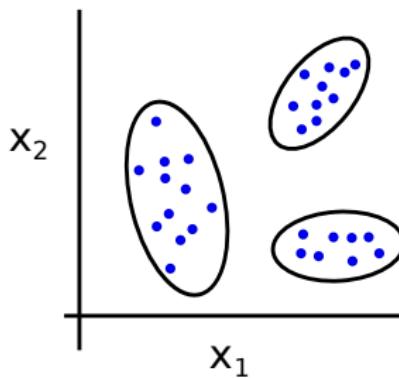


Clustering and the EM algorithm

Rich Turner and José Miguel Hernández-Lobato



What is clustering?

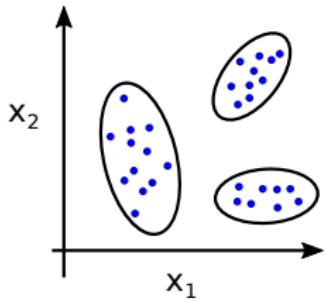
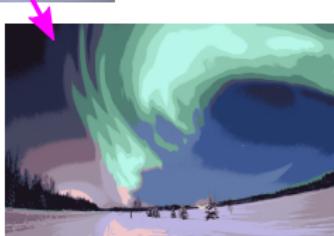
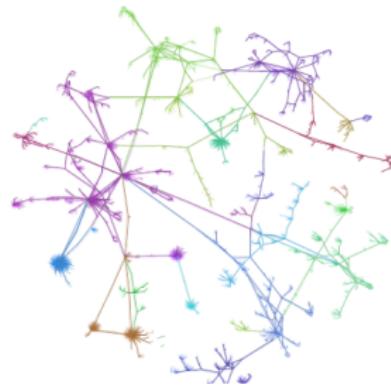


image segmentation



network community detection

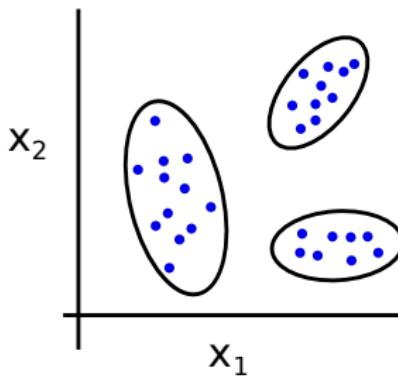


Campbell et al Social Network Analysis

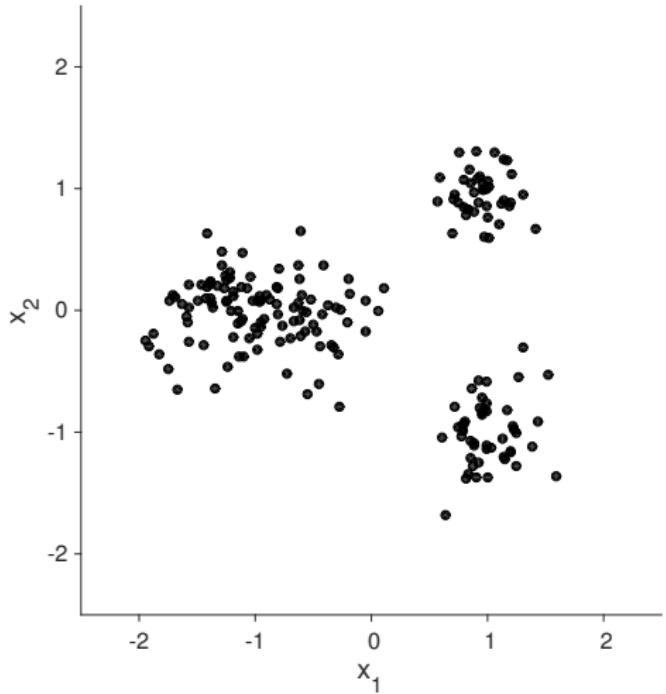
vector quantisation
genetic clustering
anomaly detection
crime analysis

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.
- ▶ $\mathcal{D} = \{x_1 \dots x_N\} \rightarrow s = \{s_1 \dots s_N\}$ where $s_n \in \{1 \dots K\}$
- ▶ Unsupervised learning problem (no labels or rewards)



A first clustering algorithm: k-means



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

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

repeat

for $n = 1 \dots N$

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

endfor

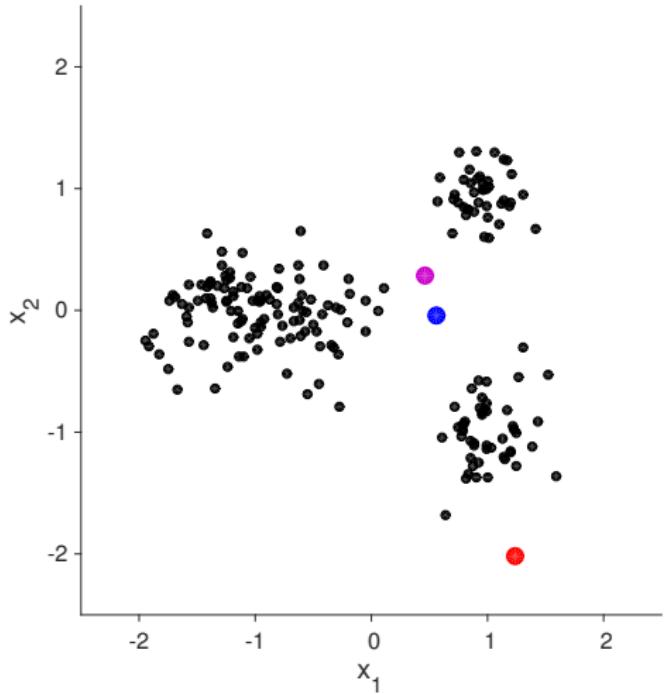
for $k = 1 \dots K$

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

endfor

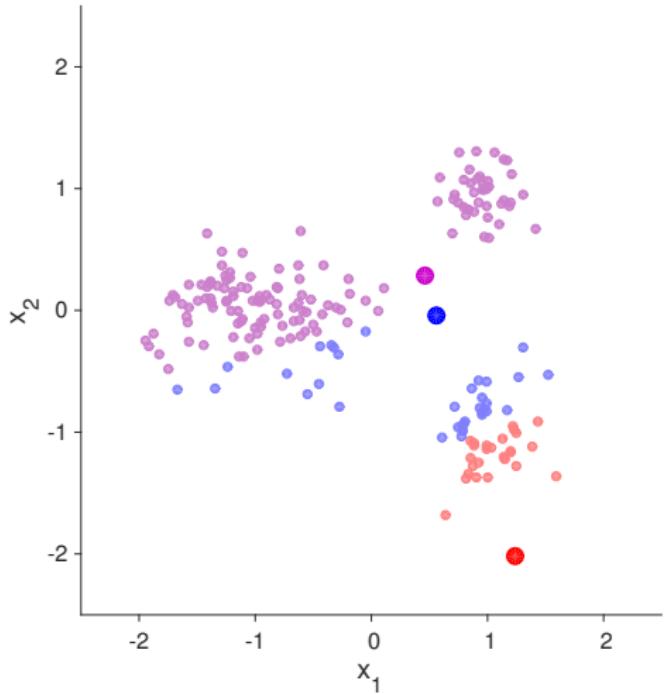
until convergence (s_n fixed)

A first clustering algorithm: k-means



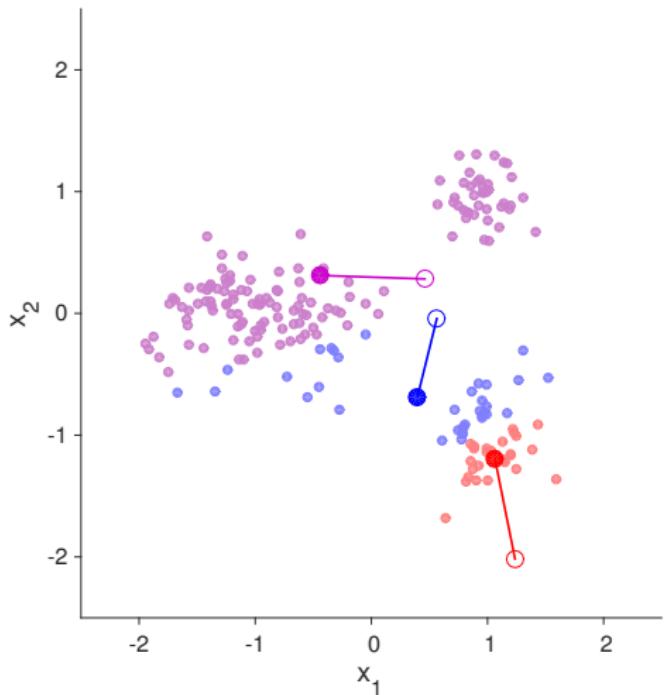
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}, \mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



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

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

repeat

for $n = 1 \dots N$

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

endfor

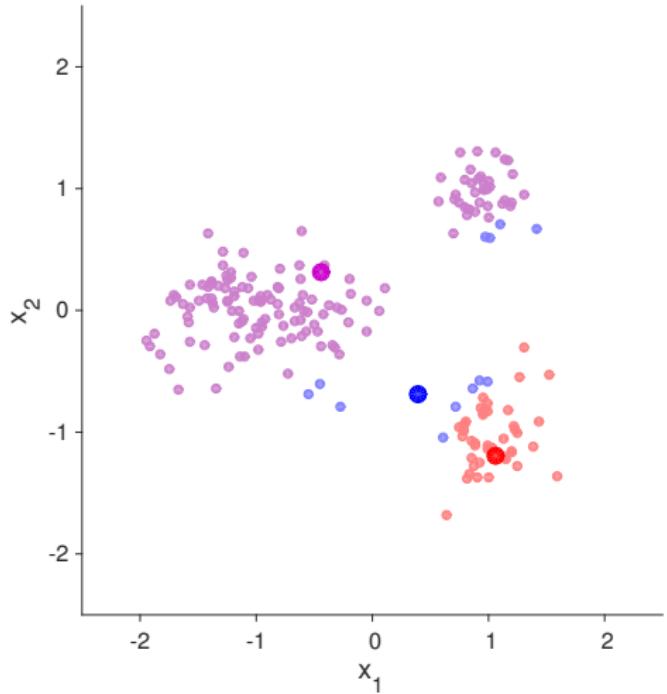
for $k = 1 \dots K$

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

endfor

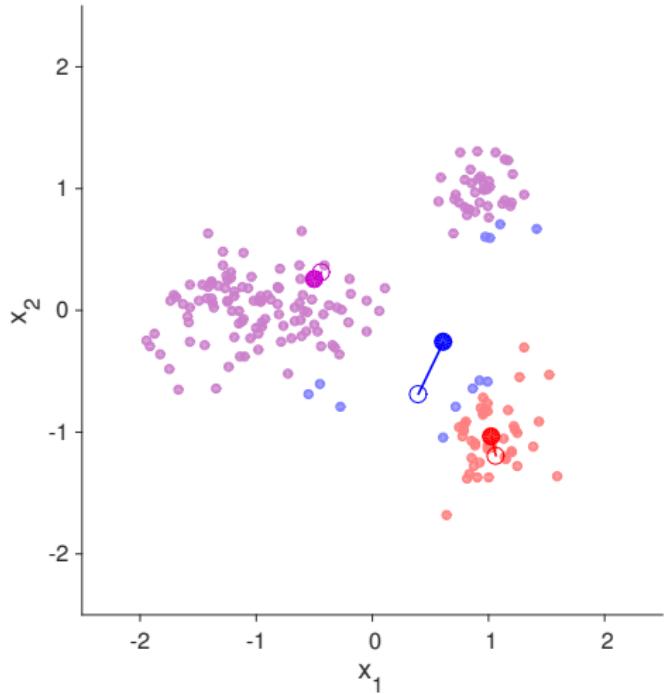
until convergence (s_n fixed)

A first clustering algorithm: k-means



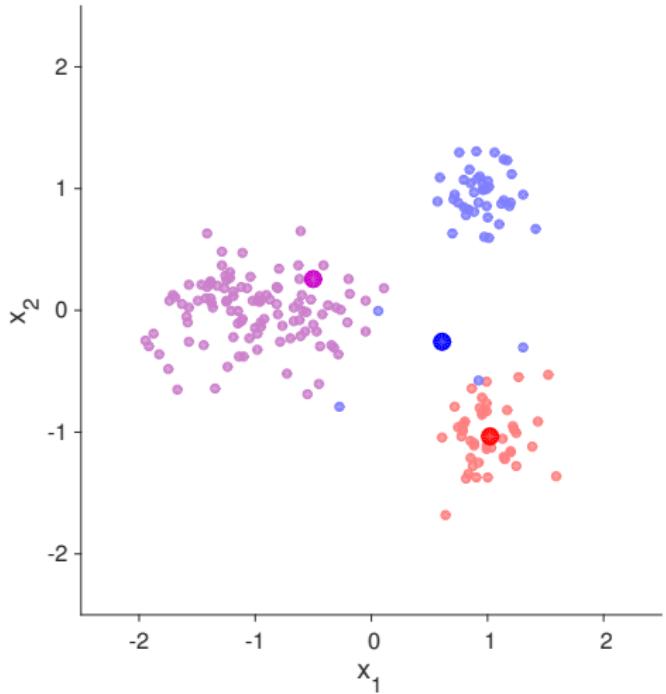
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



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

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

repeat

for $n = 1 \dots N$

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

endfor

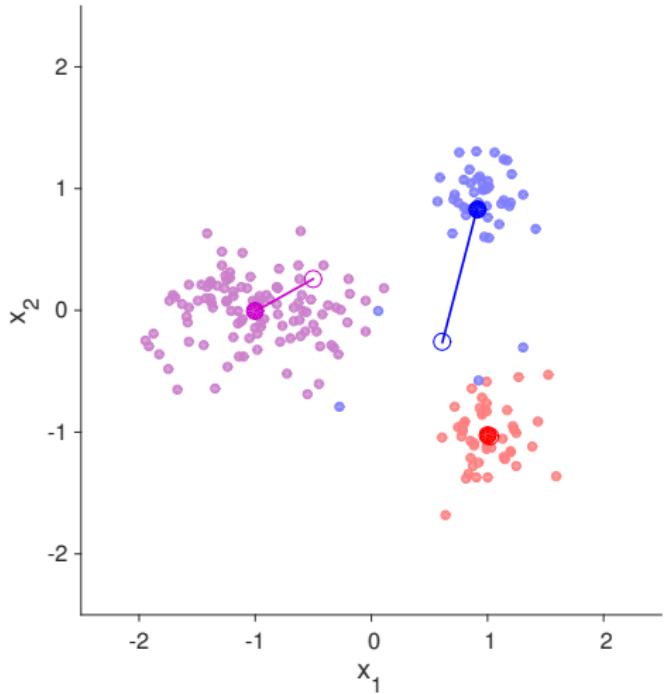
for $k = 1 \dots K$

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

endfor

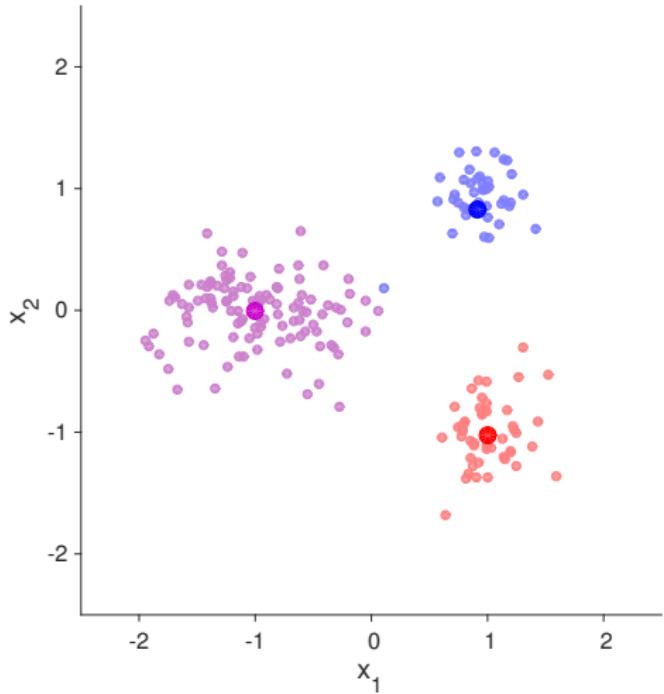
until convergence (s_n fixed)

A first clustering algorithm: k-means



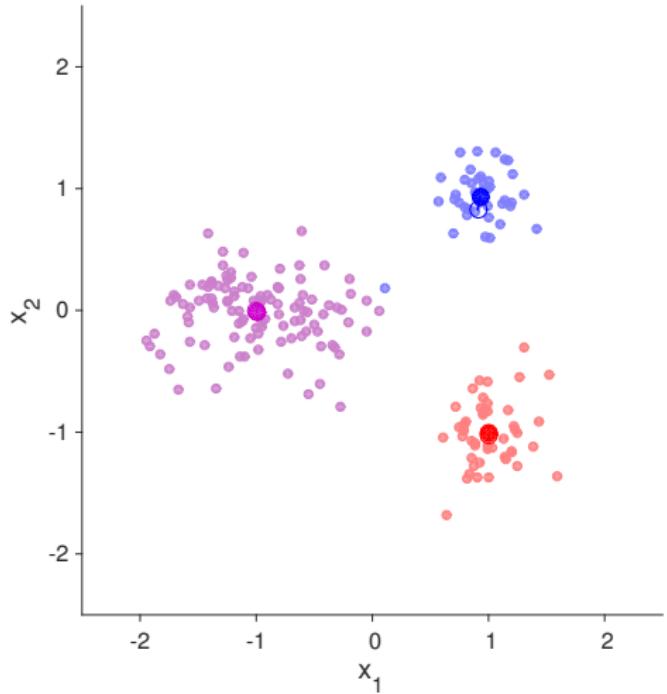
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

A first clustering algorithm: k-means



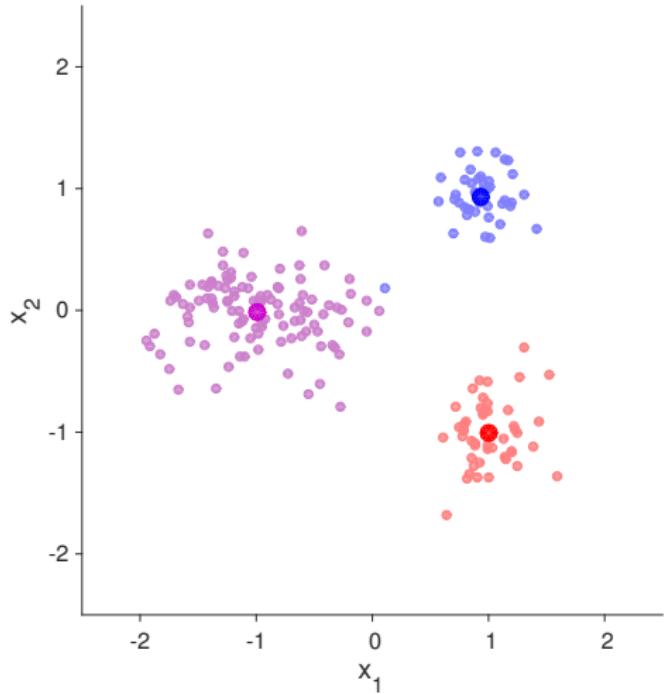
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



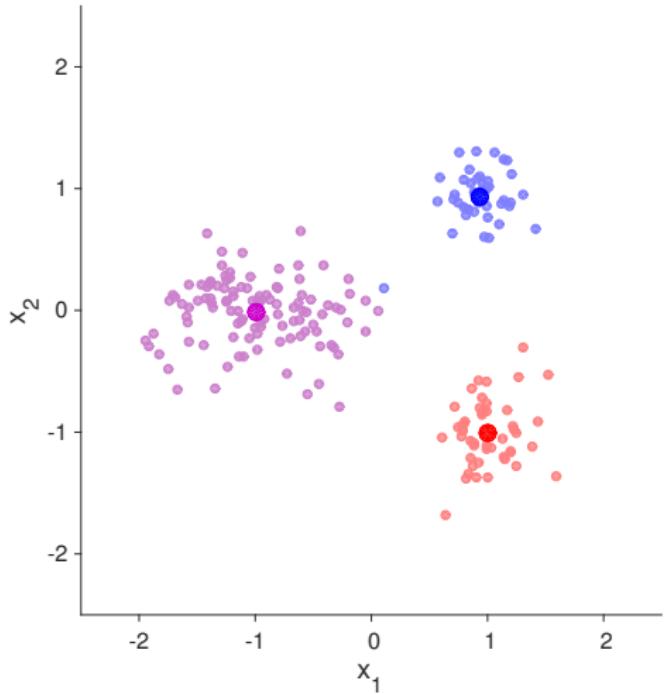
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



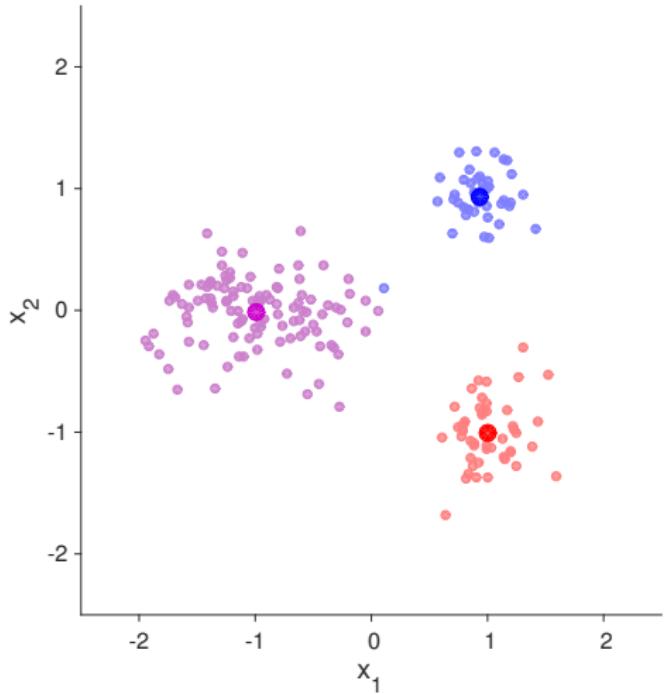
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



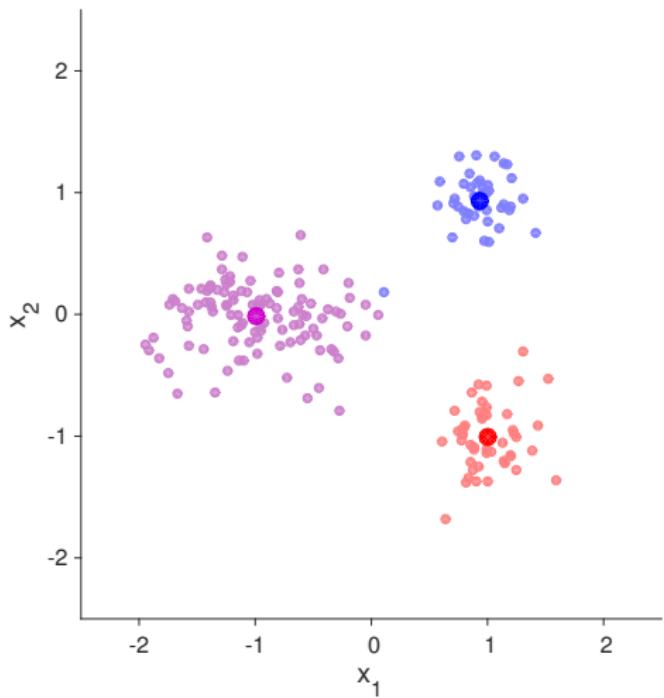
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

A first clustering algorithm: k-means



input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 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: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

K-means as optimisation

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

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

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:

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

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:

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

K-means tries to minimise the cost function \mathcal{C} with respect to $\{s_{n,k}\}$ and $\{\mathbf{m}_k\}$, 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 $\{\mathbf{m}_k\}$ fixed.
- ▶ minimises \mathcal{C} with respect to $\{\mathbf{m}_k\}$, holding $\{s_{n,k}\}$ fixed.

