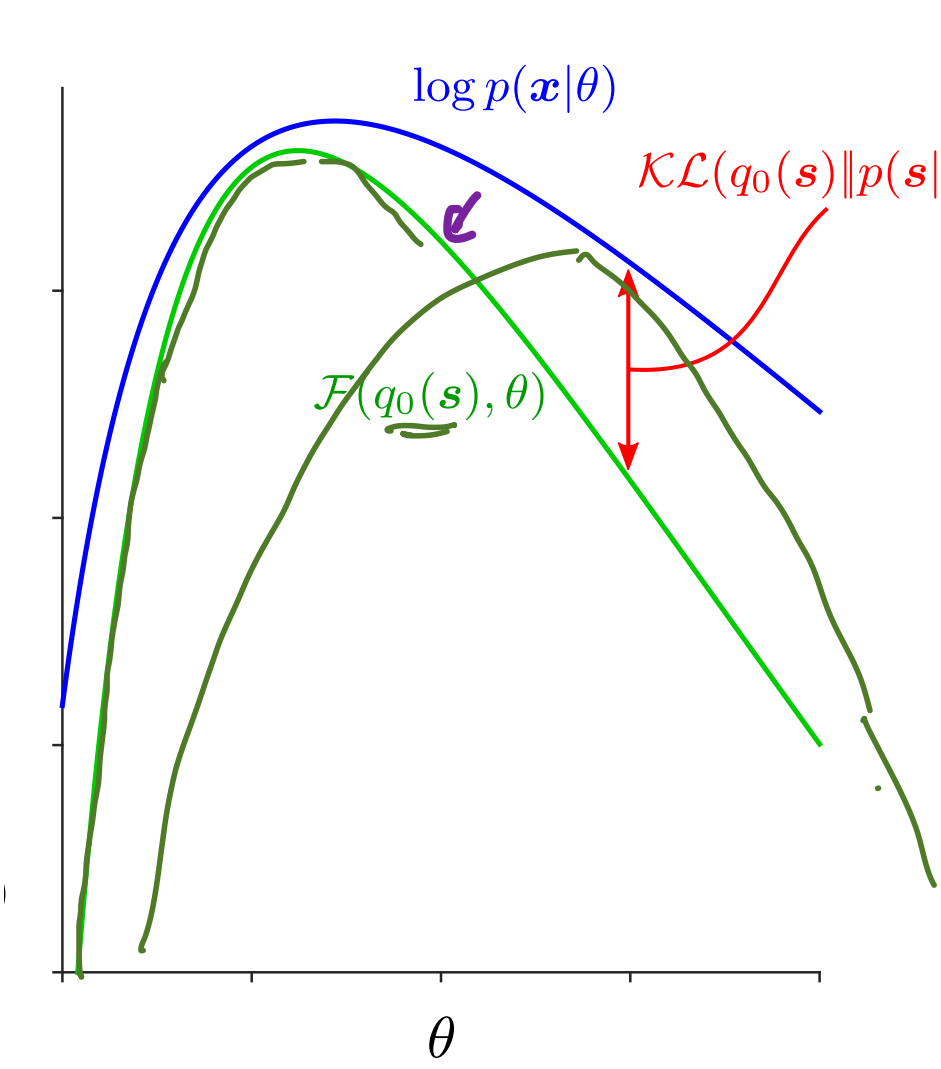
Visualising the free-energy lower bound

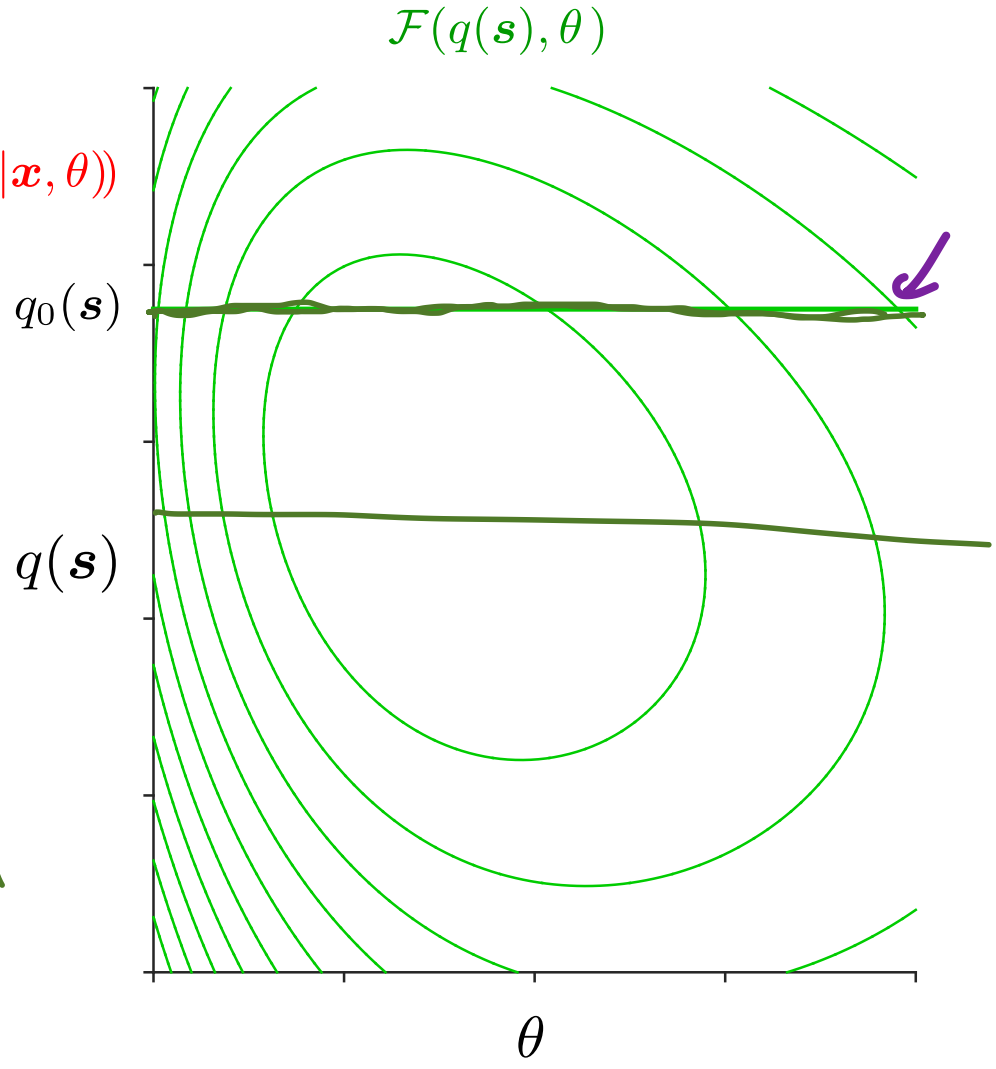
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$



Visualising the free-energy lower bound

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

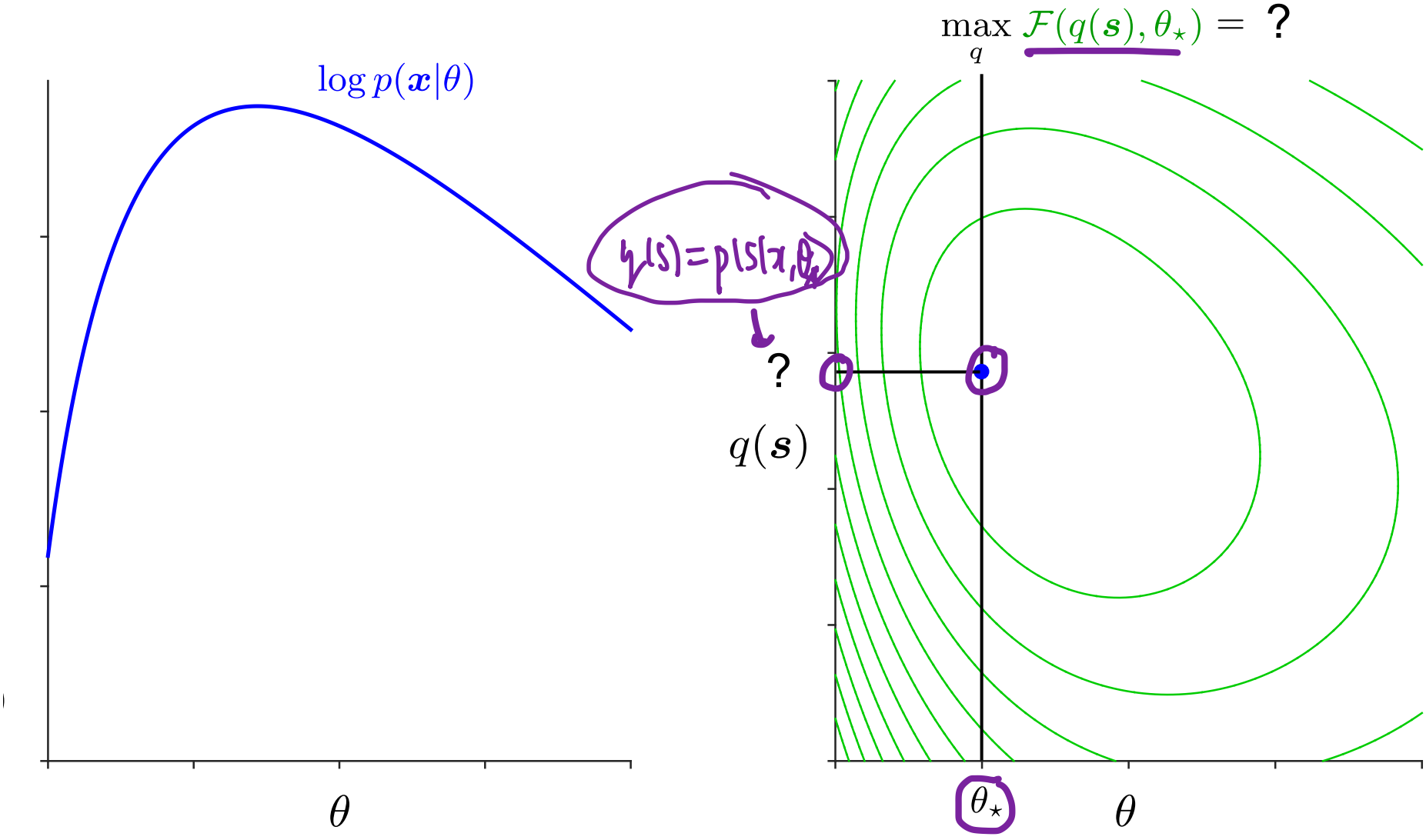




What is the maximal value of the free-energy along this vertical slice?

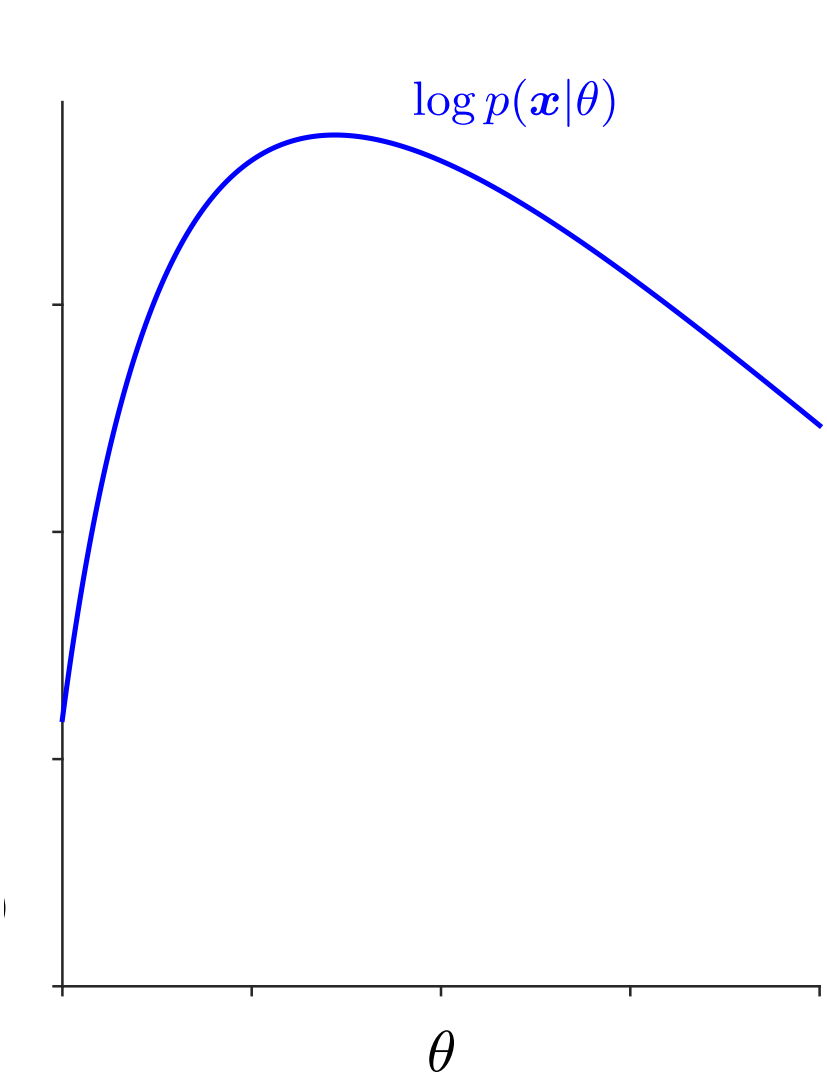
$$\Rightarrow \quad \mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta)) \qquad = 0$$

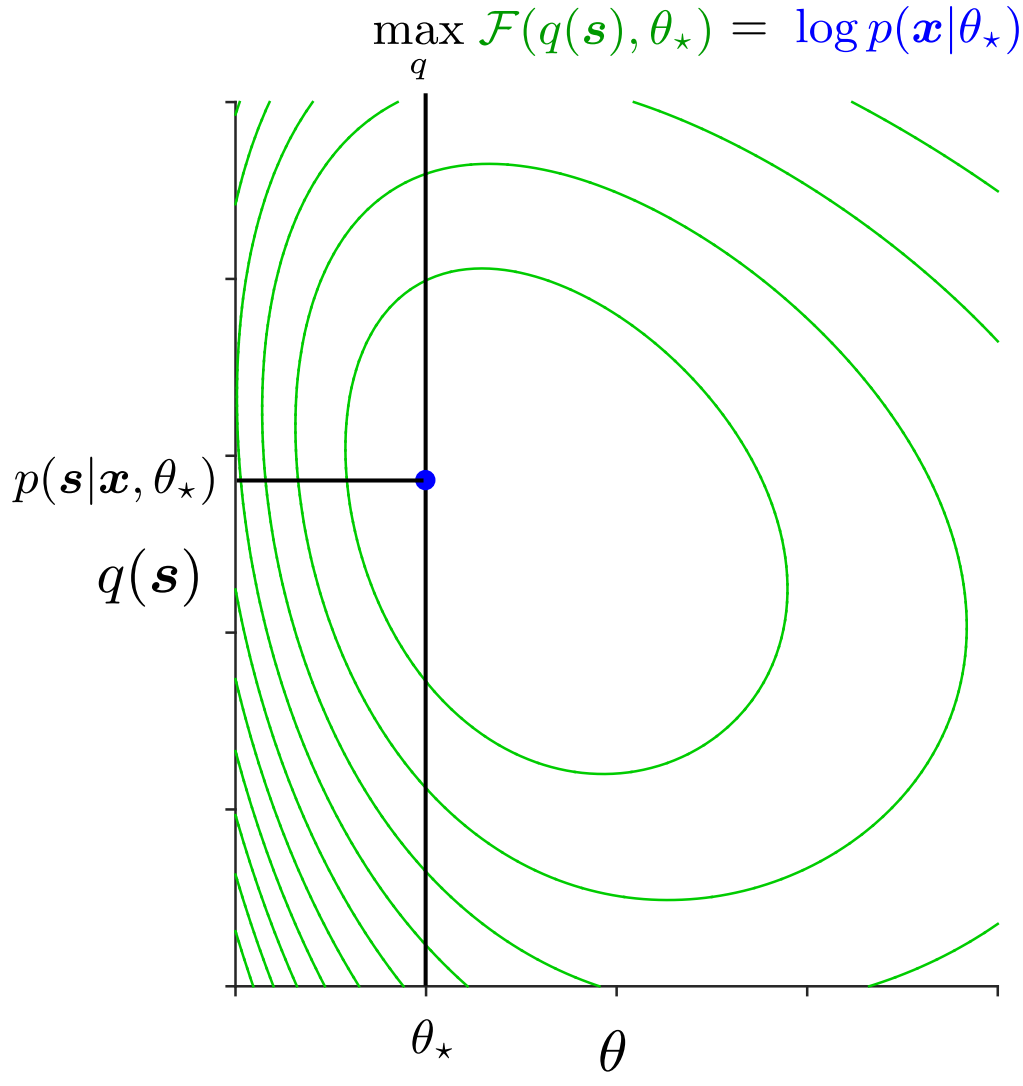
$$\max_q \mathcal{F}(q(\boldsymbol{s}), \theta_\star) = \; ?$$



What is the maximal value of the free-energy along this vertical slice?

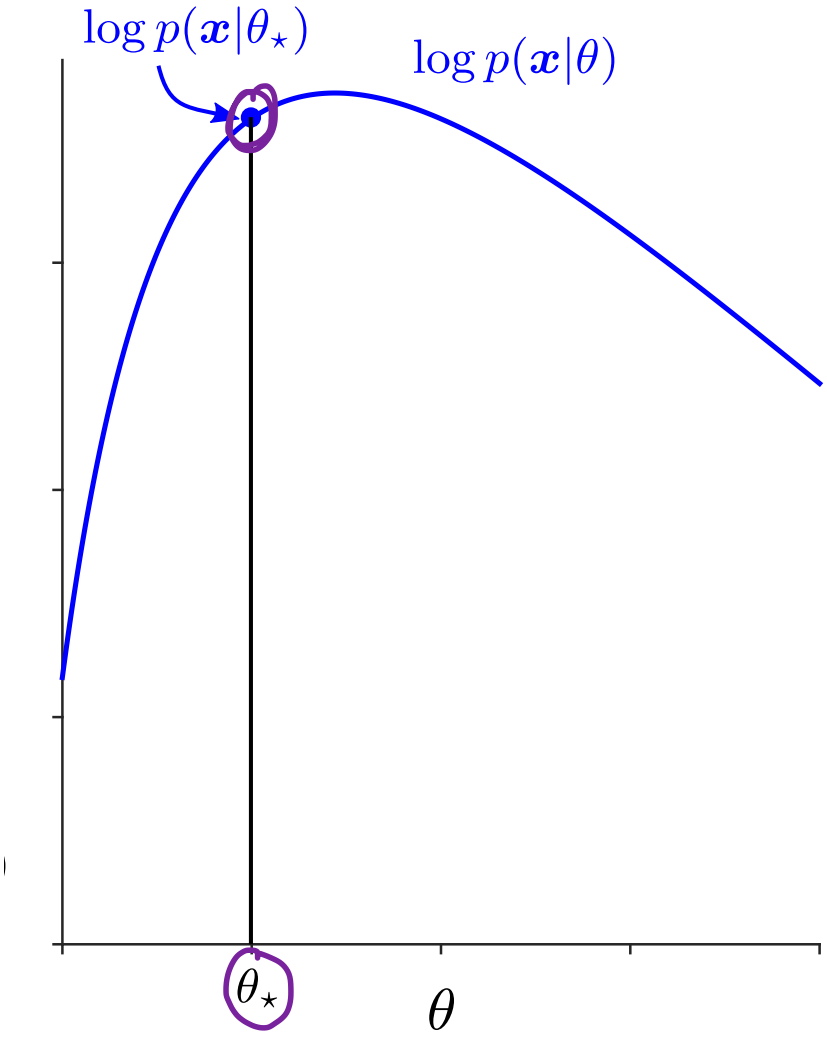
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

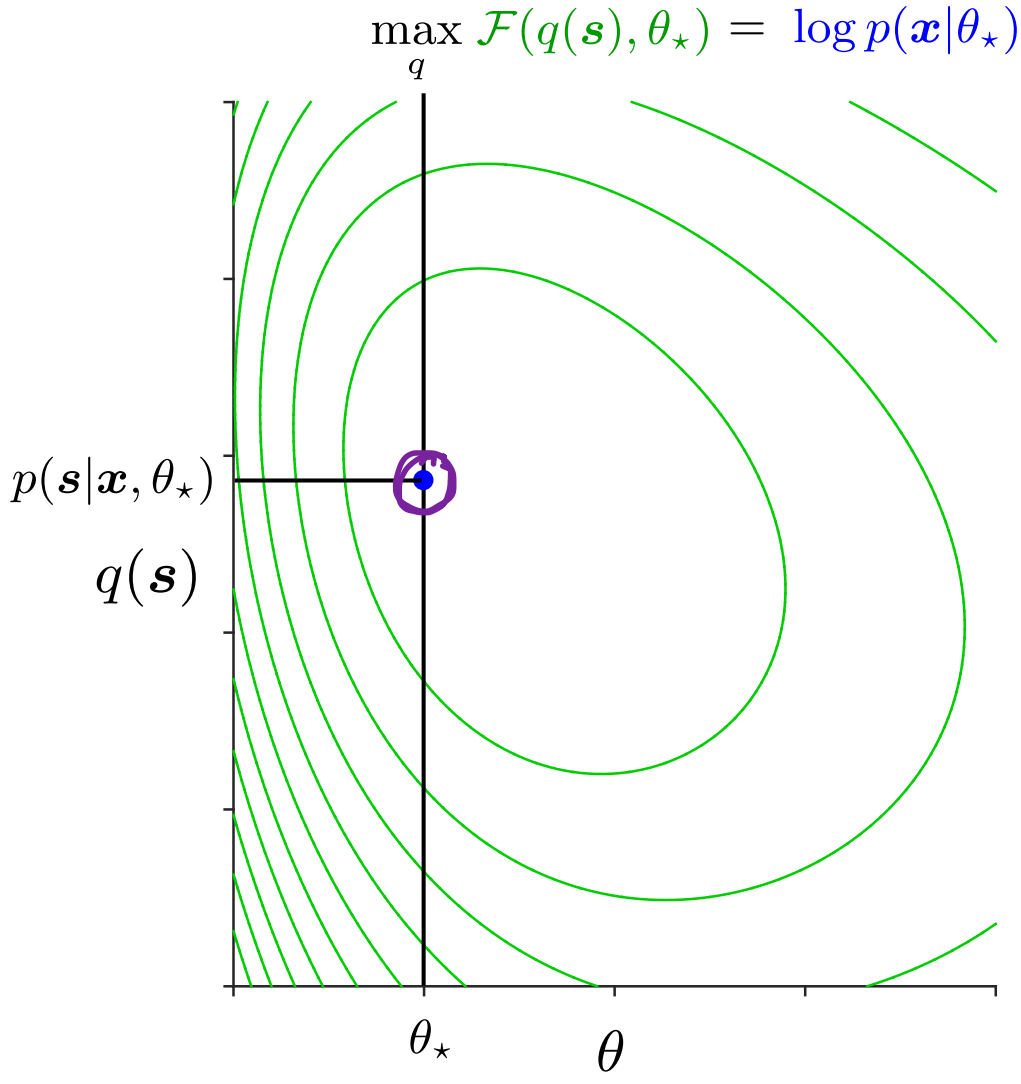




What is the maximal value of the free-energy along this vertical slice?

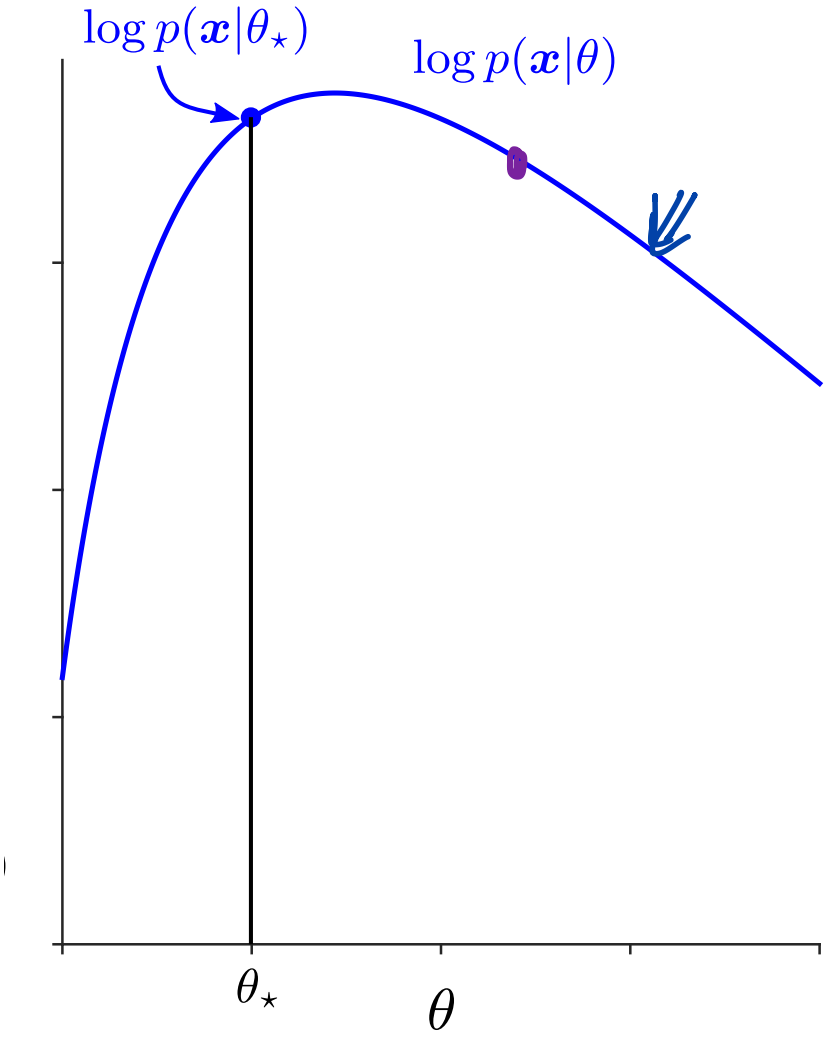
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

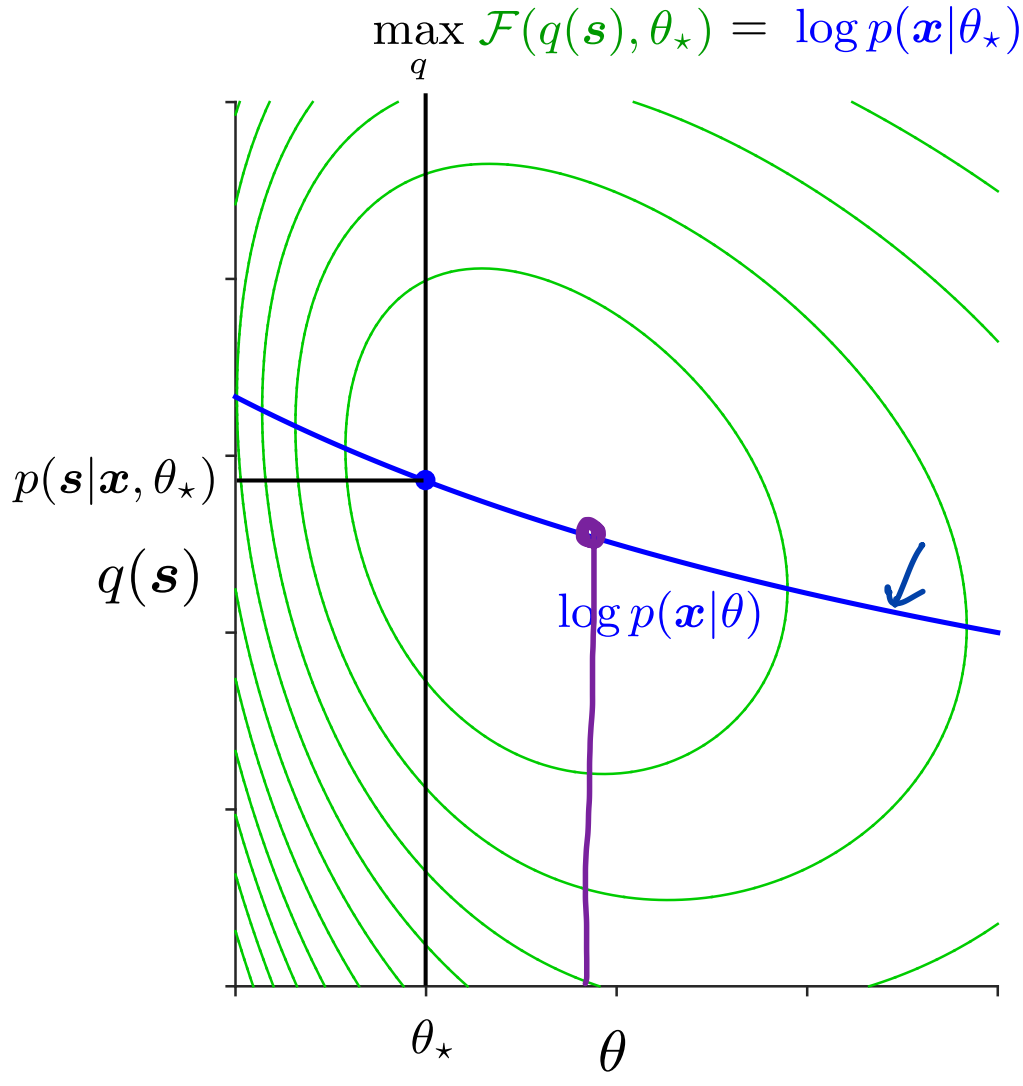




What is the maximal value of the free-energy along this vertical slice?