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:

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

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

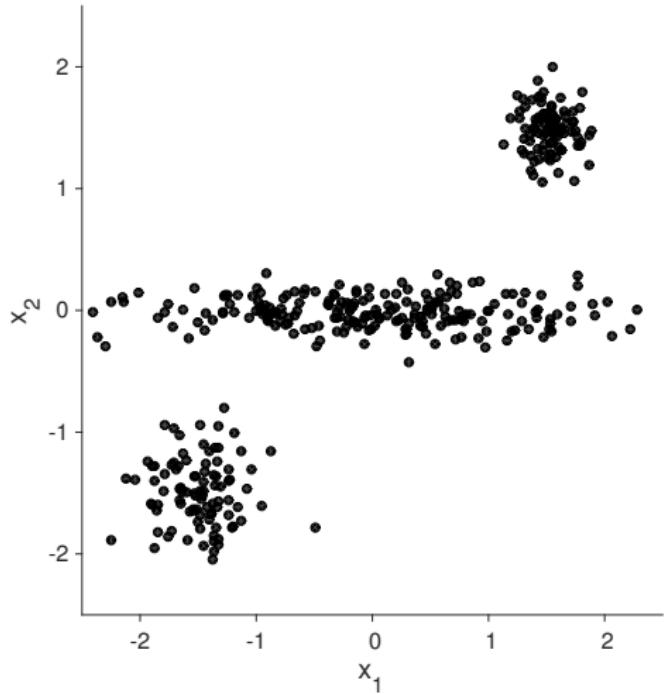
K-means sequentially:

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

\mathcal{C} is a Lyupanov function \implies K-means converges, but...

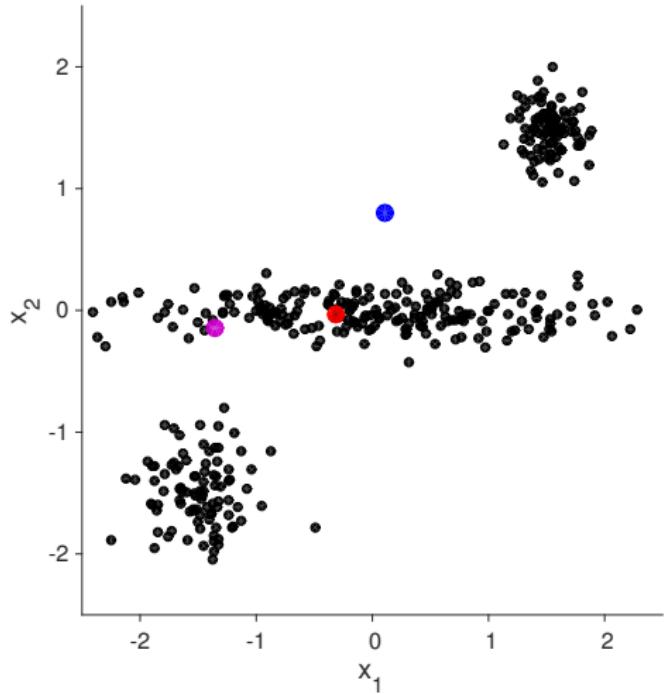
Finding the global optimum of \mathcal{C} is a hard problem.

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



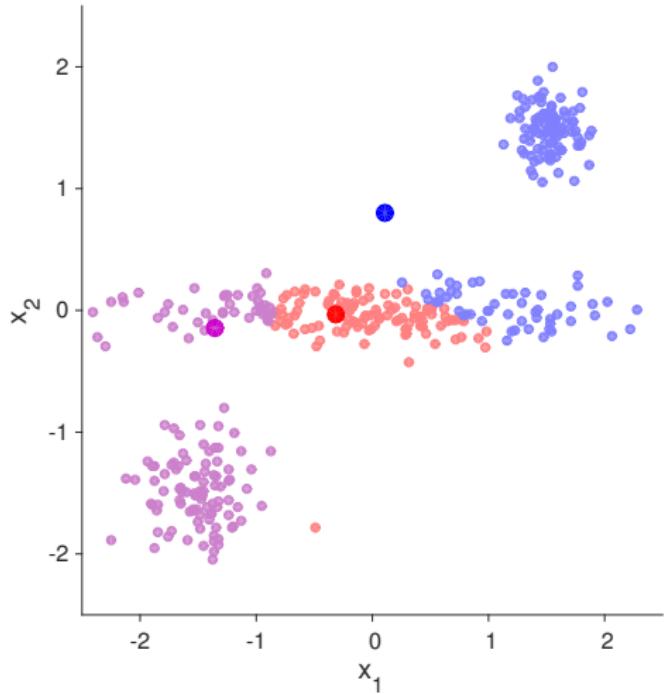
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



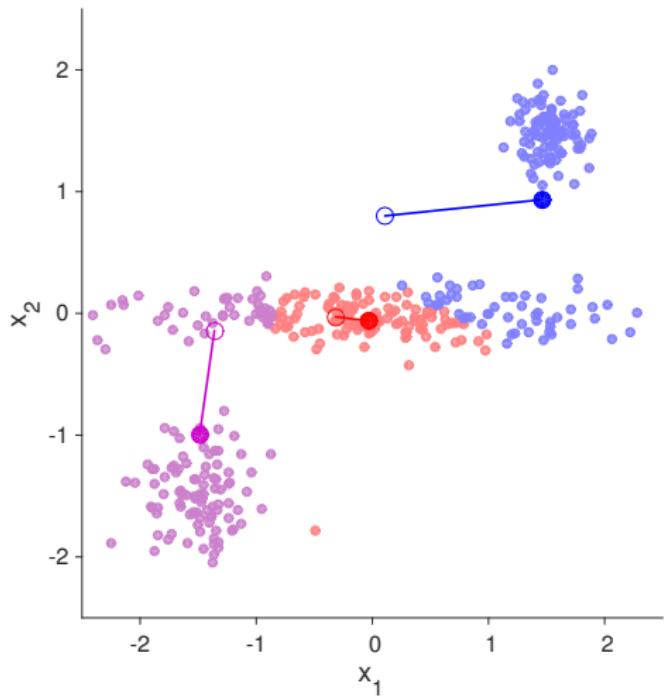
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



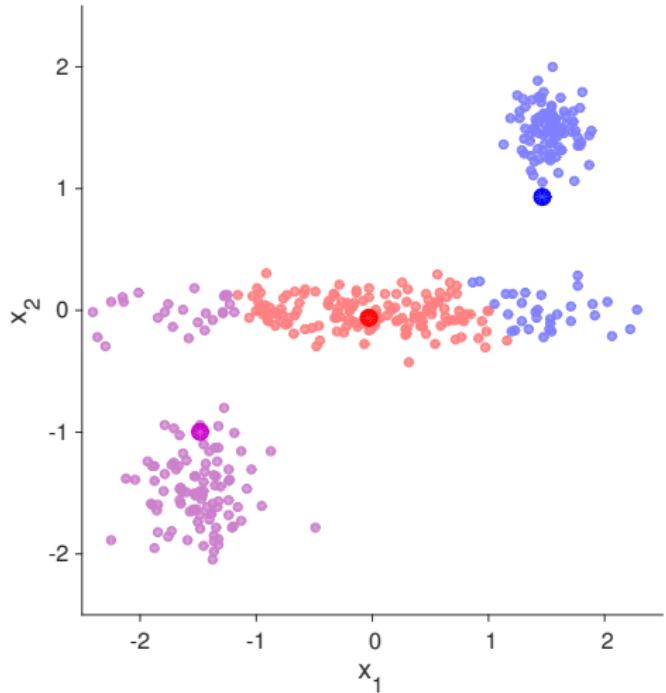
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



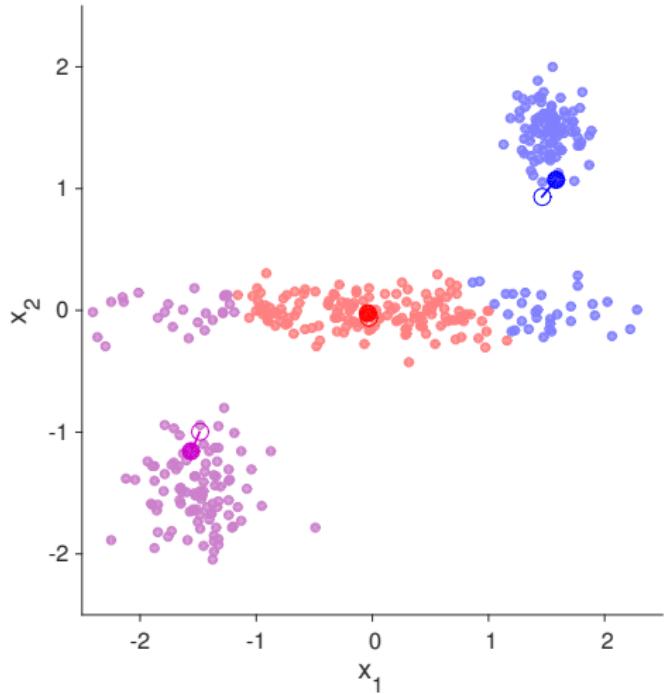
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



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

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

repeat

for $n = 1 \dots N$

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

endfor

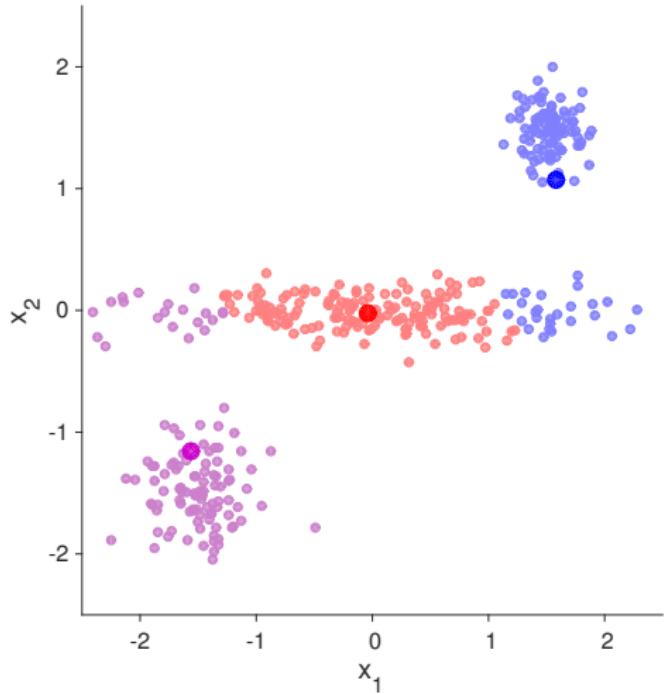
for $k = 1 \dots K$

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

endfor

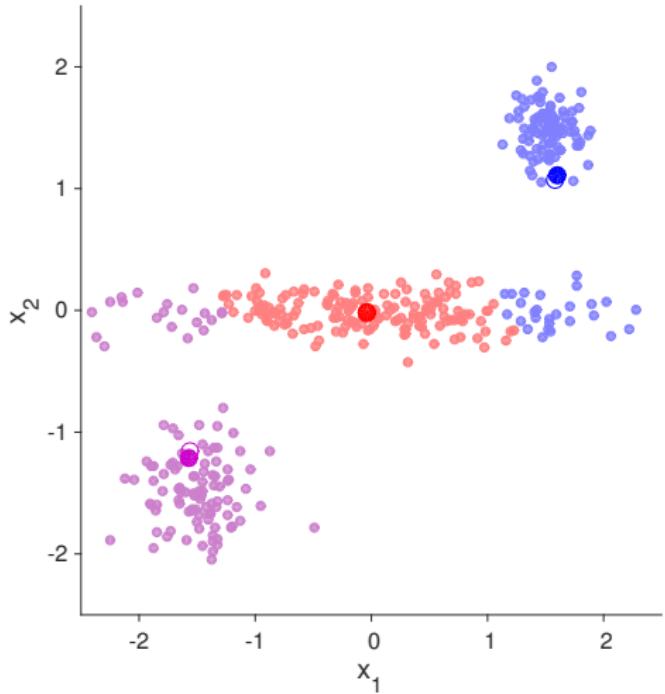
until convergence (s_n fixed)

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



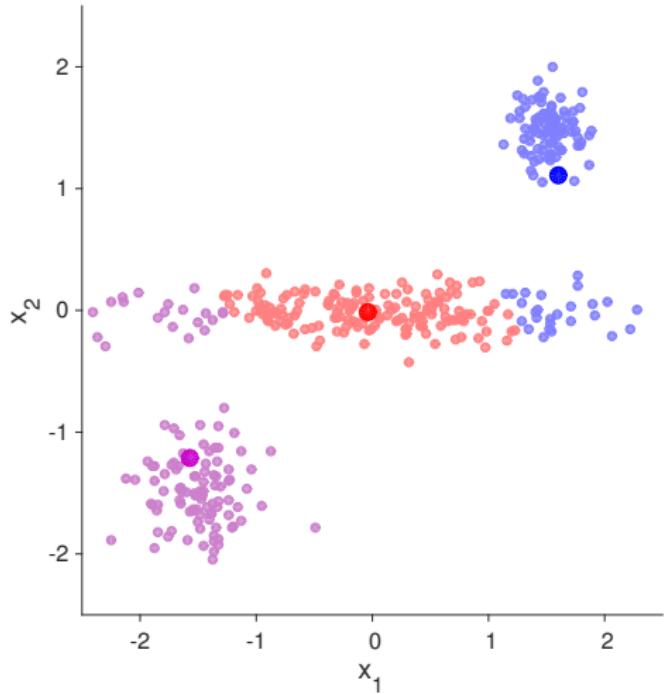
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



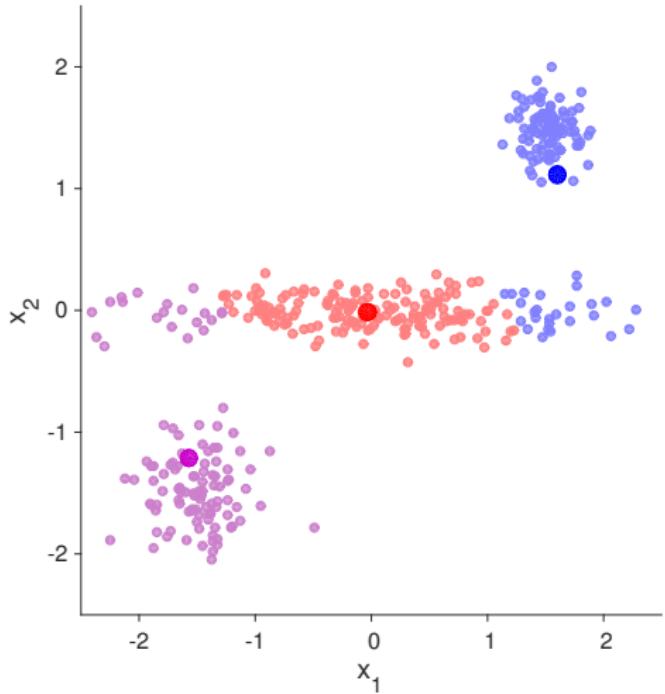
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

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



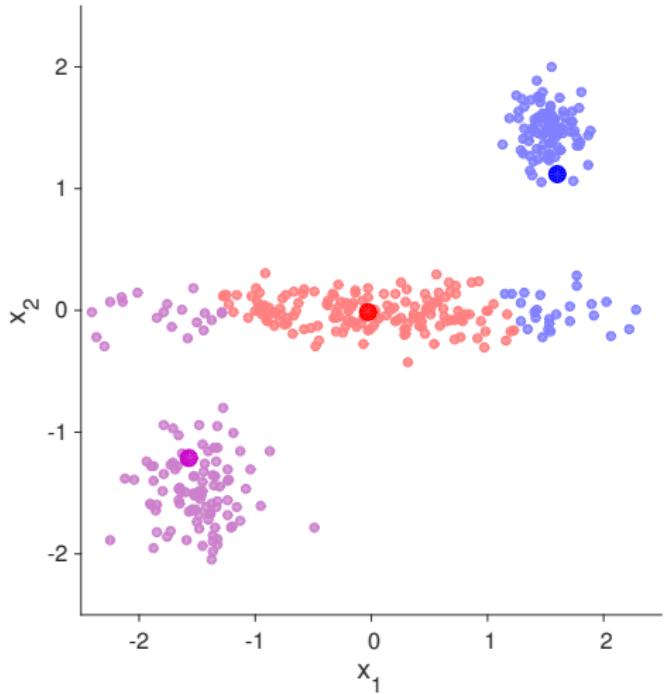
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



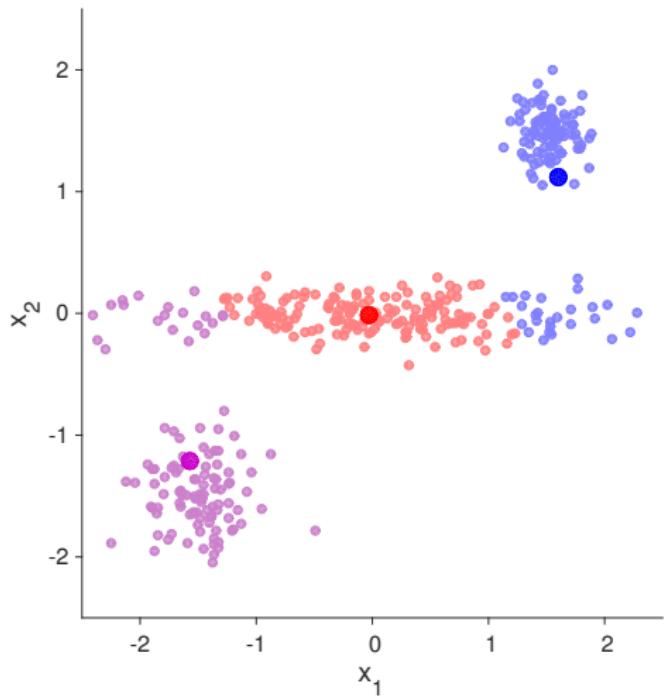
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

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



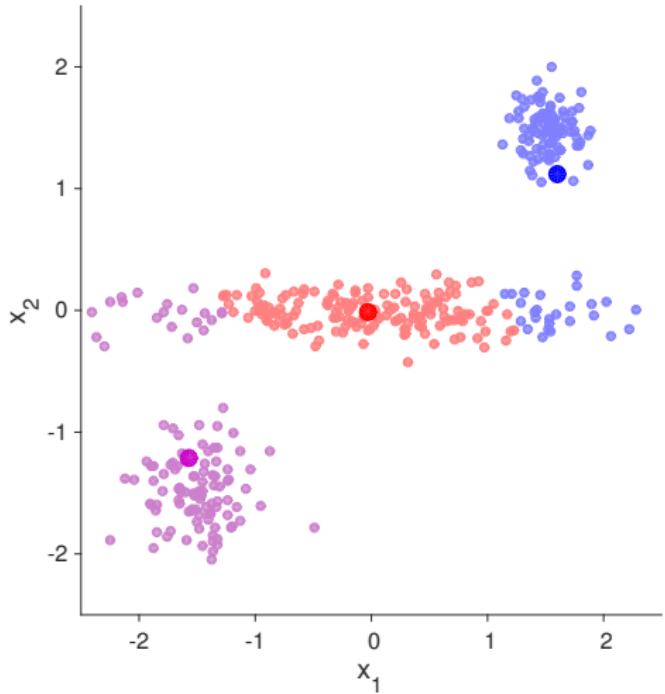
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



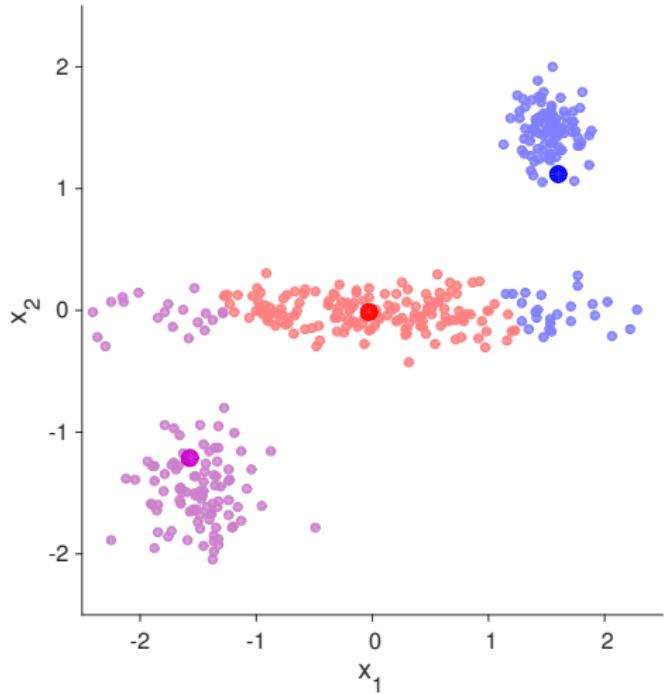
input: $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n \in \mathbb{R}^D$
initialise: $\mathbf{m}_k \in \mathbb{R}^D$ for $k = 1 \dots K$
repeat
 for $n = 1 \dots N$
 $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$
 endfor
 for $k = 1 \dots K$
 $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$
 endfor
until convergence (s_n fixed)

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



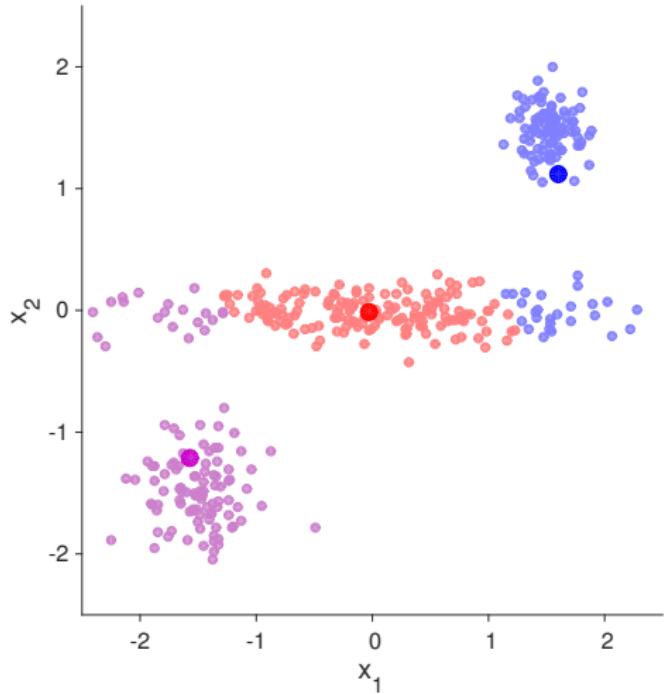
```
input:  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ,  $\mathbf{x}_n \in \mathbb{R}^D$ 
initialise:  $\mathbf{m}_k \in \mathbb{R}^D$  for  $k = 1 \dots K$ 
repeat
    for  $n = 1 \dots N$ 
         $s_n = \arg \min_k \|\mathbf{x}_n - \mathbf{m}_k\|$ 
    endfor
    for  $k = 1 \dots K$ 
         $\mathbf{m}_k = \text{mean}(\mathbf{x}_n : s_n = k)$ 
    endfor
until convergence ( $s_n$  fixed)
```