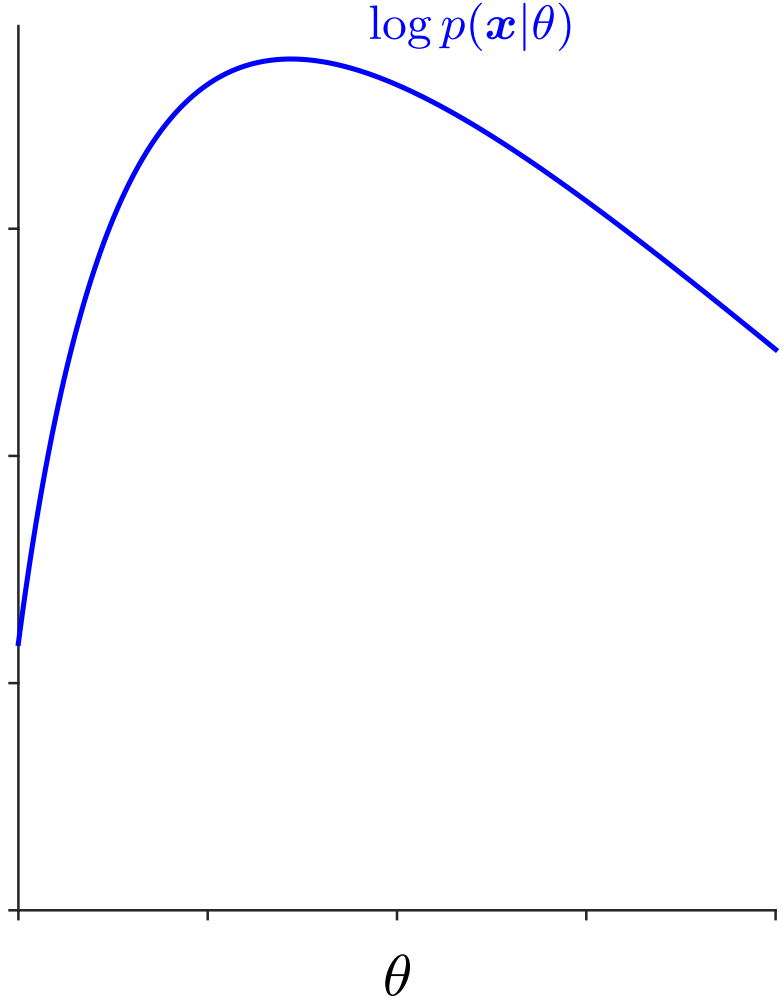
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

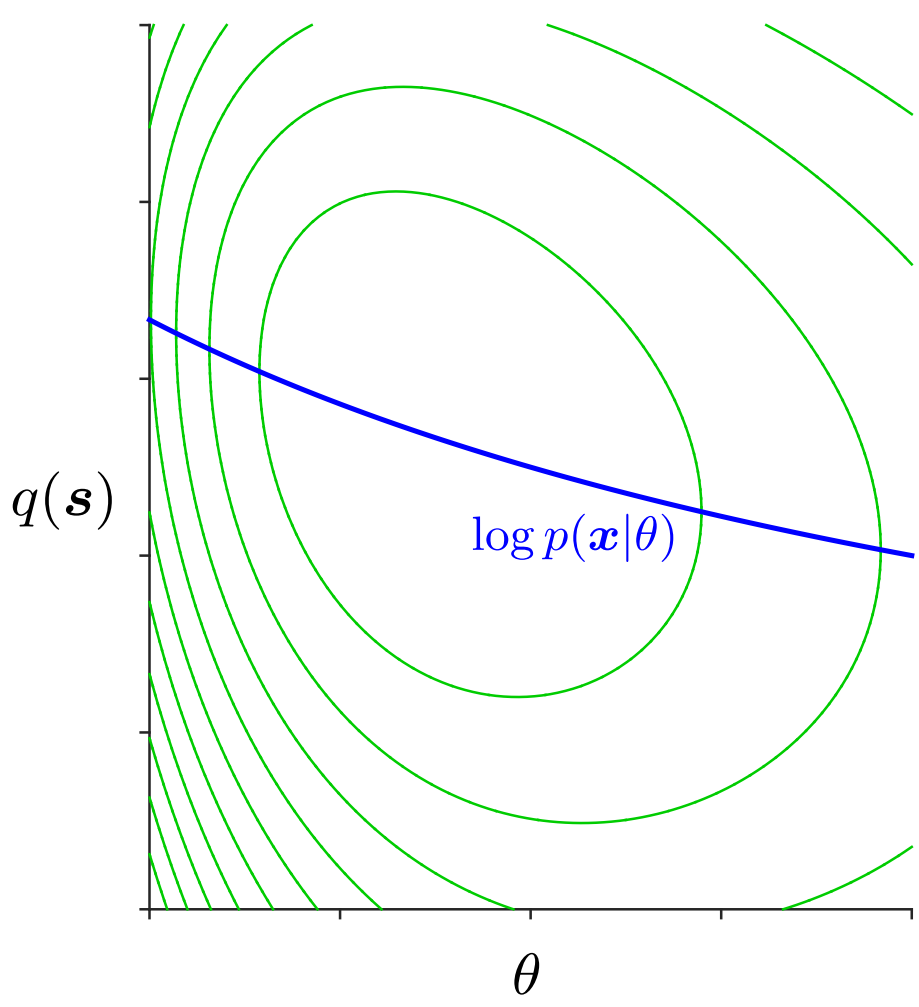




The Expectation Maximisation (EM) algorithm

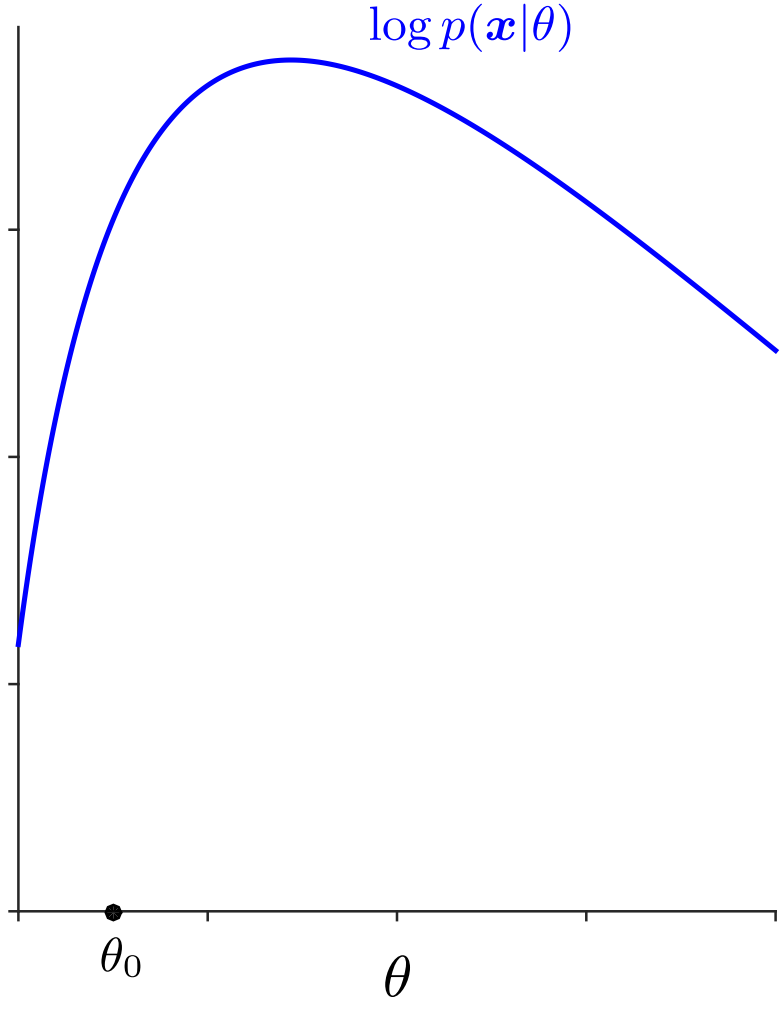
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

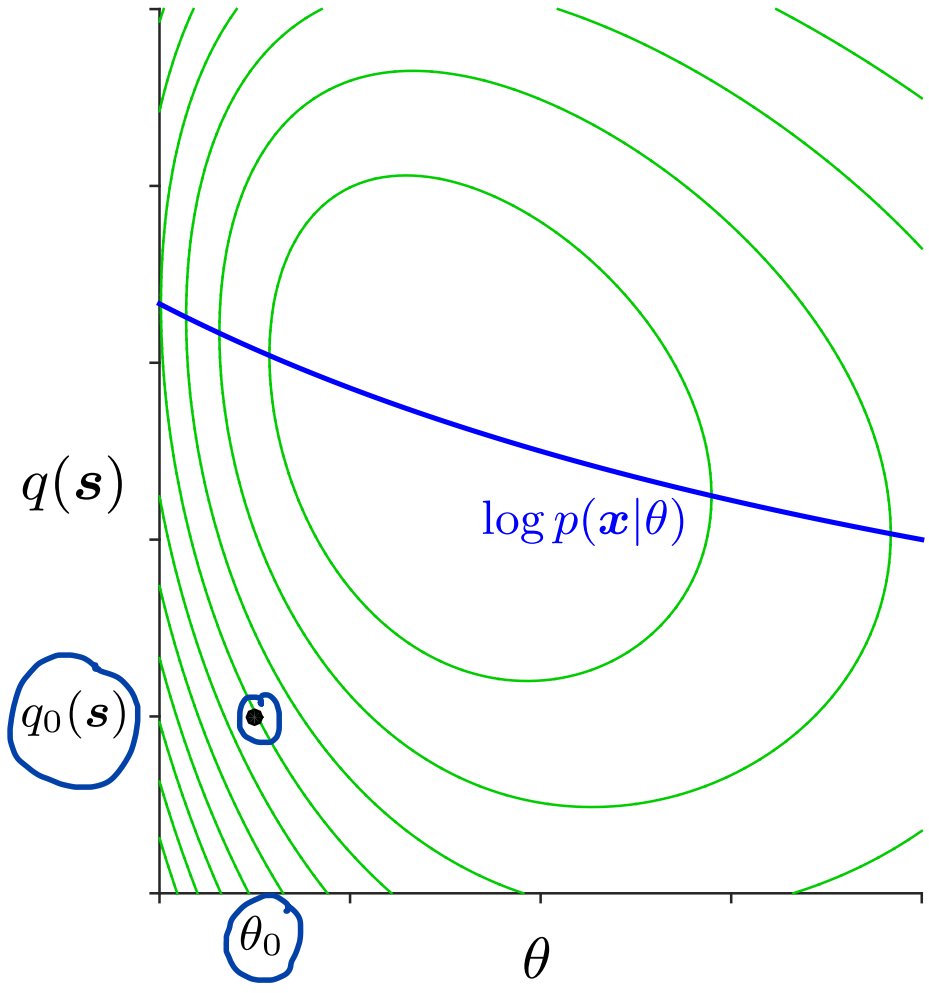




The Expectation Maximisation (EM) algorithm

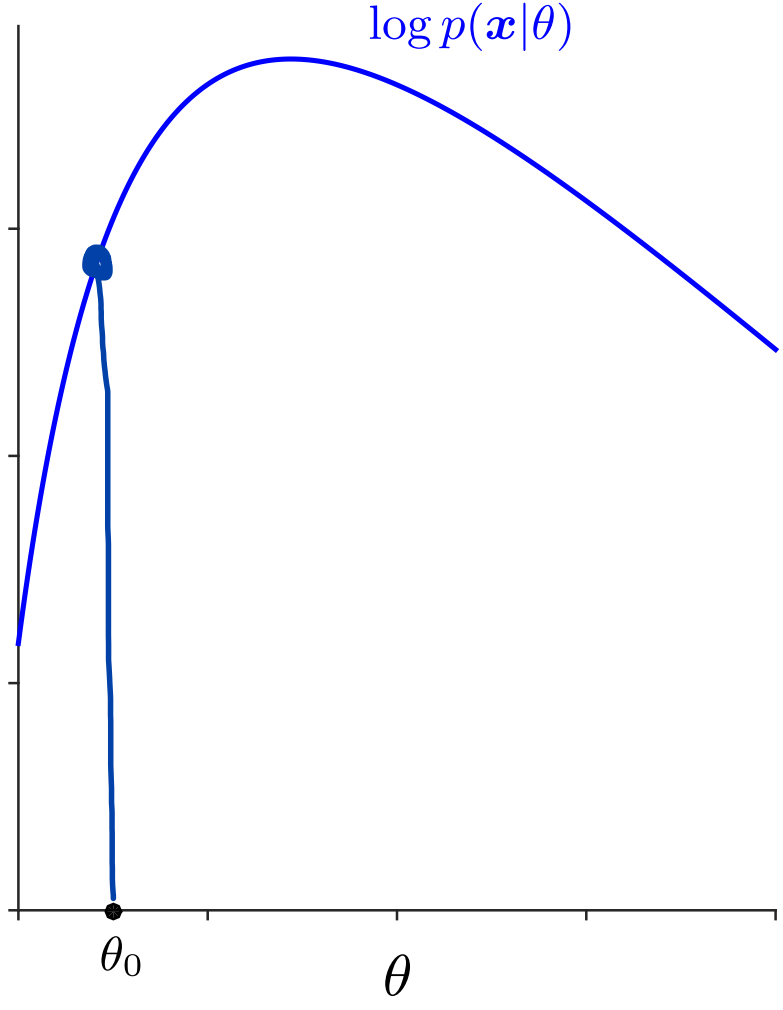
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

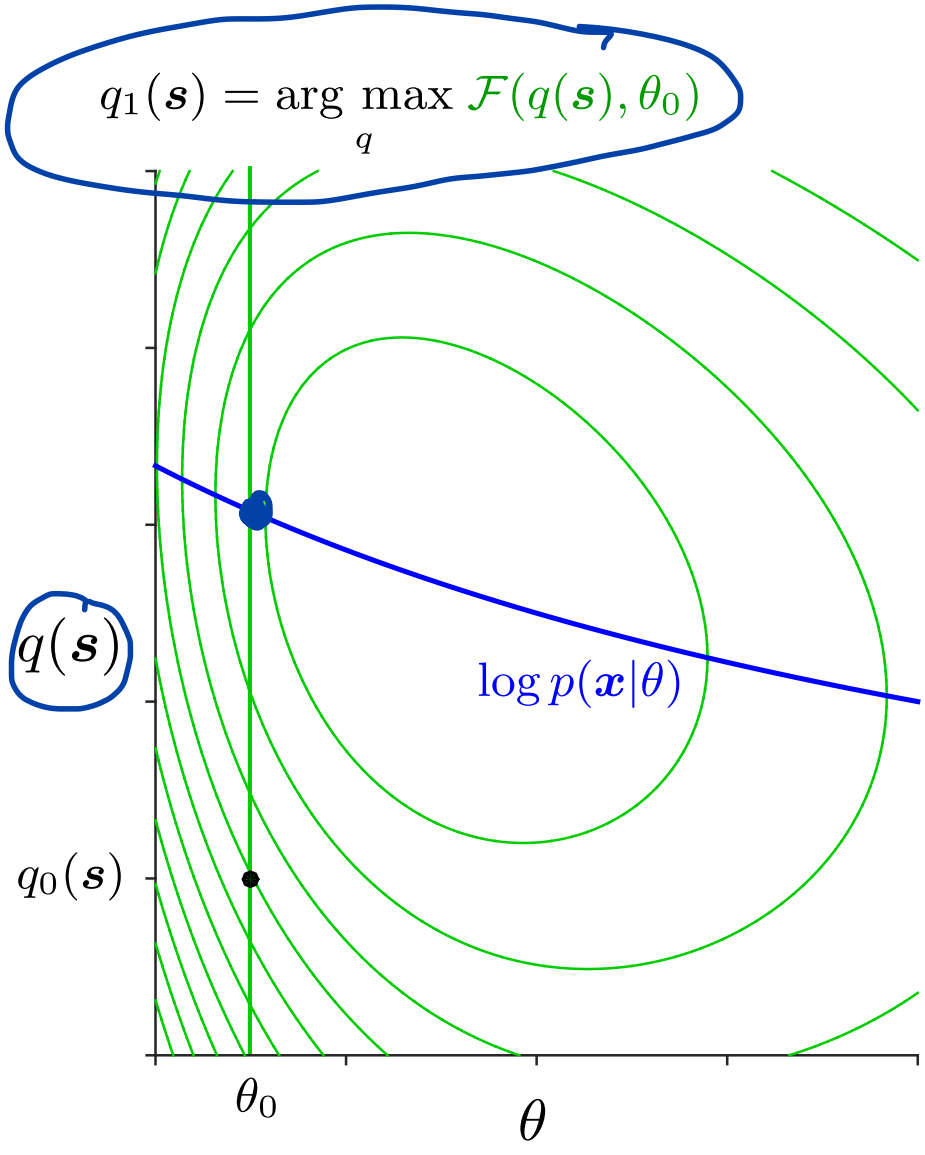




The Expectation Maximisation (EM) algorithm

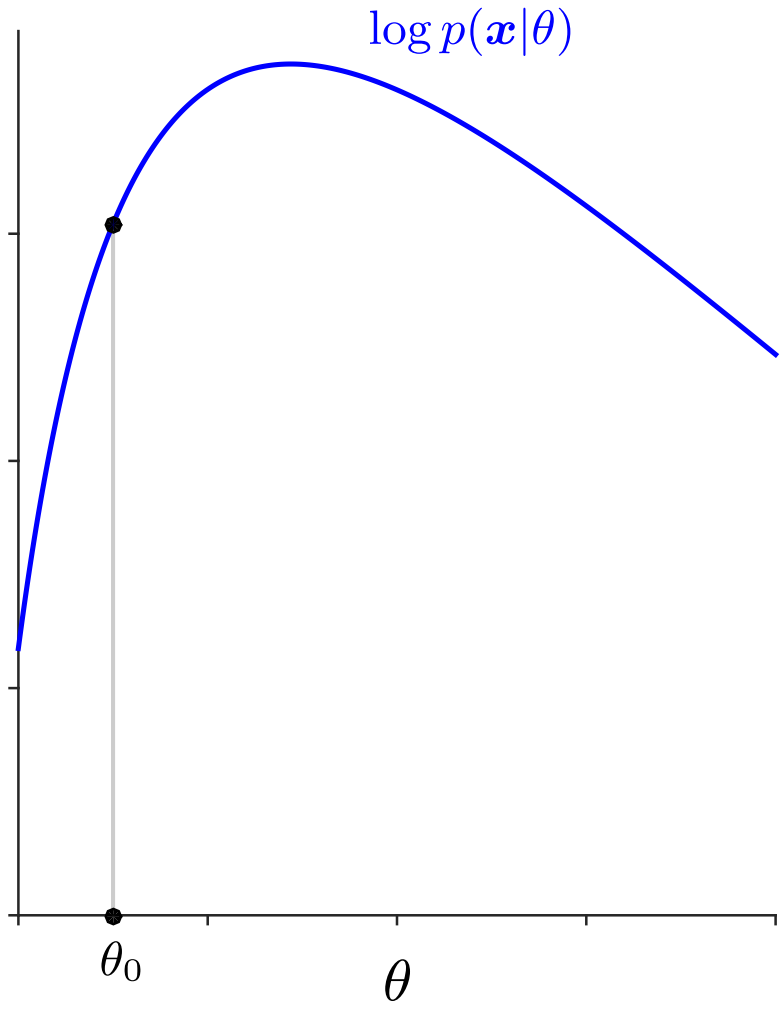
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$



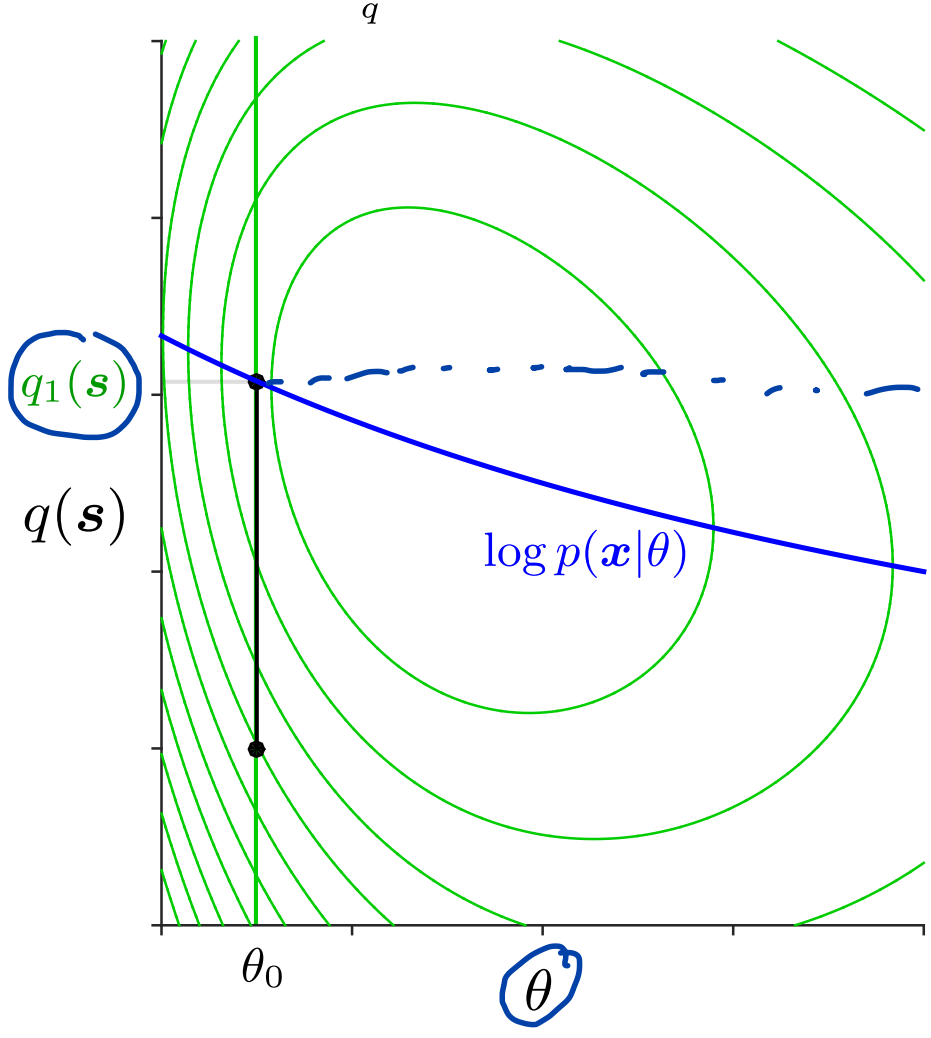


The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

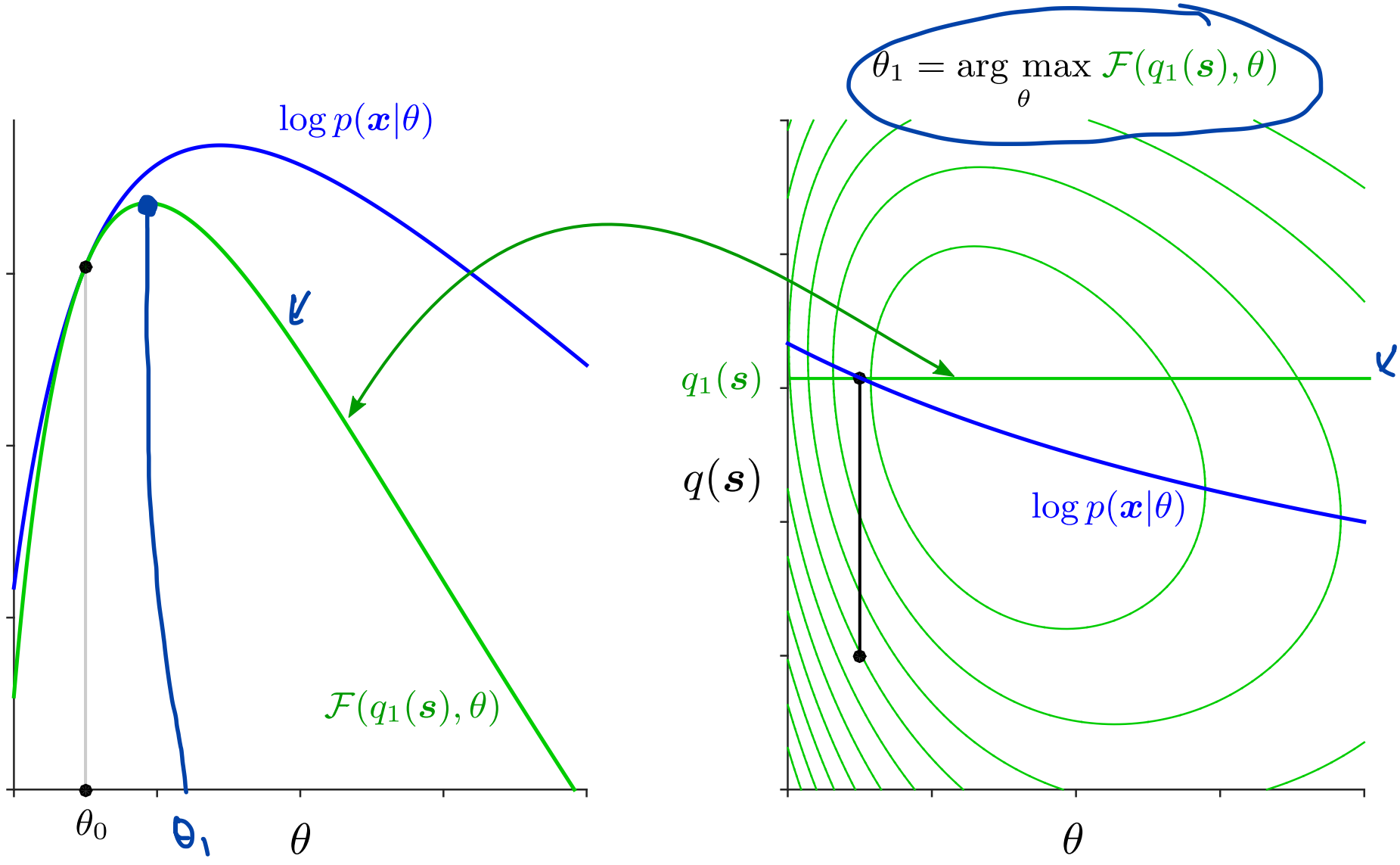


$$q_1(\boldsymbol{s}) = \arg\max_q \mathcal{F}(q(\boldsymbol{s}), \theta_0) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_0)$$



The Expectation Maximisation (EM) algorithm

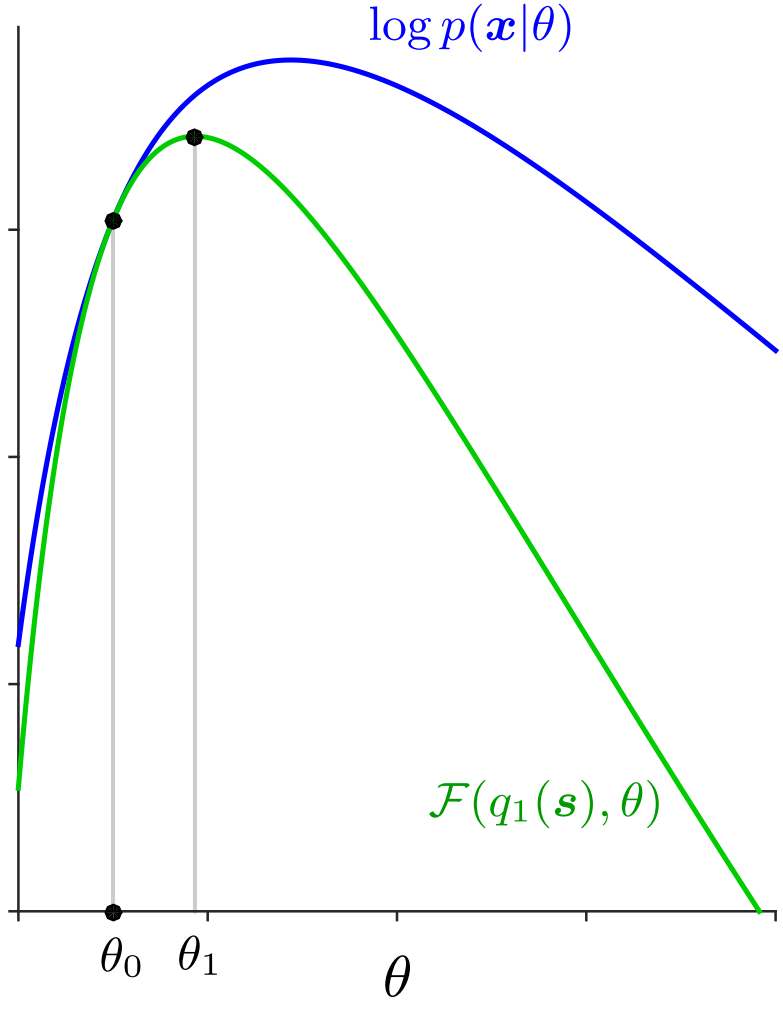
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

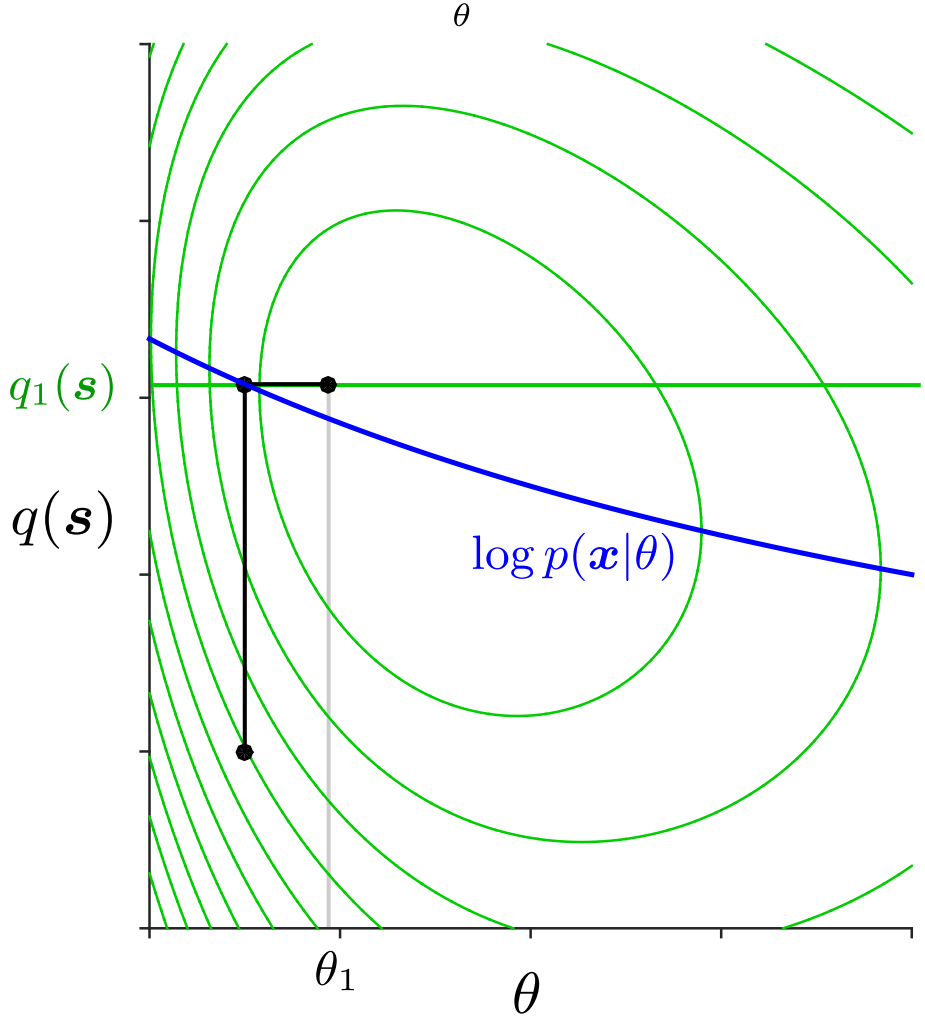


The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

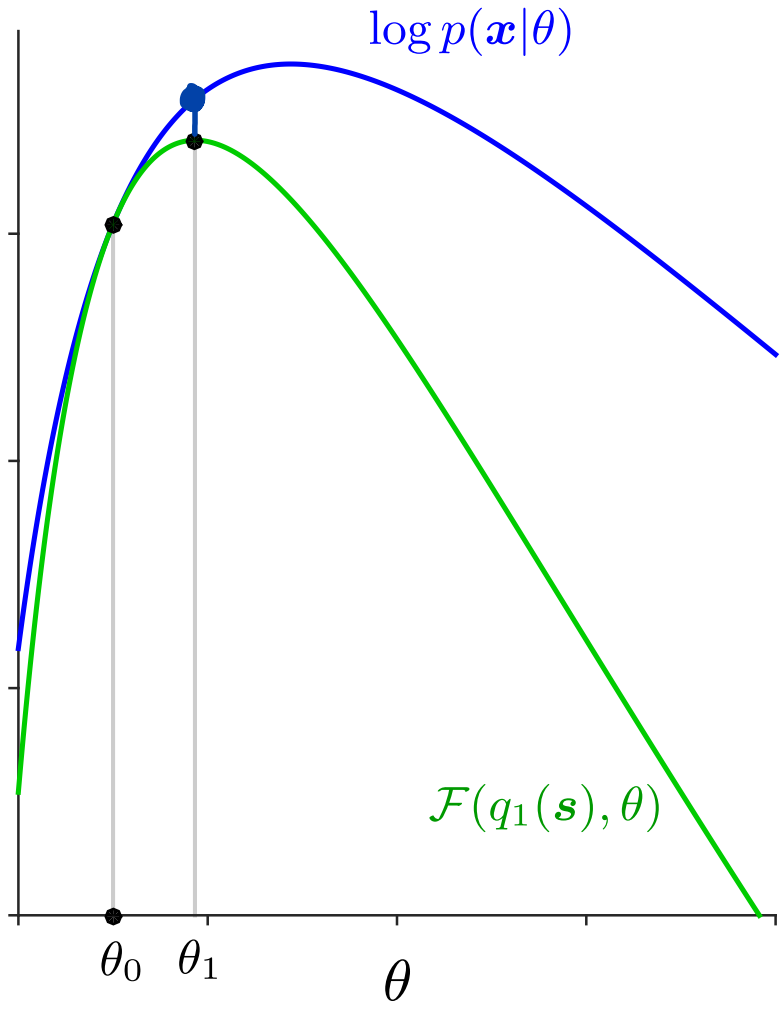
$$\theta_1 = \arg\max_{\theta} \mathcal{F}(q_1(\boldsymbol{s}), \theta)$$



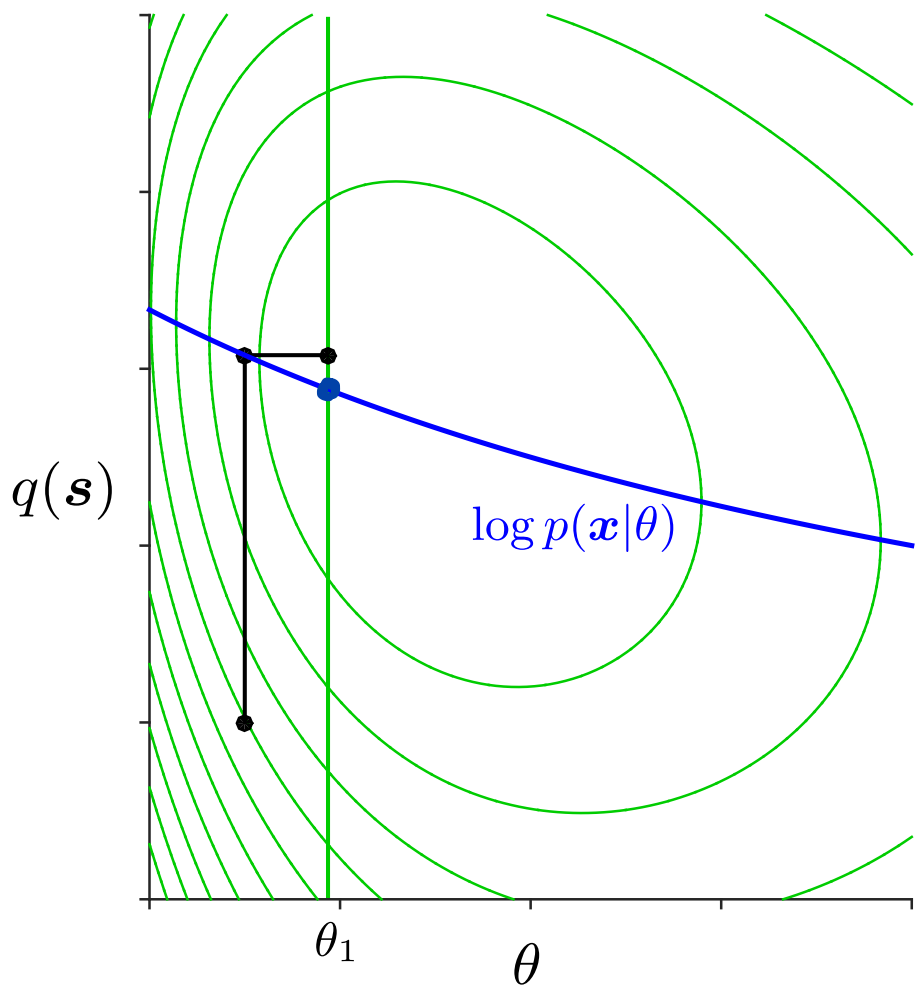


The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

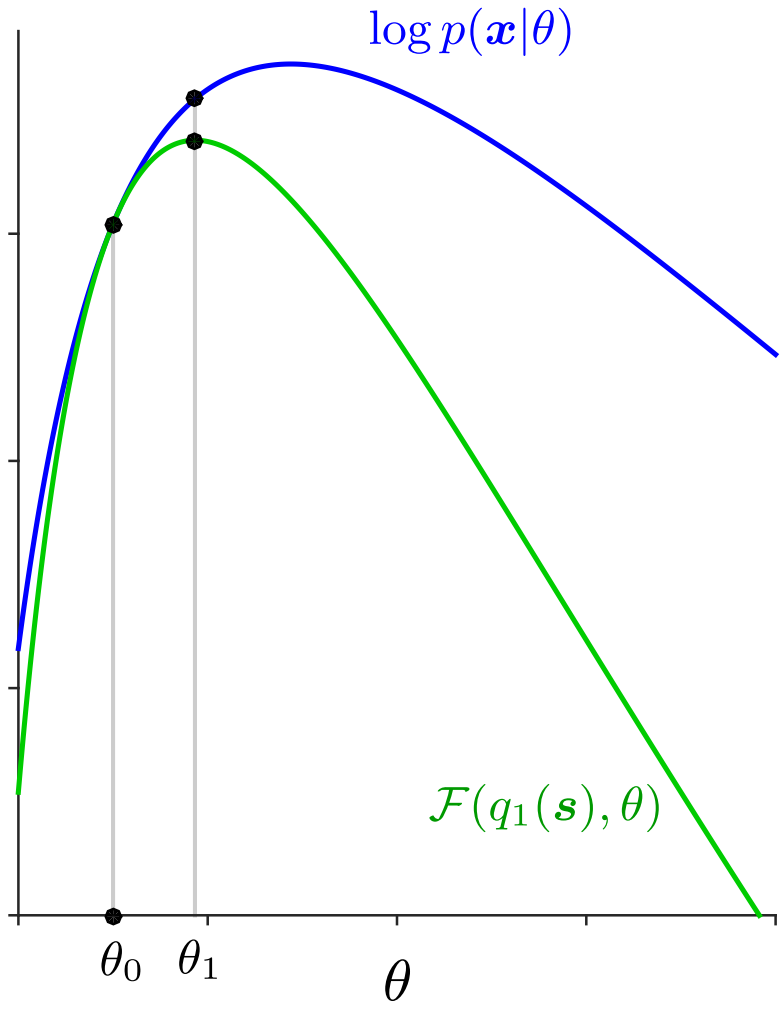


$$q_2(\boldsymbol{s}) = \arg\max_q \mathcal{F}(q(\boldsymbol{s}), \theta_1) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_1)$$