Pathologies of K-means



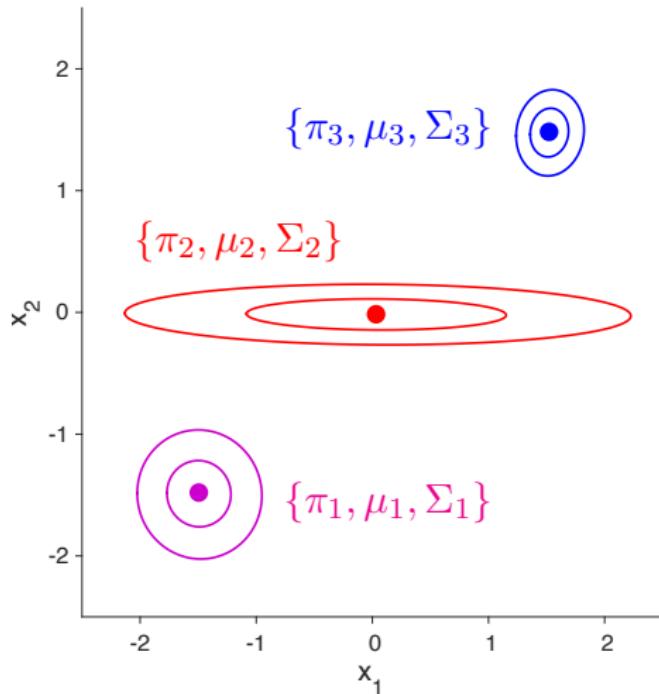
- ▶ strange results when clusters have very different shapes
- ▶ returns hard assignments
 - ▶ soft assignments more sensible
- ▶ initialisation delicate
 - ▶ k++ means algorithm: spreads out initial centres

Pathologies of K-means



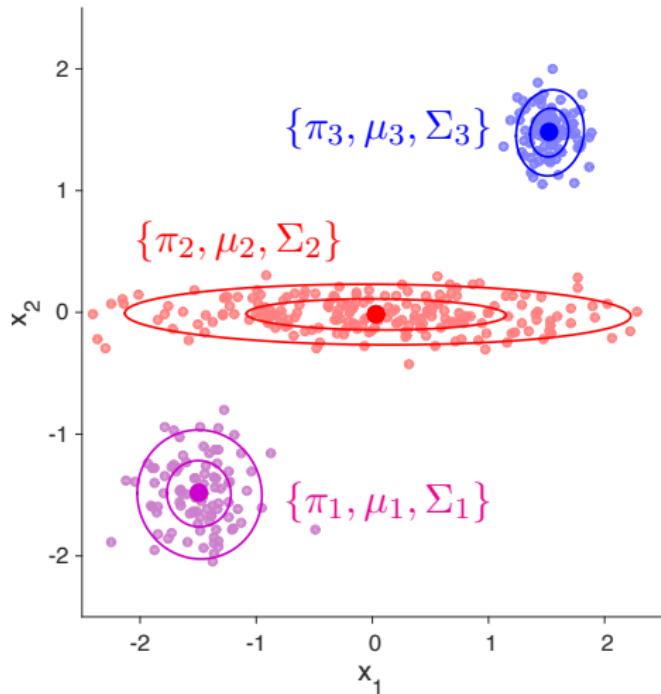
- ▶ strange results when clusters have very different shapes
- ▶ returns hard assignments
 - ▶ soft assignments more sensible
- ▶ initialisation delicate
 - ▶ k++ means algorithm: spreads out initial centres

Mixture of Gaussians: Generative Model



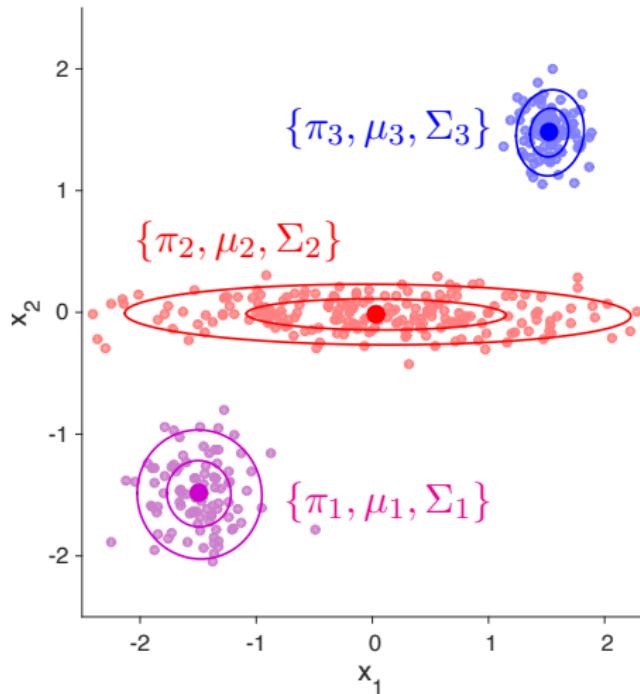
For each data point $1 \dots N$
sample cluster membership:
 $p(s_n = k|\theta) = \pi_k$ note $\sum_{k=1}^K \pi_k = 1$
sample data-value given cluster mem.:
 $p(\mathbf{x}_n | s_n = k, \theta) = \mathcal{N}(\mathbf{x}_n; \mathbf{m}_k, \Sigma_k)$

Mixture of Gaussians: Generative Model



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

Mixture of Gaussians: Generative Model



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

How can we learn the parameters of this model from data using maximum-likelihood?

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

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

$$\overbrace{\log p(\mathbf{x}|\theta)}^{\text{log-likelihood}}$$

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

arbitrary distribution
over class memberships

posterior distribution
over class memberships

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

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

arbitrary distribution
over class memberships

posterior distribution
over class memberships

$$\log p(\mathbf{x}|\theta) - \underbrace{\sum_{\mathbf{s}} q(\mathbf{s}) \log \frac{q(\mathbf{s})}{p(\mathbf{s}|\mathbf{x}, \theta)}}_{\text{KL-Divergence}} \quad \mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta))$$

log-likelihood

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

$$\mathcal{F}(q(\mathbf{s}), \theta) = \underbrace{\log p(\mathbf{x}|\theta)}_{\text{log-likelihood}} - \underbrace{\sum_{\mathbf{s}} q(\mathbf{s}) \log \frac{q(\mathbf{s})}{p(\mathbf{s}|\mathbf{x}, \theta)}}_{\text{KL-Divergence}} \quad \begin{matrix} \text{arbitrary distribution} \\ \text{over class memberships} \end{matrix} \quad \begin{matrix} \text{posterior distribution} \\ \text{over class memberships} \end{matrix}$$

free-energy log-likelihood

KL-Divergence
 $\mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta))$

A brief introduction to the Kullback-Leibler divergence

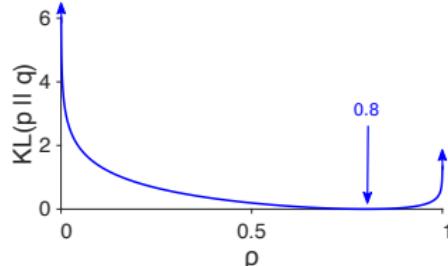
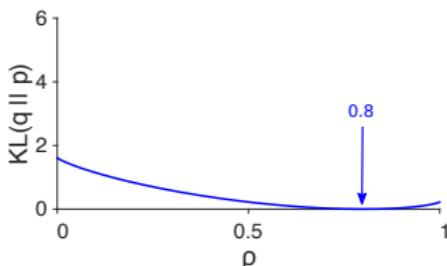
$$\mathcal{KL}(p_1(z)||p_2(z)) = \sum_z p_1(z) \log \frac{p_1(z)}{p_2(z)}$$

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 $q(z=1) = \rho$



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

$$\mathcal{F}(q(\mathbf{s}), \theta) = \underbrace{\log p(\mathbf{x}|\theta)}_{\text{log-likelihood}} - \underbrace{\sum_{\mathbf{s}} q(\mathbf{s}) \log \frac{q(\mathbf{s})}{p(\mathbf{s}|\mathbf{x}, \theta)}}_{\text{KL-Divergence}} \quad \begin{matrix} \text{arbitrary distribution} \\ \text{over class memberships} \end{matrix} \quad \begin{matrix} \text{posterior distribution} \\ \text{over class memberships} \end{matrix}$$

free-energy log-likelihood

KL-Divergence
 $\mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta))$

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

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

arbitrary distribution over class memberships posterior distribution over class memberships

Non-negativity
of KL-Divergence

Free-energy is lower bound on
log-likelihood

$$\mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta)) \geq 0 \implies \mathcal{F}(q(\mathbf{s}), \theta) \leq \log p(\mathbf{x}|\theta)$$

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

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

arbitrary distribution over class memberships posterior distribution over class memberships

Non-negativity
of KL-Divergence

$$\mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta)) \geq 0 \implies \mathcal{F}(q(\mathbf{s}), \theta) \leq \log p(\mathbf{x}|\theta)$$

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

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

Free-energy is lower bound on
log-likelihood

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

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

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

arbitrary distribution over class memberships posterior distribution over class memberships

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

Non-negativity
of KL-Divergence

Free-energy is lower bound on
log-likelihood

$$\mathcal{KL}(q(\mathbf{s}) \| p(\mathbf{s}|\mathbf{x}, \theta)) \geq 0 \implies \mathcal{F}(q(\mathbf{s}), \theta) \leq \log p(\mathbf{x}|\theta)$$

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

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

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

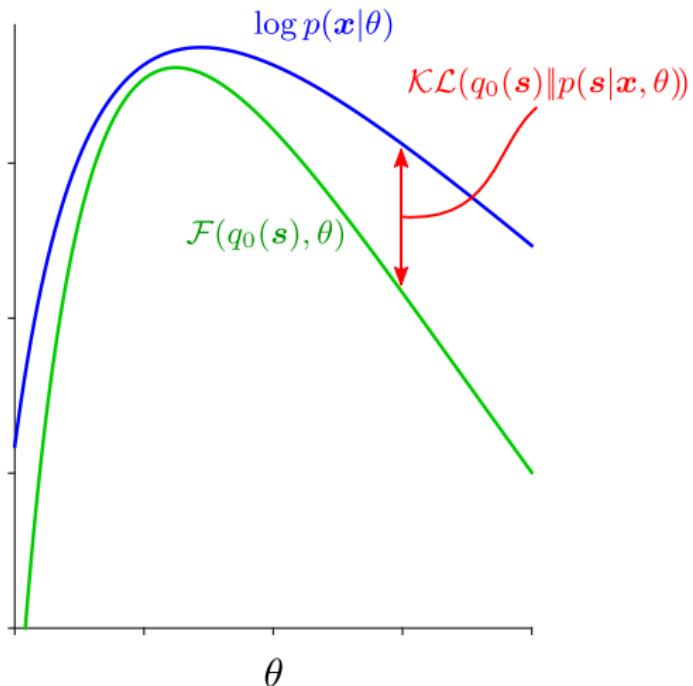
Visualising the free-energy lower bound

Visualising the free-energy lower bound

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

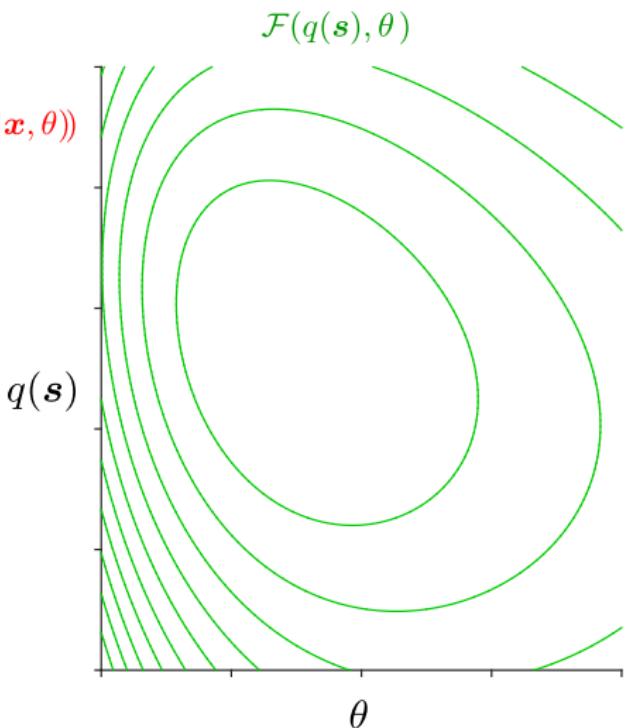
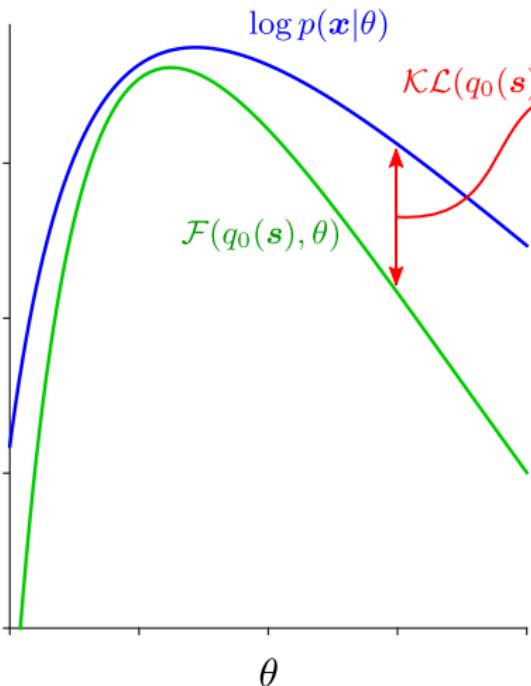
Visualising the free-energy lower bound

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



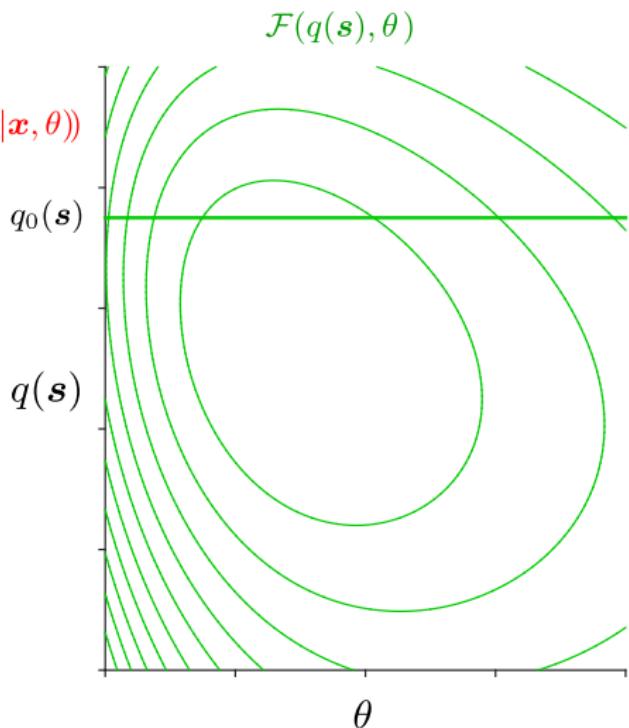
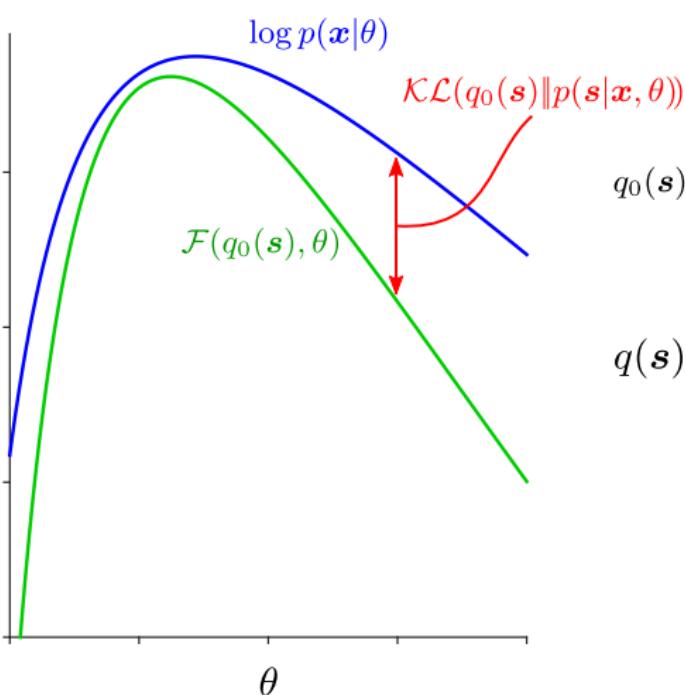
Visualising the free-energy lower bound

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



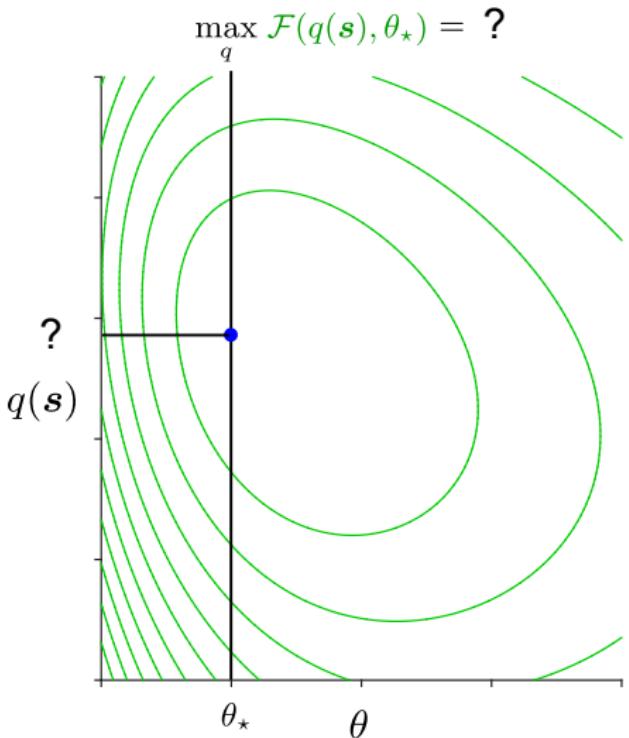
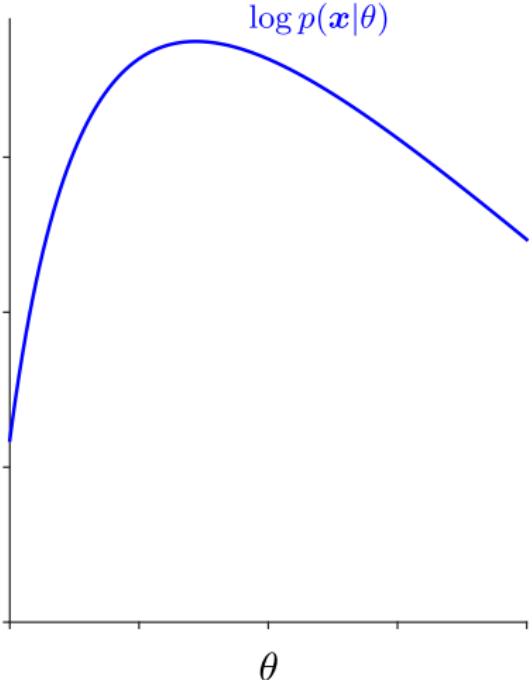
Visualising the free-energy lower bound

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



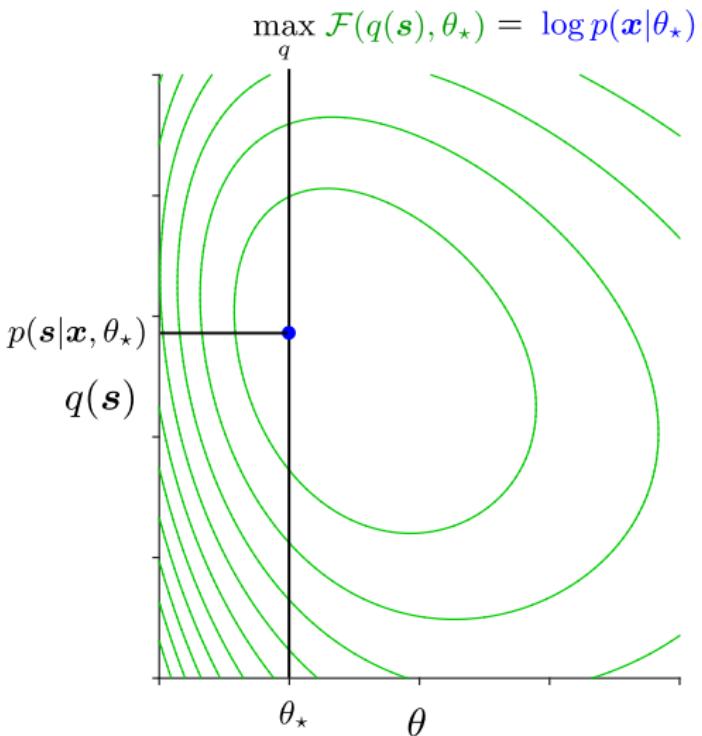
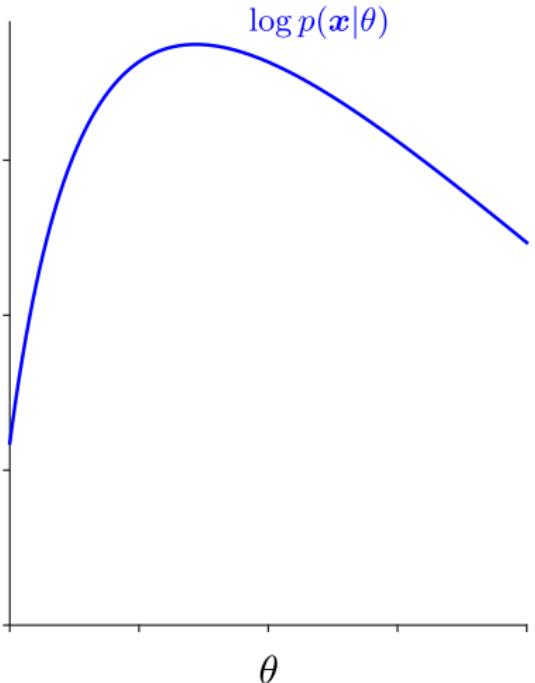
What is the maximal value of the free-energy along this vertical slice?

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



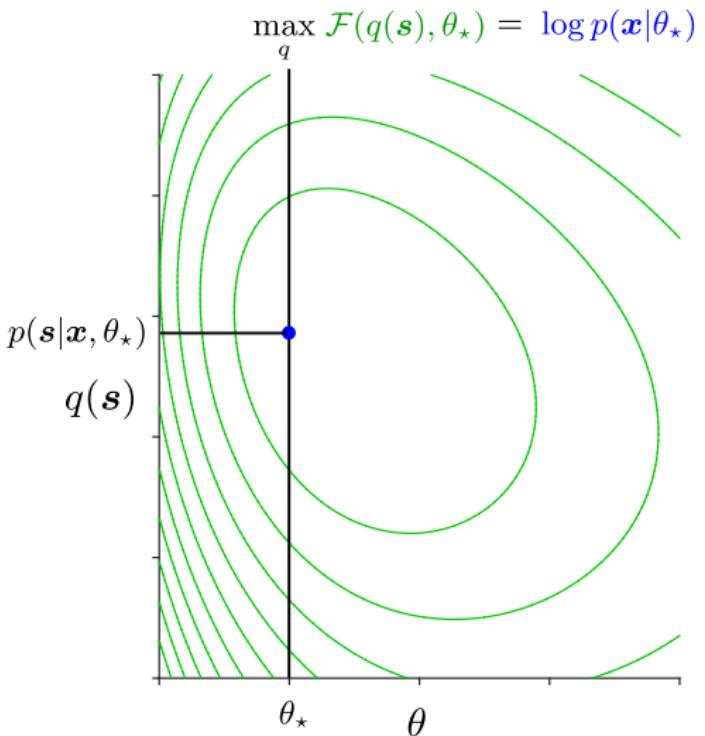
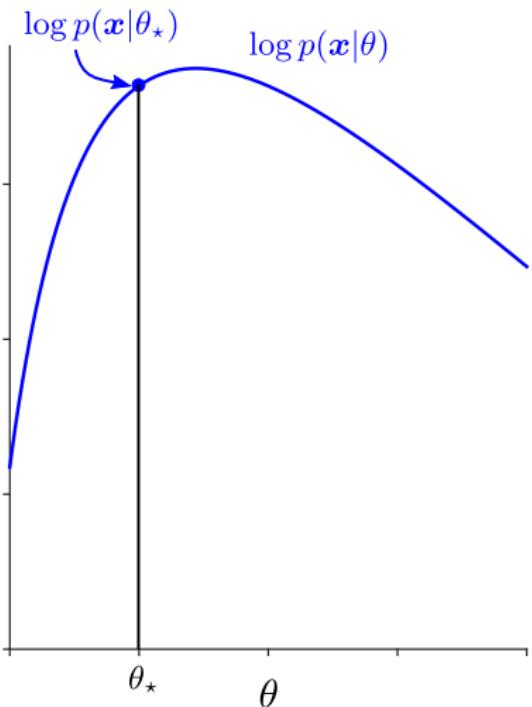
What is the maximal value of the free-energy along this vertical slice?

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



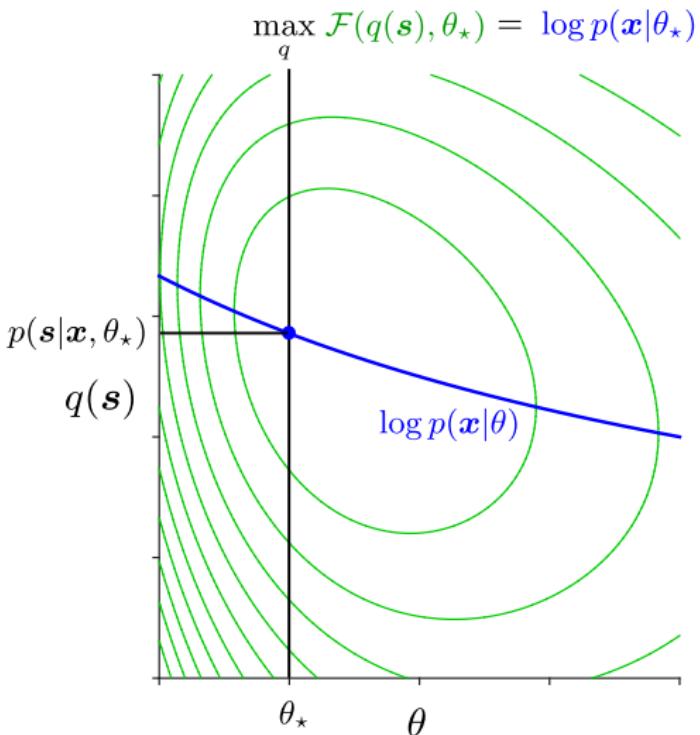
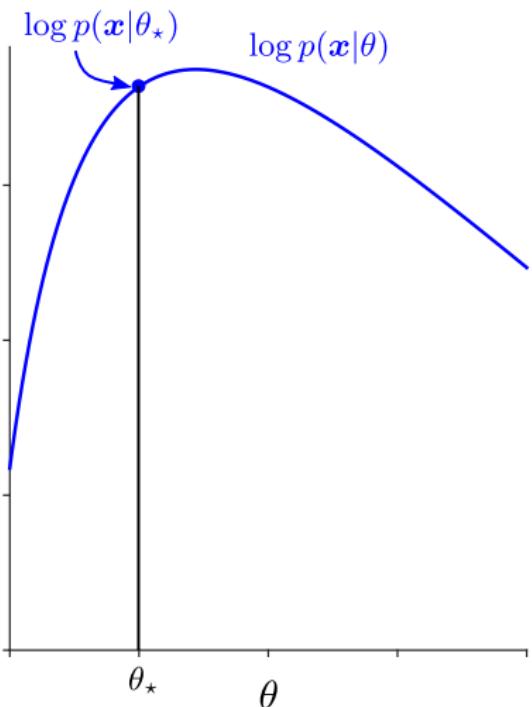
What is the maximal value of the free-energy along this vertical slice?

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



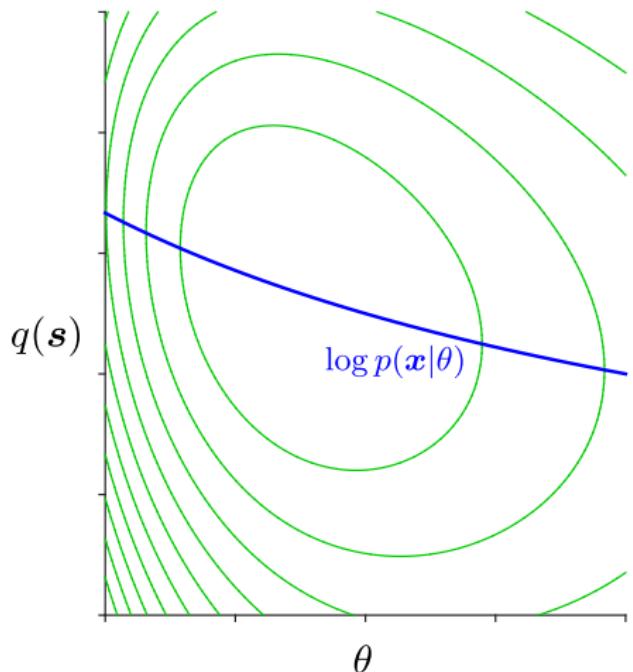
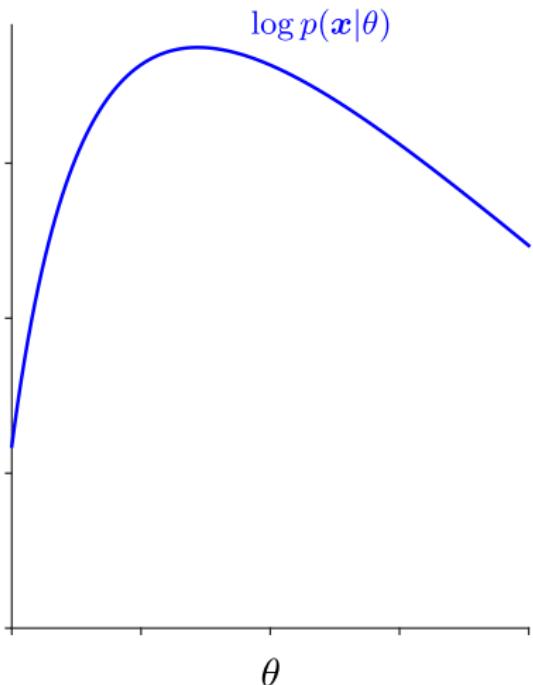
What is the maximal value of the free-energy along this vertical slice?

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



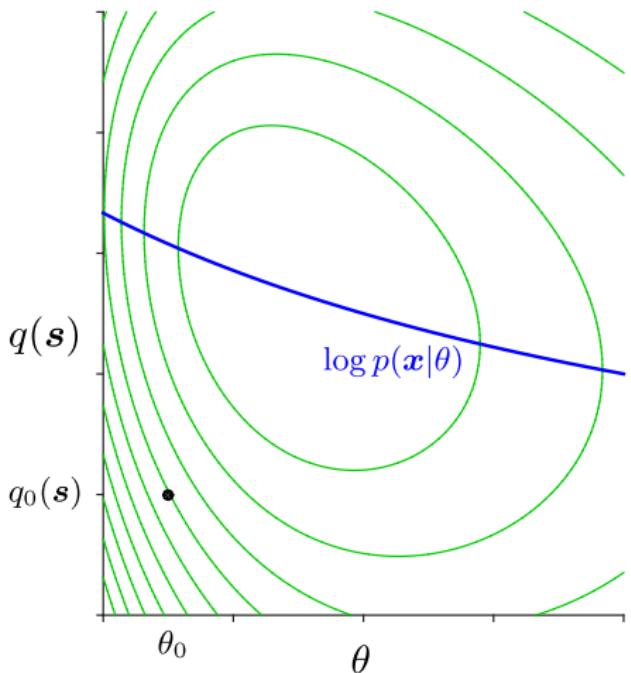
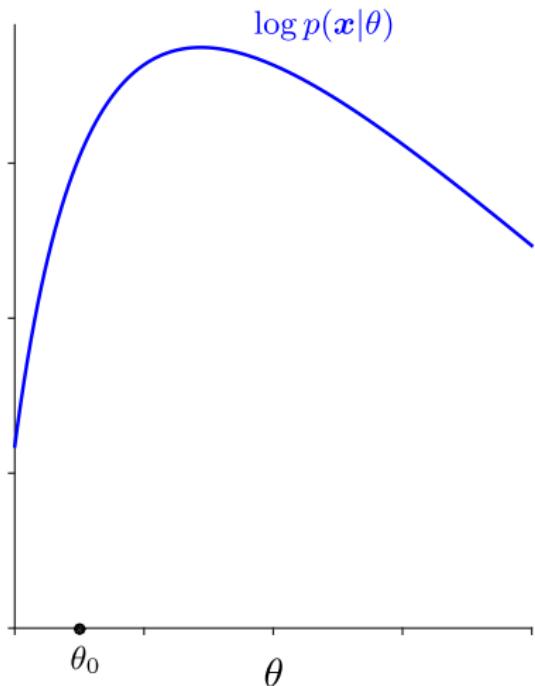
The Expectation Maximisation (EM) algorithm

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



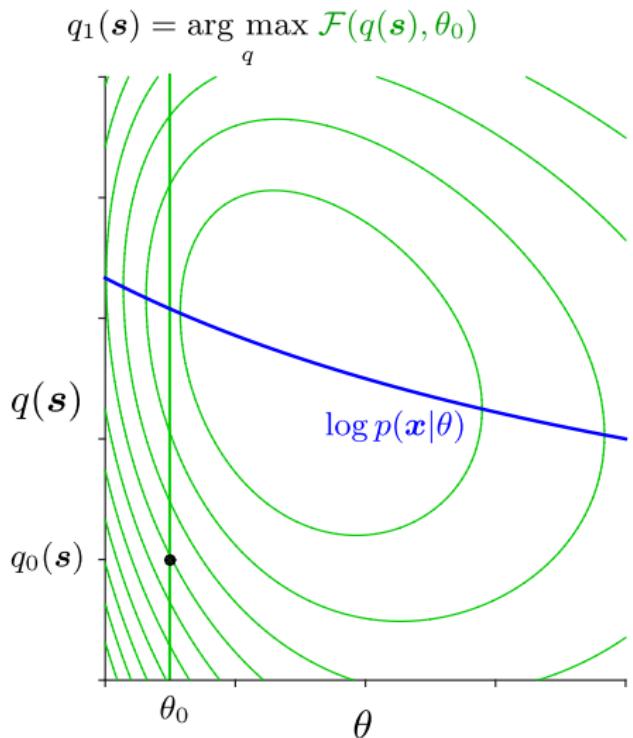
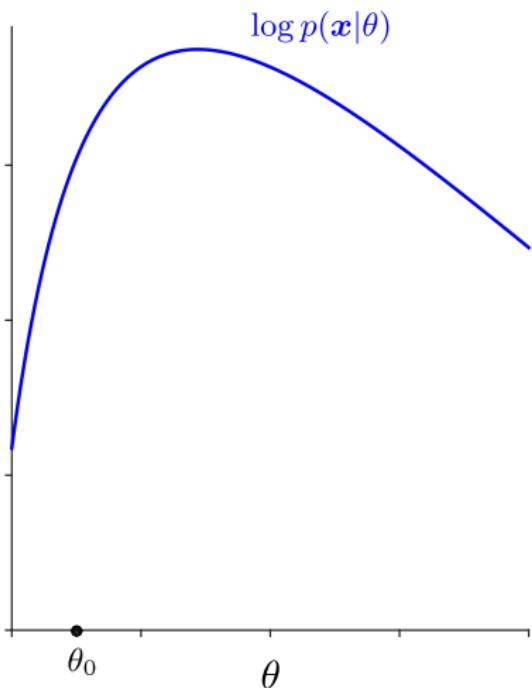
The Expectation Maximisation (EM) algorithm

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



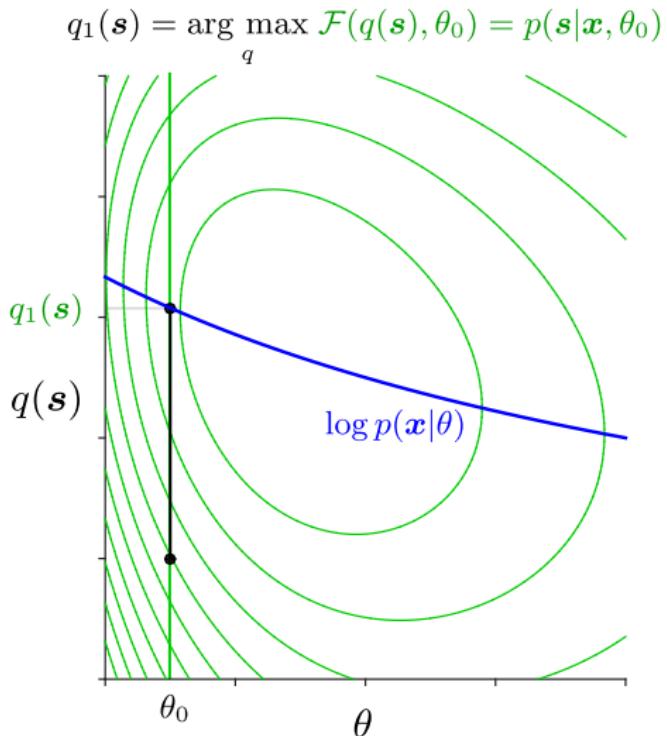
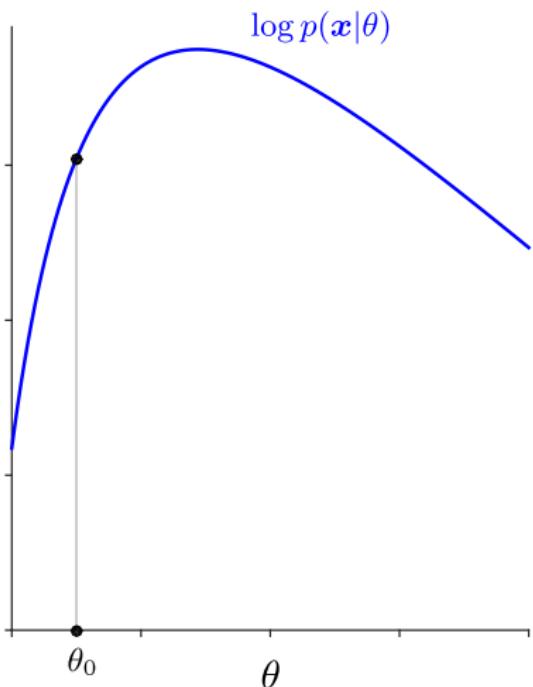
The Expectation Maximisation (EM) algorithm

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



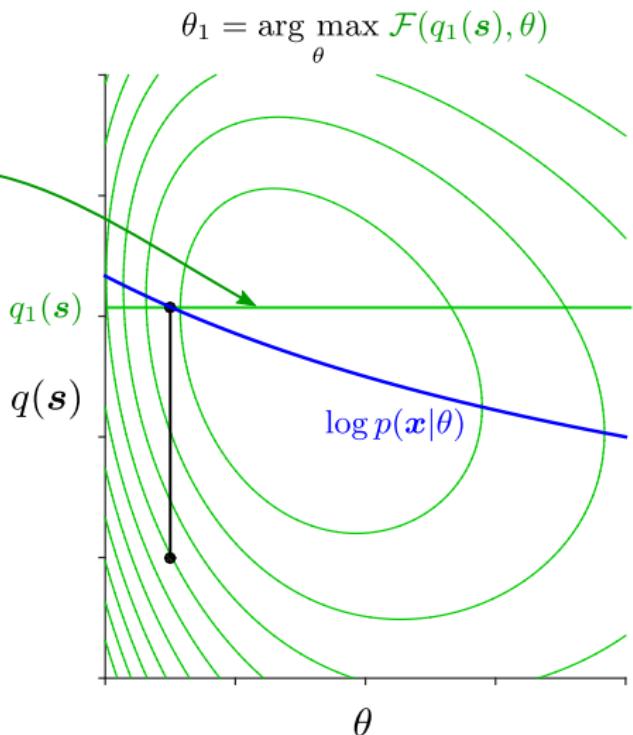
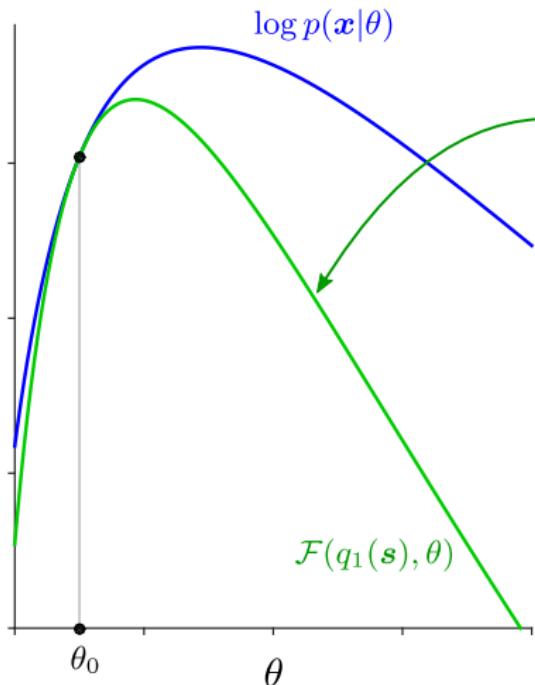
The Expectation Maximisation (EM) algorithm

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



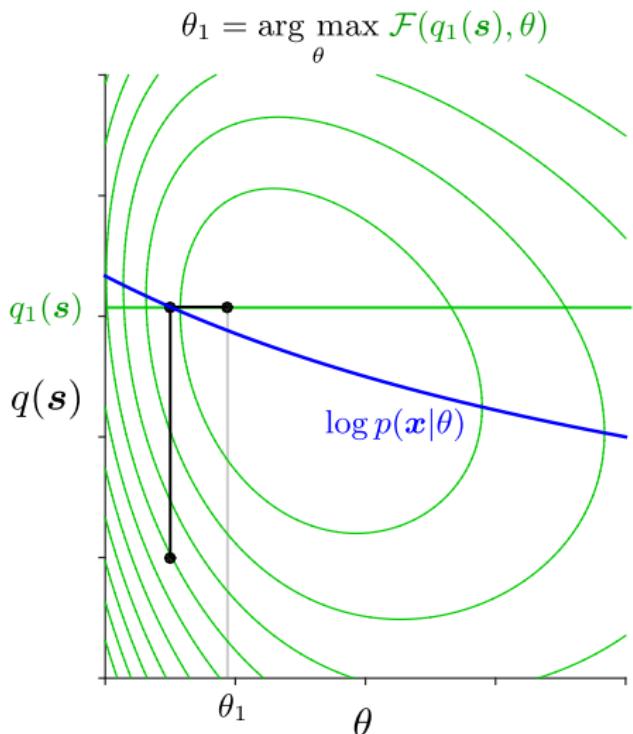
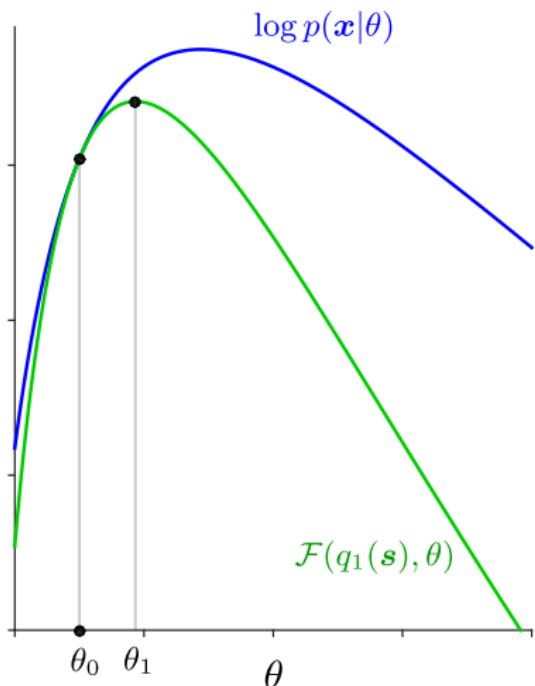
The Expectation Maximisation (EM) algorithm

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



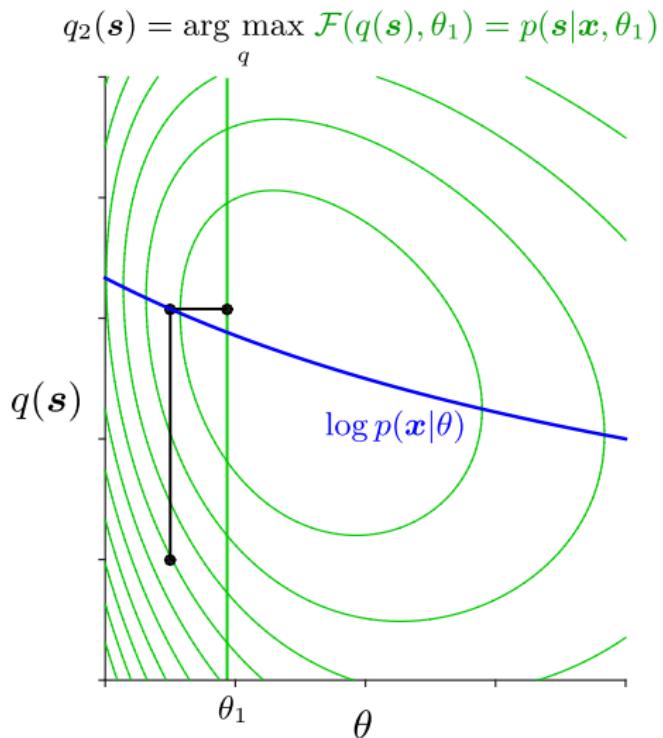
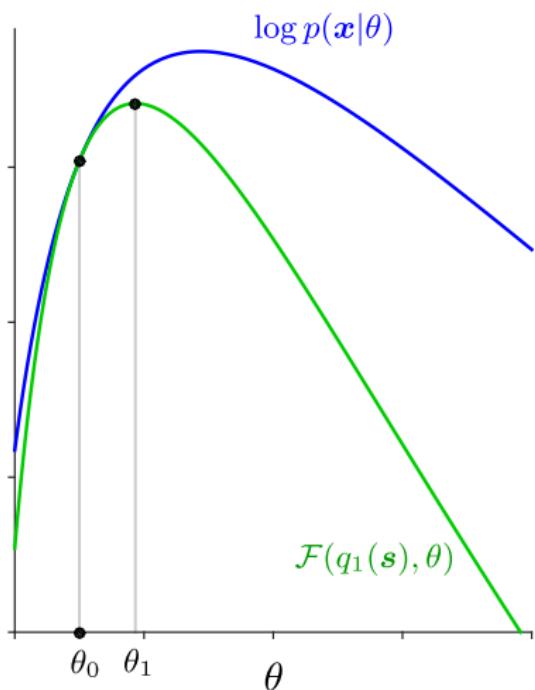
The Expectation Maximisation (EM) algorithm