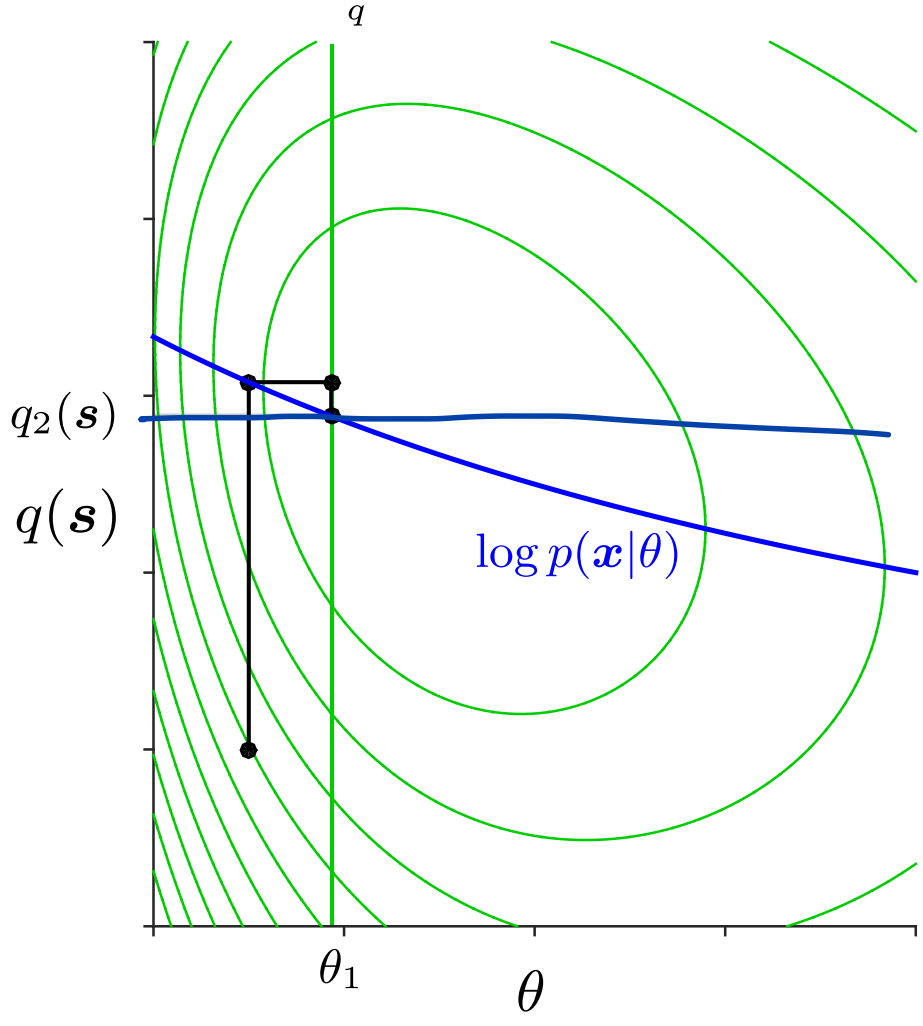


The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$

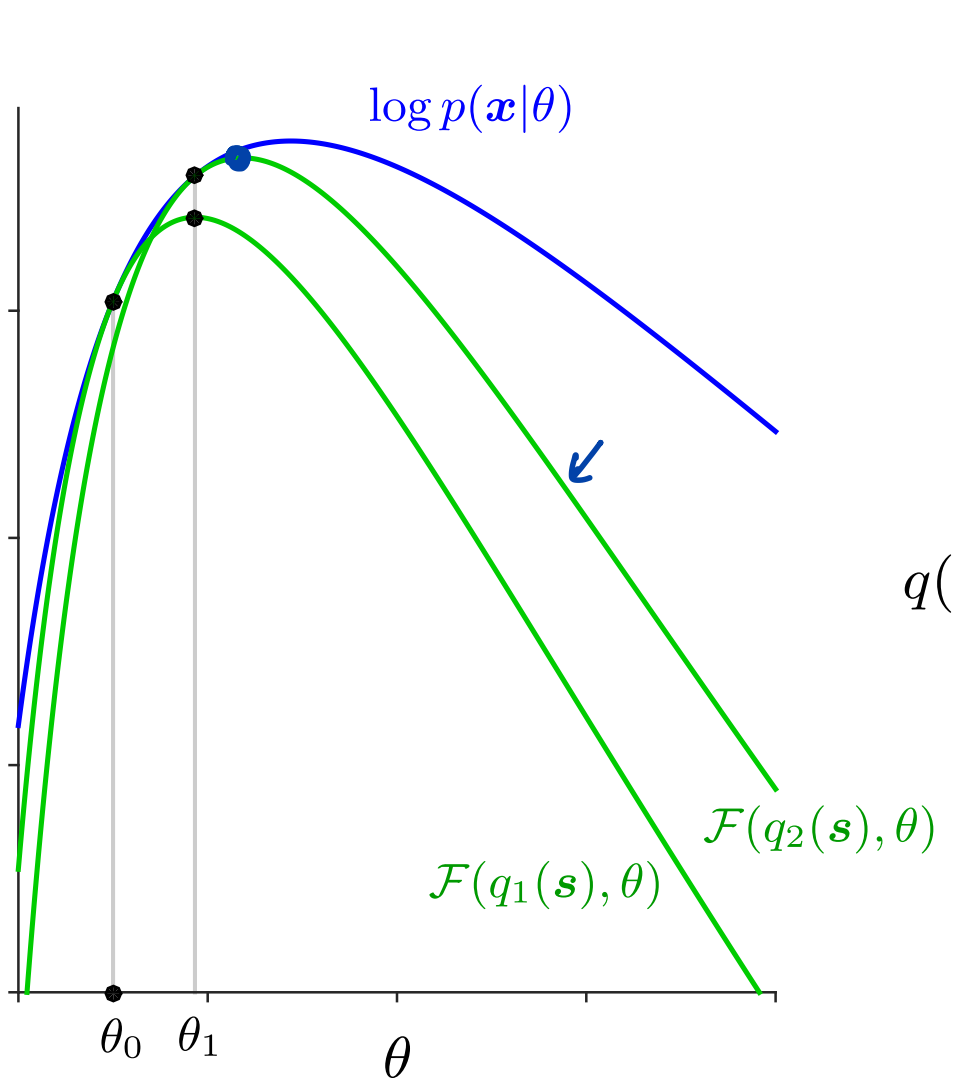


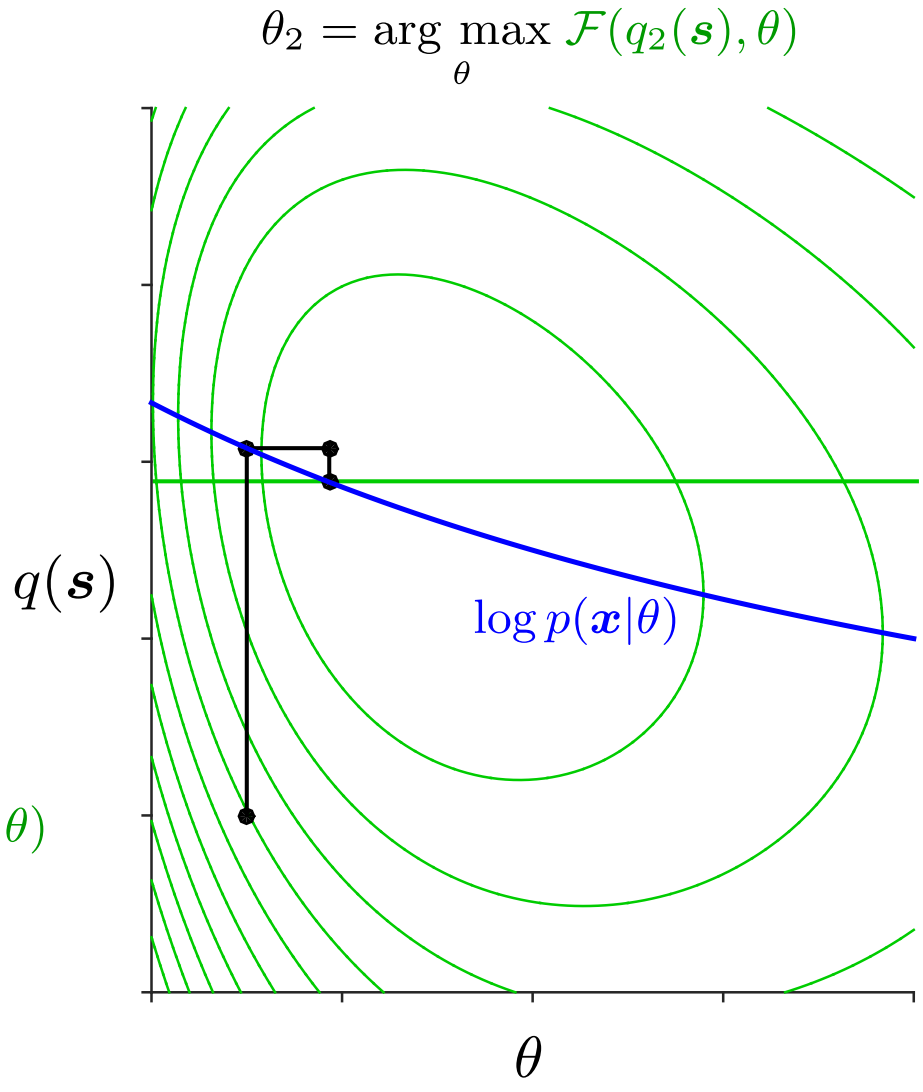
$$q_2(\boldsymbol{s}) = \arg\max_q \mathcal{F}(q(\boldsymbol{s}), \theta_1) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_1)$$



The Expectation Maximisation (EM) algorithm

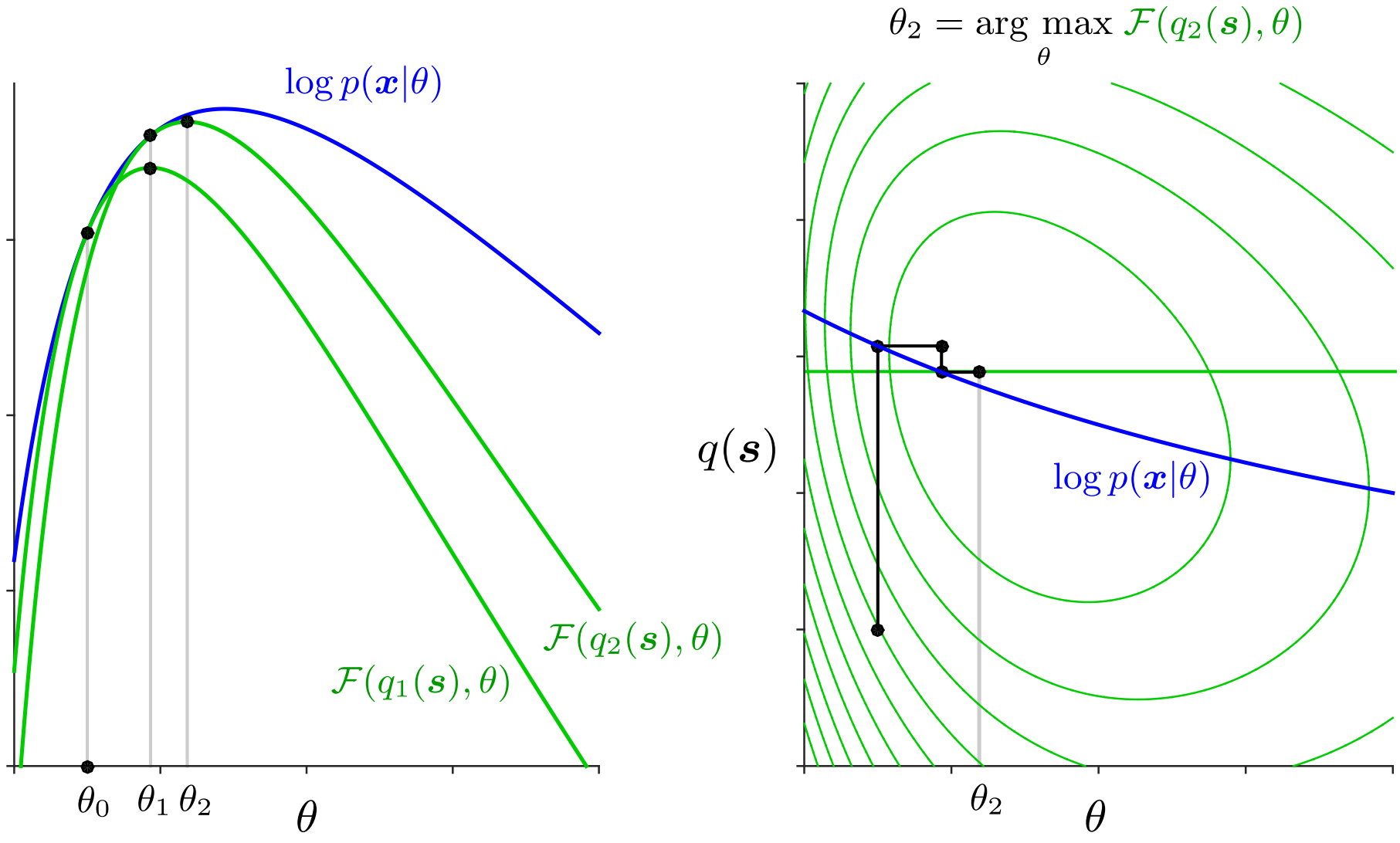
$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$





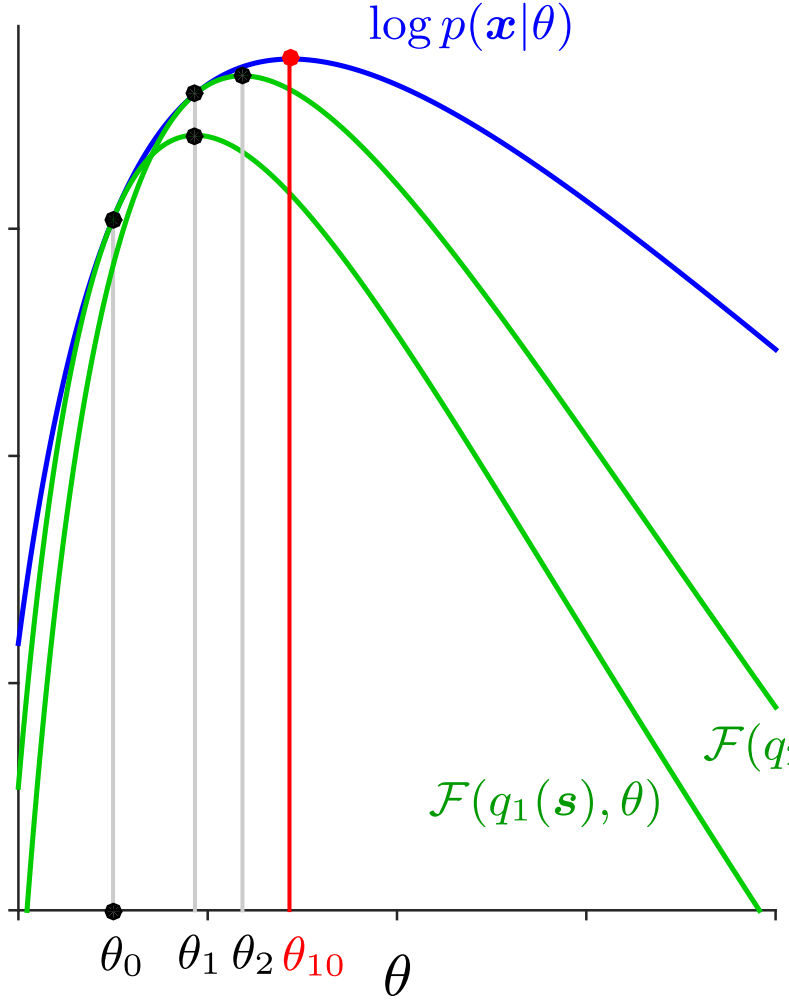
The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$



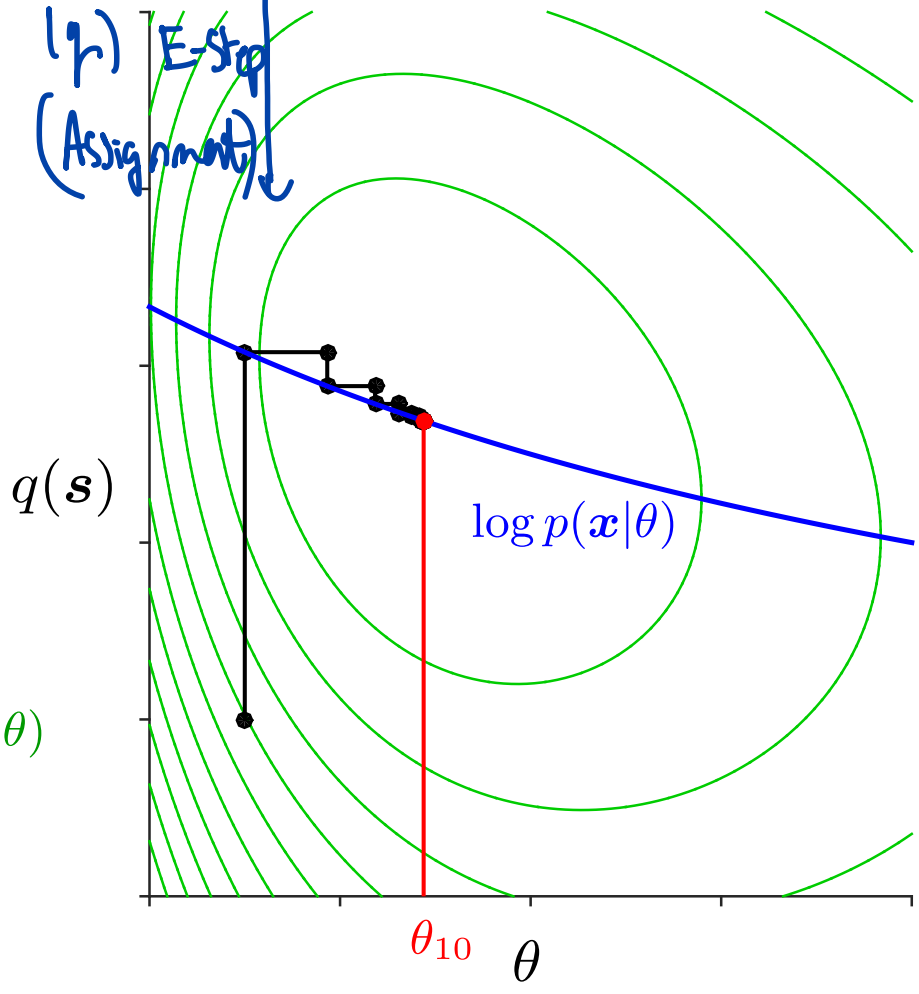
The Expectation Maximisation (EM) algorithm

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \log p(\boldsymbol{x}|\theta) - \mathcal{KL}(q(\boldsymbol{s}) \| p(\boldsymbol{s}|\boldsymbol{x}, \theta))$$



M-Step (θ)

E-Step (Assignment)



The Expectation Maximisation (EM) algorithm

From initial (random) parameters θ_0 iterate $t = 1, \ldots, T$ the two steps:

E step: for fixed θ_{t-1}, maximize lower bound $\mathcal{F}(q(\boldsymbol{s}), \theta_{t-1})$ wrt $q(\boldsymbol{s})$.
As log likelihood $\log p(\boldsymbol{x}|\theta)$ is independent of $q(\boldsymbol{s})$ this is equivalent to minimizing $\mathcal{KL}(q(\boldsymbol{s})||p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1}))$, so $q_t(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1})$.

M step: for fixed $q_t(\boldsymbol{s})$ maximize the lower bound $\mathcal{F}(q_t(\boldsymbol{s}), \theta)$ wrt θ.

$$\Rightarrow \mathcal{F}(q(\boldsymbol{s}), \theta) = \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \left(p(\boldsymbol{x}|\boldsymbol{s}, \theta) p(\boldsymbol{s}|\theta) \right) - \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log q(\boldsymbol{s})$$

$$\mathcal{F}(q(s), \theta) = \log p(x|\theta) - KL\left(q(s) \,||\, p(s|x, \theta)\right)$$

$$= \log p(x|\theta) - \sum_s q(s) \log q(s)$$

Entropy

product rule: $p(x, s|\theta) = p(s|\theta) \cdot p(x|s, \theta)$

model specifies this

$$= \sum_s q(s) \log p(x|\theta) + \sum_s q(s) \log p(s|x, \theta) - \sum_s q(s) \log q(s)$$

($\sum_s q(s) = 1$; $p(s|x, \theta)$)

$$= \sum_s q(s) \log \left[p(x|\theta) p(s|x, \theta) \right] - \sum_s q(s) \log q(s)$$

The Expectation Maximisation (EM) algorithm

From initial (random) parameters θ_0 iterate $t = 1, \ldots, T$ the two steps:

E step: for fixed θ_{t-1}, maximize lower bound $\mathcal{F}(q(\boldsymbol{s}), \theta_{t-1})$ wrt $q(\boldsymbol{s})$.
As log likelihood $\log p(\boldsymbol{x}|\theta)$ is independent of $q(\boldsymbol{s})$ this is equivalent to minimizing $\mathcal{KL}(q(\boldsymbol{s})||p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1}))$, so $q_t(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1})$.

M step: for fixed $q_t(\boldsymbol{s})$ maximize the lower bound $\mathcal{F}(q_t(\boldsymbol{s}), \theta)$ wrt θ.

$$\mathcal{F}(q(\boldsymbol{s}), \theta) \;=\; \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \left(p(\boldsymbol{x}|\boldsymbol{s}, \theta) p(\boldsymbol{s}|\theta)\right) - \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log q(\boldsymbol{s}),$$

the second term is the entropy of $q(\boldsymbol{s})$, independent of θ, so the M step is

$$\theta_t \;=\; \operatorname*{argmax}_{\theta} \sum_{\boldsymbol{s}} q_t(\boldsymbol{s}) \log \left(p(\boldsymbol{x}|\boldsymbol{s}, \theta) p(\boldsymbol{s}|\theta)\right).$$

The Expectation Maximisation (EM) algorithm

From initial (random) parameters θ_0 iterate $t = 1, \ldots, T$ the two steps:

E step: for fixed θ_{t-1}, maximize lower bound $\mathcal{F}(q(\boldsymbol{s}), \theta_{t-1})$ wrt $q(\boldsymbol{s})$.
As log likelihood $\log p(\boldsymbol{x}|\theta)$ is independent of $q(\boldsymbol{s})$ this is equivalent to minimizing $\mathcal{KL}(q(\boldsymbol{s})||p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1}))$, so $q_t(\boldsymbol{s}) = p(\boldsymbol{s}|\boldsymbol{x}, \theta_{t-1})$.

M step: for fixed $q_t(\boldsymbol{s})$ maximize the lower bound $\mathcal{F}(q_t(\boldsymbol{s}), \theta)$ wrt θ.

$$\mathcal{F}(q(\boldsymbol{s}), \theta) = \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log \left(p(\boldsymbol{x}|\boldsymbol{s}, \theta) p(\boldsymbol{s}|\theta) \right) - \sum_{\boldsymbol{s}} q(\boldsymbol{s}) \log q(\boldsymbol{s}),$$

the second term is the entropy of $q(\boldsymbol{s})$, independent of θ, so the M step is

$$\theta_t = \operatorname*{argmax}_{\theta} \sum_{\boldsymbol{s}} q_t(\boldsymbol{s}) \log \left(p(\boldsymbol{x}|\boldsymbol{s}, \theta) p(\boldsymbol{s}|\theta) \right).$$

Although the steps work with the lower bound (Lyupanov* function), each iteration cannot decrease the log likelihood as

$$\log p(\boldsymbol{x}|\theta_{t-1}) \overset{\text{E step}}{=} \mathcal{F}(q_t(\boldsymbol{s}), \theta_{t-1}) \overset{\text{M step}}{\leq} \mathcal{F}(q_t(\boldsymbol{s}), \theta_t) \overset{\text{lower bound}}{\leq} \log p(\boldsymbol{x}|\theta_t)$$

Application of EM to Mixture of Gaussians (E Step)

- Assume $D = 1$ dimensional data for x simplicity
- Gaussian mixture model parameters: $\theta = \{\mu_k, \sigma_k^2, \pi_k\}_{k=1\ldots K}$
- One latent variable per datapoint $s_n, n = 1 \ldots N$ takes values $1 \ldots K$.

Probability of the observations given the latent variables and the parameters, and the prior on latent variables are:

$$p(x_n|s_n = k, \theta) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \mathrm{e}^{-\frac{1}{2\sigma_k^2}(x_n - \mu_k)^2} \qquad p(s_n = k|\theta) = \pi_k$$

Application of EM to Mixture of Gaussians (E Step)

- Assume $D = 1$ dimensional data for x simplicity
- Gaussian mixture model parameters: $\theta = \{\mu_k, \sigma_k^2, \pi_k\}_{k=1\ldots K}$
- One latent variable per datapoint $s_n, n = 1 \ldots N$ takes values $1 \ldots K$.

Probability of the observations given the latent variables and the parameters, and the prior on latent variables are:

$$p(x_n|s_n = k, \theta) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{1}{2\sigma_k^2}(x_n - \mu_k)^2} \qquad p(s_n = k|\theta) = \pi_k$$

so the E step becomes:

$$q(s_n = k) = p(s_n = k|x_n, \theta) \propto p(x_n, s_n = k|\theta) = \frac{\pi_k}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{1}{2\sigma_k^2}(x_n - \mu_k)^2} = u_{nk}$$

$p(s_n = k|x_n, \theta) \propto p(s_n = k|\theta)\, p(x_n|s_n = k, \theta)$ (Bayes' Rule) $\Rightarrow p(\cancel{x_n}|\theta)$

That is: $q(s_n = k) = r_{nk} = \frac{u_{nk}}{u_n}$ where $u_n = \sum_{k=1}^{K} u_{nk}$ — $p(x_n|\theta)$

responsibility

Application of EM to Mixture of Gaussians (E Step)

- Assume $D = 1$ dimensional data for x simplicity
- Gaussian mixture model parameters: $\theta = \{\mu_k, \sigma_k^2, \pi_k\}_{k=1\ldots K}$
- One latent variable per datapoint $s_n, n = 1 \ldots N$ takes values $1 \ldots K$.

Probability of the observations given the latent variables and the parameters, and the prior on latent variables are:

$$p(x_n|s_n = k, \theta) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \mathrm{e}^{-\frac{1}{2\sigma_k^2}(x_n - \mu_k)^2} \qquad p(s_n = k|\theta) = \pi_k$$

so the E step becomes:

$$q(s_n = k) = p(s_n = k|x_n, \theta) \propto p(x_n, s_n = k|\theta) = \frac{\pi_k}{\sqrt{2\pi\sigma_k^2}} \mathrm{e}^{-\frac{1}{2\sigma_k^2}(x_n - \mu_k)^2} = u_{nk}$$

That is: $q(s_n = k) = r_{nk} = \frac{u_{nk}}{u_n}$ where $u_n = \sum_{k=1}^{K} u_{nk}$

Posterior for each latent variable, s_n follows a categorical distribution with probability given by the product of the prior and likelihood, renormalised. r_{nk} is called the *responsibility* that component k takes for data point n.

Application of EM to Mixture of Gaussians (M Step)

The lower bound is ① $\sum_s q(s) \log [p(s|\theta)\, p(x|s,\theta)] - \sum_s q(s) \log q(s)$

② $$\mathcal{F}(q(\boldsymbol{s}), \theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} q(s_n = k) \left[\log(\pi_k) - \frac{1}{2\sigma_k^2}(x_n - \mu_k)^2 - \frac{1}{2}\log(\sigma_k^2) \right] + \text{const.}$$

π $\mathcal{N}(x_n; \mu_k, \sigma_k^2)$

Ⓐ iid $q(\underline{s}) = q(s_1, s_2, s_3, \dots s_N) = p(\underline{s} | \underline{x}, \theta)$

$$q(\underline{s}) = \prod_n q(s_n)$$

$$p(\underline{s}|\underline{x},\theta) = \prod_{n=1}^{N} p(s_n | \underline{x}_n, \theta)$$

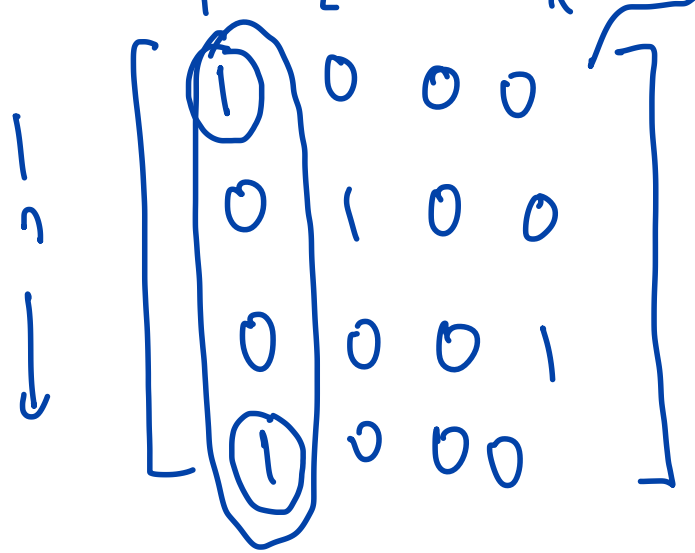
Application of EM to Mixture of Gaussians (M Step)

The lower bound is

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \sum_{n=1}^{N}\sum_{k=1}^{K} q(s_n=k)\left[\log(\pi_k) - \frac{1}{2\sigma_k^2}(x_n-\mu_k)^2 - \frac{1}{2}\log(\sigma_k^2)\right] + \text{const.}$$

The M step, optimizing $\mathcal{F}(q(\boldsymbol{s}),\theta)$ wrt the parameters, θ

$$\frac{\partial \mathcal{F}}{\partial \mu_j} = \sum_{n=1}^{N} q(s_n=j)\frac{x_n-\mu_j}{\sigma_j^2} = 0 \Rightarrow \mu_j = \frac{\sum_{n=1}^{N} q(s_n=j)\, x_n}{\sum_{n=1}^{N} q(s_n=j)},$$



$$\mu_1 = \frac{x_1 + x_4}{2}$$

= mean of the data points that are assigned to this cluster

Application of EM to Mixture of Gaussians (M Step)

The lower bound is

$$\mathcal{F}(q(\boldsymbol{s}),\theta) \;=\; \sum_{n=1}^{N}\sum_{k=1}^{K} q(s_n=k)\big[\log(\pi_k) - \tfrac{1}{2\sigma_k^2}(x_n-\mu_k)^2 - \tfrac{1}{2}\log(\sigma_k^2)\big] + \text{const.}$$

The M step, optimizing $\mathcal{F}(q(\boldsymbol{s}),\theta)$ wrt the parameters, θ

$$\frac{\partial \mathcal{F}}{\partial \mu_j} = \sum_{n=1}^{N} q(s_n=j)\frac{x_n-\mu_j}{\sigma_j^2} \;=\; 0 \Rightarrow \mu_j = \frac{\sum_{n=1}^{N} q(s_n=j)x_n}{\sum_{n=1}^{N} q(s_n=j)},$$

E step fills in the values of the hidden variables: M step just like performing **supervised learning** with known (soft) cluster assignments.

Application of EM to Mixture of Gaussians (M Step)

The lower bound is

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \sum_{n=1}^{N}\sum_{k=1}^{K} q(s_n=k)\left[\log(\pi_k) - \frac{1}{2\sigma_k^2}(x_n-\mu_k)^2 - \frac{1}{2}\log(\sigma_k^2)\right] + \text{const.}$$

The M step, optimizing $\mathcal{F}(q(\boldsymbol{s}),\theta)$ wrt the parameters, θ

$$\frac{\partial \mathcal{F}}{\partial \mu_j} = \sum_{n=1}^{N} q(s_n=j)\frac{x_n-\mu_j}{\sigma_j^2} = 0 \Rightarrow \mu_j = \frac{\sum_{n=1}^{N} q(s_n=j)x_n}{\sum_{n=1}^{N} q(s_n=j)},$$

$$\frac{\partial \mathcal{F}}{\partial \sigma_j^2} = \sum_{n=1}^{N} q(s_n=j)\left[\frac{(x_n-\mu_j)^2}{2\sigma_j^4} - \frac{1}{2\sigma_j^2}\right] = 0 \Rightarrow \sigma_j^2 = \frac{\sum_{n=1}^{N} q(s_n=j)(x_n-\mu_j)^2}{\sum_{n=1}^{N} q(s_n=j)}.$$

E step fills in the values of the hidden variables: M step just like performing **supervised learning** with known (soft) cluster assignments.