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



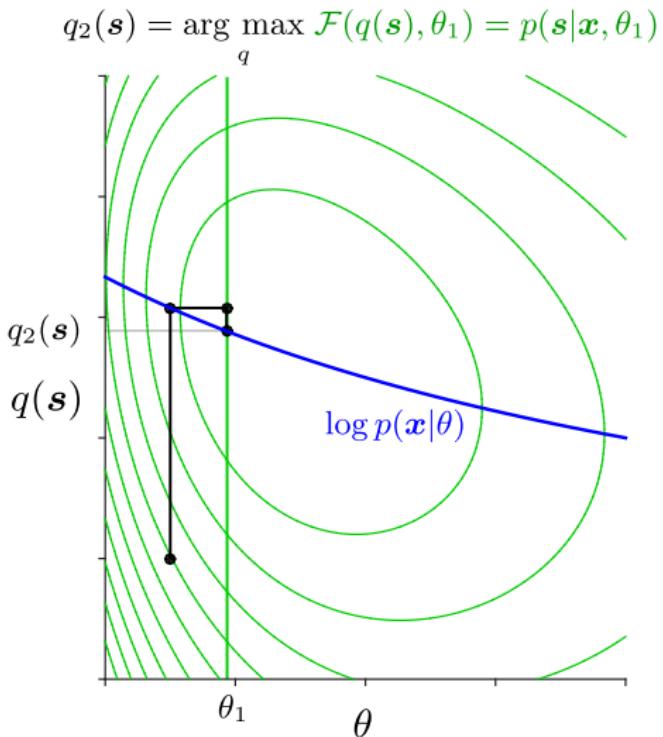
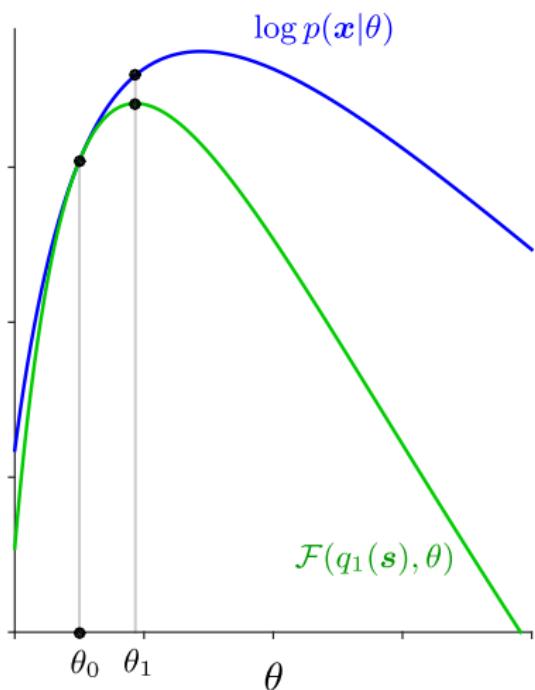
The Expectation Maximisation (EM) algorithm

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



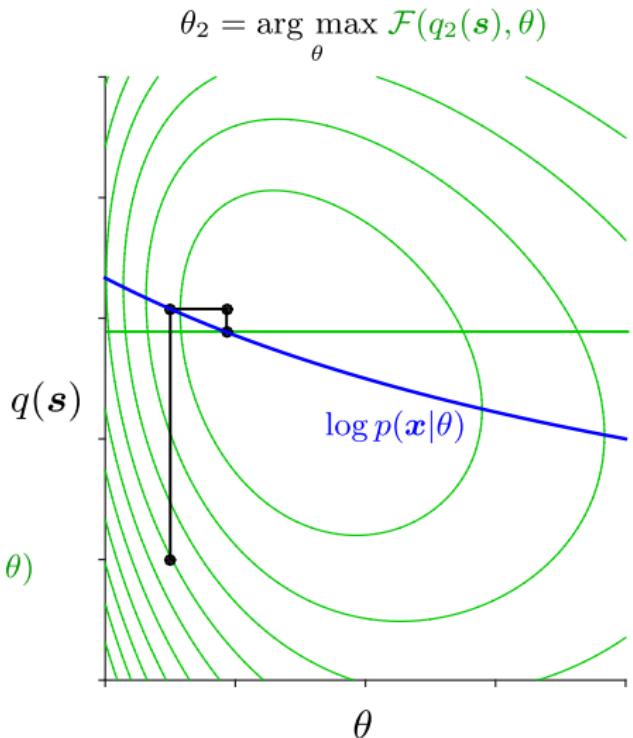
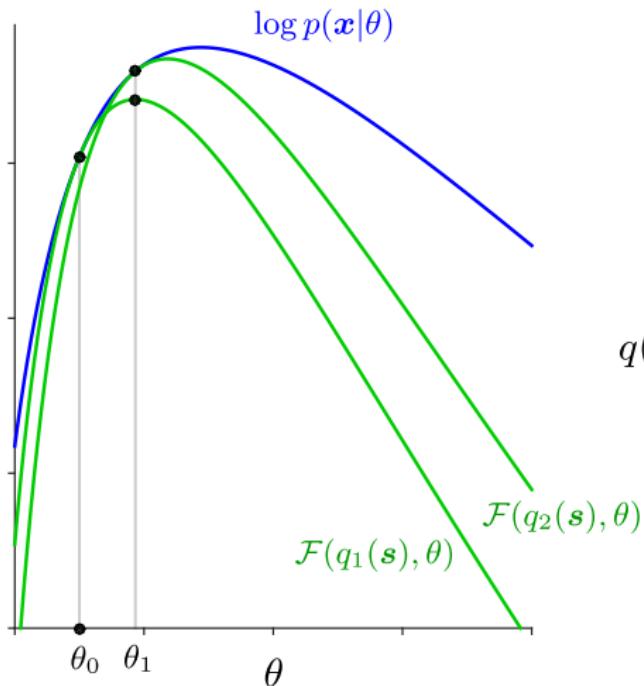
The Expectation Maximisation (EM) algorithm

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



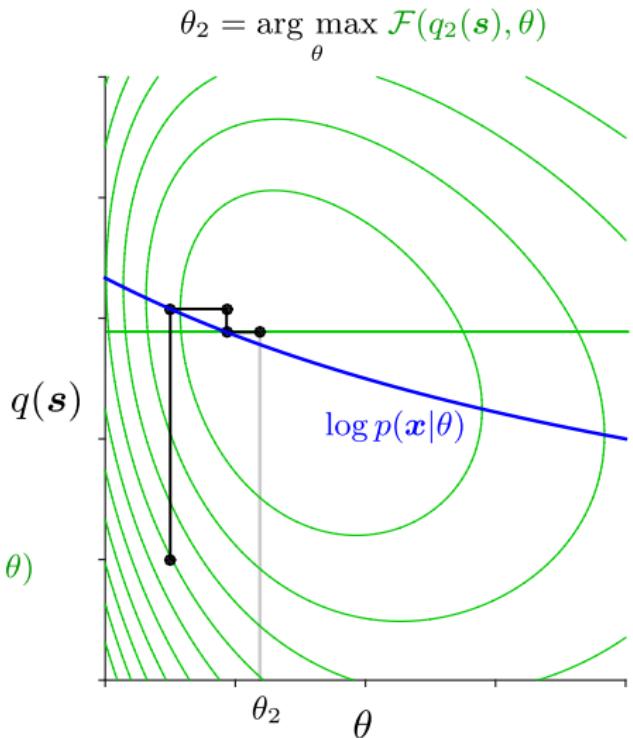
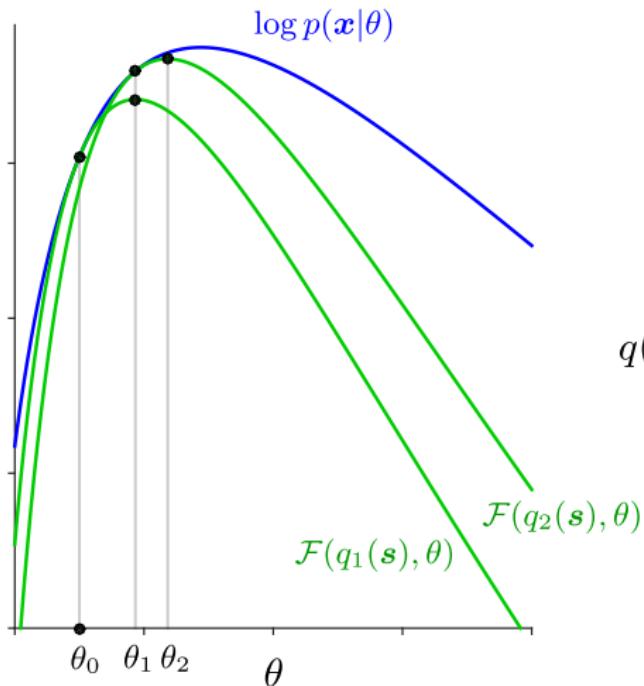
The Expectation Maximisation (EM) algorithm

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



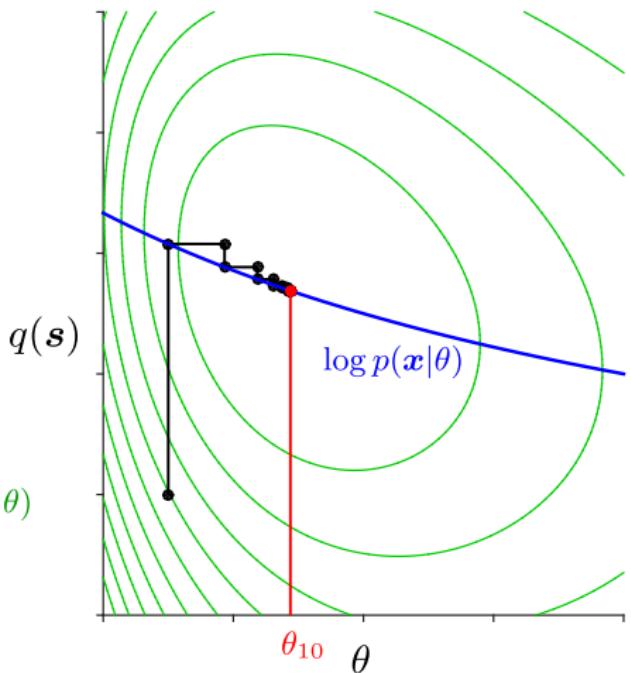
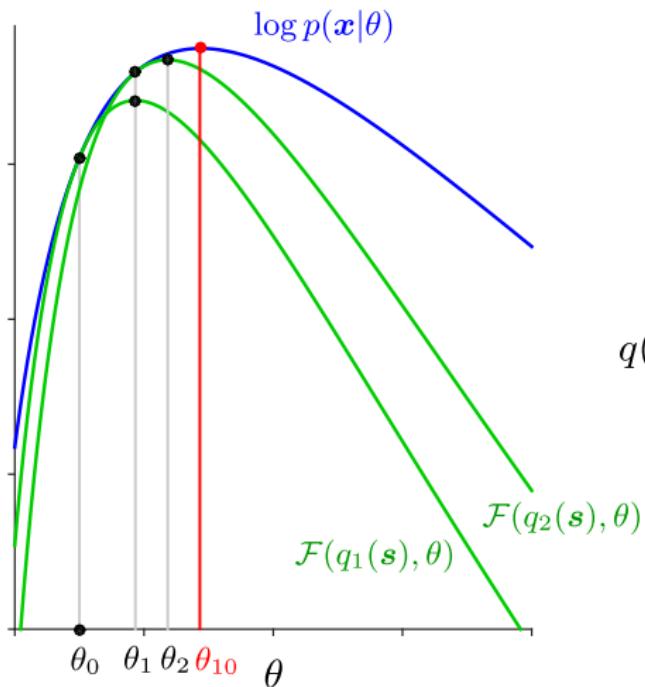
The Expectation Maximisation (EM) algorithm

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



The Expectation Maximisation (EM) algorithm

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



The Expectation Maximisation (EM) algorithm

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

E step: for fixed θ_{t-1} , maximize lower bound $\mathcal{F}(q(s), \theta_{t-1})$ wrt $q(s)$.

As log likelihood $\log p(\mathbf{x}|\theta)$ is independent of $q(s)$ this is equivalent to minimizing $\mathcal{KL}(q(s)||p(s|\mathbf{x}, \theta_{t-1}))$, so $q_t(s) = p(s|\mathbf{x}, \theta_{t-1})$.

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

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

The Expectation Maximisation (EM) algorithm

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

E step: for fixed θ_{t-1} , maximize lower bound $\mathcal{F}(q(s), \theta_{t-1})$ wrt $q(s)$.

As log likelihood $\log p(\mathbf{x}|\theta)$ is independent of $q(s)$ this is equivalent to minimizing $\mathcal{KL}(q(s)||p(s|\mathbf{x}, \theta_{t-1}))$, so $q_t(s) = p(s|\mathbf{x}, \theta_{t-1})$.

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

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

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

$$\theta_t = \operatorname{argmax}_{\theta} \sum_s q_t(s) \log (p(\mathbf{x}|s, \theta)p(s|\theta)).$$

The Expectation Maximisation (EM) algorithm

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

E step: for fixed θ_{t-1} , maximize lower bound $\mathcal{F}(q(s), \theta_{t-1})$ wrt $q(s)$.

As log likelihood $\log p(\mathbf{x}|\theta)$ is independent of $q(s)$ this is equivalent to minimizing $\mathcal{KL}(q(s)||p(s|\mathbf{x}, \theta_{t-1}))$, so $q_t(s) = p(s|\mathbf{x}, \theta_{t-1})$.

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

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

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

$$\theta_t = \operatorname{argmax}_{\theta} \sum_s q_t(s) \log(p(\mathbf{x}|s, \theta)p(s|\theta)).$$

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

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

Application of EM to Mixture of Gaussians (E Step)

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

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

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

Application of EM to Mixture of Gaussians (E Step)

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

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

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

so the E step becomes:

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

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

Application of EM to Mixture of Gaussians (E Step)

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

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

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

so the E step becomes:

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

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

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

Application of EM to Mixture of Gaussians (M Step)

The lower bound is

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

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

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

Application of EM to Mixture of Gaussians (M Step)

The lower bound is

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

The M step, optimizing $\mathcal{F}(q(s), \theta)$ 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

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

The M step, optimizing $\mathcal{F}(q(s), \theta)$ 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

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

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

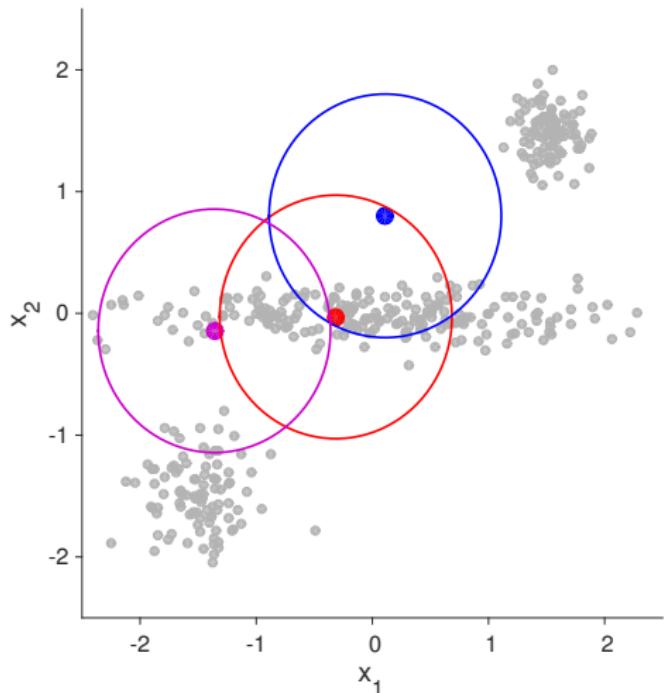
$$\frac{\partial \mathcal{F}}{\partial \mu_j} = \sum_{n=1}^N q(s_n = j) \frac{x_n - \mu_j}{\sigma_j^2} = 0 \Rightarrow \mu_j = \frac{\sum_{n=1}^N q(s_n = j) x_n}{\sum_{n=1}^N q(s_n = j)},$$

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

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

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

repeat

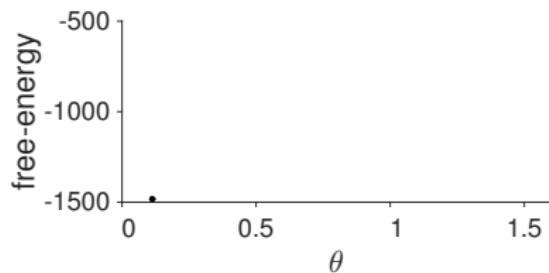
E-Step:

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

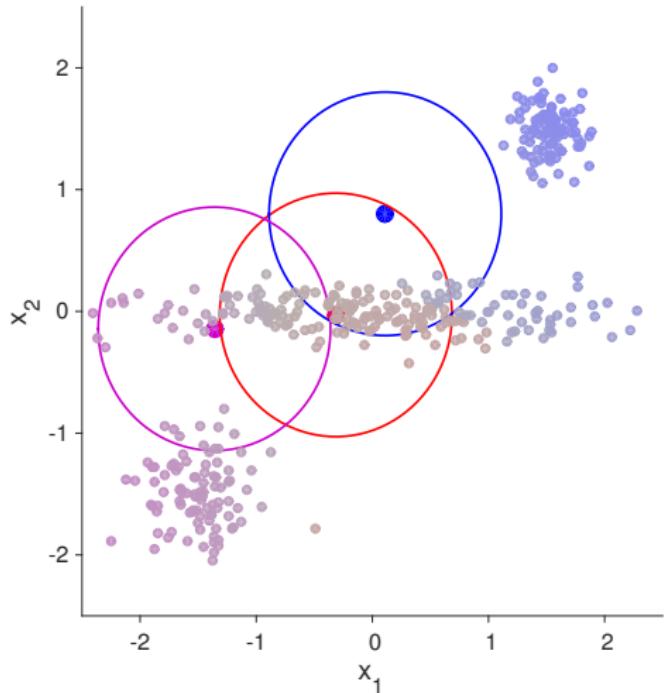
M-Step:

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

until convergence



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



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

repeat

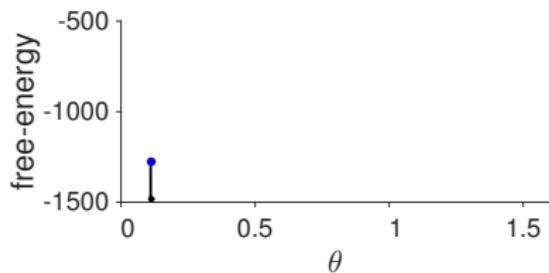
E-Step:

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

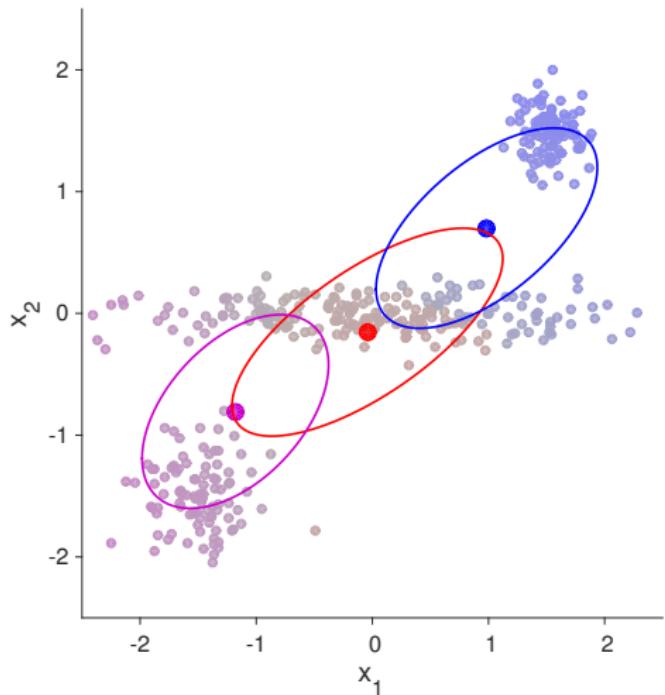
M-Step:

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

until convergence



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



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

repeat

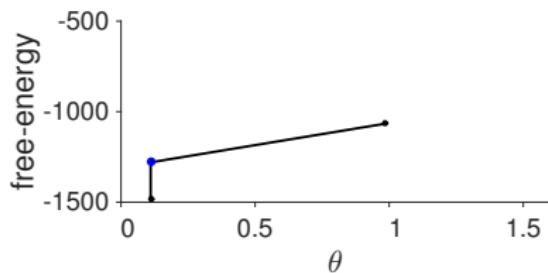
E-Step:

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

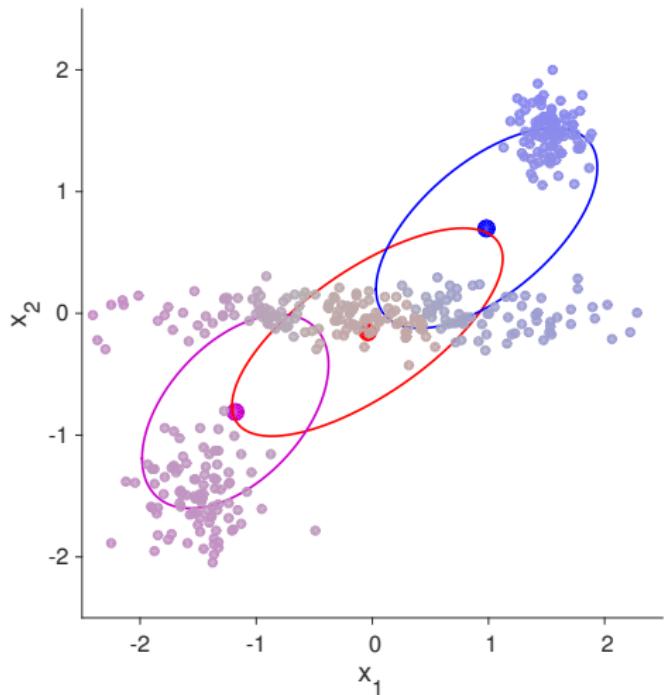
M-Step:

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

until convergence



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



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

repeat

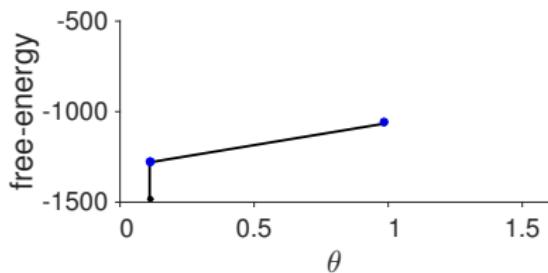
E-Step:

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

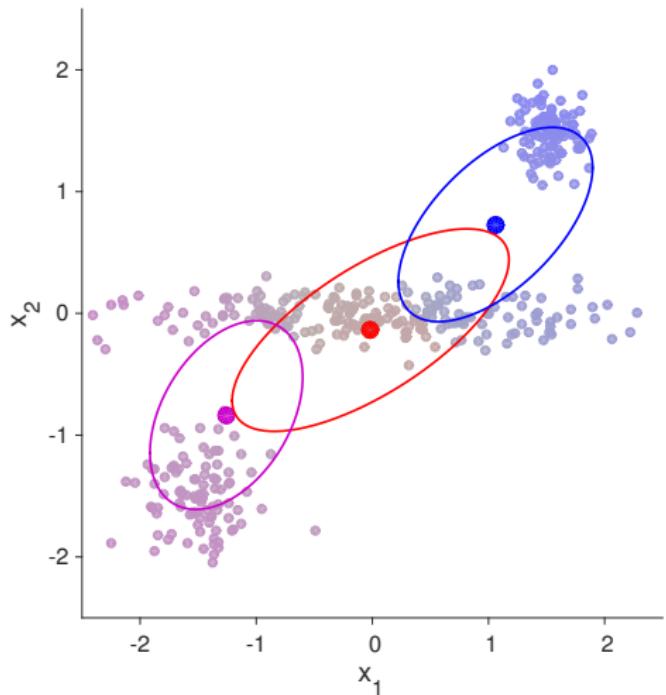
M-Step:

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

until convergence



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



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

repeat

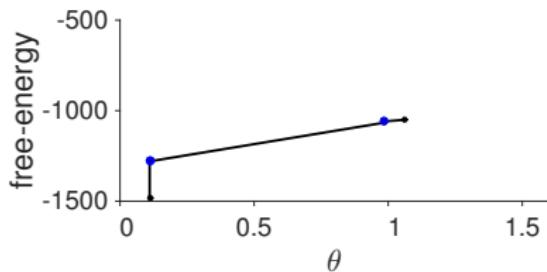
E-Step:

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

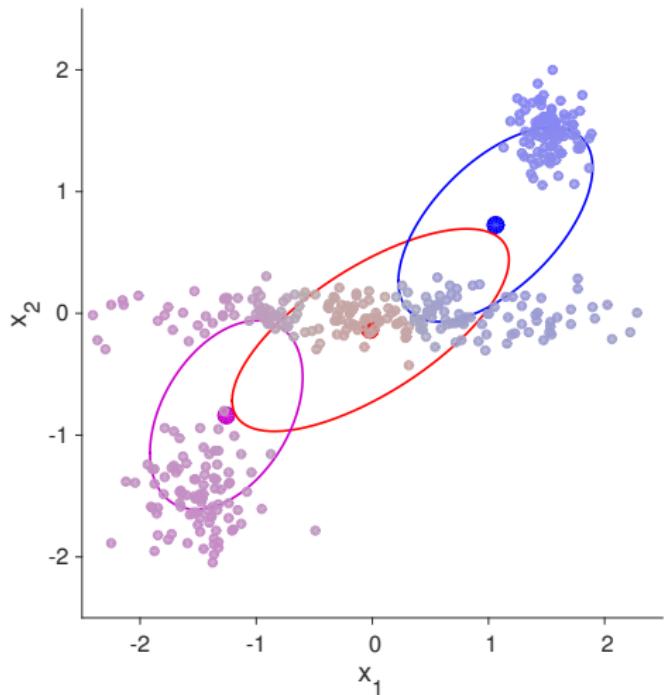
M-Step:

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

until convergence



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



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

repeat

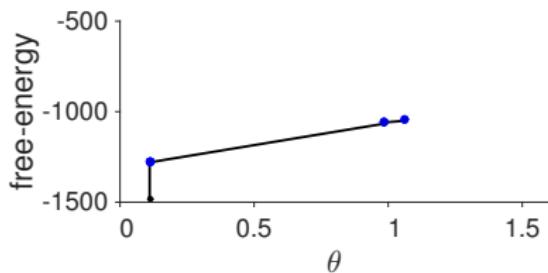
E-Step:

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

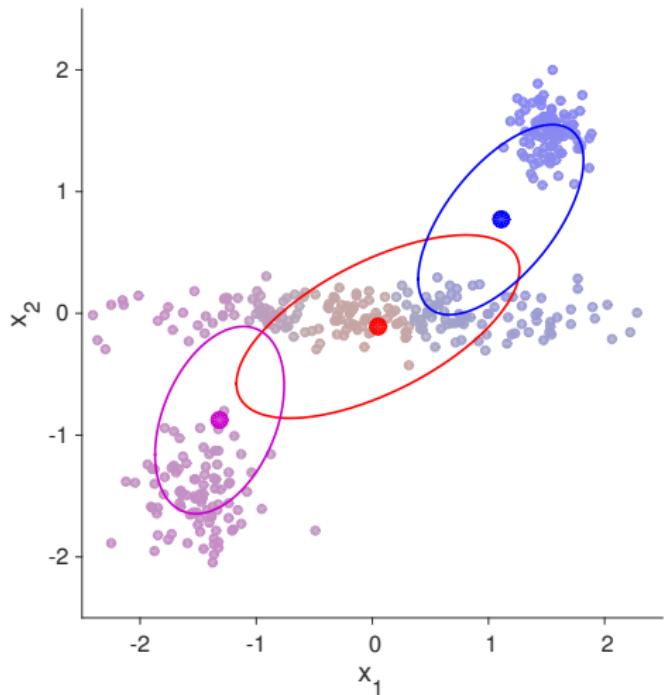
M-Step:

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

until convergence



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



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

repeat

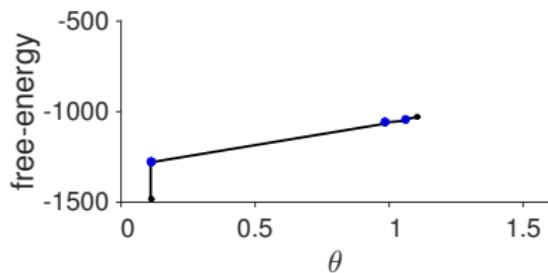
E-Step:

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

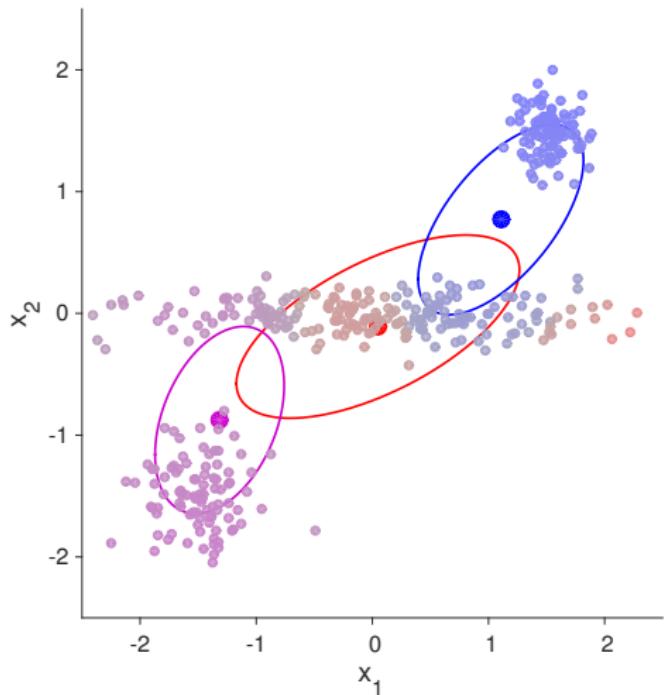
M-Step:

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

until convergence



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



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

repeat

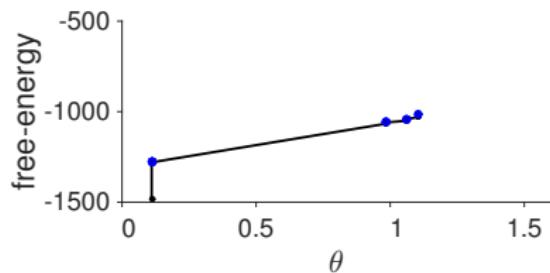
E-Step:

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

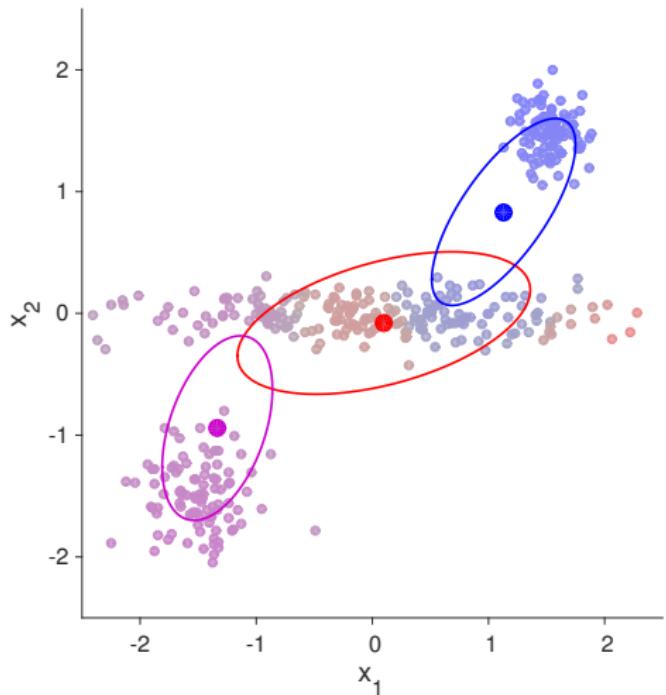
M-Step:

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

until convergence



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



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

repeat

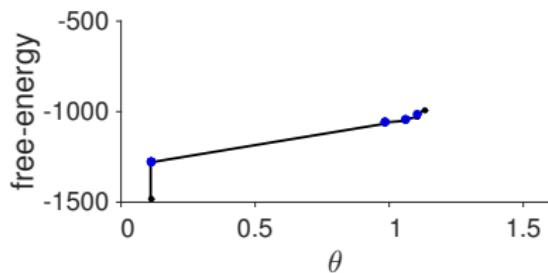
E-Step:

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

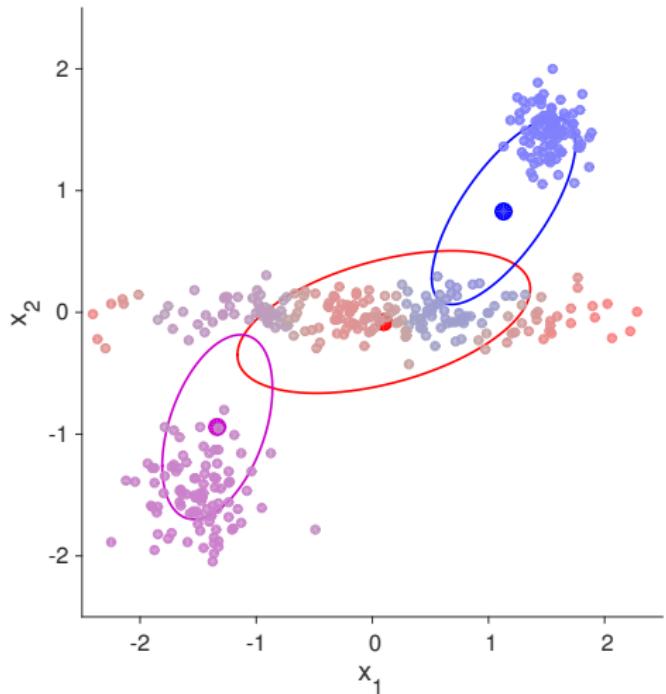
M-Step:

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

until convergence



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



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

repeat

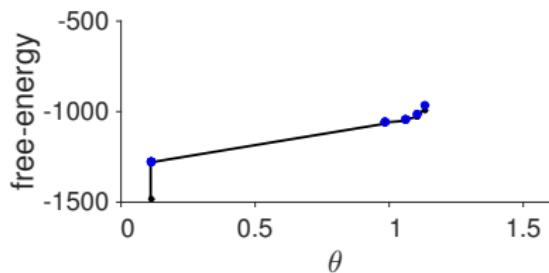
E-Step:

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

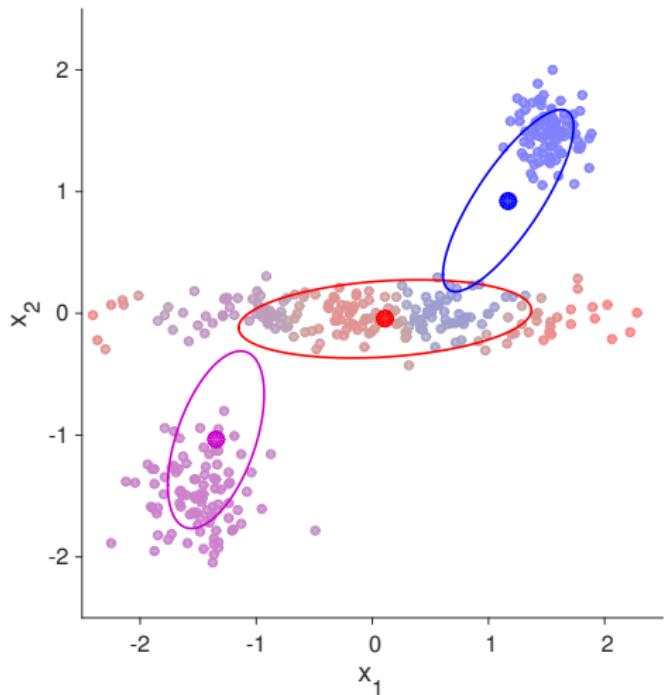
M-Step:

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

until convergence



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



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

repeat

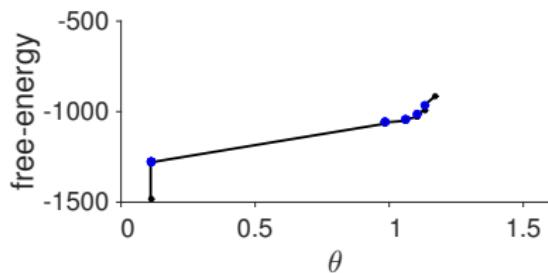
E-Step:

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

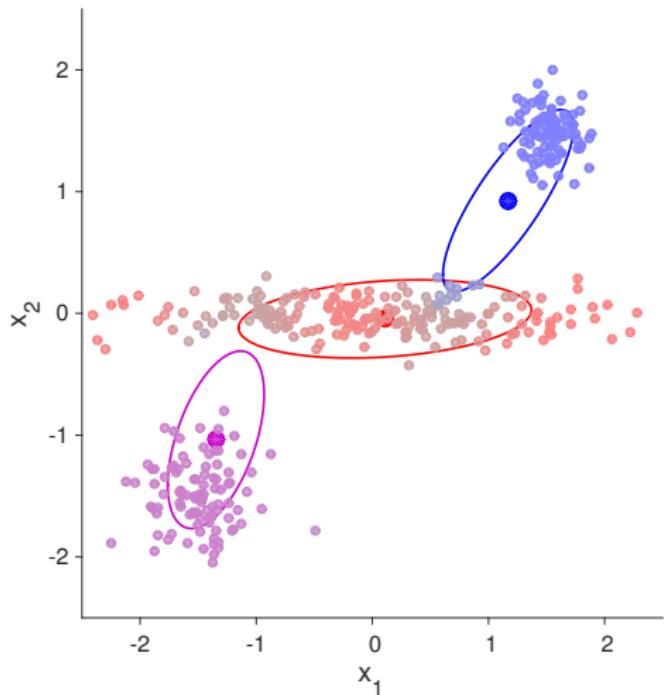
M-Step:

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

until convergence



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



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

repeat

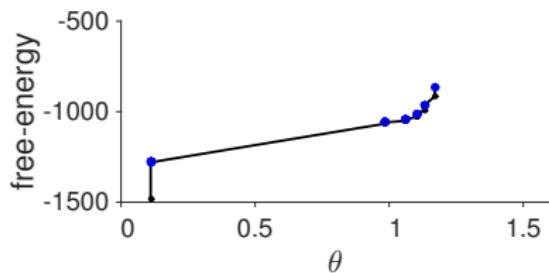
E-Step:

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

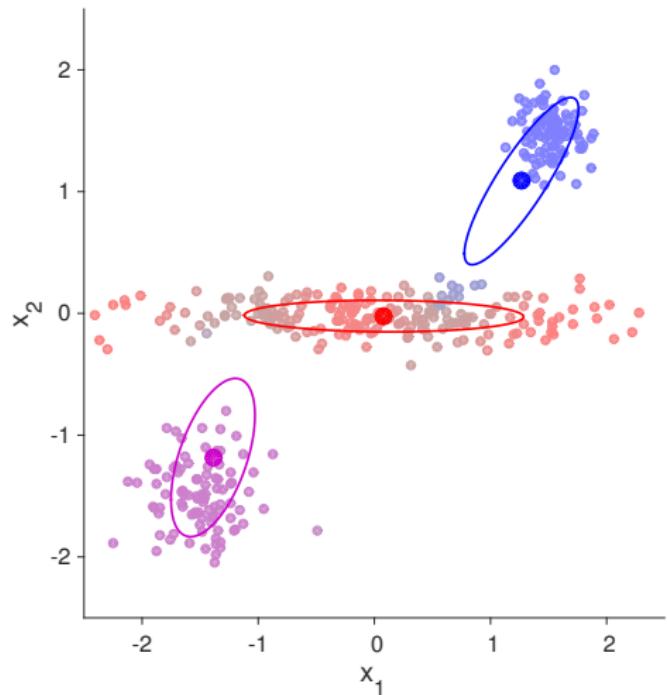
M-Step:

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

until convergence



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



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

repeat

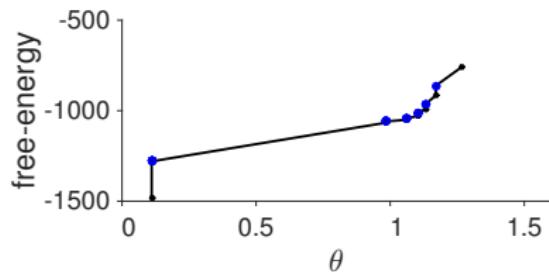
E-Step:

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

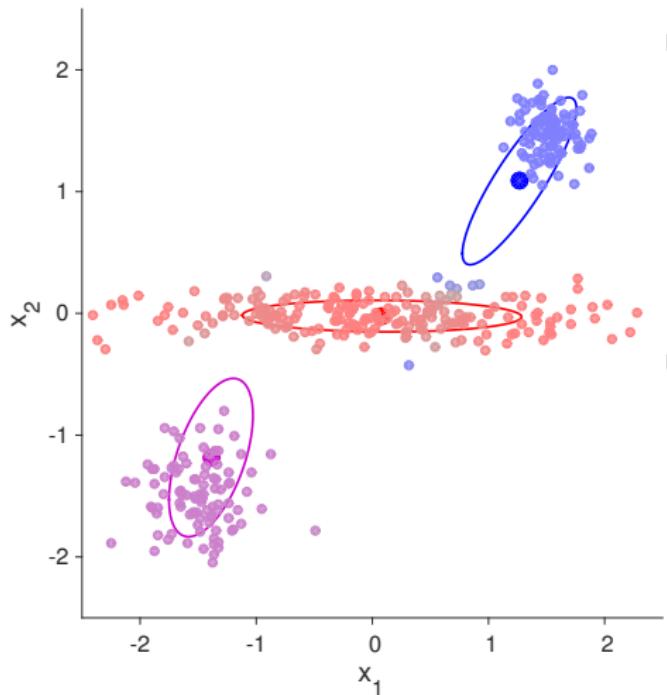
M-Step:

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

until convergence



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



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

repeat

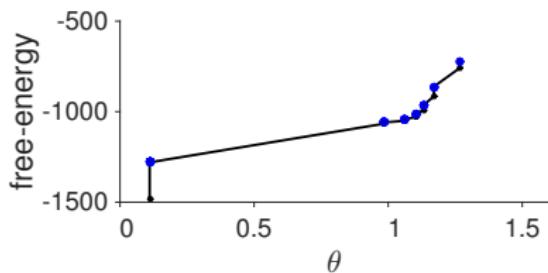
E-Step:

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

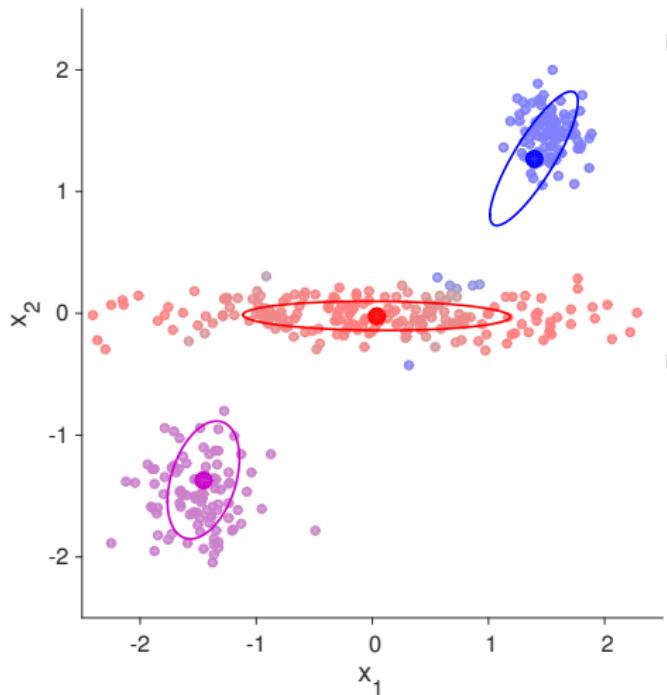
M-Step:

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

until convergence



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



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

repeat

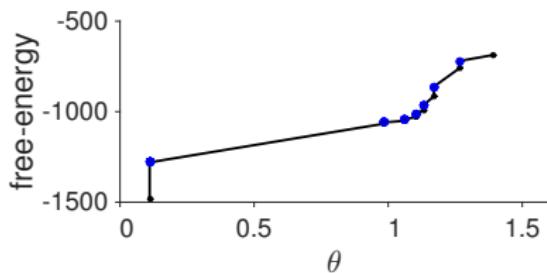
E-Step:

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

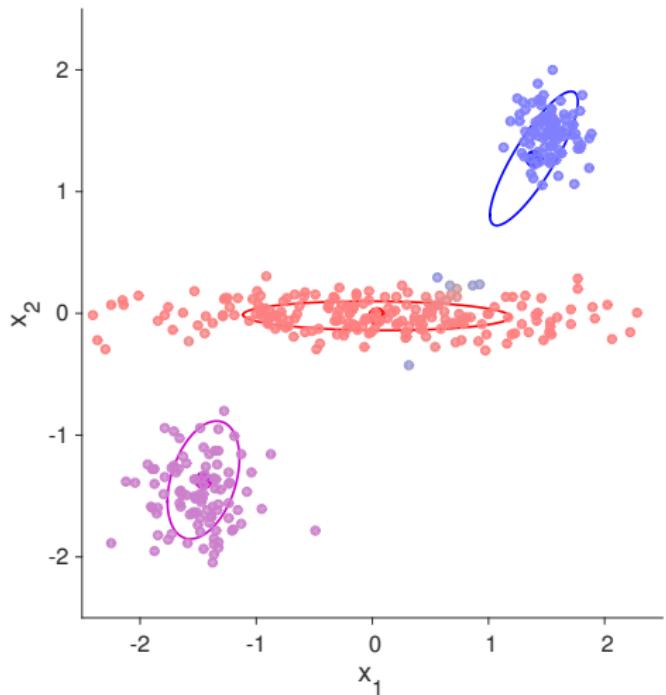
M-Step:

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

until convergence



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



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

repeat

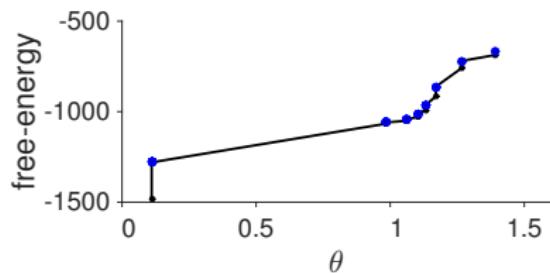
E-Step:

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

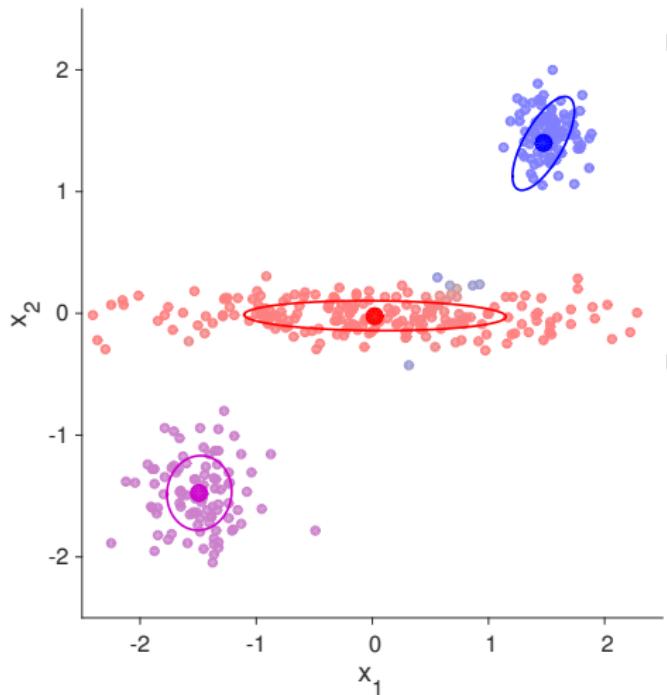
M-Step:

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

until convergence



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



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

repeat

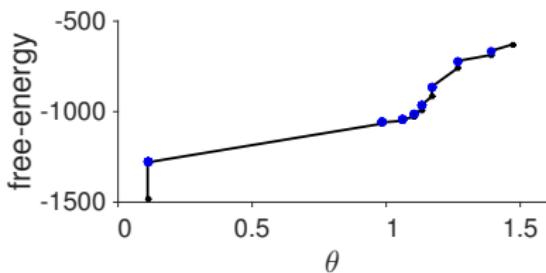
E-Step:

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

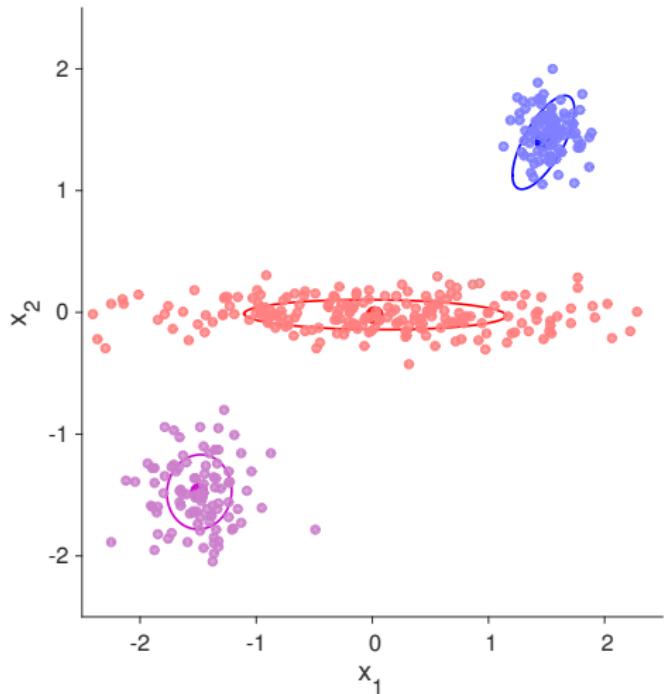
M-Step:

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

until convergence



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



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

repeat

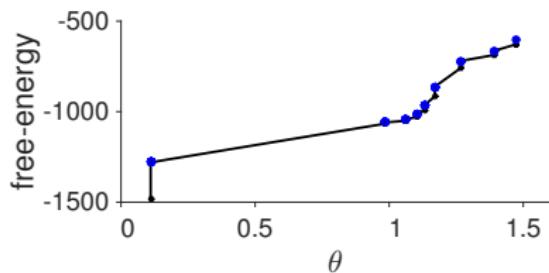
E-Step:

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

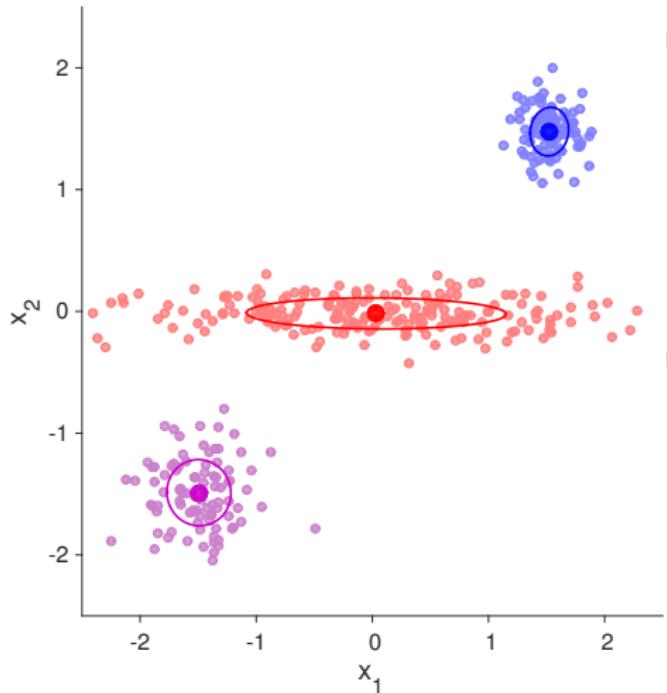
M-Step:

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

until convergence



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



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

repeat

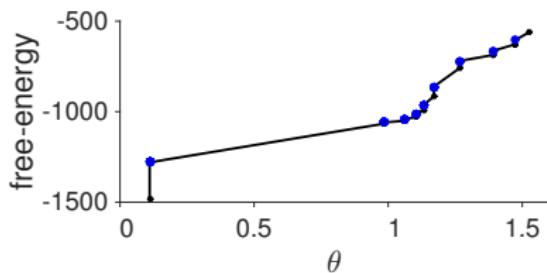
E-Step:

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

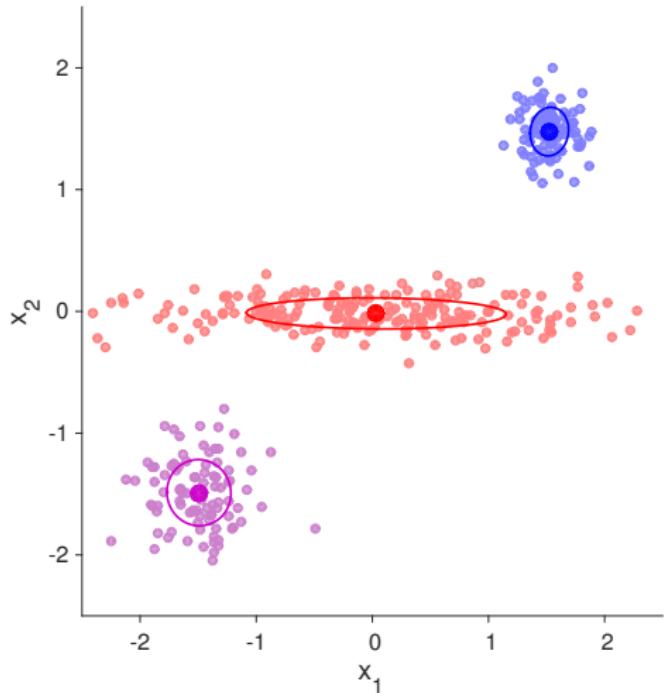
M-Step:

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

until convergence



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



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

repeat

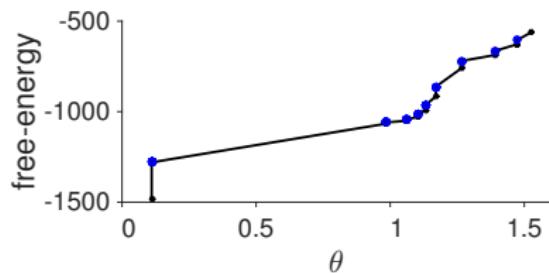
E-Step:

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

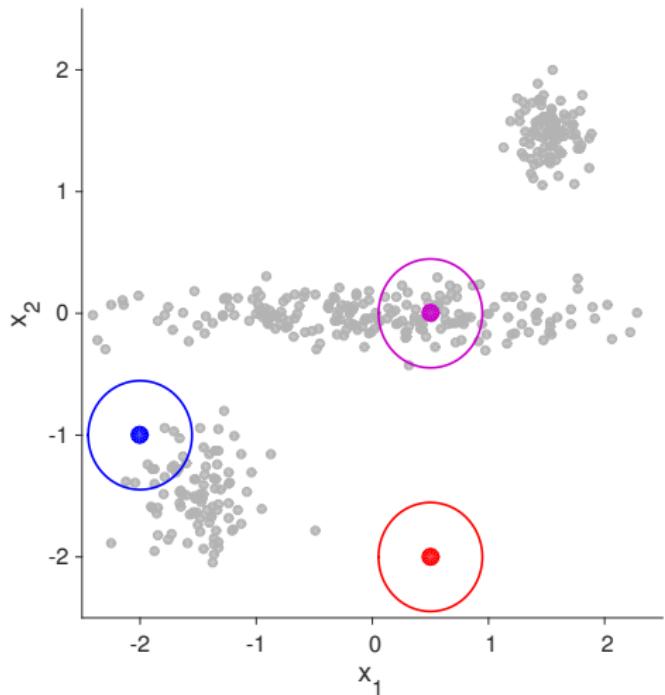
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

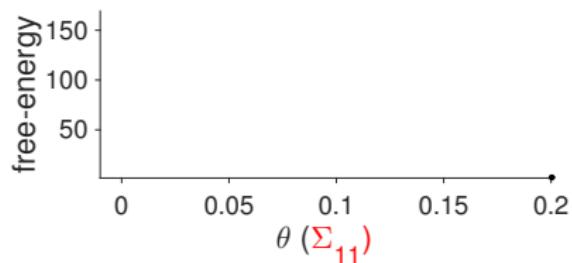
E-Step:

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

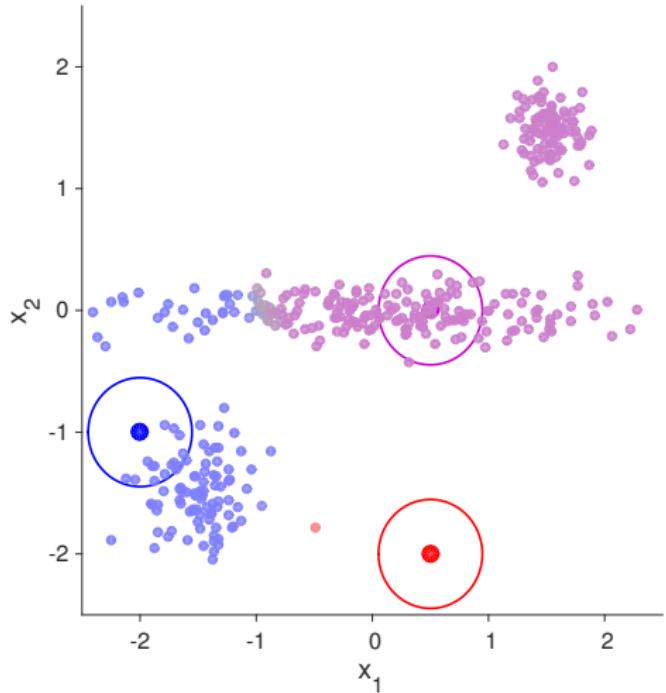
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

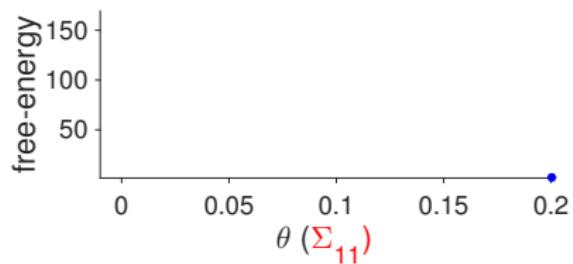
E-Step:

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

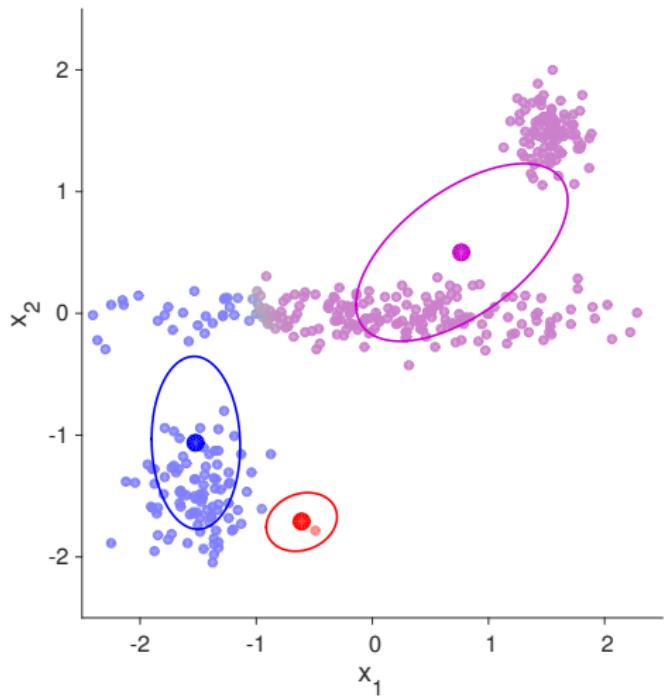
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

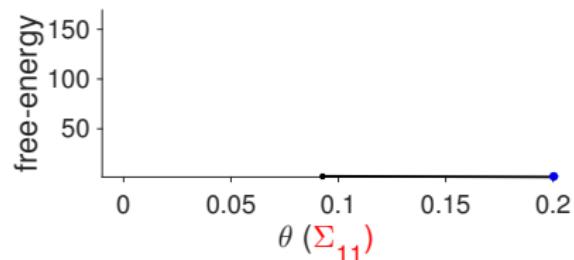
E-Step:

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

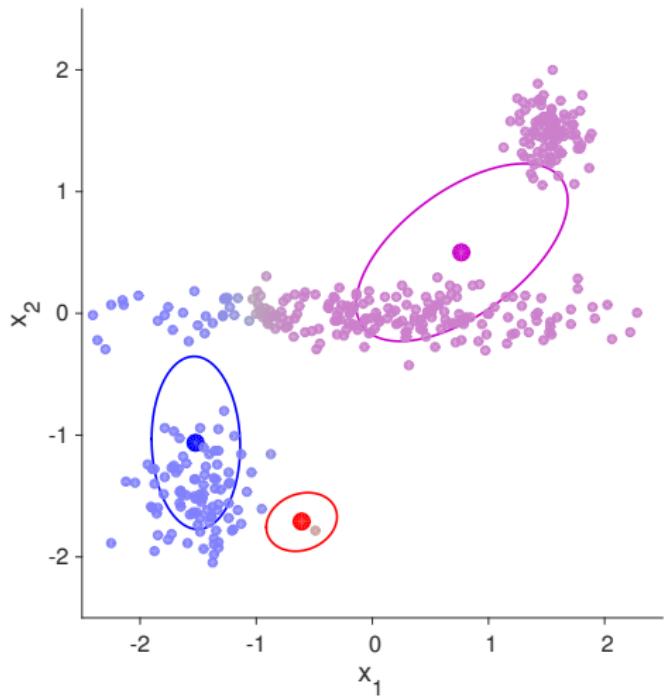
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

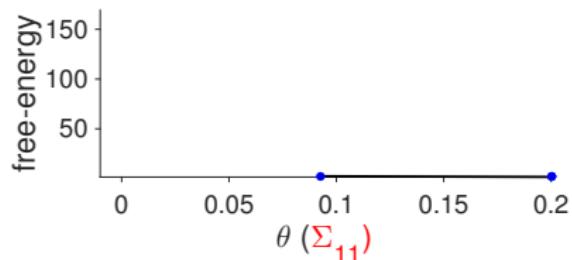
E-Step:

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

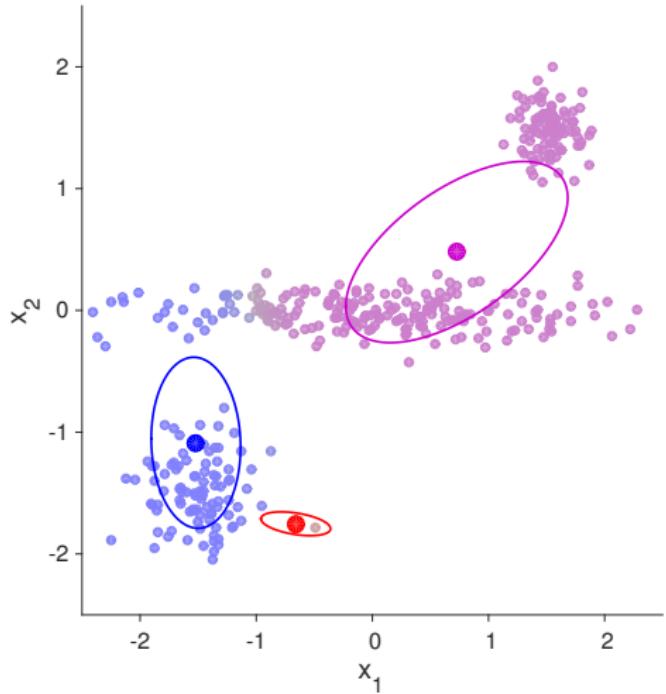
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

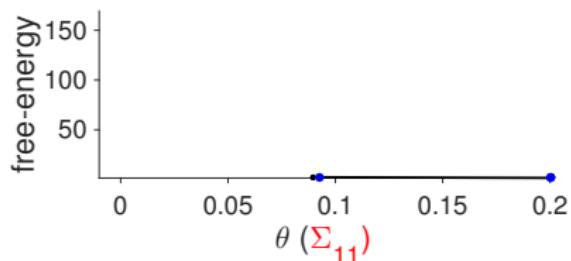
E-Step:

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

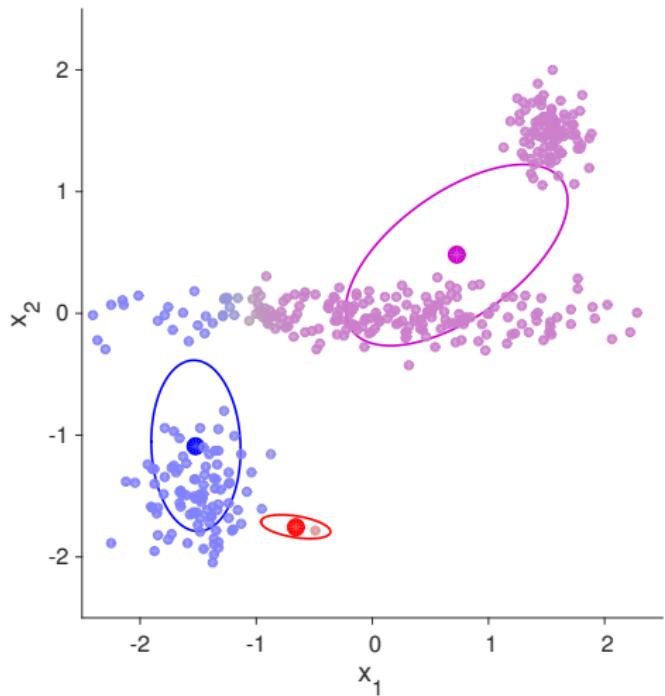
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

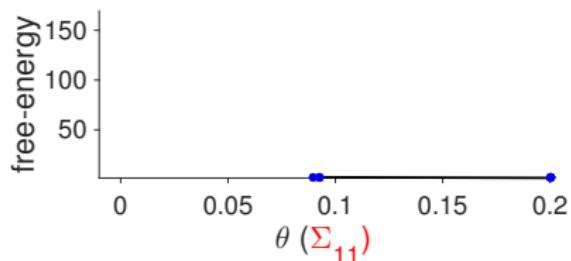
E-Step:

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

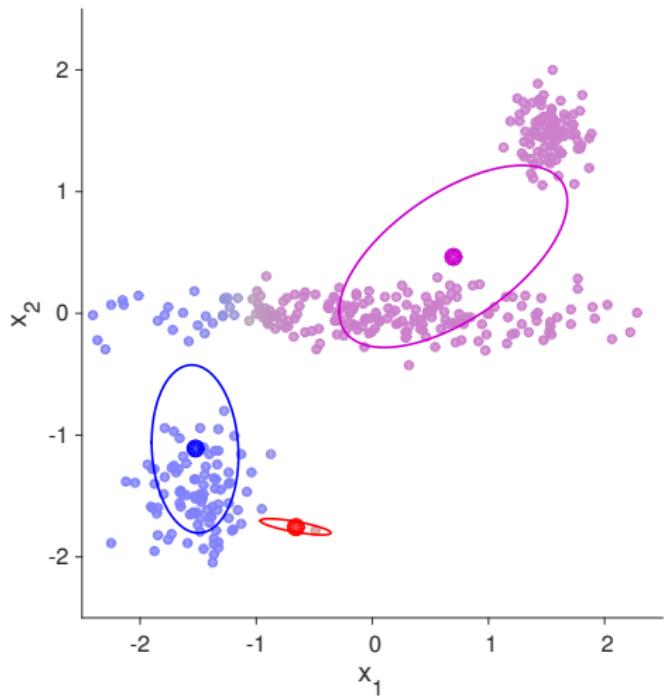
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

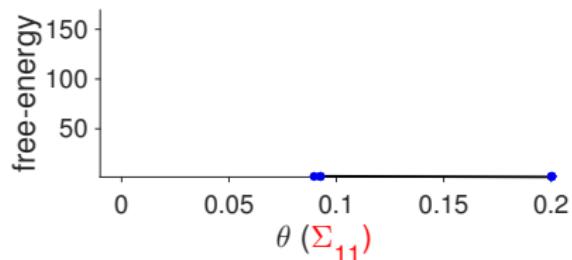
E-Step:

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

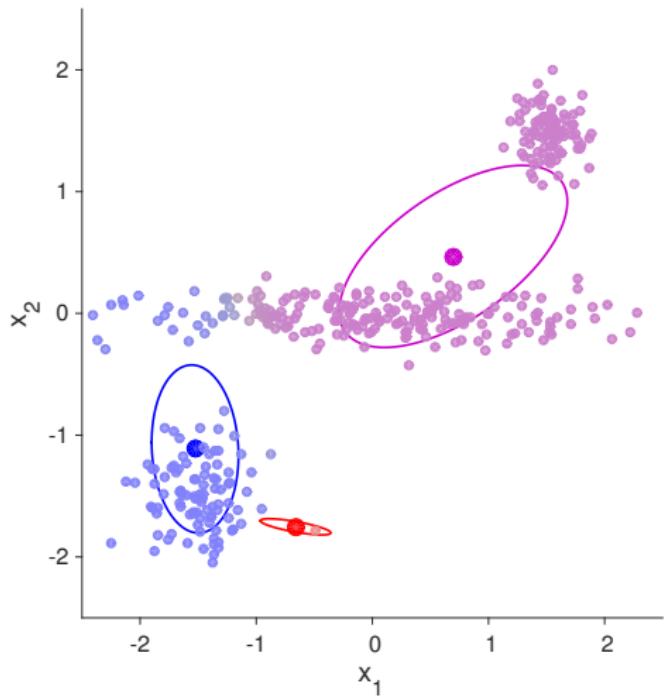
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

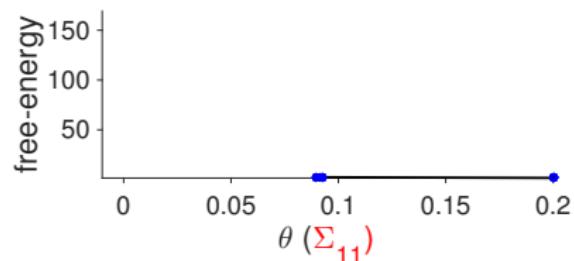
E-Step:

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

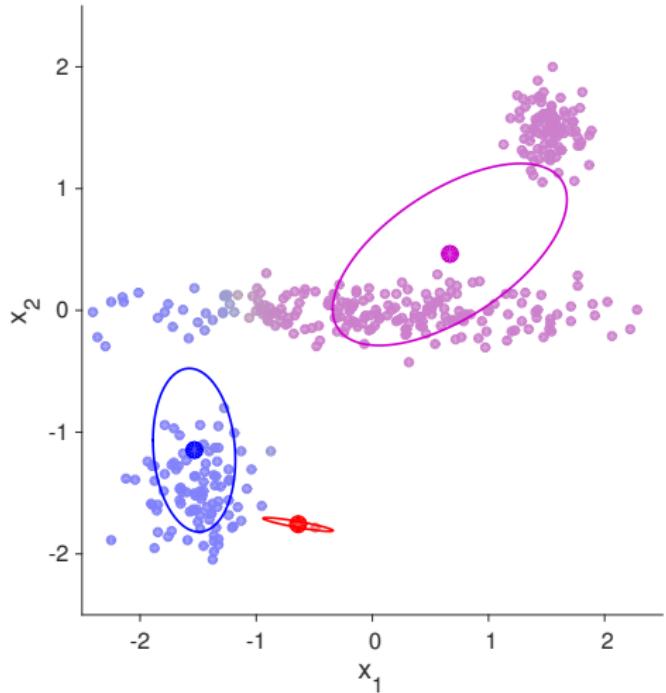
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

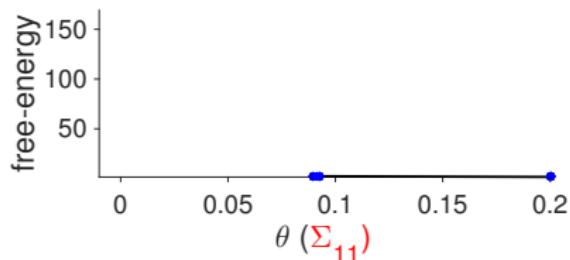
E-Step:

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

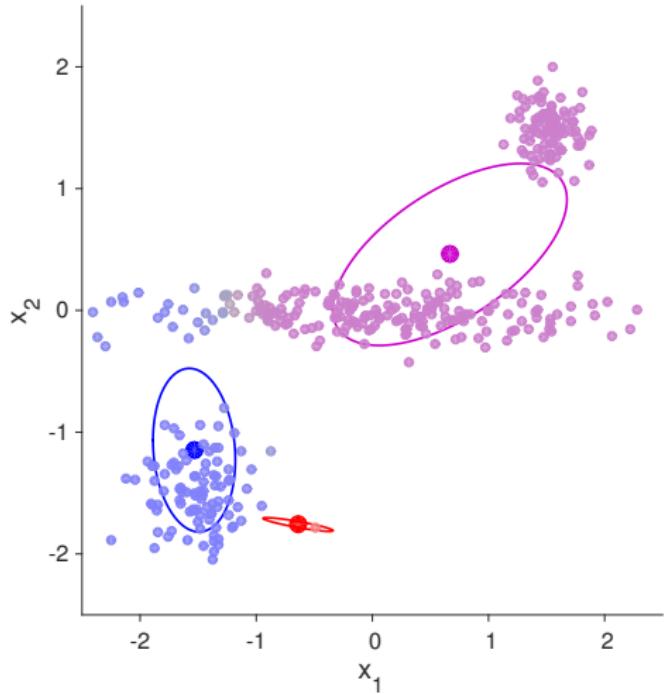
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

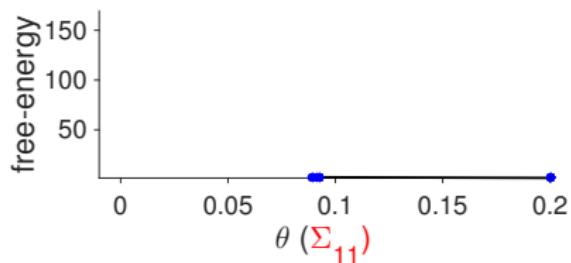
E-Step:

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

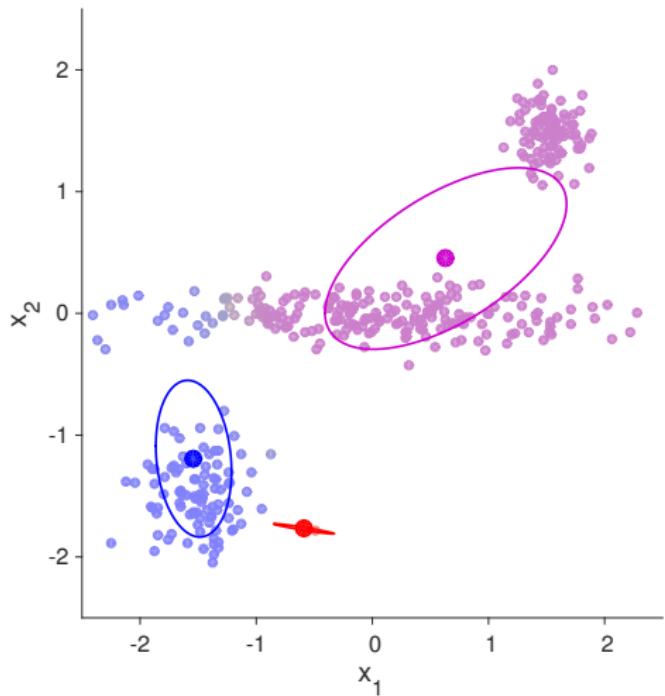
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

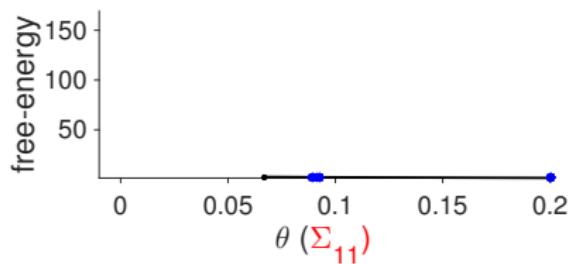
E-Step:

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

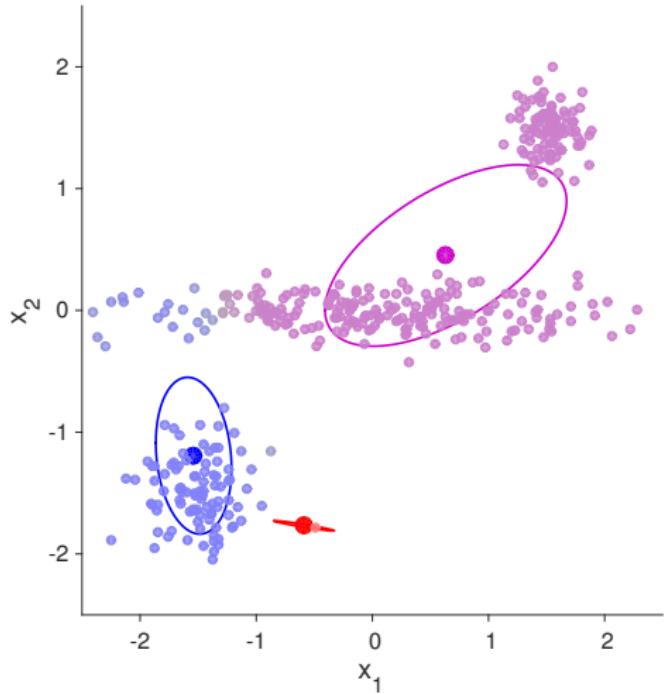
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

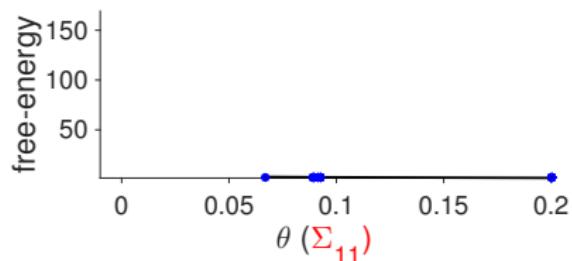
E-Step:

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

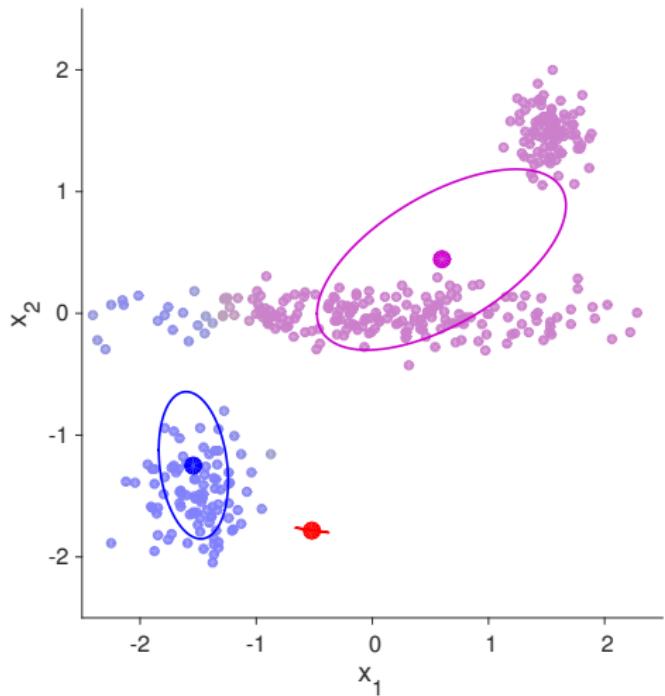
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

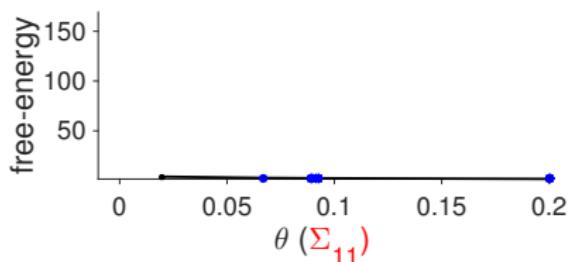
E-Step:

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

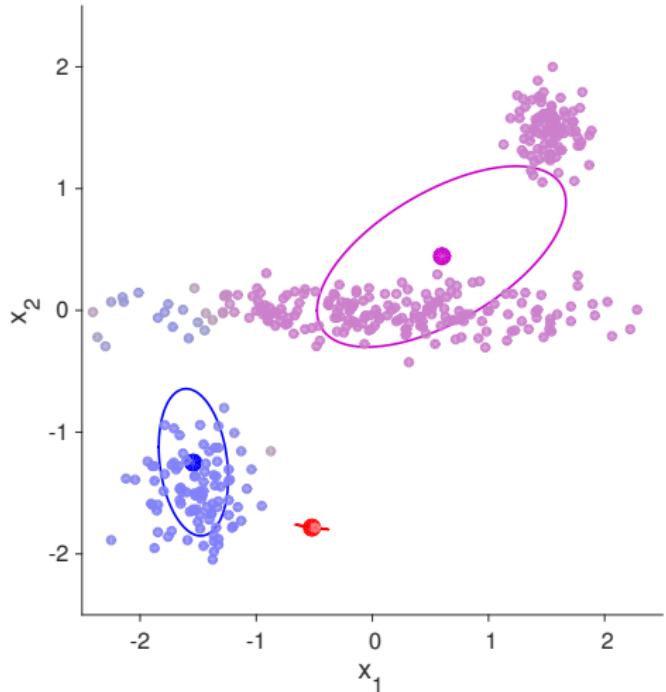
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

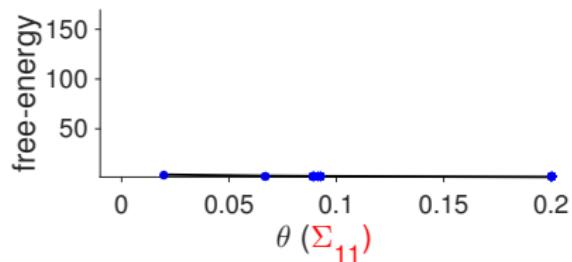
E-Step:

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

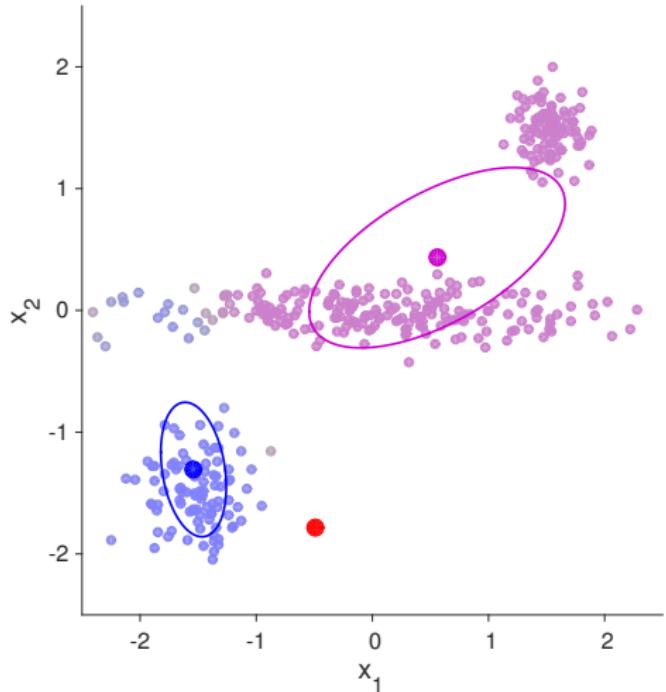
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

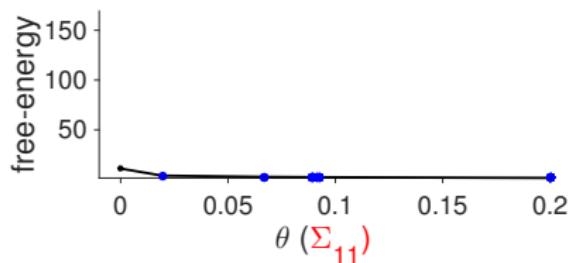
E-Step:

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

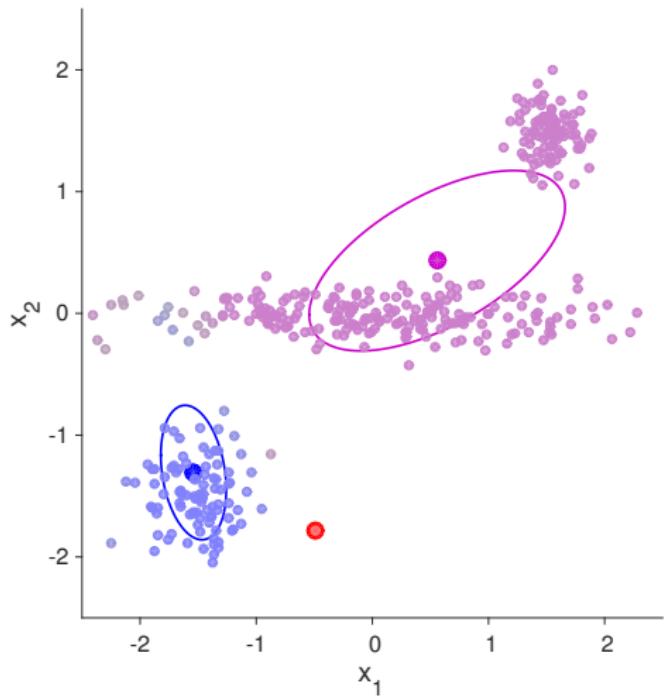
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

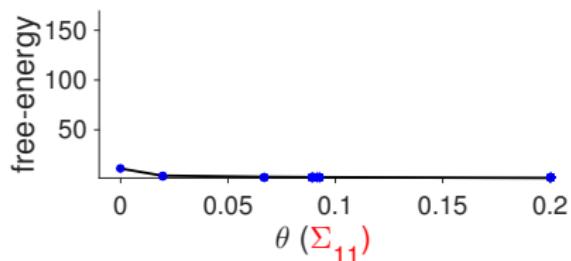
E-Step:

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

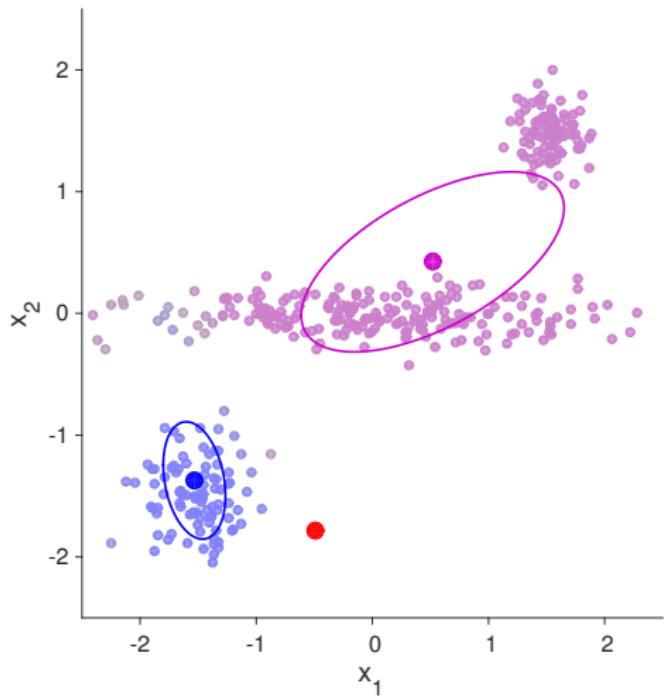
M-Step:

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

until convergence



What will happen with this initialisation?



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

repeat

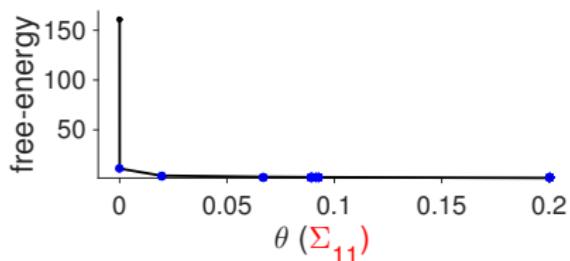
E-Step:

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

M-Step:

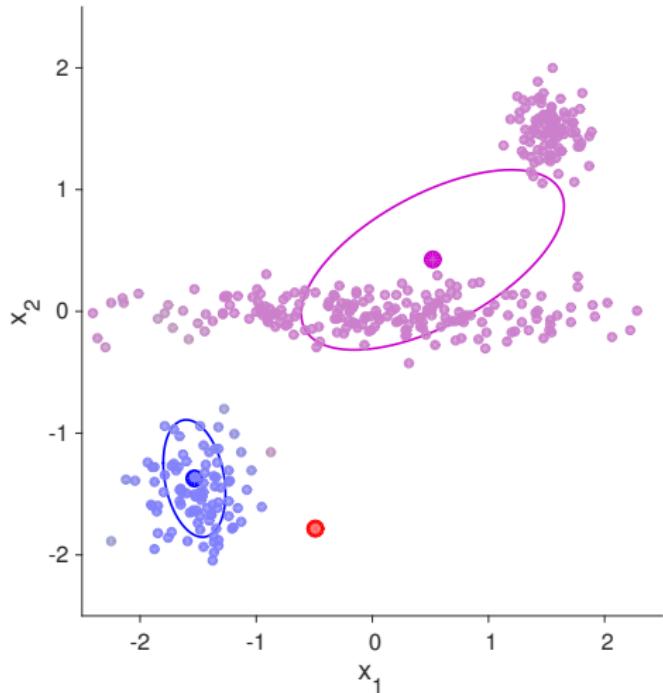
$$\arg \max_{\theta} \sum_{n,k} r_{nk} \log p(s_n = k, \mathbf{x} | \theta)$$

until convergence



What will happen with this initialisation?

KABOOM!



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

repeat

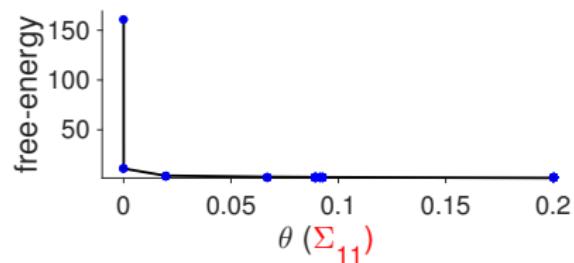
E-Step:

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

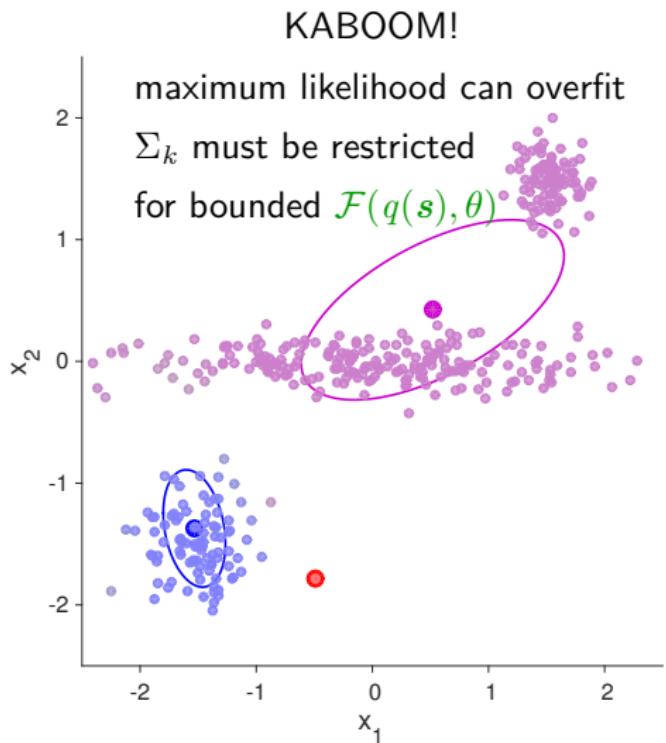
M-Step:

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

until convergence



What will happen with this initialisation?



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

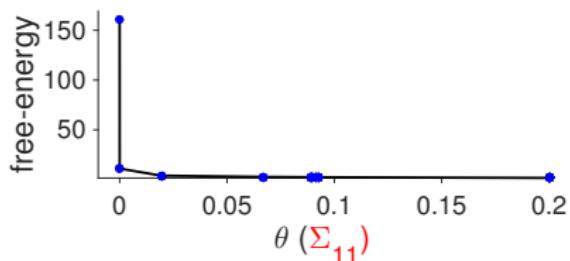
E-Step:

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

M-Step:

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

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 $p(s|x, \theta)$ entropy and average log-joint $\mathbb{E}_{q(s|\theta)} [\log p(s, x|\theta)]$

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 $p(\theta, s|x)$
- ▶ co-ordinate ascent is often slow to converge (lots of iterations required)
 - ▶ joint optimisation of $\mathcal{F}(q(s), \theta)$ faster
 - ▶ direct optimisation of log-likelihood $\log p(x|\theta)$

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:

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

Find stationary point by setting the derivative to zero:

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

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