Application of EM to Mixture of Gaussians (M Step)

The lower bound is

$$\mathcal{F}(q(\boldsymbol{s}),\theta) = \sum_{n=1}^{N}\sum_{k=1}^{K} q(s_n=k)\left[\log(\pi_k) - \frac{1}{2\sigma_k^2}(x_n-\mu_k)^2 - \frac{1}{2}\log(\sigma_k^2)\right] + \text{const.}$$

The M step, optimizing $\mathcal{F}(q(\boldsymbol{s}),\theta)$ wrt the parameters, θ

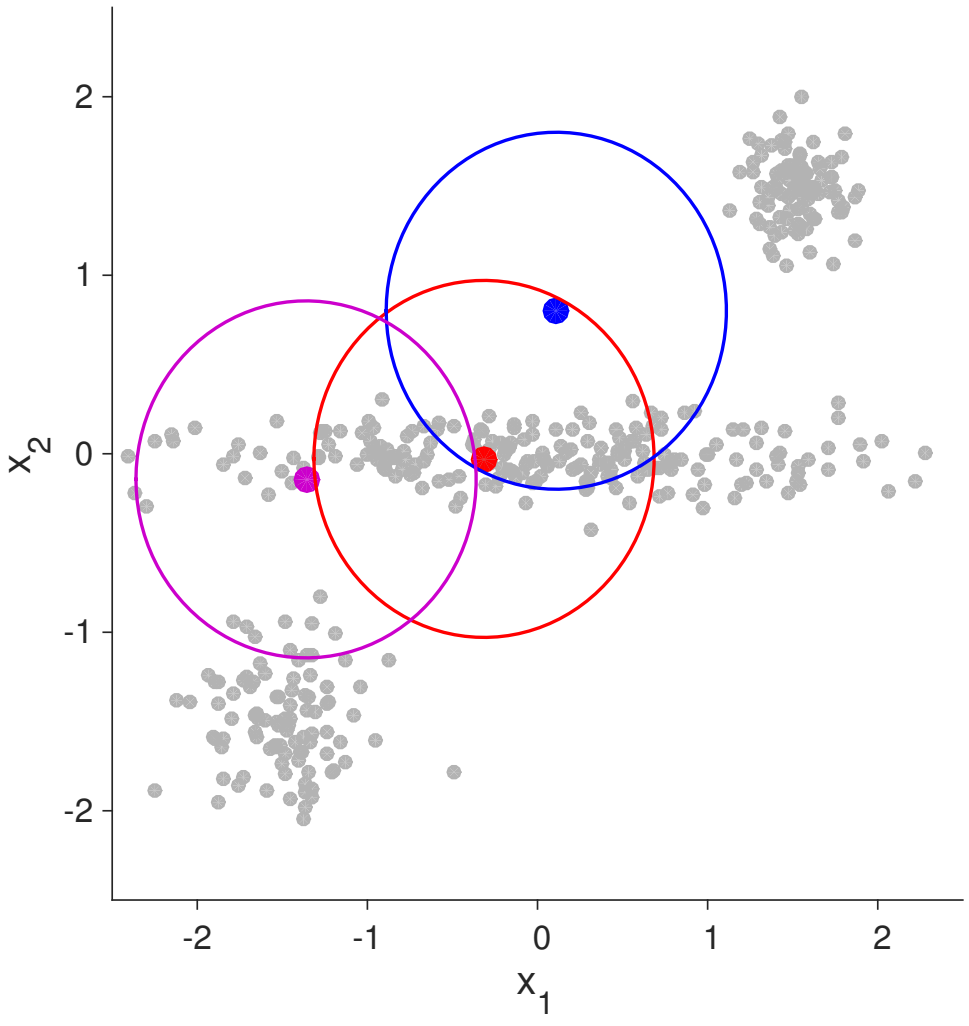
$$\frac{\partial \mathcal{F}}{\partial \mu_j} = \sum_{n=1}^{N} q(s_n=j)\frac{x_n-\mu_j}{\sigma_j^2} = 0 \Rightarrow \mu_j = \frac{\sum_{n=1}^{N} q(s_n=j)x_n}{\sum_{n=1}^{N} q(s_n=j)},$$

$$\frac{\partial \mathcal{F}}{\partial \sigma_j^2} = \sum_{n=1}^{N} q(s_n=j)\left[\frac{(x_n-\mu_j)^2}{2\sigma_j^4} - \frac{1}{2\sigma_j^2}\right] = 0 \Rightarrow \sigma_j^2 = \frac{\sum_{n=1}^{N} q(s_n=j)(x_n-\mu_j)^2}{\sum_{n=1}^{N} q(s_n=j)},$$

$$\frac{\partial[\mathcal{F} + \lambda(1-\sum_k \pi_k)]}{\partial \pi_j} = \sum_{n=1}^{N} \frac{q(s_n=j)}{\pi_j} - \lambda = 0 \Rightarrow \pi_j = \frac{1}{N}\sum_{n=1}^{N} q(s_n=j)$$

E step fills in the values of the hidden variables: M step just like performing **supervised learning** with known (soft) cluster assignments.

EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

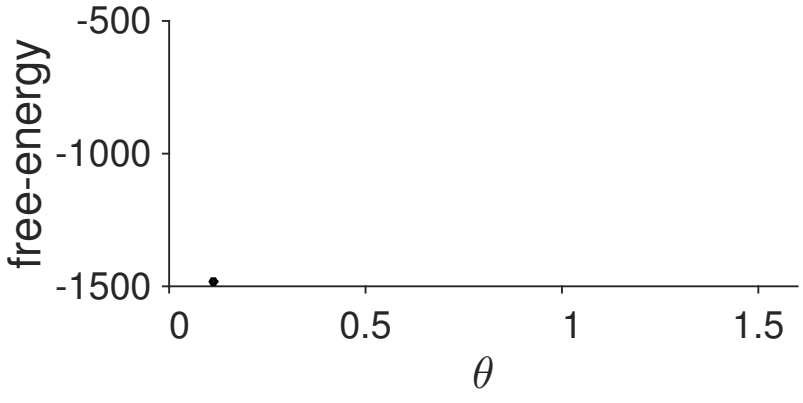
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

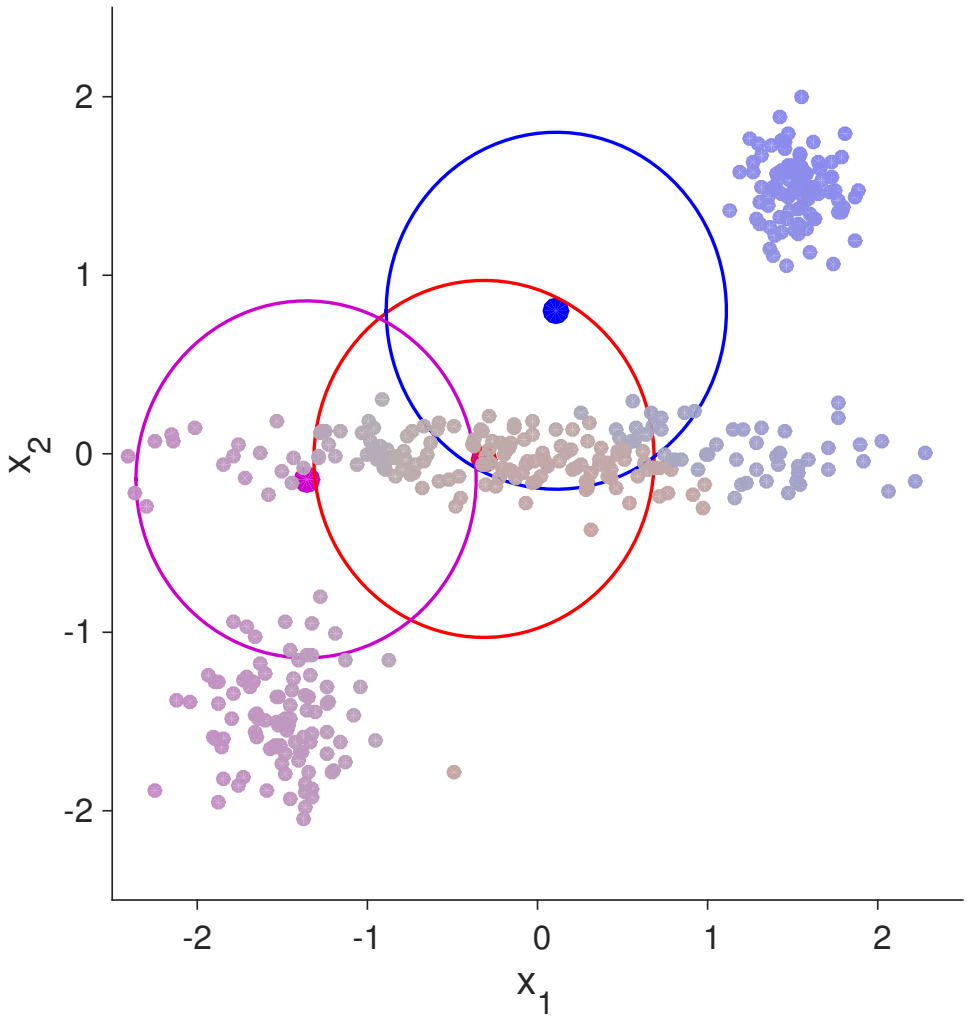
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

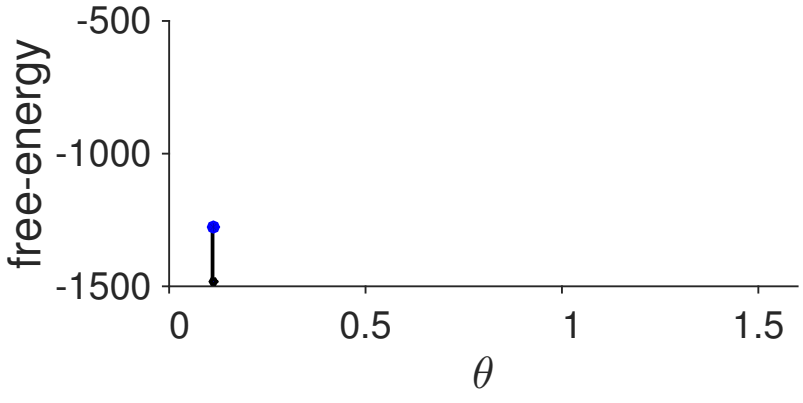
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

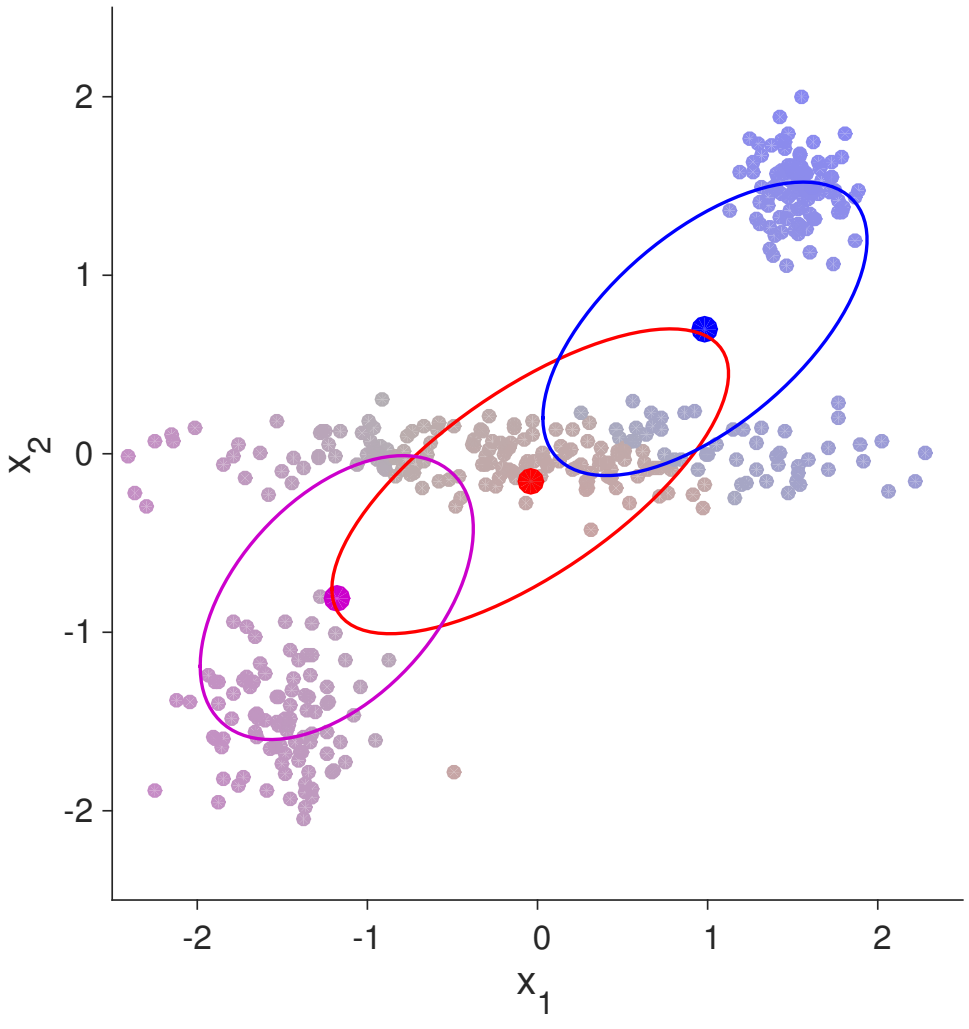
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

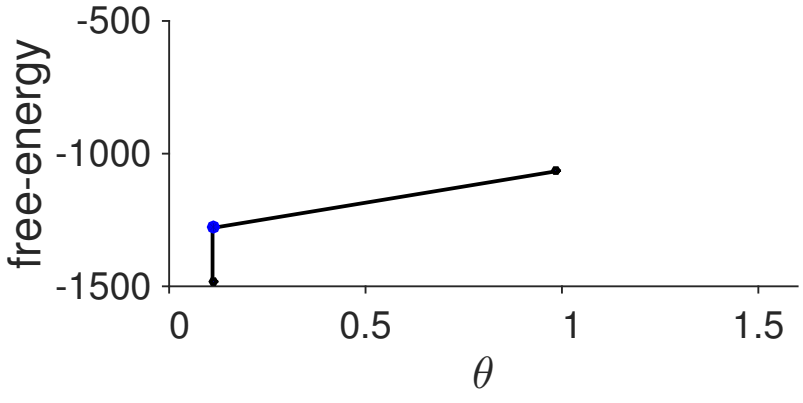
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

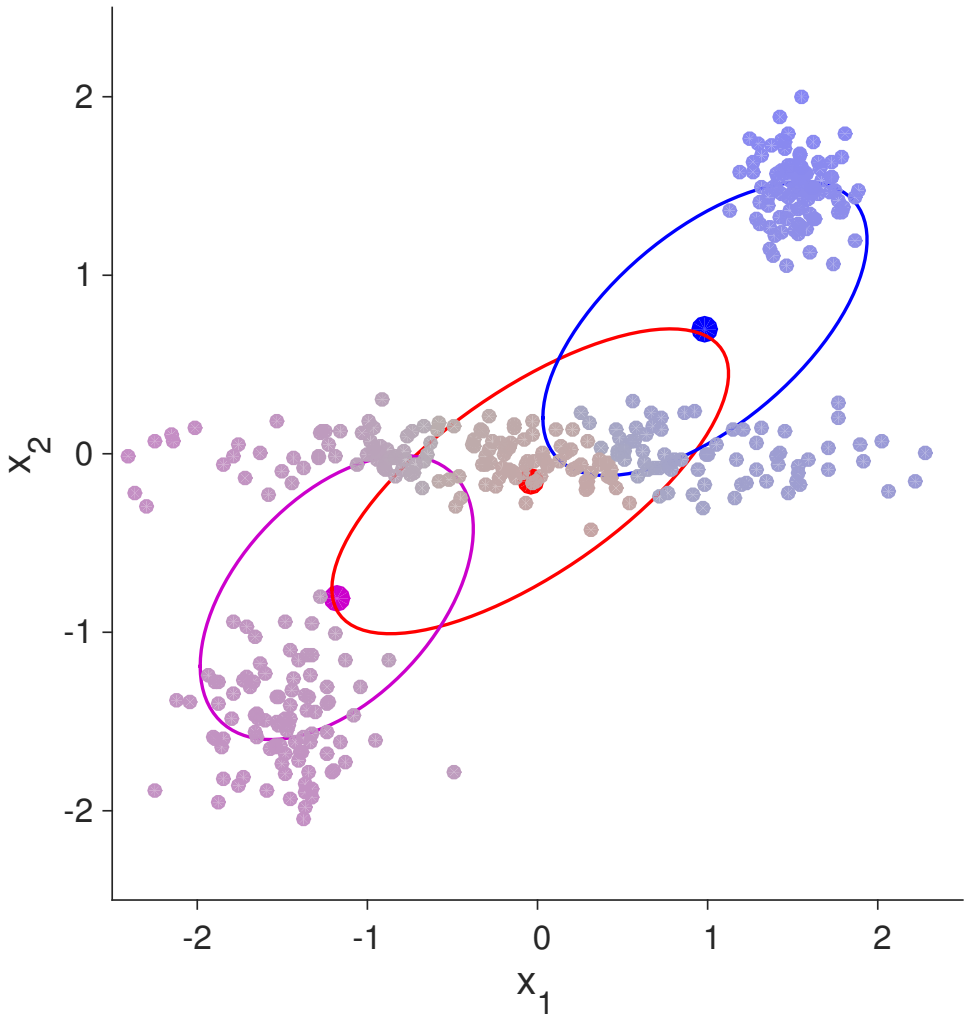
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

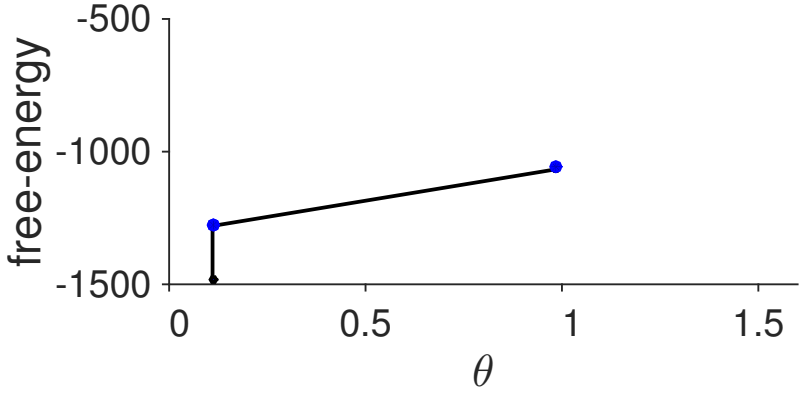
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

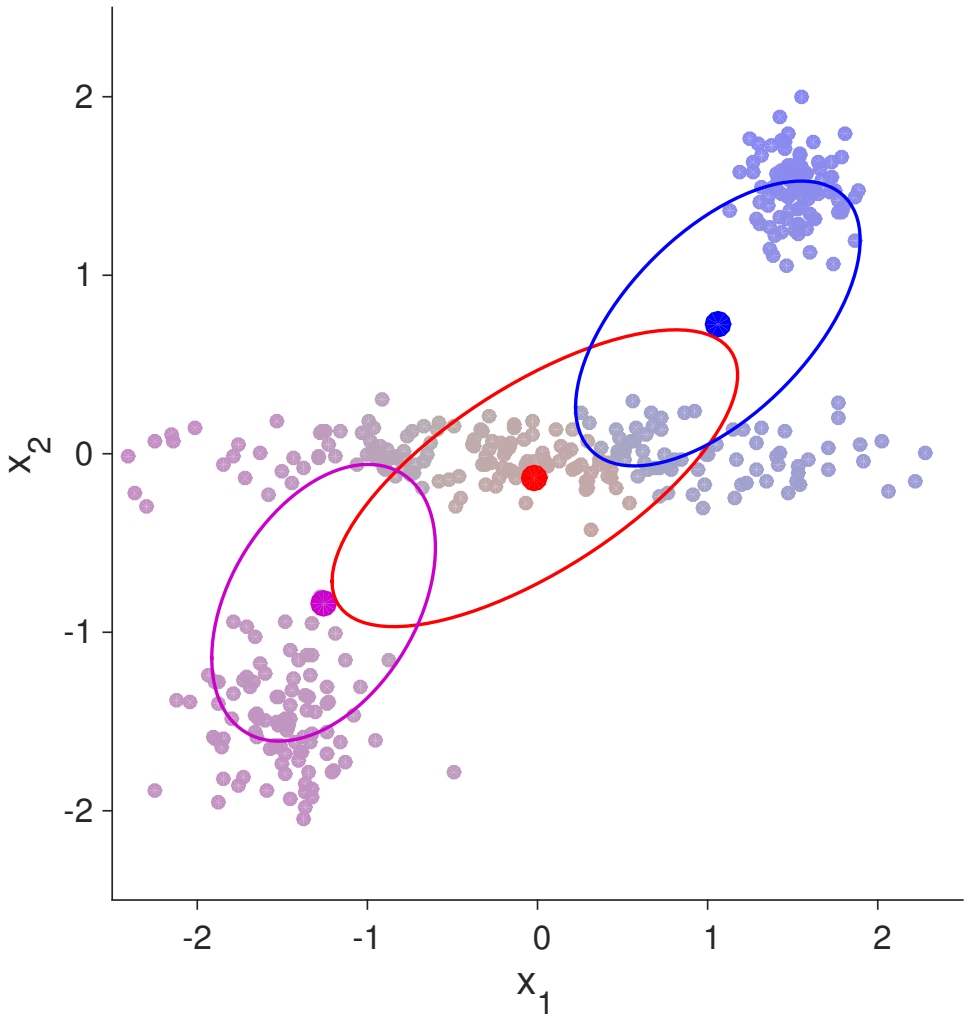
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

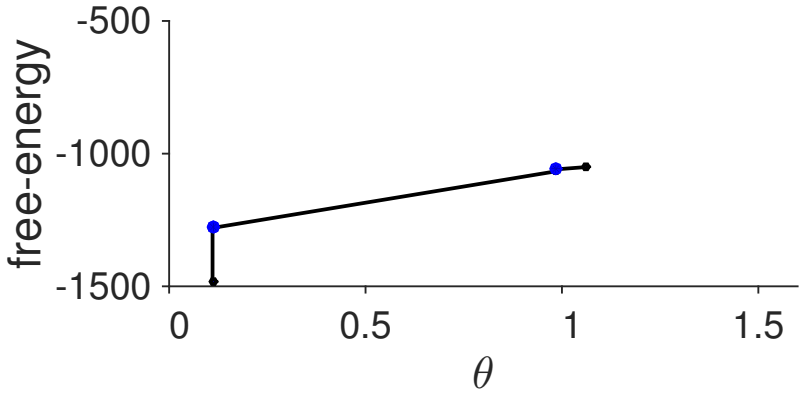
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

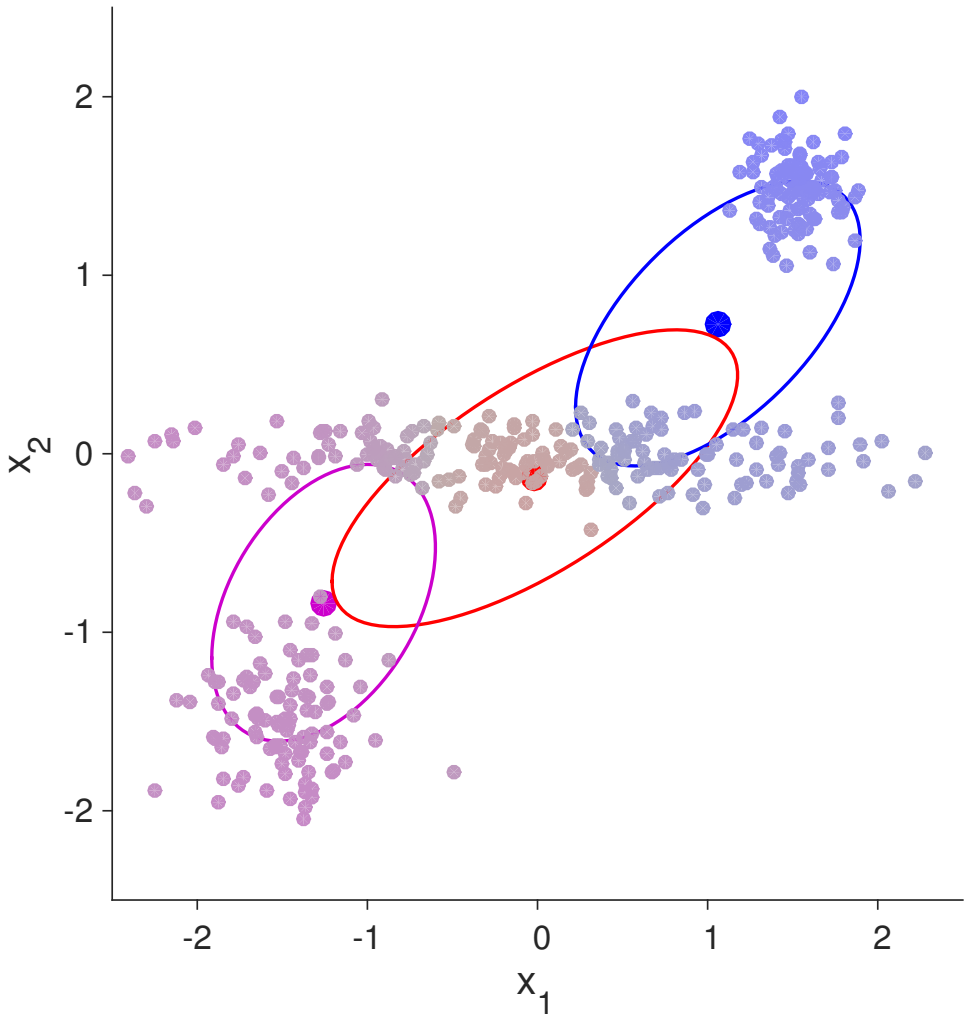
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

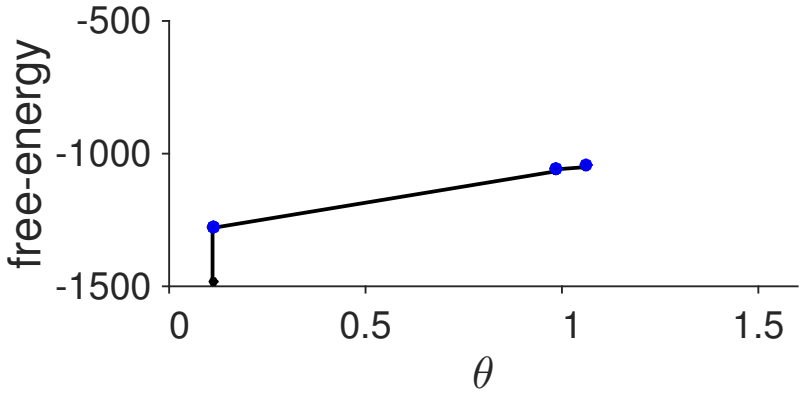
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

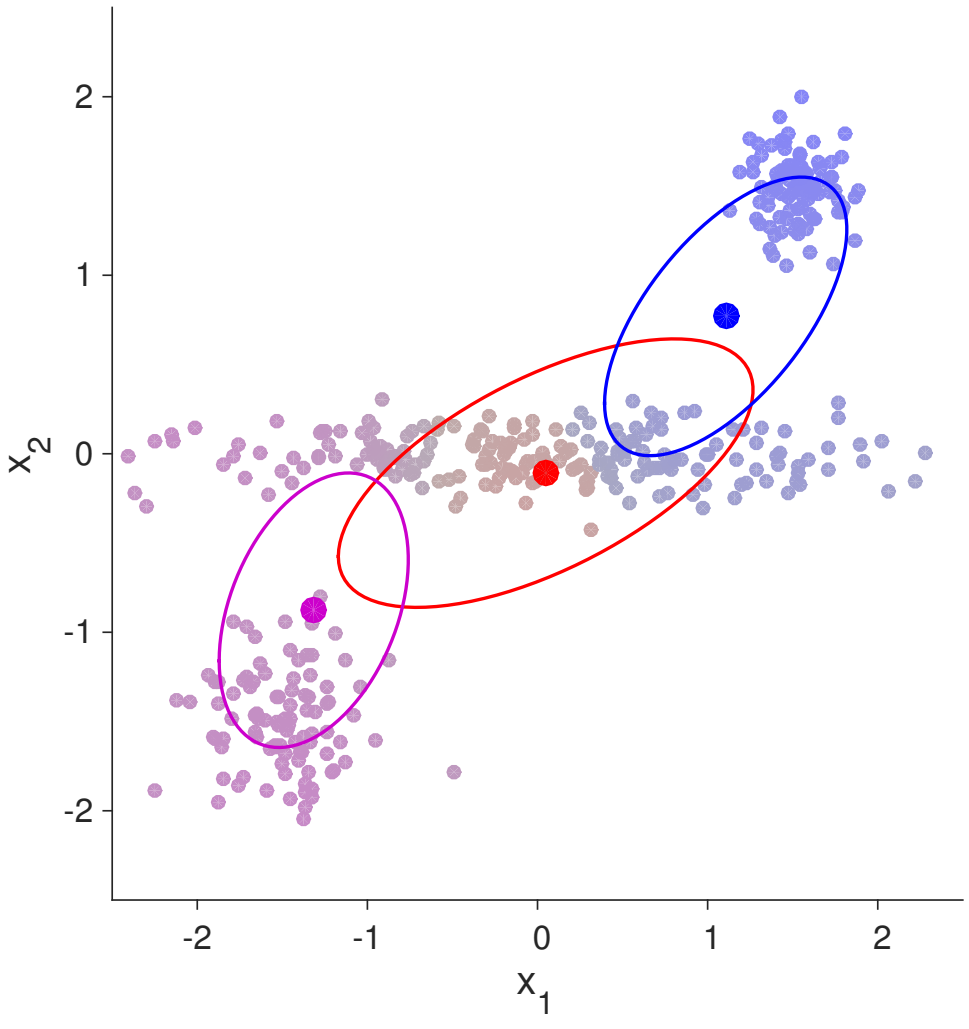
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

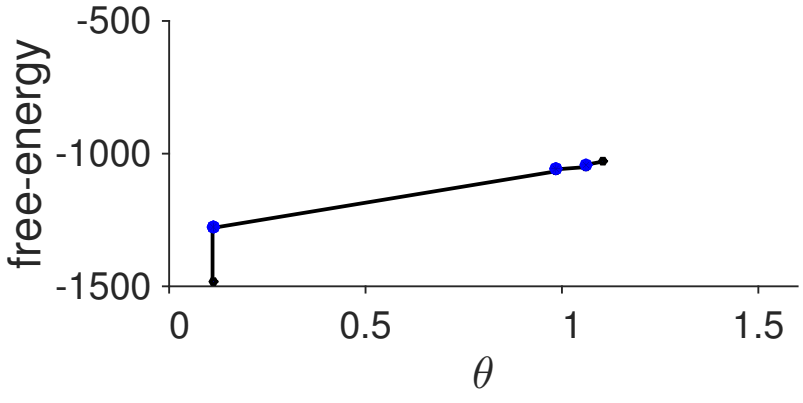
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

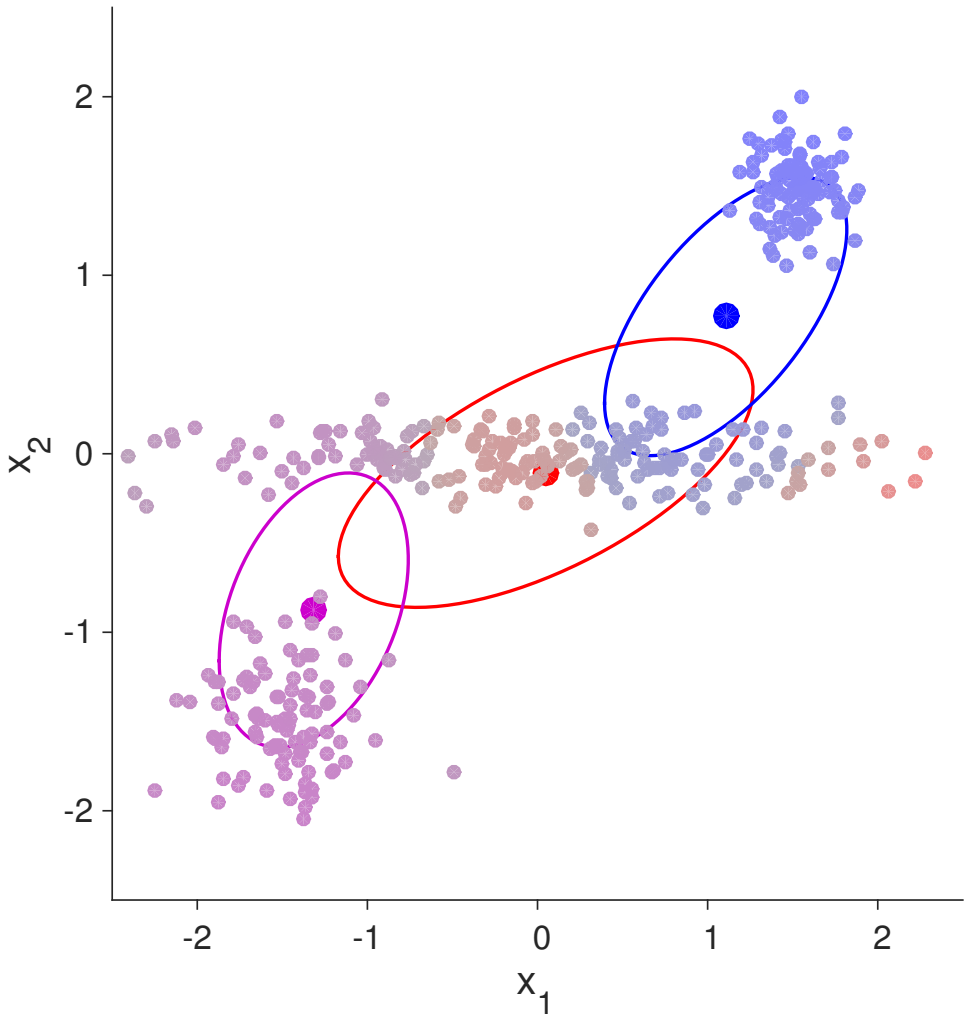
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

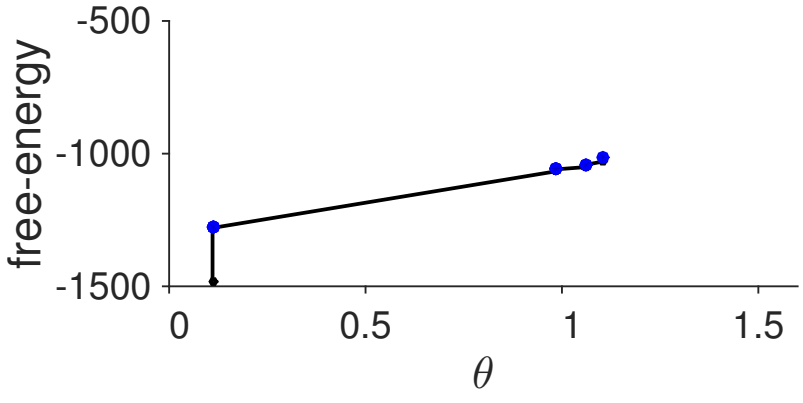
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

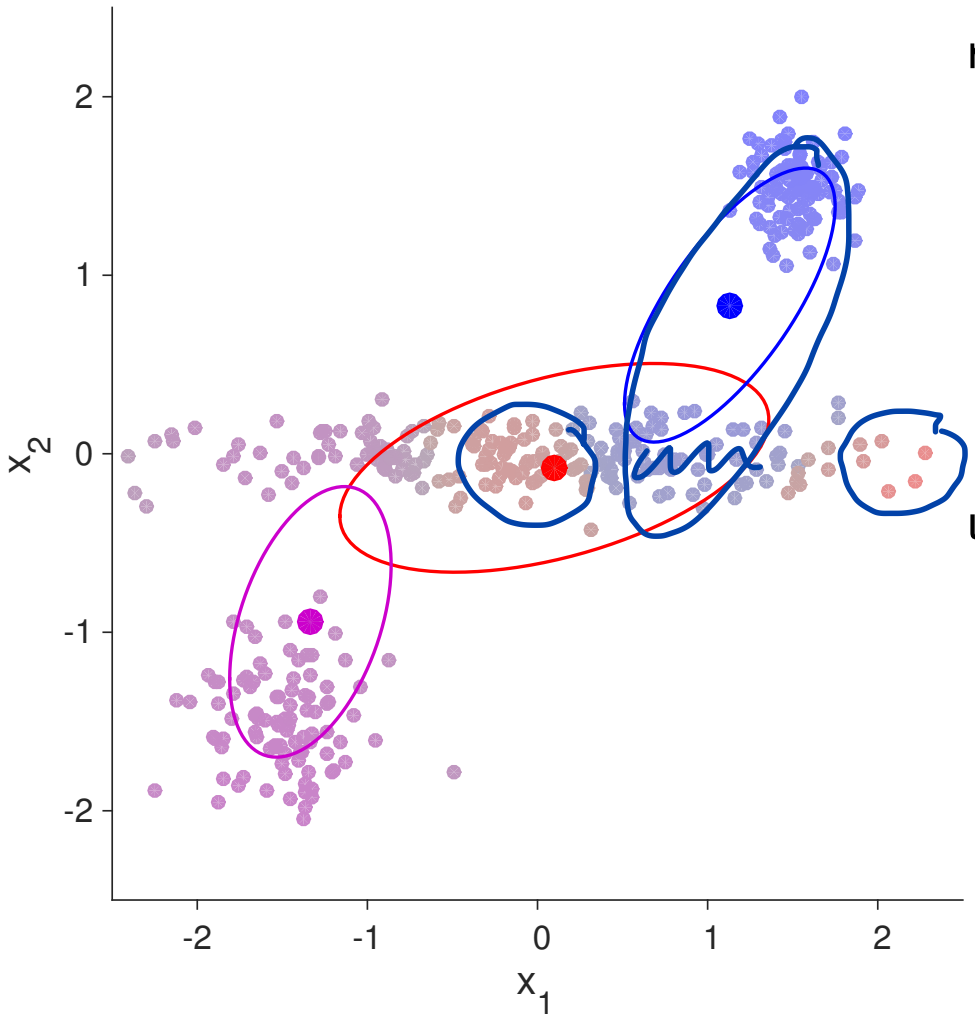
M-Step:

$$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

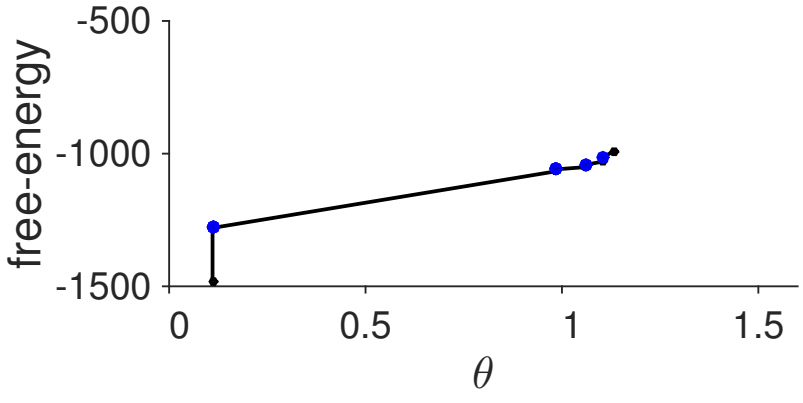
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

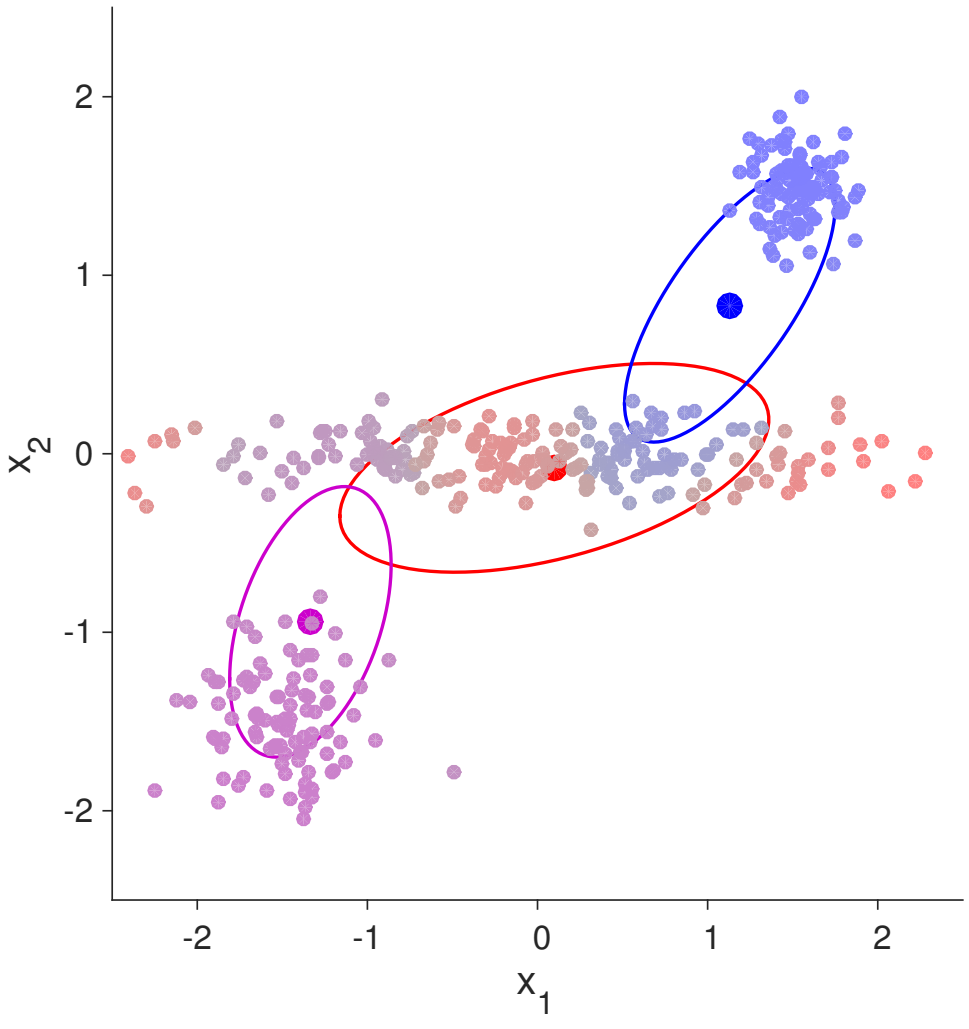
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

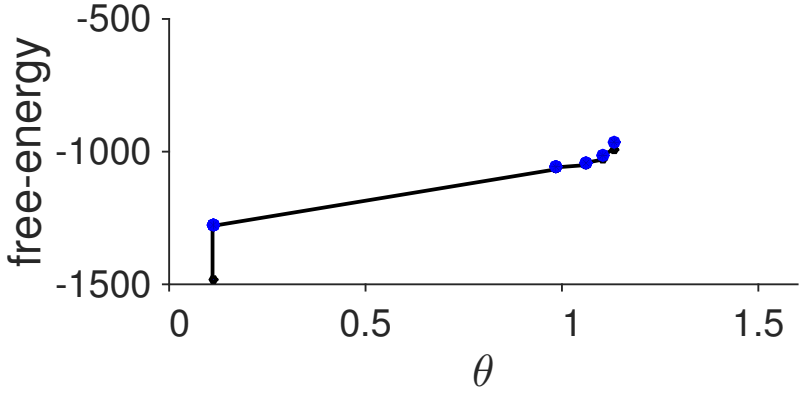
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

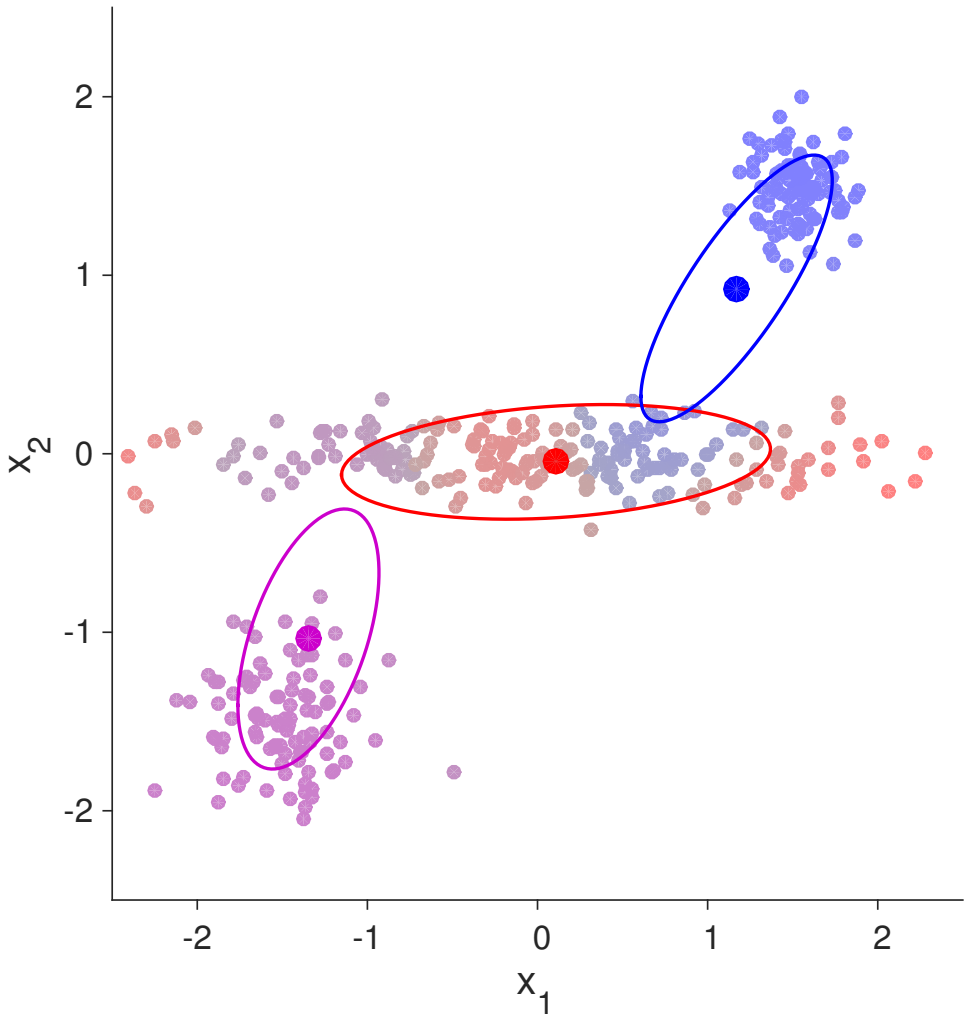
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

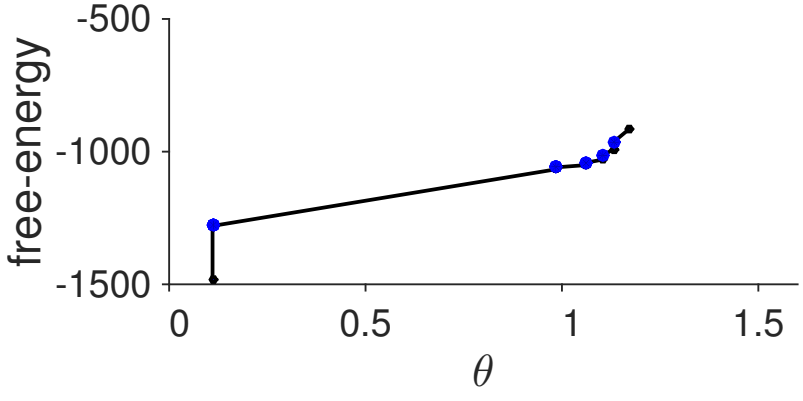
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

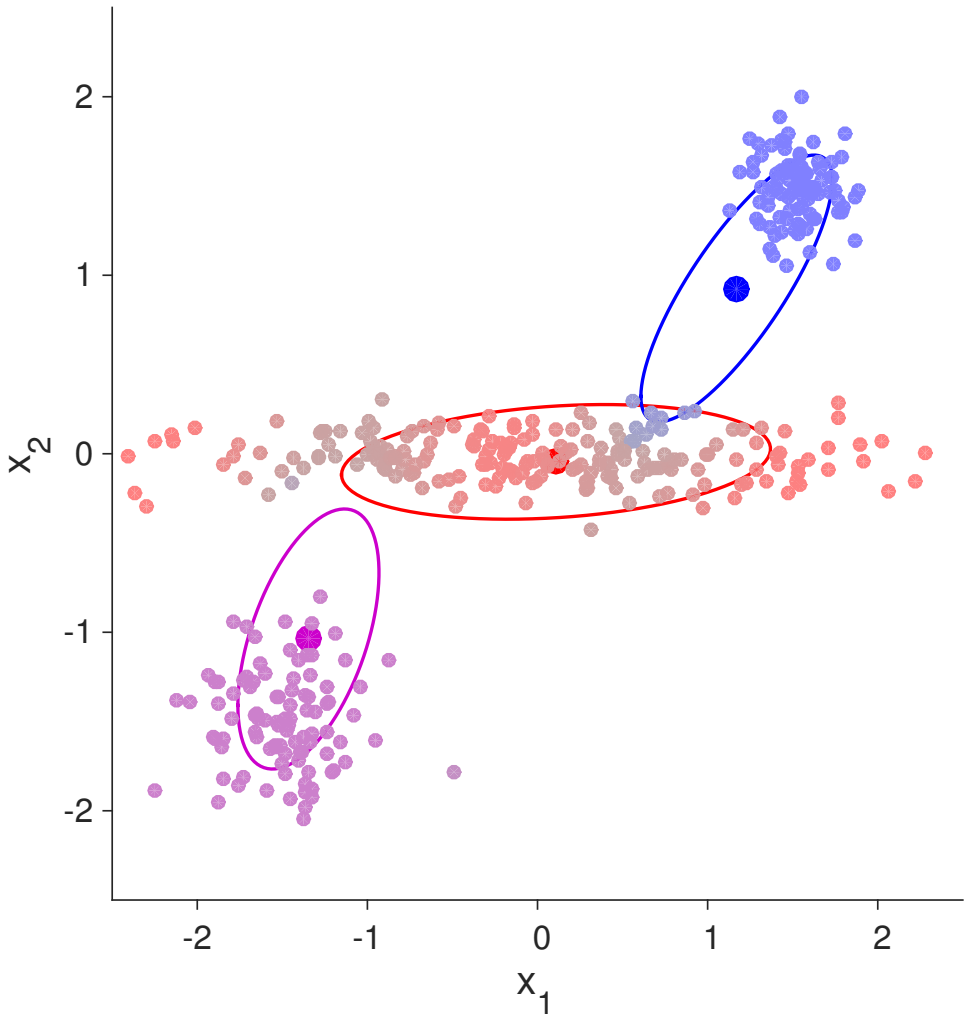
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

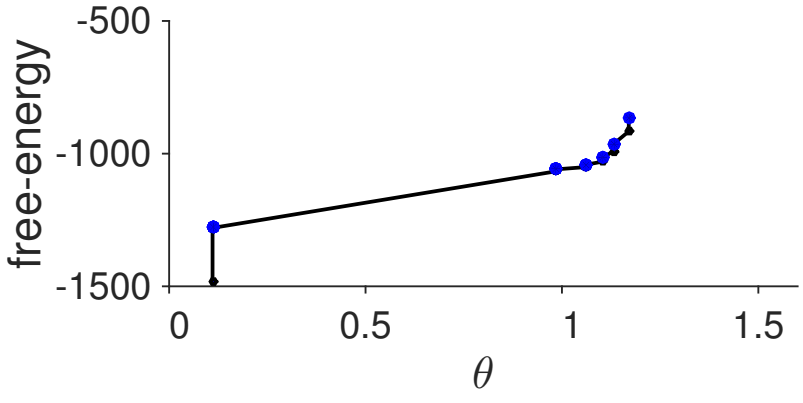
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

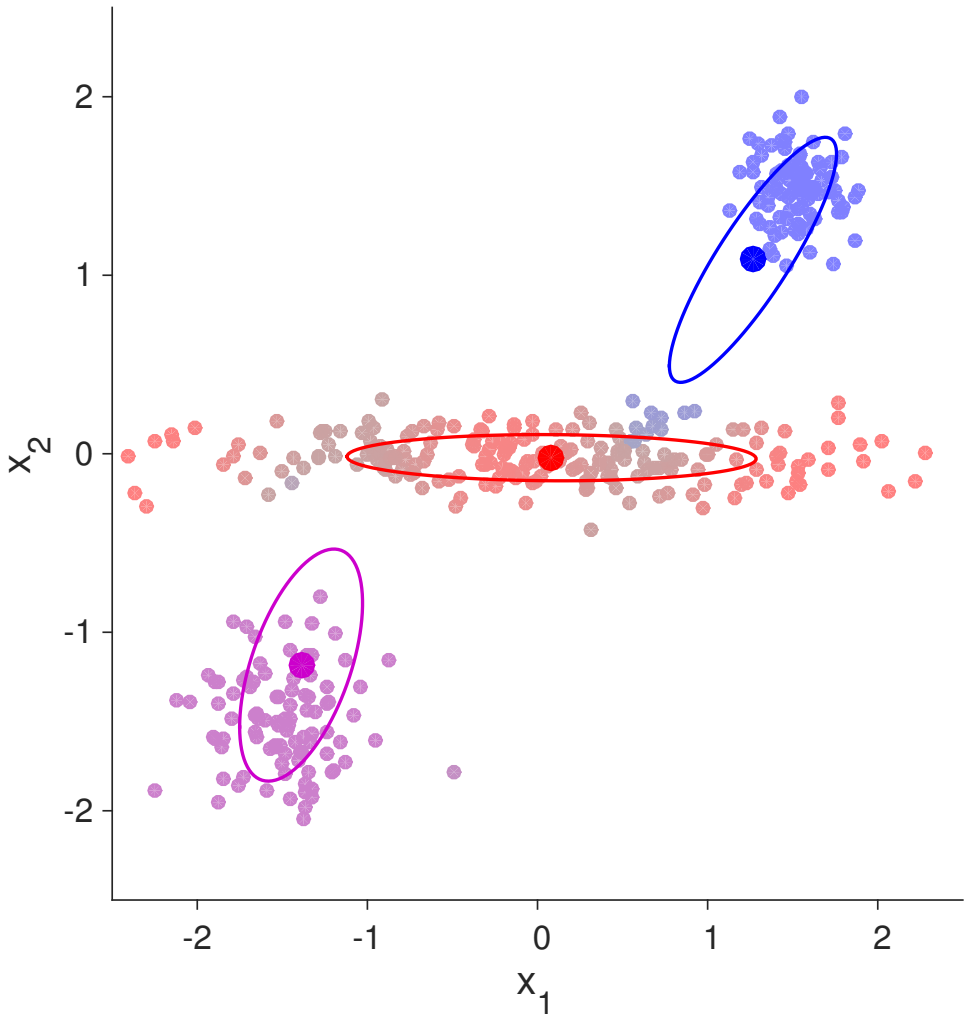
M-Step:

$$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

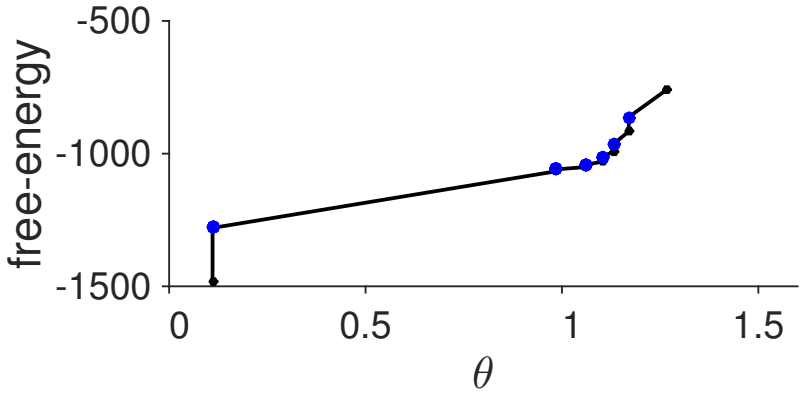
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

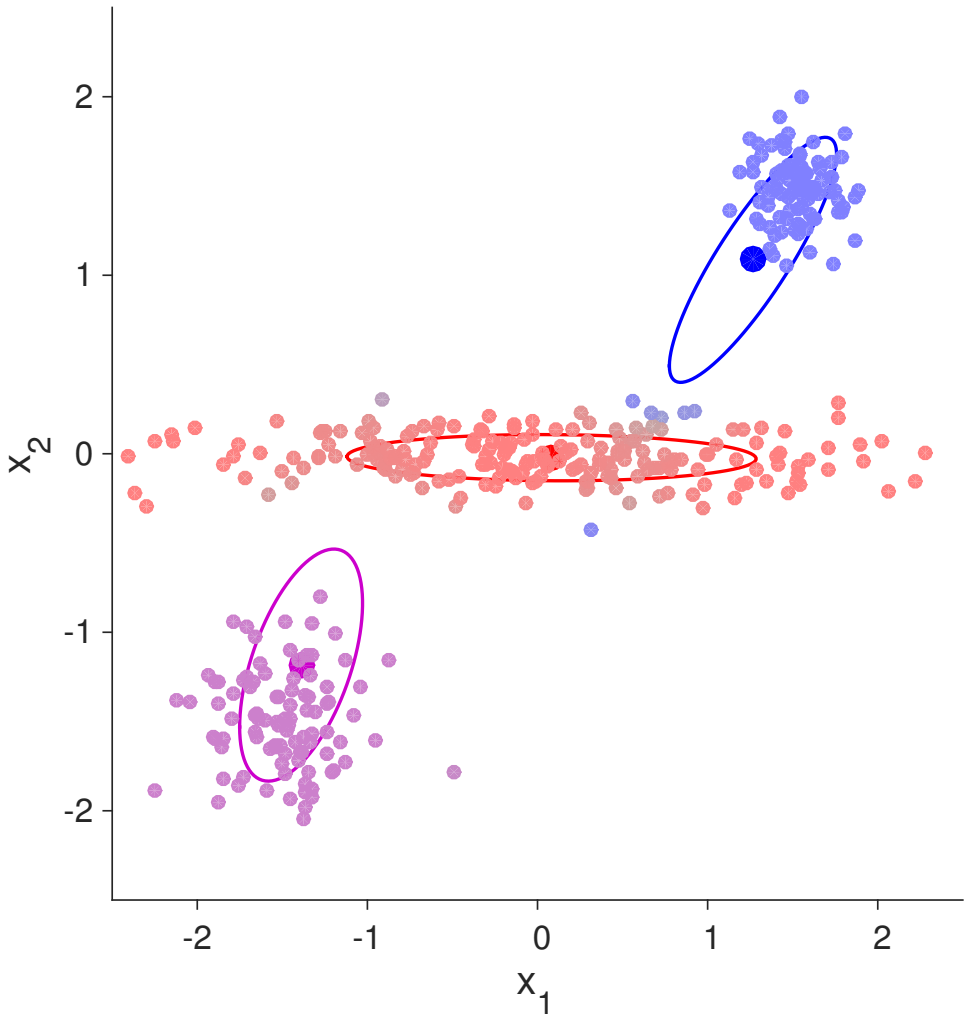
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

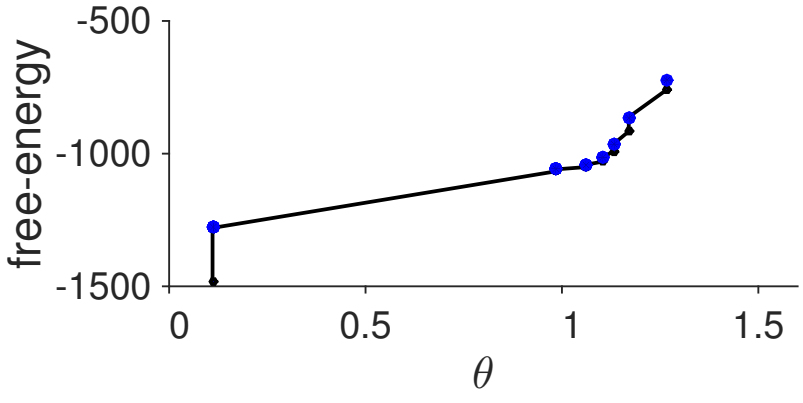
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

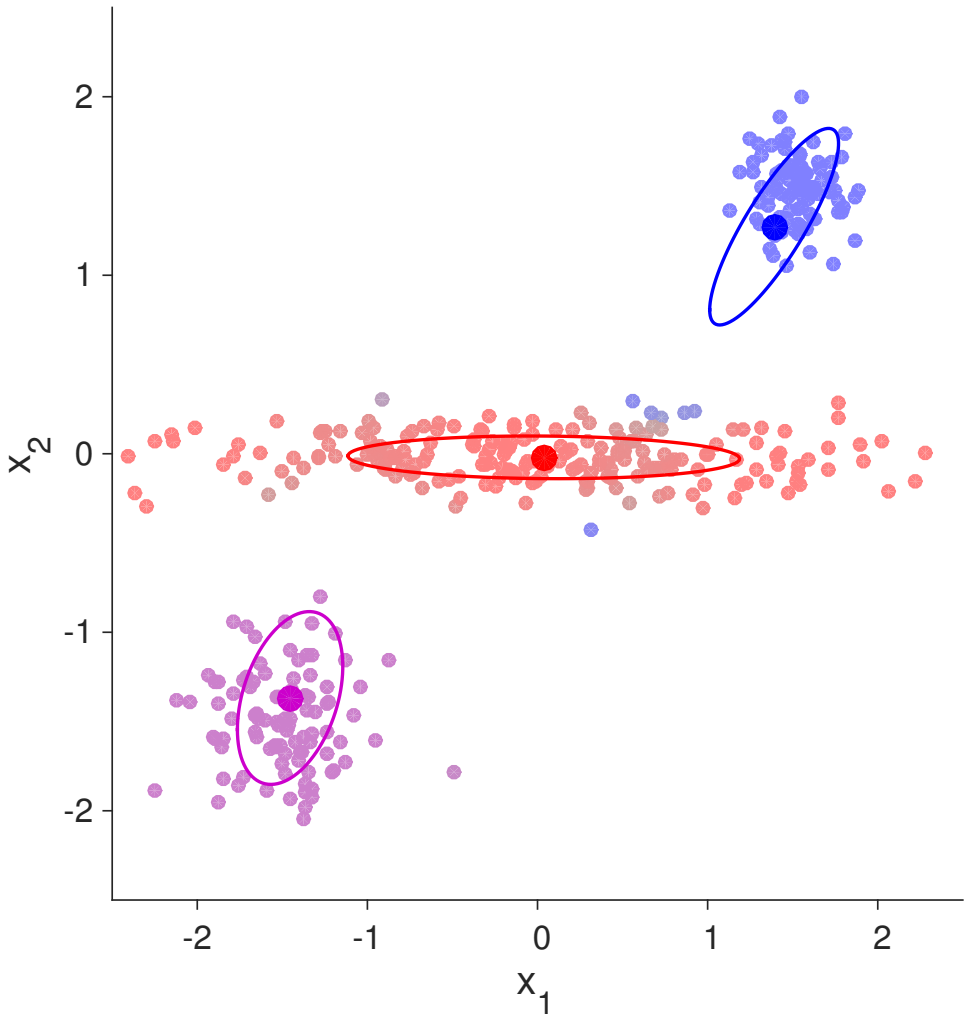
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

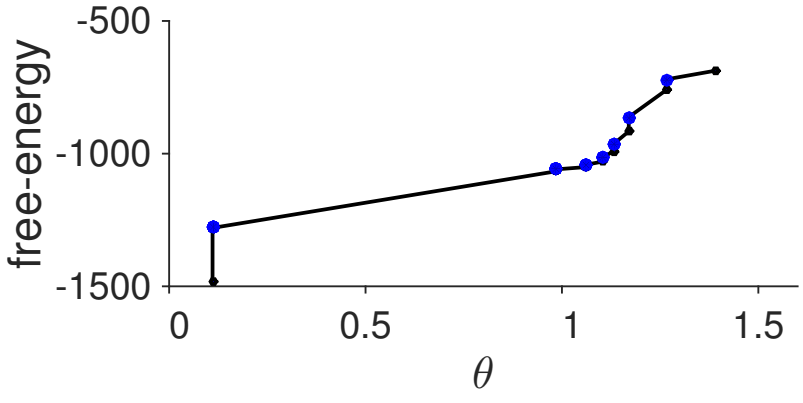
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

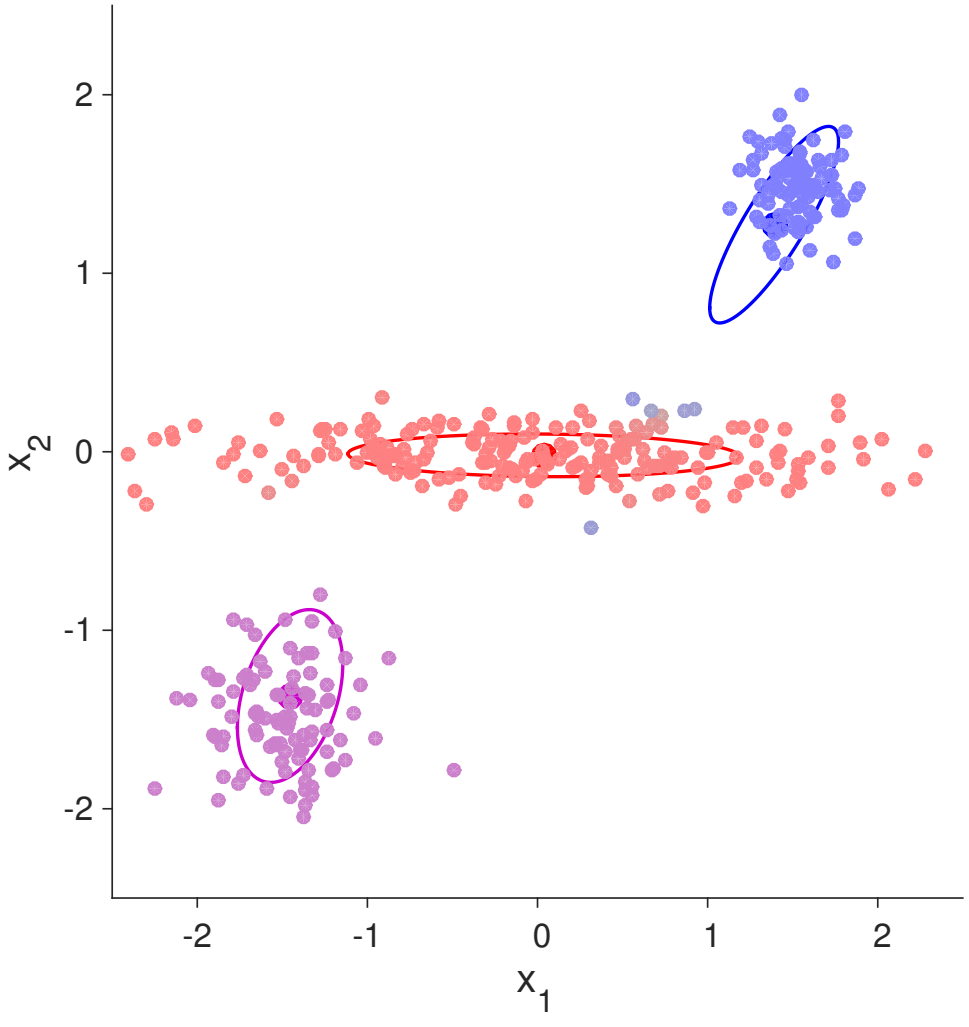
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

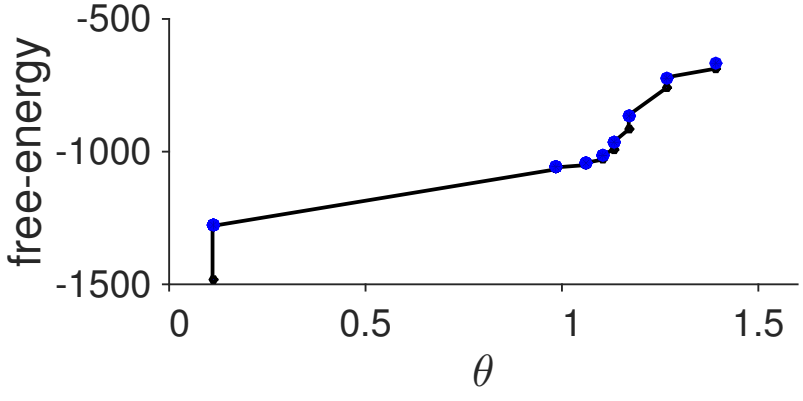
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

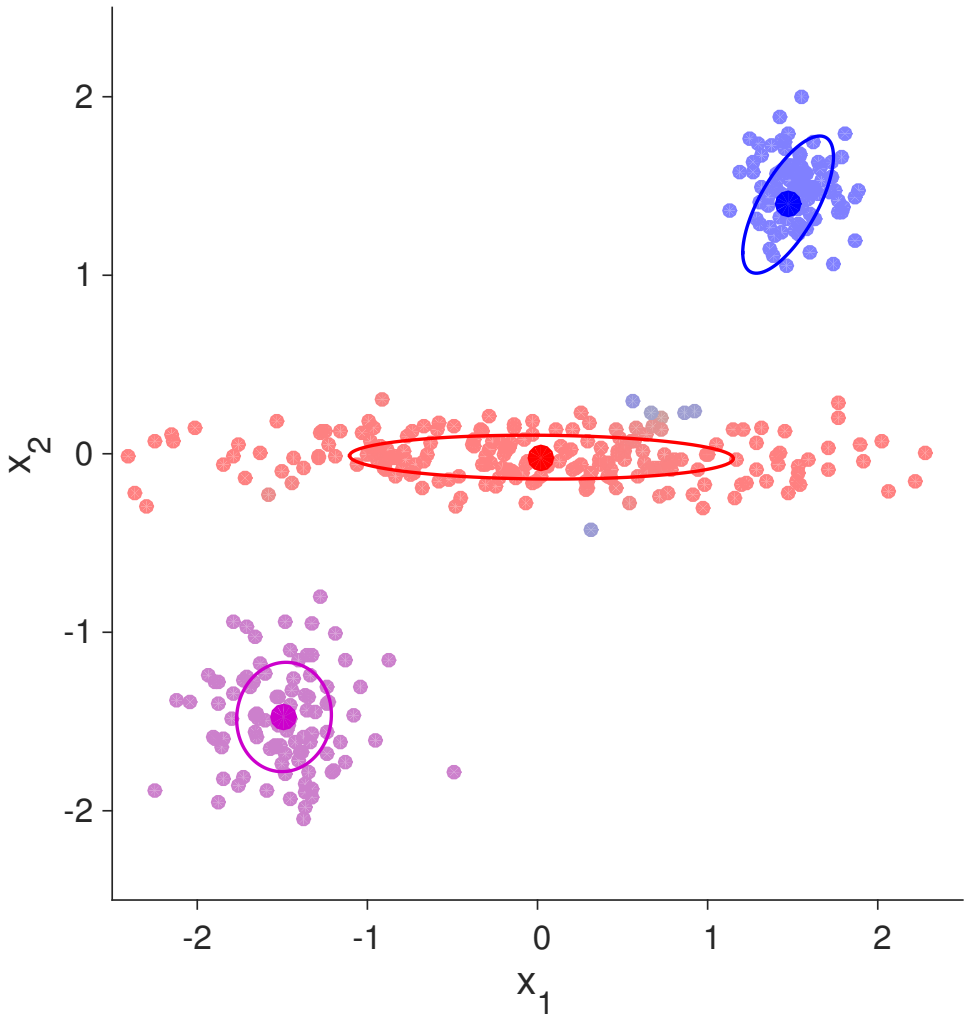
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

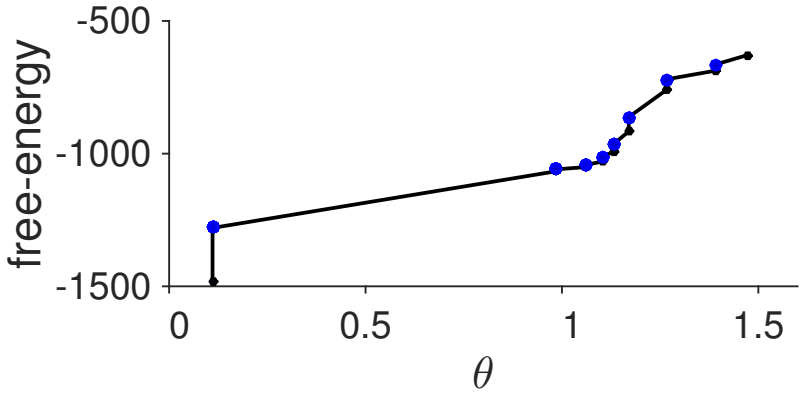
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

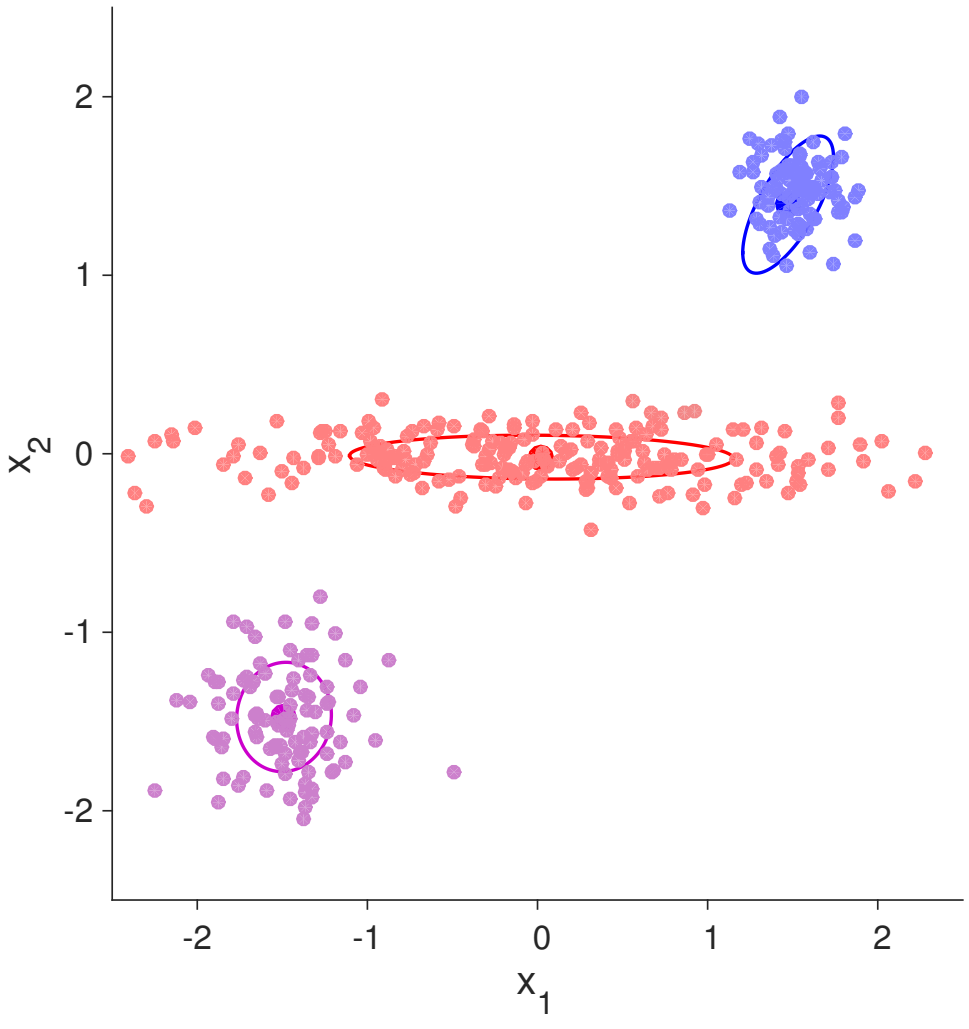
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

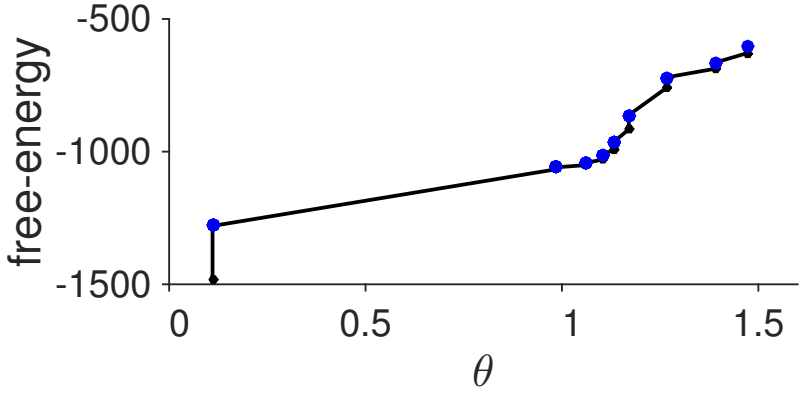
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

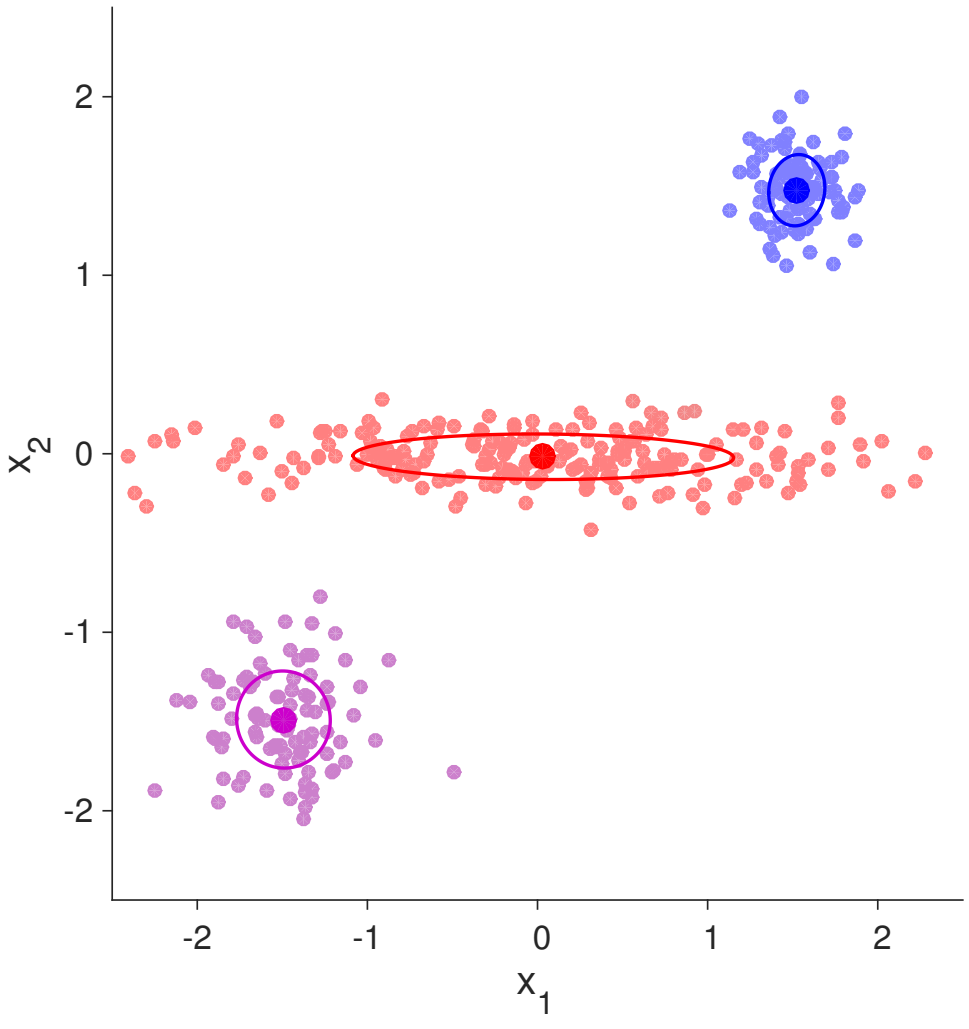
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

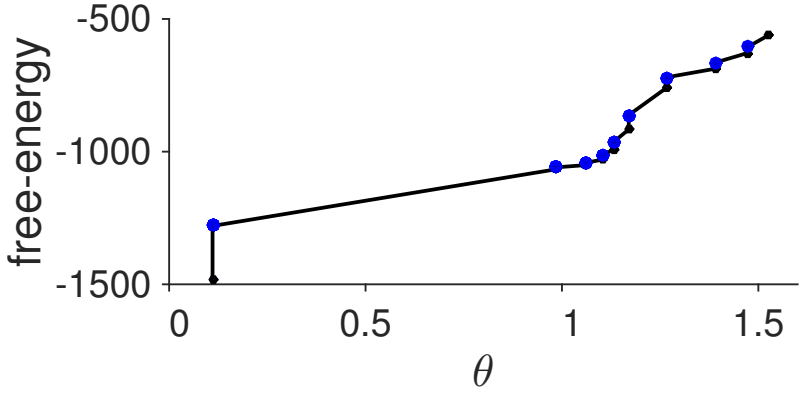
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

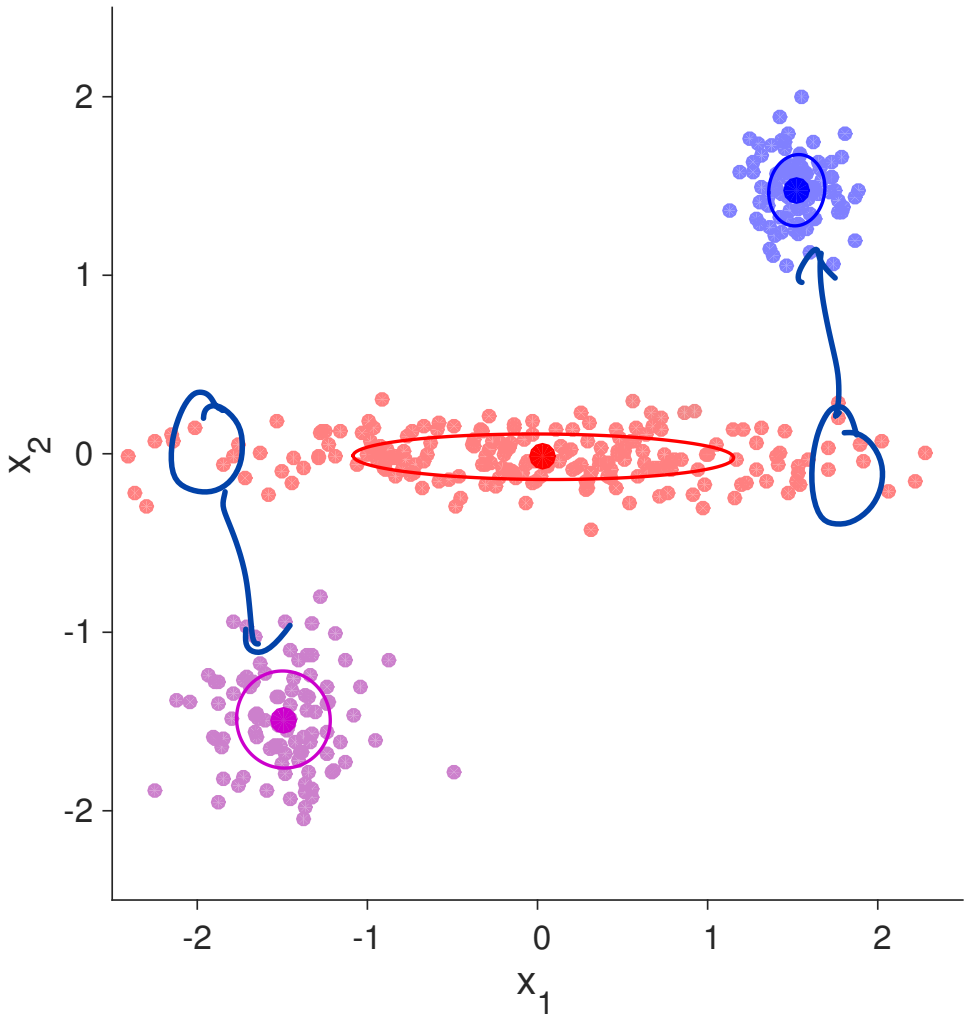
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



EM for MoGs: soft, non-axis aligned K-means



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

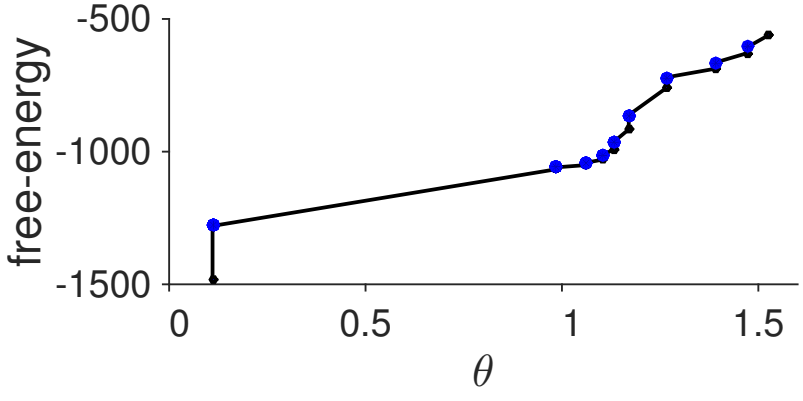
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

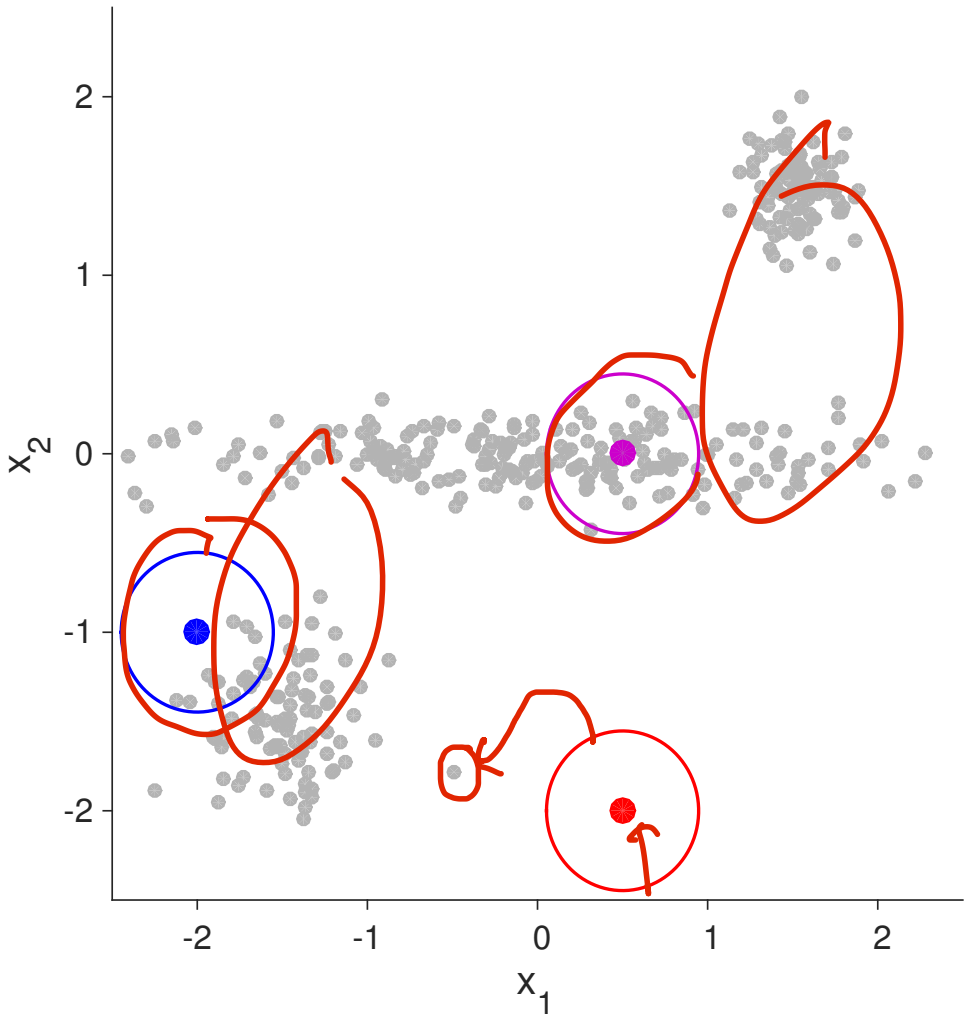
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

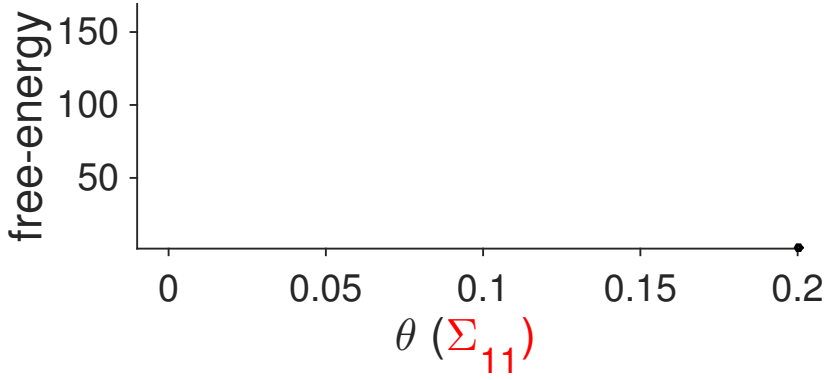
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

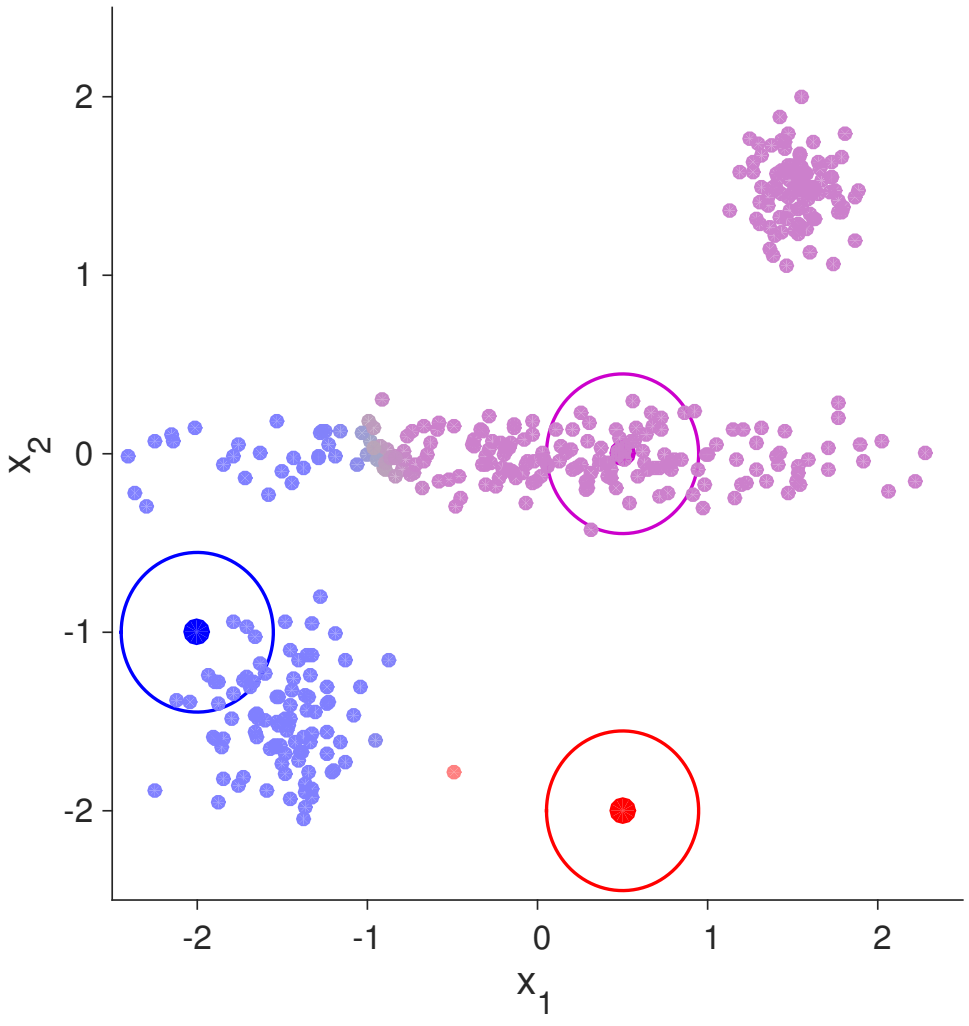
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

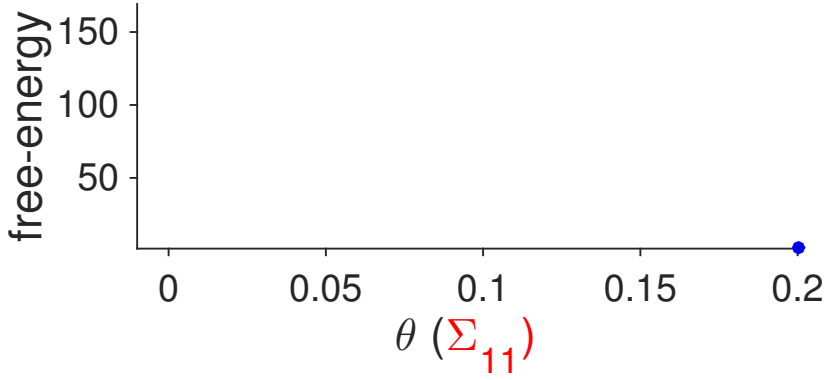
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

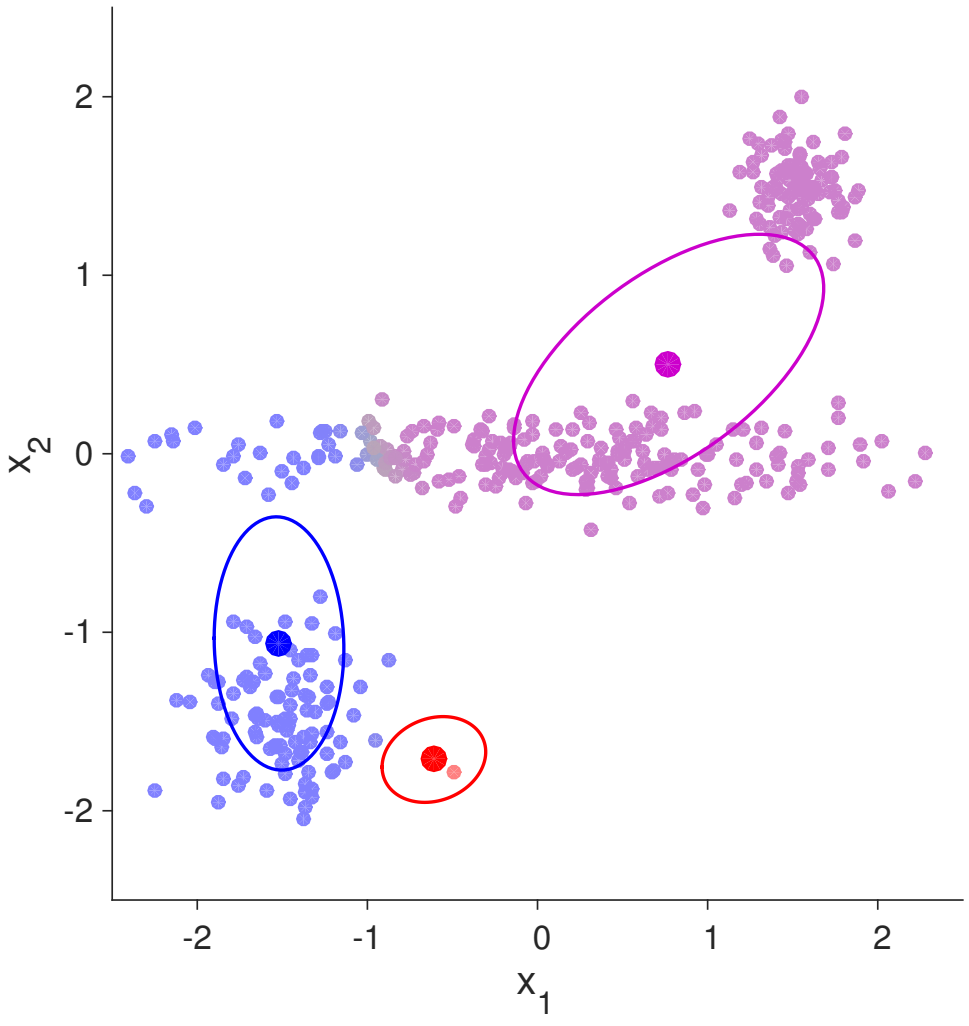
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

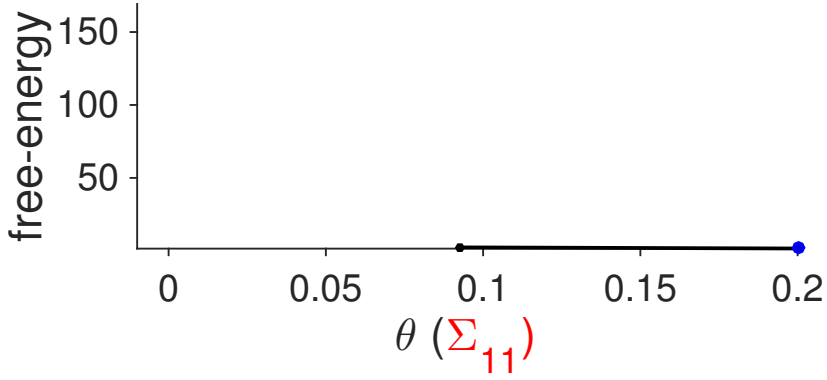
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

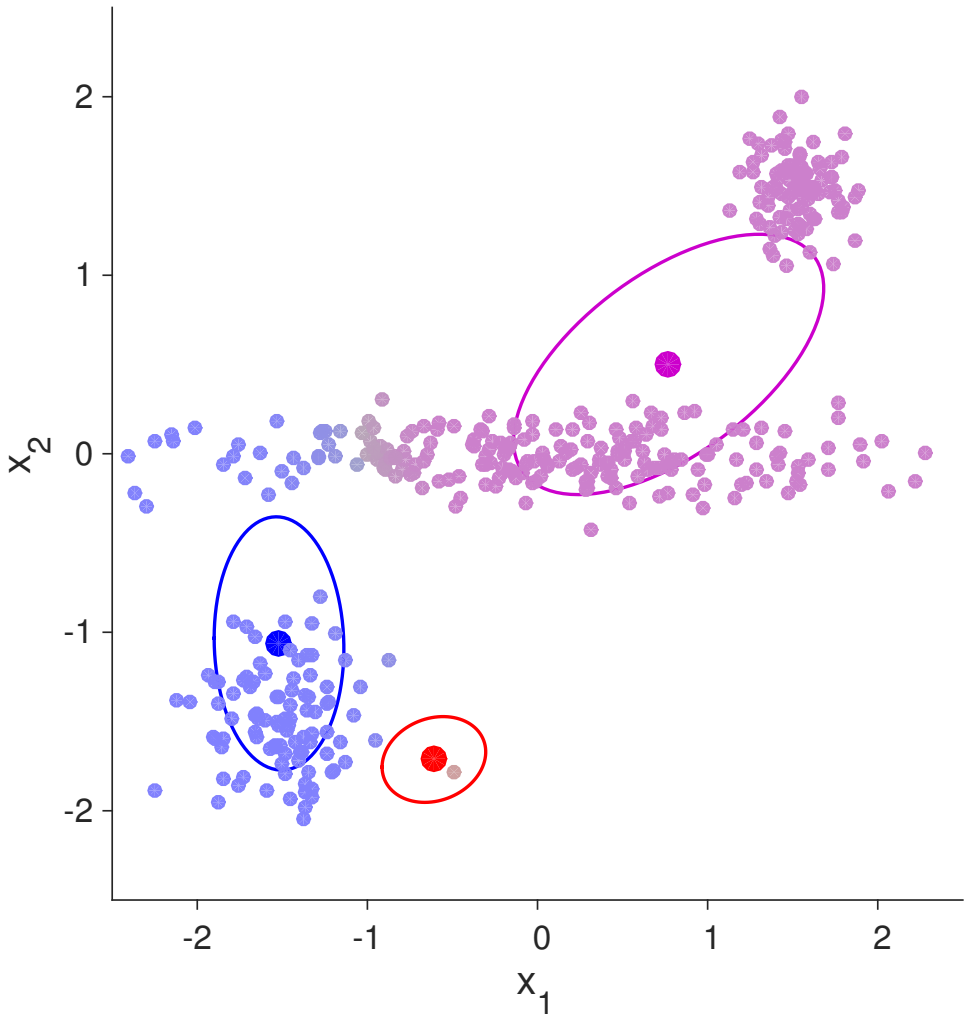
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

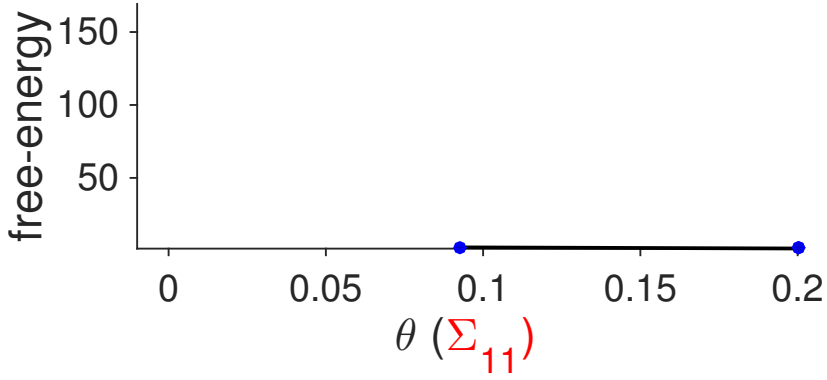
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

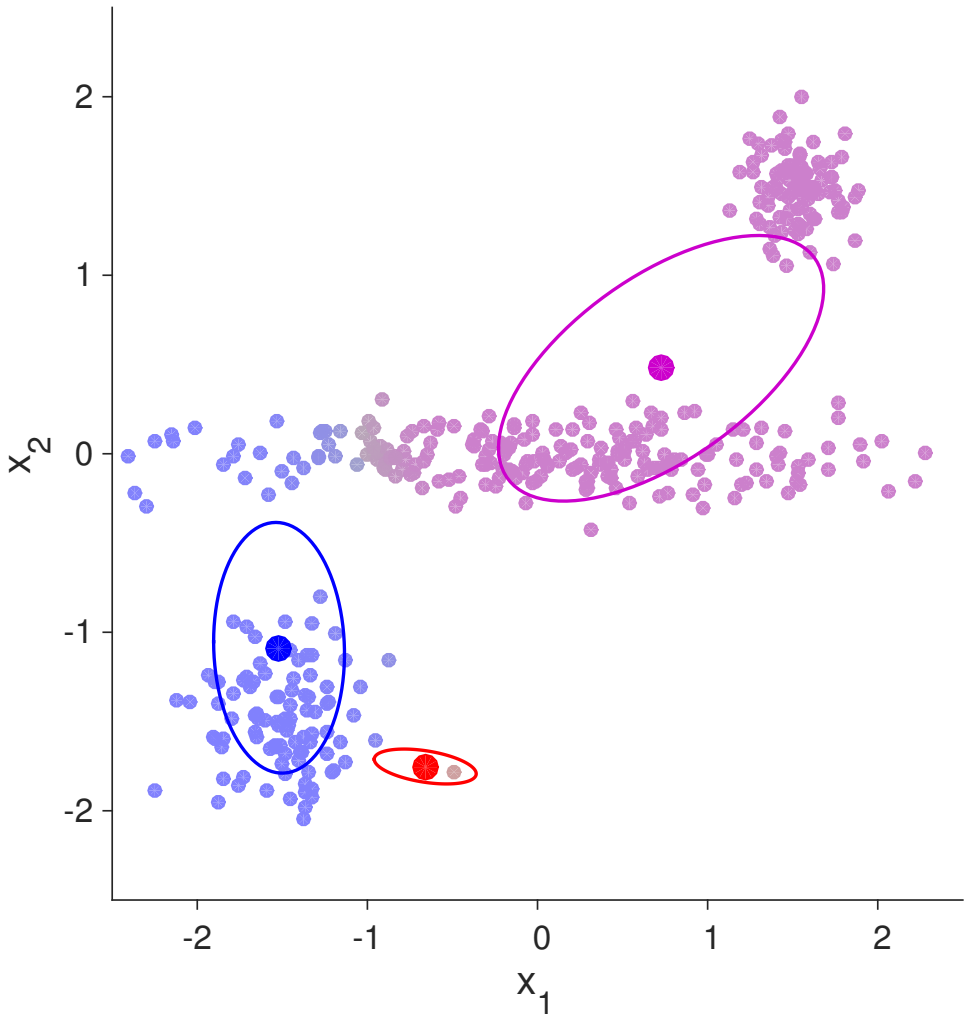
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

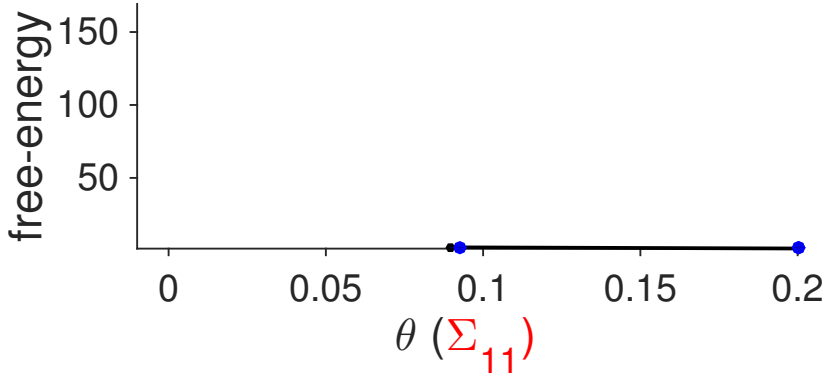
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

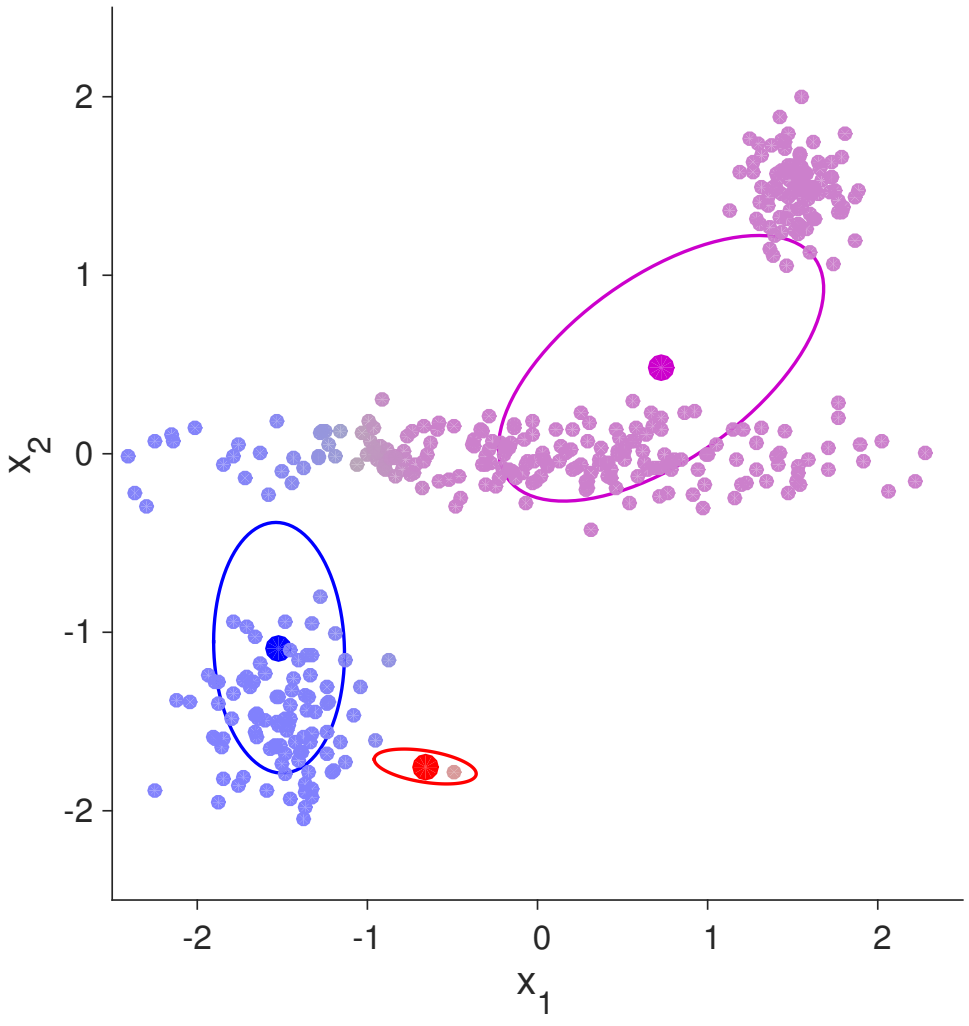
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

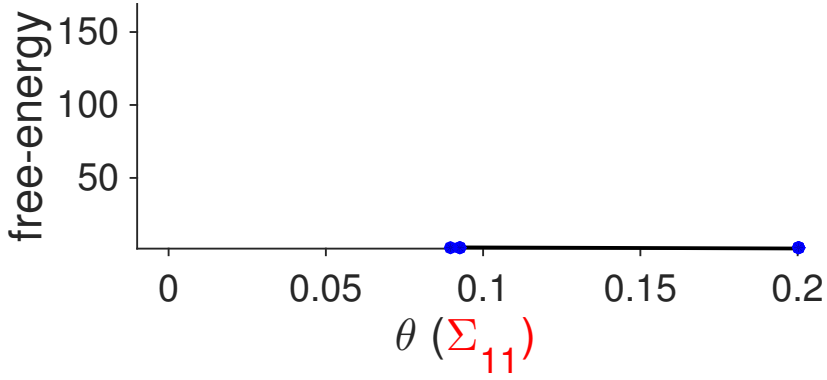
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

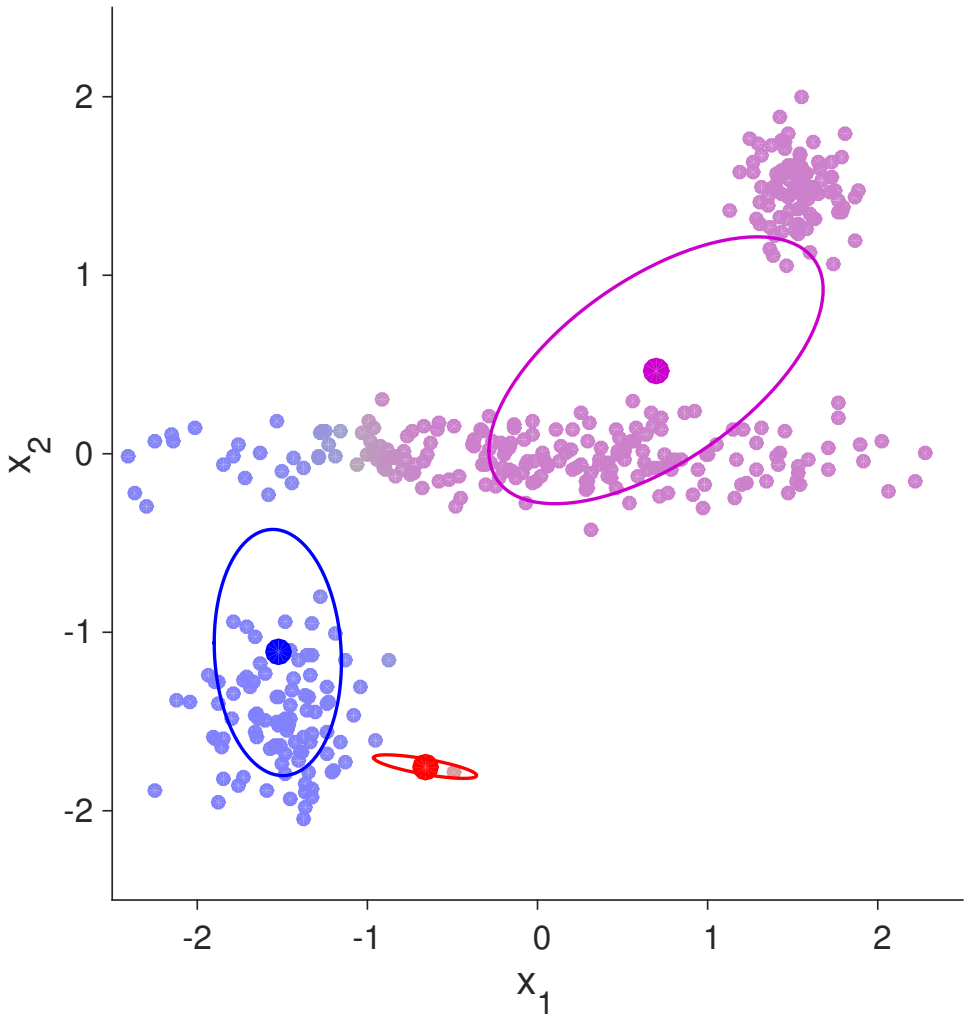
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

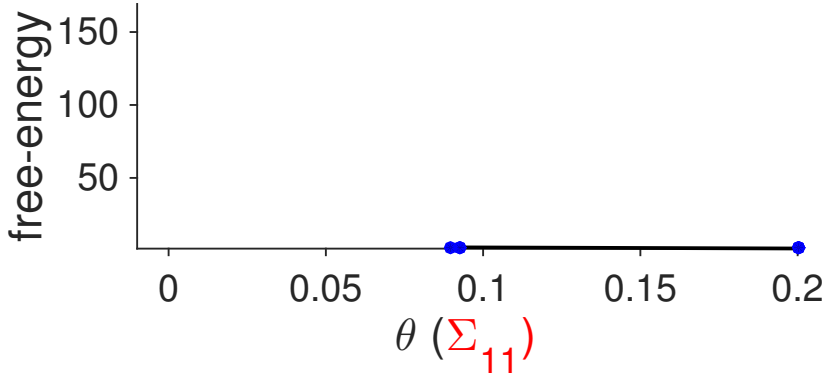
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

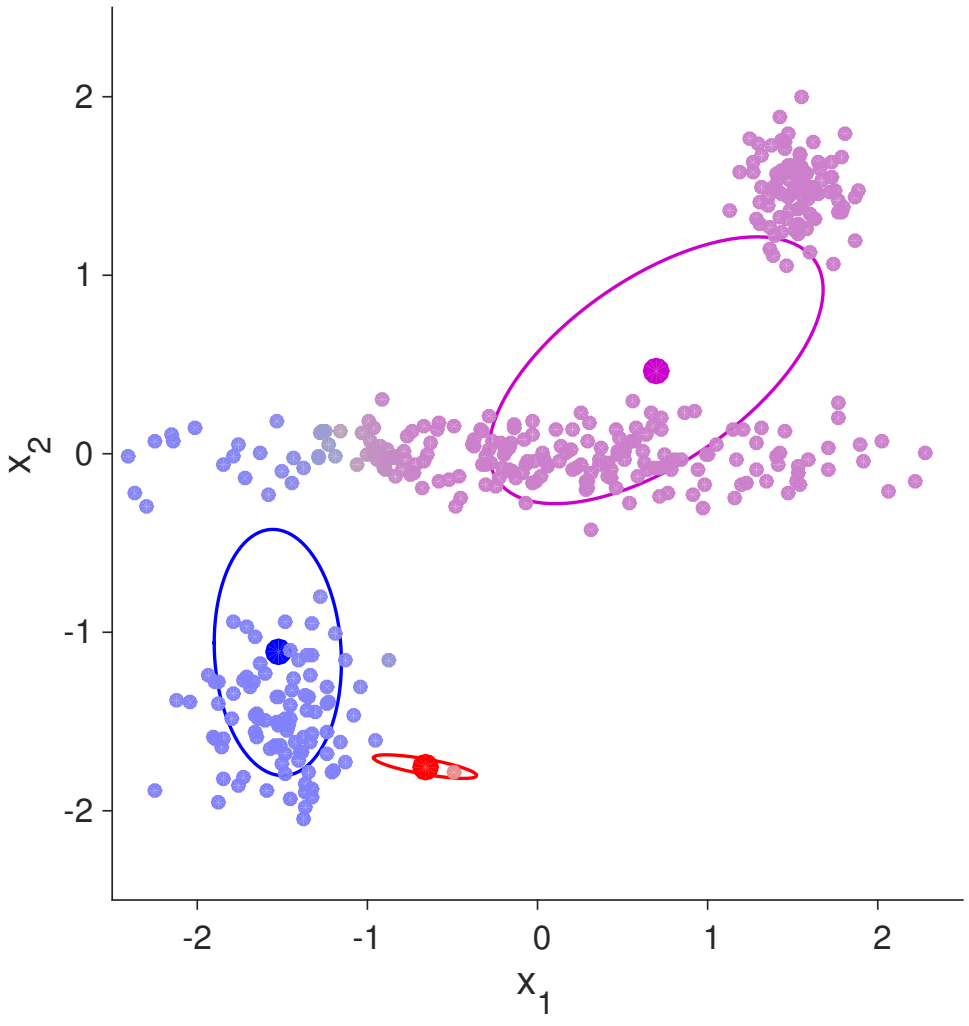
M-Step:

$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

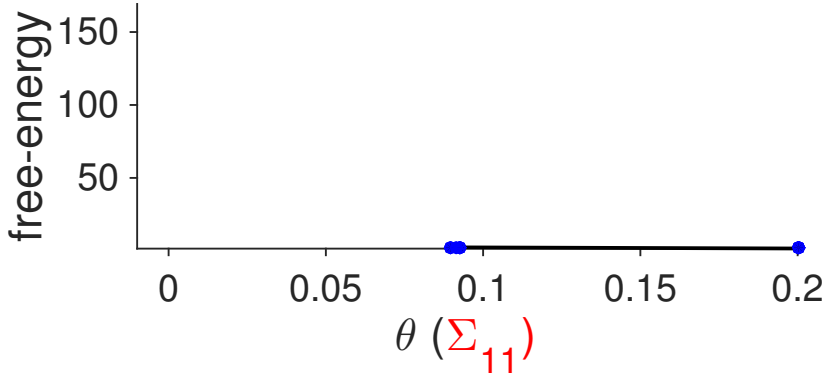
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

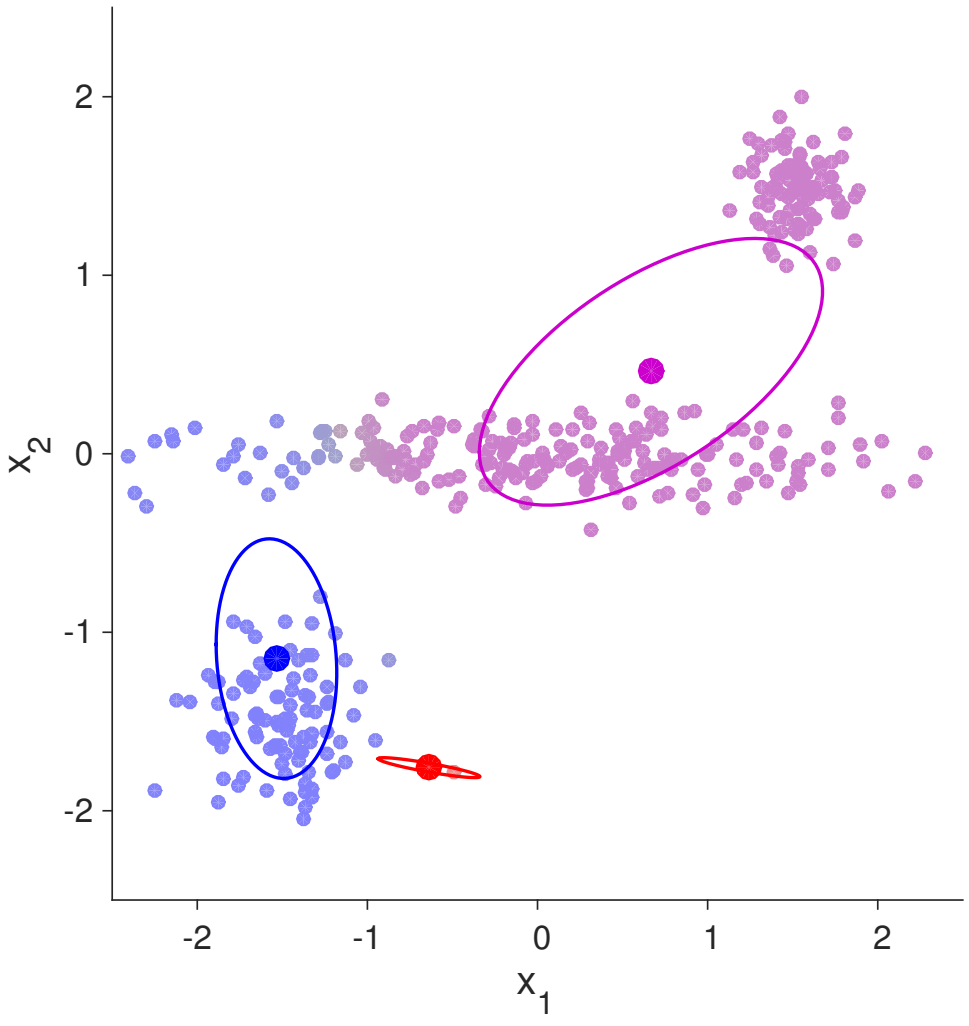
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

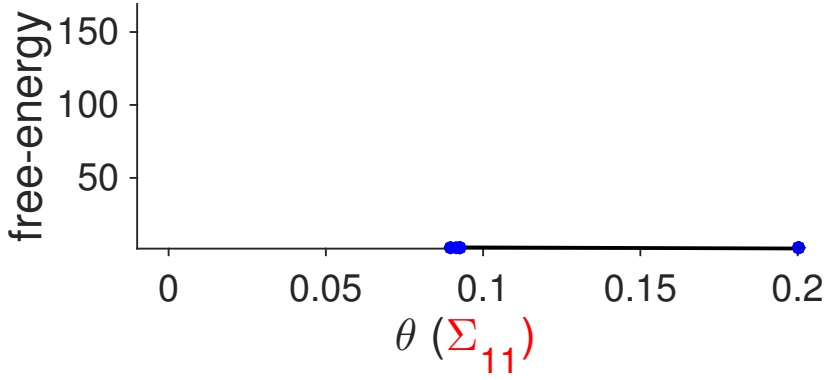
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

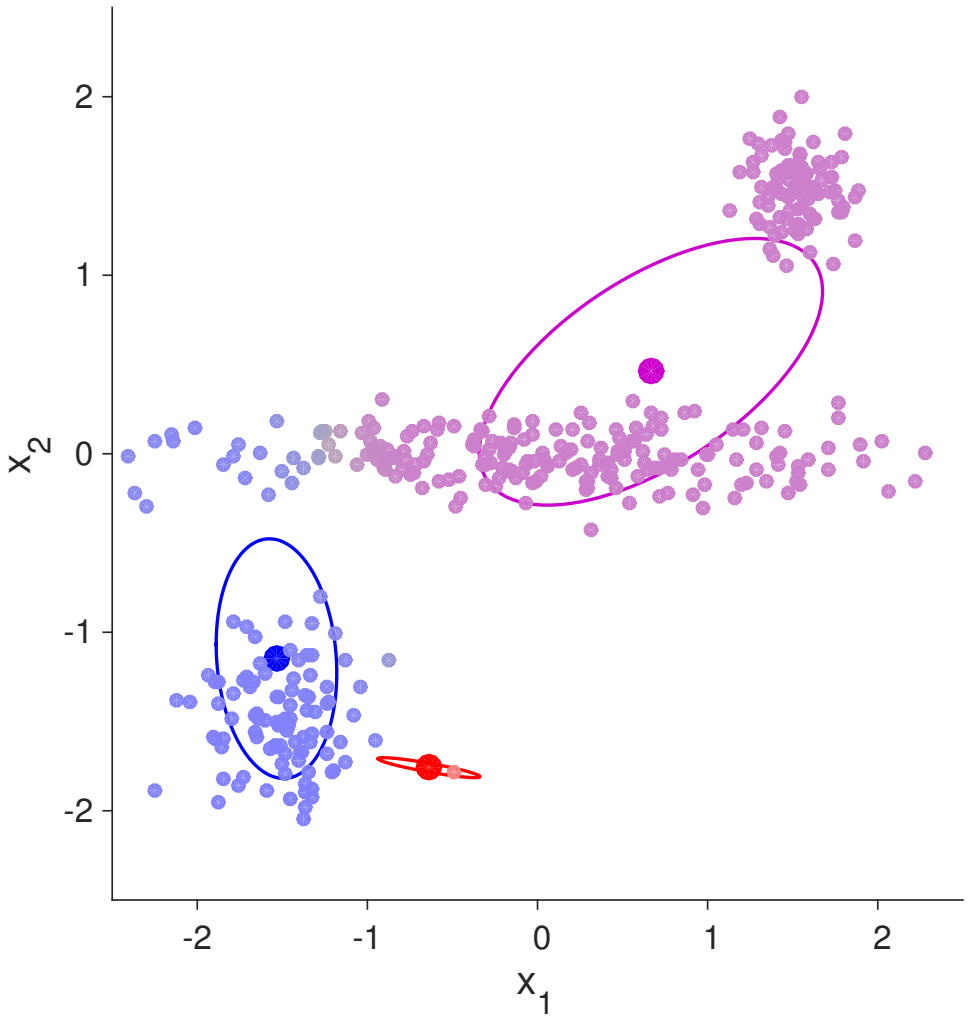
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

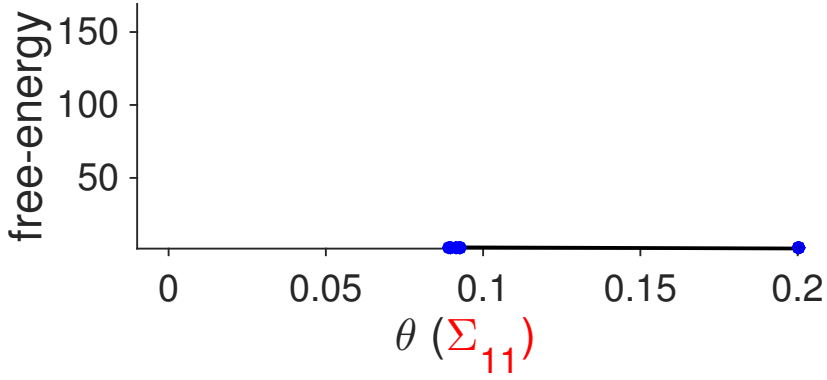
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

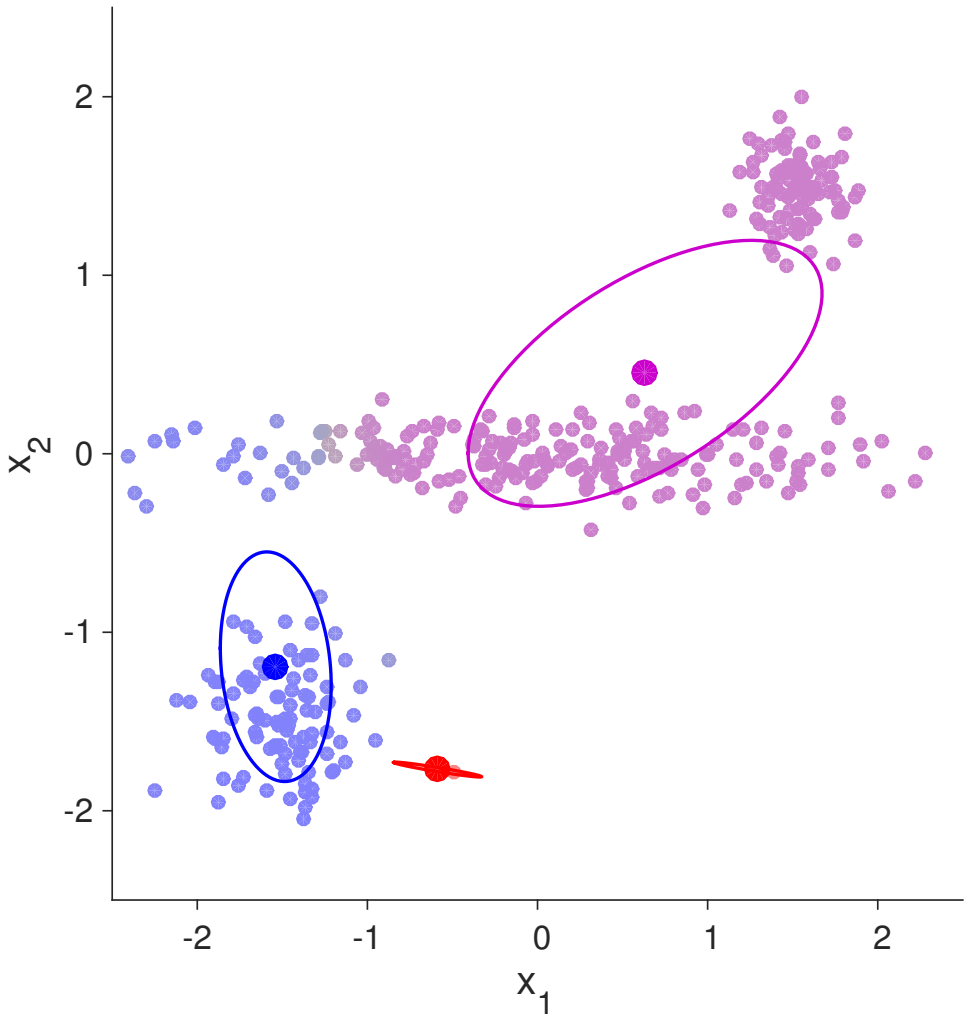
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

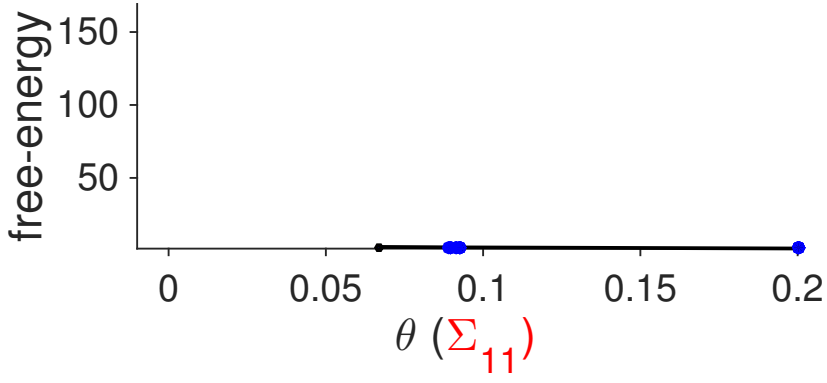
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

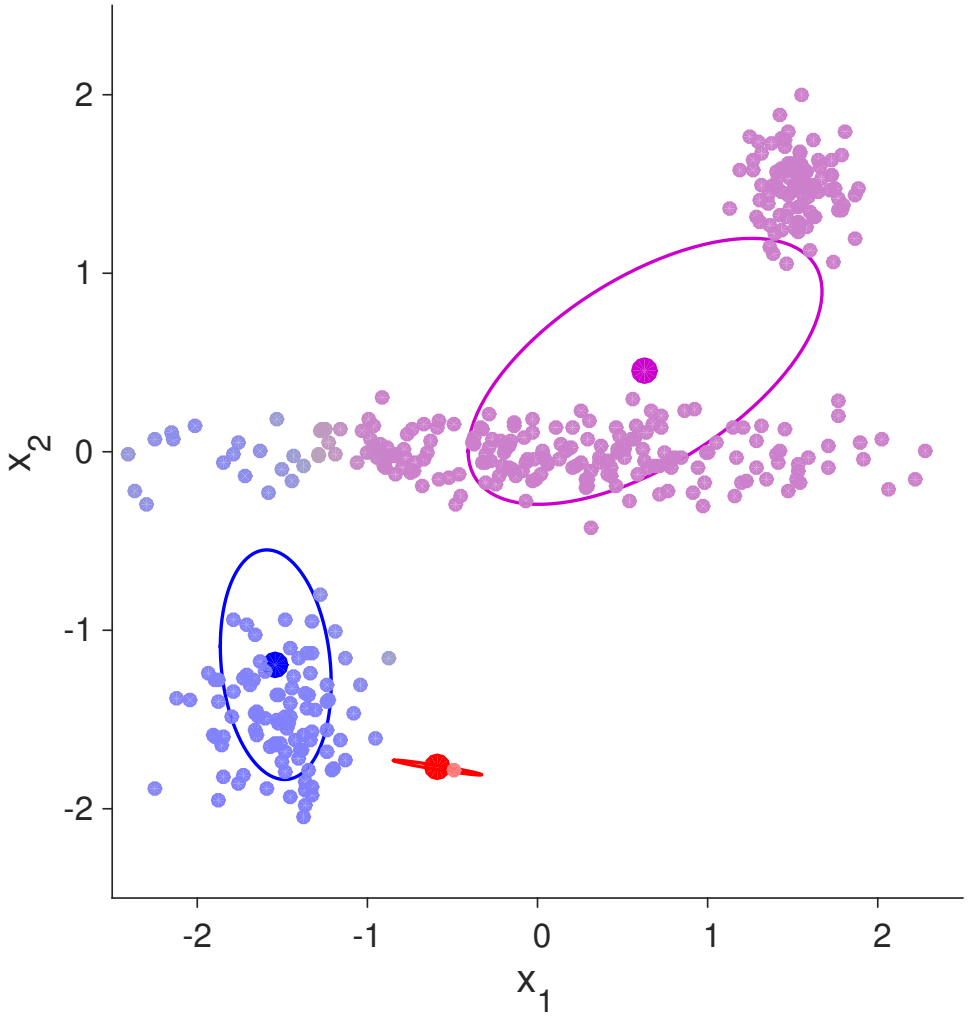
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

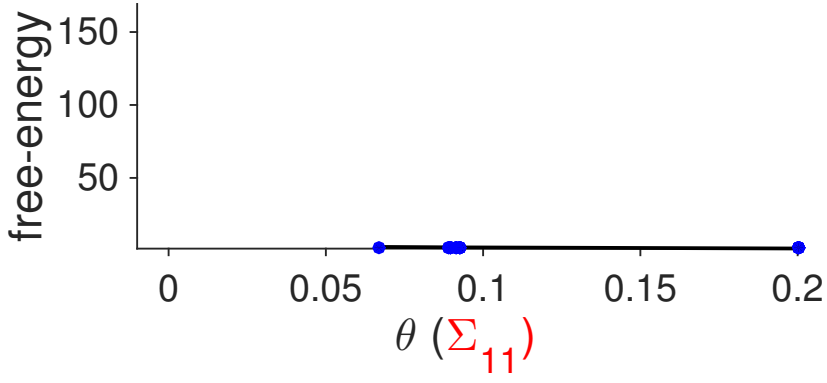
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

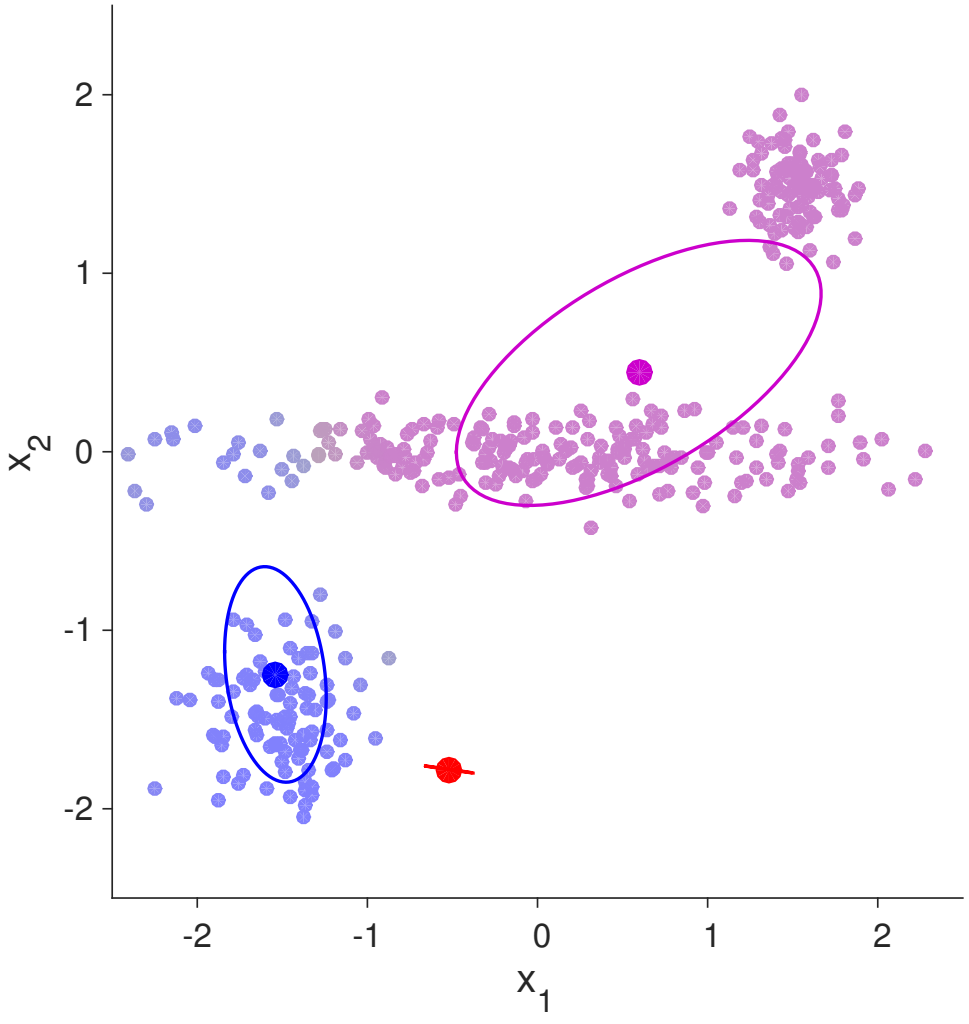
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

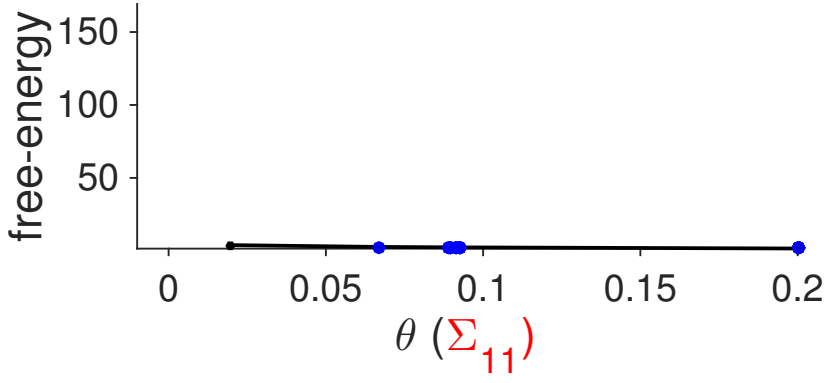
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

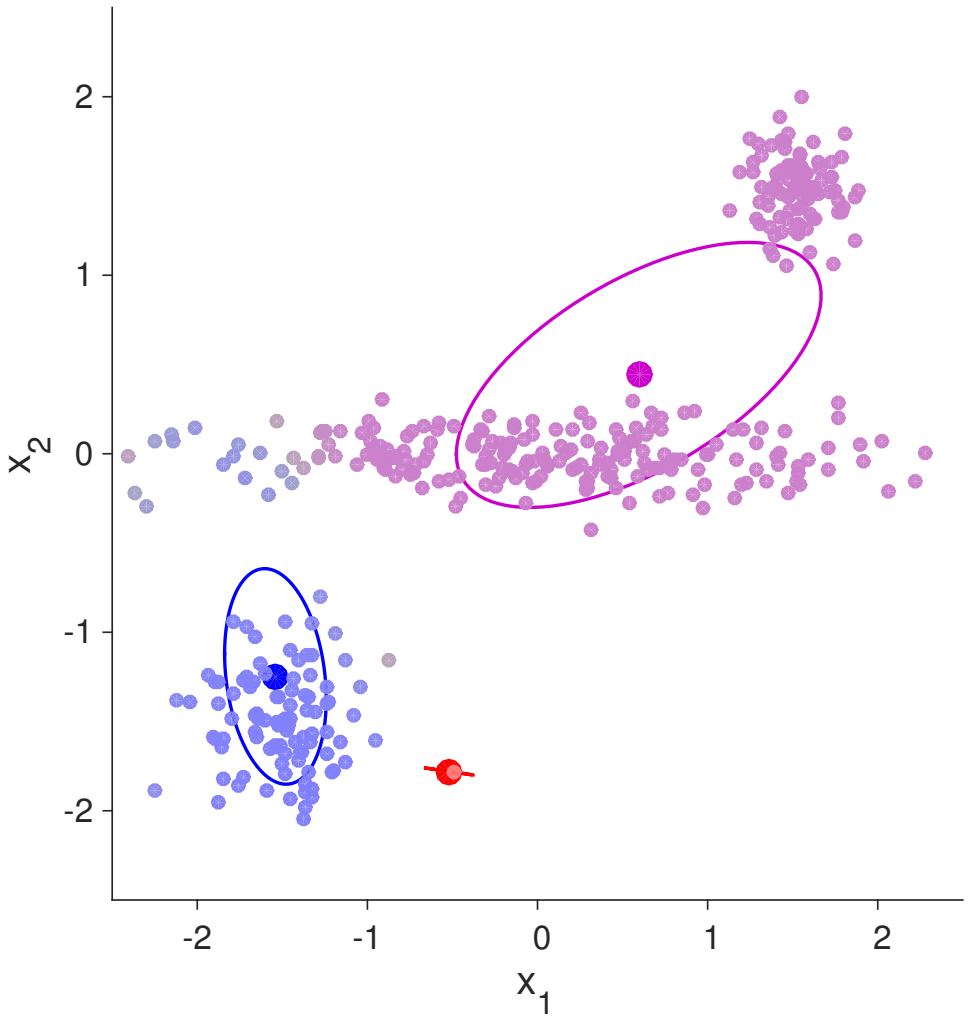
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

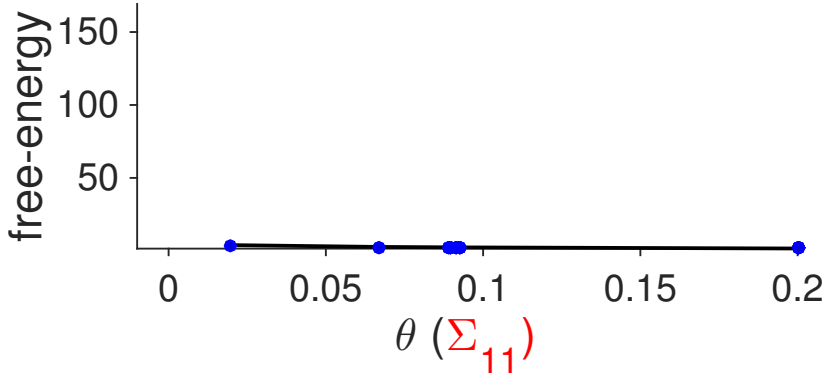
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

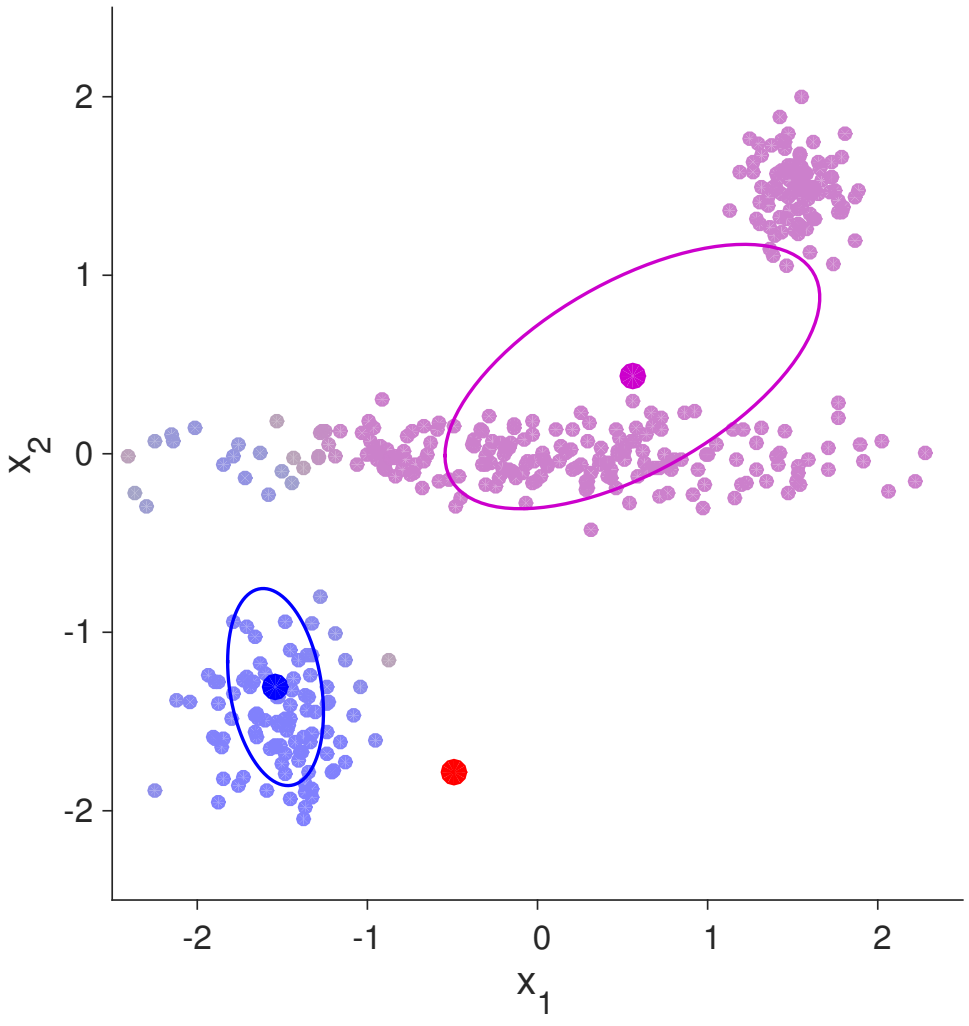
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

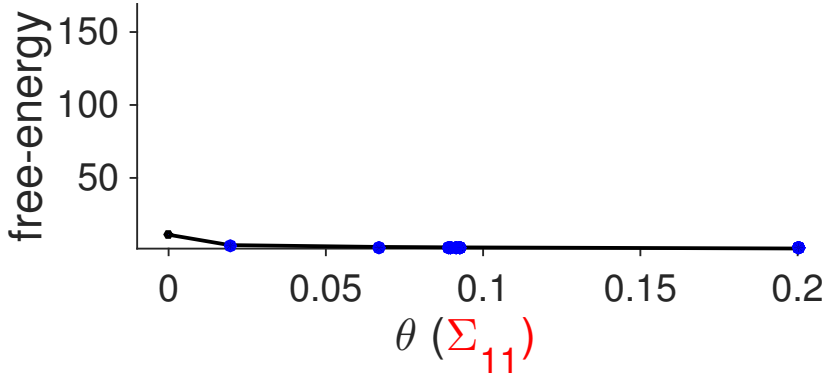
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

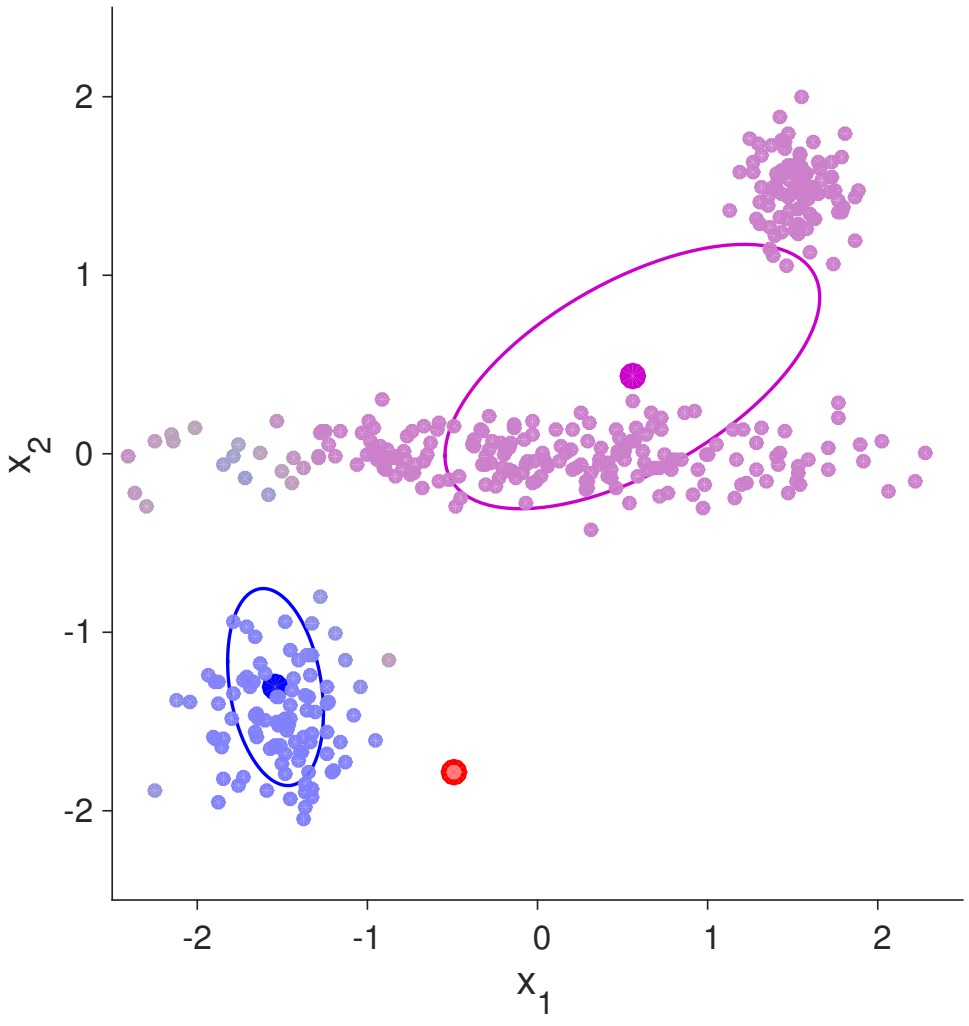
M-Step:

$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

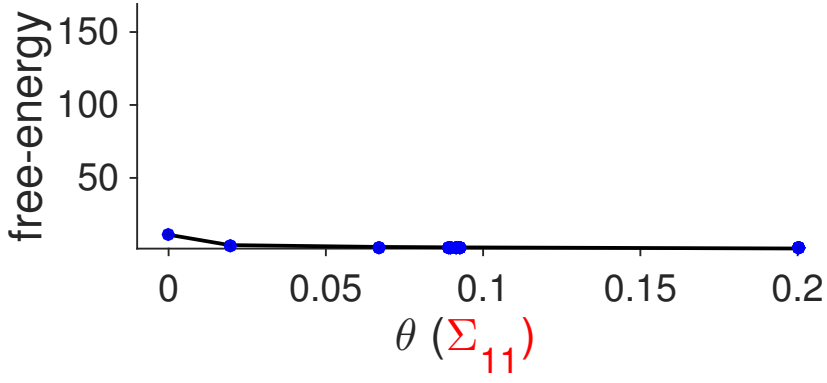
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

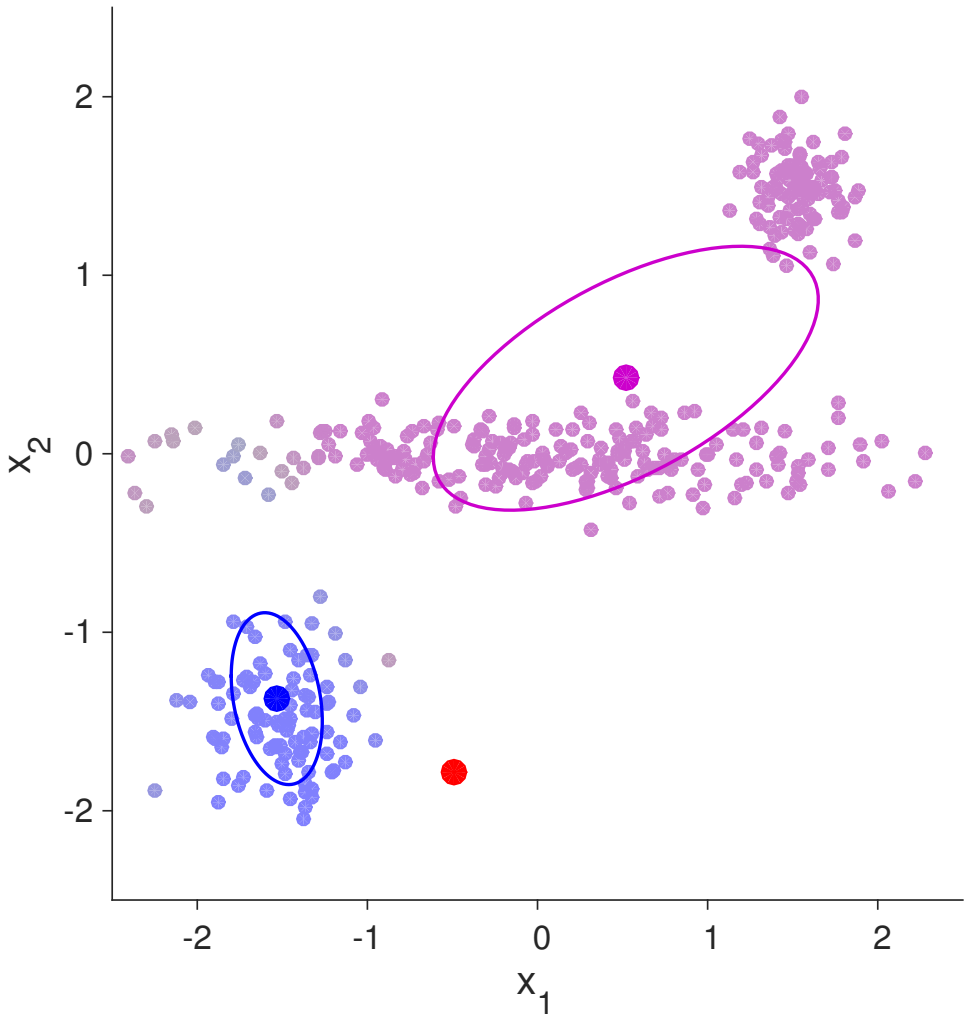
M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



What will happen with this initialisation?



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

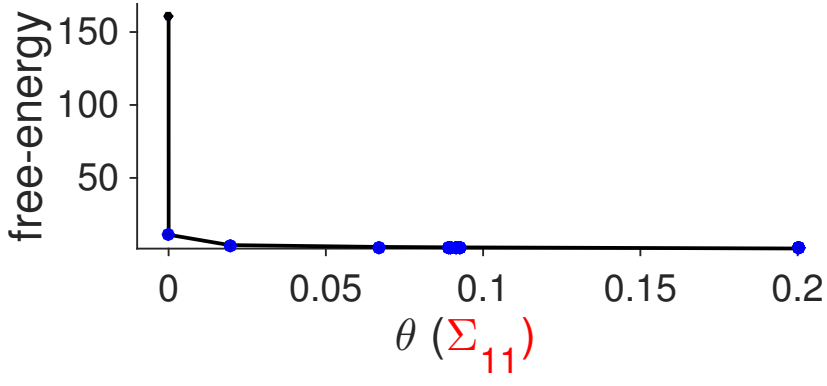
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

M-Step:

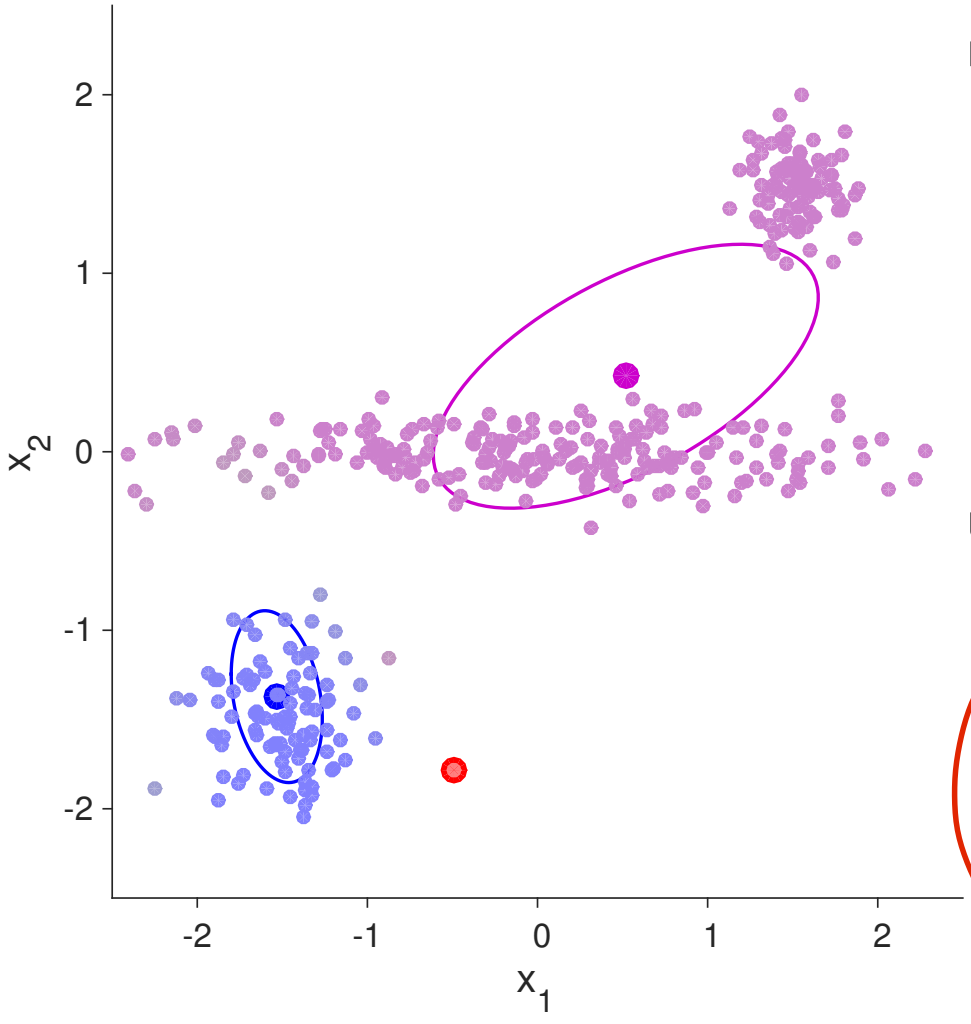
$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$

until convergence



What will happen with this initialisation?

KABOOM!



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^{K}$

repeat

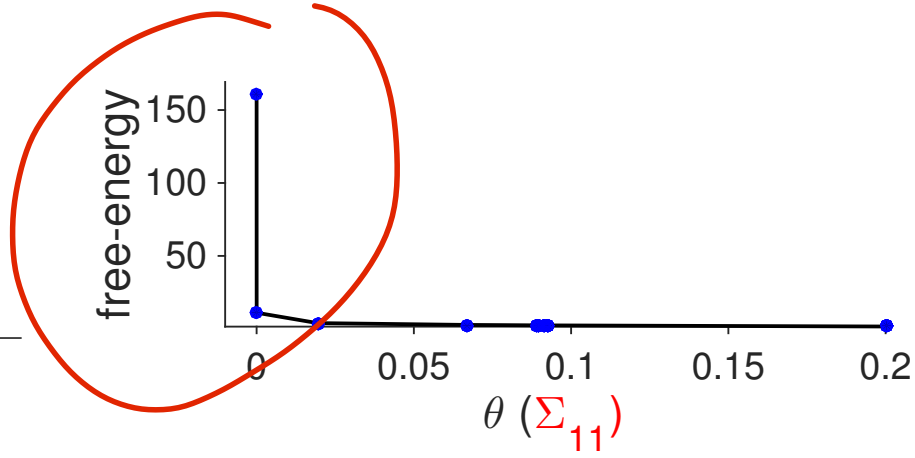
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

M-Step:

$$\arg\max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



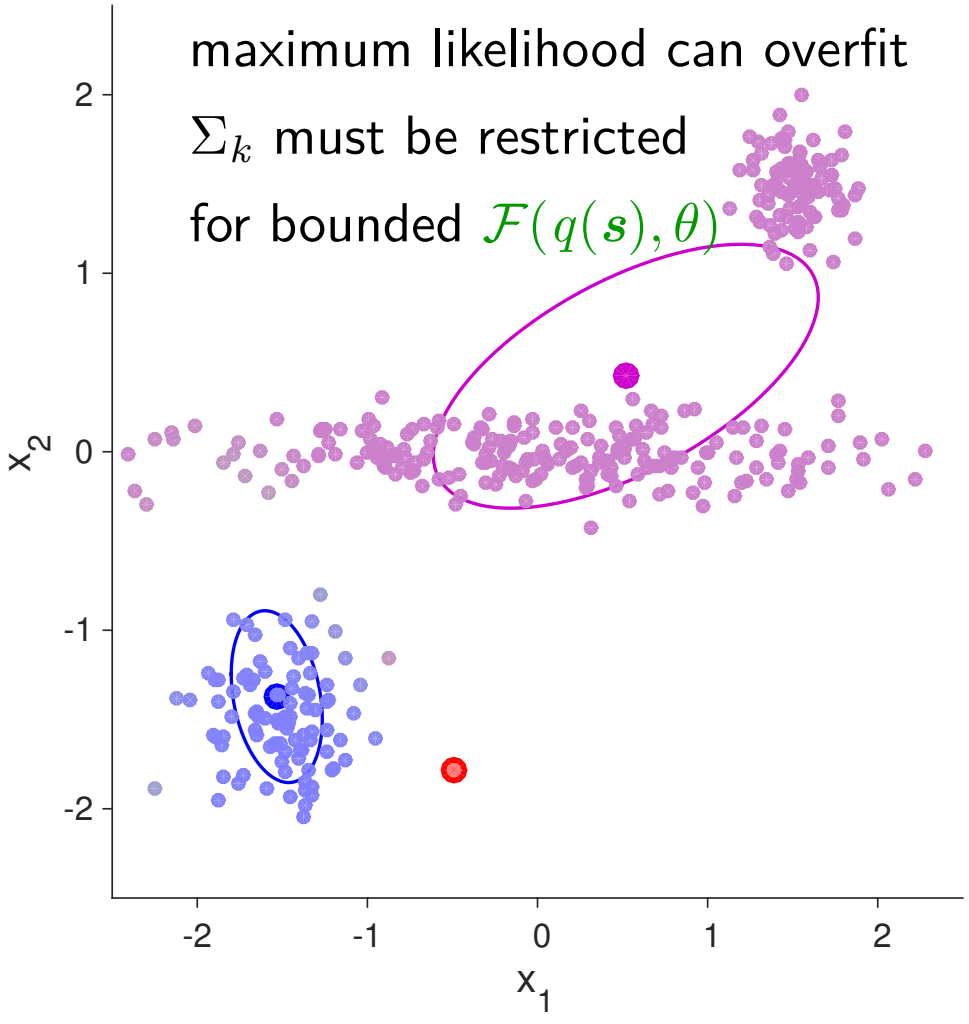
What will happen with this initialisation?

KABOOM!

maximum likelihood can overfit

Σ_k must be restricted

for bounded $\mathcal{F}(q(\boldsymbol{s}), \theta)$



initialise: $\theta = \{\pi_k, \boldsymbol{m}_k, \Sigma_k\}_{k=1}^K$

repeat

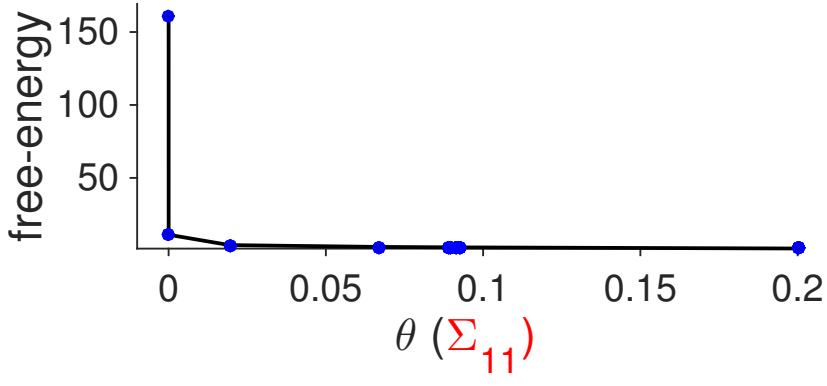
E-Step:

$r_{nk} = p(s_n = k|\boldsymbol{x}_n, \theta)$ for $n = 1 \ldots N$

M-Step:

$$\arg\max_\theta \sum_{n,k} r_{nk} \log p(s_n = k, \boldsymbol{x}|\theta)$$

until convergence



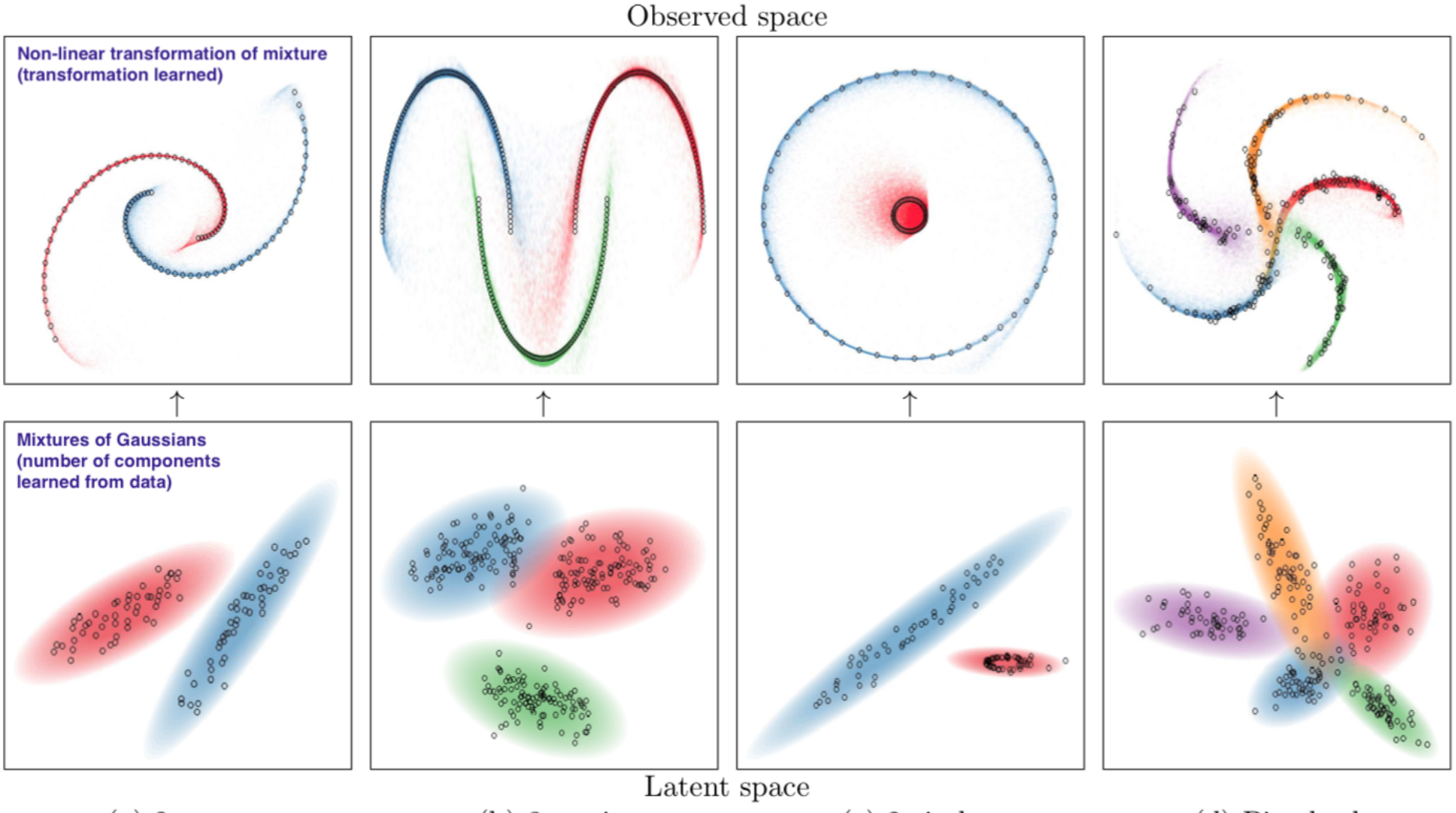
Summary

- MoG + EM algorithm = soft k-means clustering with non-axis aligned, non-equally weighted clusters
- EM can be used to fit **latent variable models** e.g. PCA, Factor analysis, MoGs, HMMs, ...
  - requires tractable posterior $p(\boldsymbol{s}|\boldsymbol{x}, \theta)$ entropy and average log-joint $\mathbb{E}_{q(\boldsymbol{s}|\theta)}\left[\log p(\boldsymbol{s}, \boldsymbol{x}|\theta)\right]$

Limitations

- MoG clusters still have simple shapes (ellipses)
  - a single real cluster might be described by many components
  - more complex cluster models have been developed
- maximum-likelihood can overfit
  - Bayesian approaches avoid overfitting $p(\theta, \boldsymbol{s}|\boldsymbol{x})$
- co-ordinate ascent is often slow to converge (lots of iterations required)
  - joint optimisation of $\mathcal{F}(q(\boldsymbol{s}), \theta)$ faster (variational inference methods)
  - direct optimisation of log-likelihood $\log p(\boldsymbol{x}|\theta)$

Research frontiers: variable numbers of non-Gaussian clusters



Warped Mixtures for Nonparametric Cluster Shapes, Iwata et al., UAI, 2013

Jupyter Notebooks

https://github.com/cambridge-mlg/mphil-intro-module

Appendix: proof of KL divergence properties

Minimise Kullback Leibler divergence (relative entropy) $\mathcal{KL}(q(x)||p(x))$: add Lagrange multiplier (enforce $q(x)$ normalises), take variational derivatives:

$$\frac{\delta}{\delta q(x)}\Big[\int q(x)\log\frac{q(x)}{p(x)}dx+\lambda(1-\int q(x)dx)\Big] \;=\; \log\frac{q(x)}{p(x)}+1-\lambda.$$

Find staionary point by setting the derivative to zero:

$$q(x) \;=\; \exp(\lambda-1)p(x), \quad \text{normalization conditon } \lambda \;=\; 1, \quad \text{so } q(x) \;=\; p(x),$$

which corresponds to a minimum, since the second derivative is positive:

$$\frac{\delta^2}{\delta q(x)\delta q(x)}\mathcal{KL}(q(x)||p(x)) \;=\; \frac{1}{q(x)} \;>\; 0.$$

The minimum value attained at $q(x)=p(x)$ is $\mathcal{KL}(p(x)||p(x))=0$, showing that $\mathcal{KL}(q(x)||p(x))$

- is non-negative
- attains its minimum 0 when $p(x)$ and $q(x)$ are equal