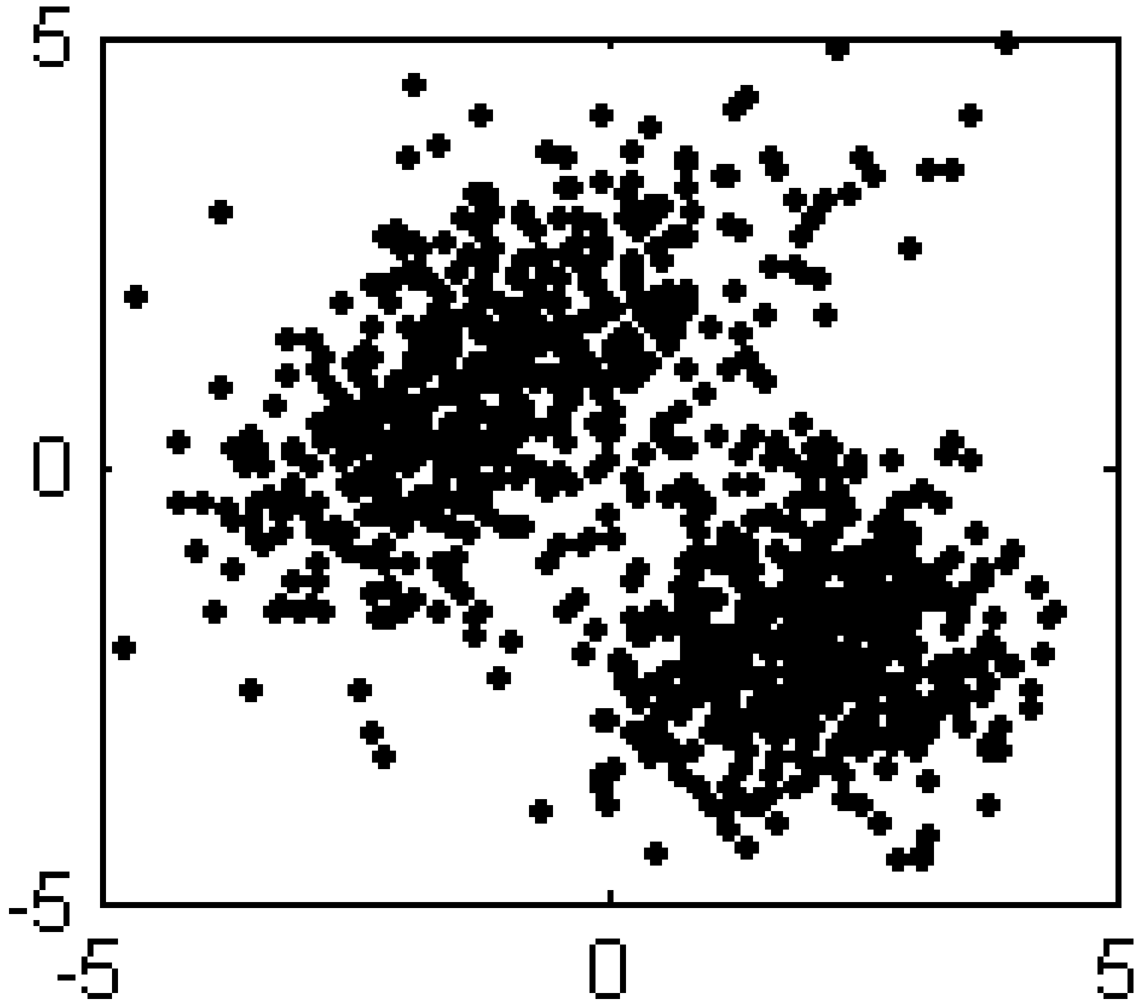
# Machine Learning

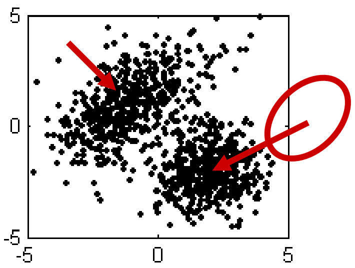
4771

Instructor: Itsik Pe’er

# Reminder: Mixture of Gaussians



# Kmeans: Notation Reminder




$\vec{z} \in \boldsymbol{B}^K, \sum_{i=1}^{K} \vec{z}(i) = 1$ or

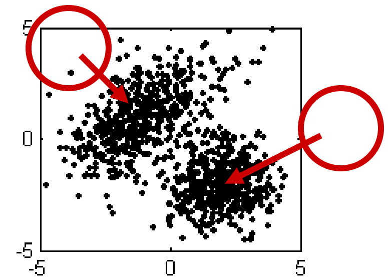
$\vec{z} \in \{\vec{\delta}_1, \dots, \vec{\delta}_K\}$ where $\vec{\delta}_i(i) = 1, \vec{\delta}_i(j) = 0 \; for \; i \neq j$

**mixing proportions (prior) =** $\boldsymbol{\pi} = p\left(\vec{z} = \vec{\delta}_i \middle| \pi\right)$

**mixture components =** $p\left(\vec{x} \middle| \vec{z} = \vec{\delta}_i, \theta\right)$

**posteriors (responsibilities) =** $\tau_{n,i} = p\left(\vec{z} = \vec{\delta}_i \middle| \vec{x}_n, \theta\right) = \frac{p\left(\vec{x}_n \middle| \vec{z} = \vec{\delta}_i, \theta\right) p\left(\vec{z} = \vec{\delta}_i \middle| \theta\right)}{p(\vec{x}_n | \theta)}$

# K-Means Clustering




- An old "heuristic" clustering algorithm
- Gobble up data with a divide & conquer scheme
- Assume each point $x$ has an discrete multinomial vector $z$
- Chicken and Egg problem:

  If know classes, we can get model (max likelihood!)

  If know the model, we can predict the classes (classifier!)

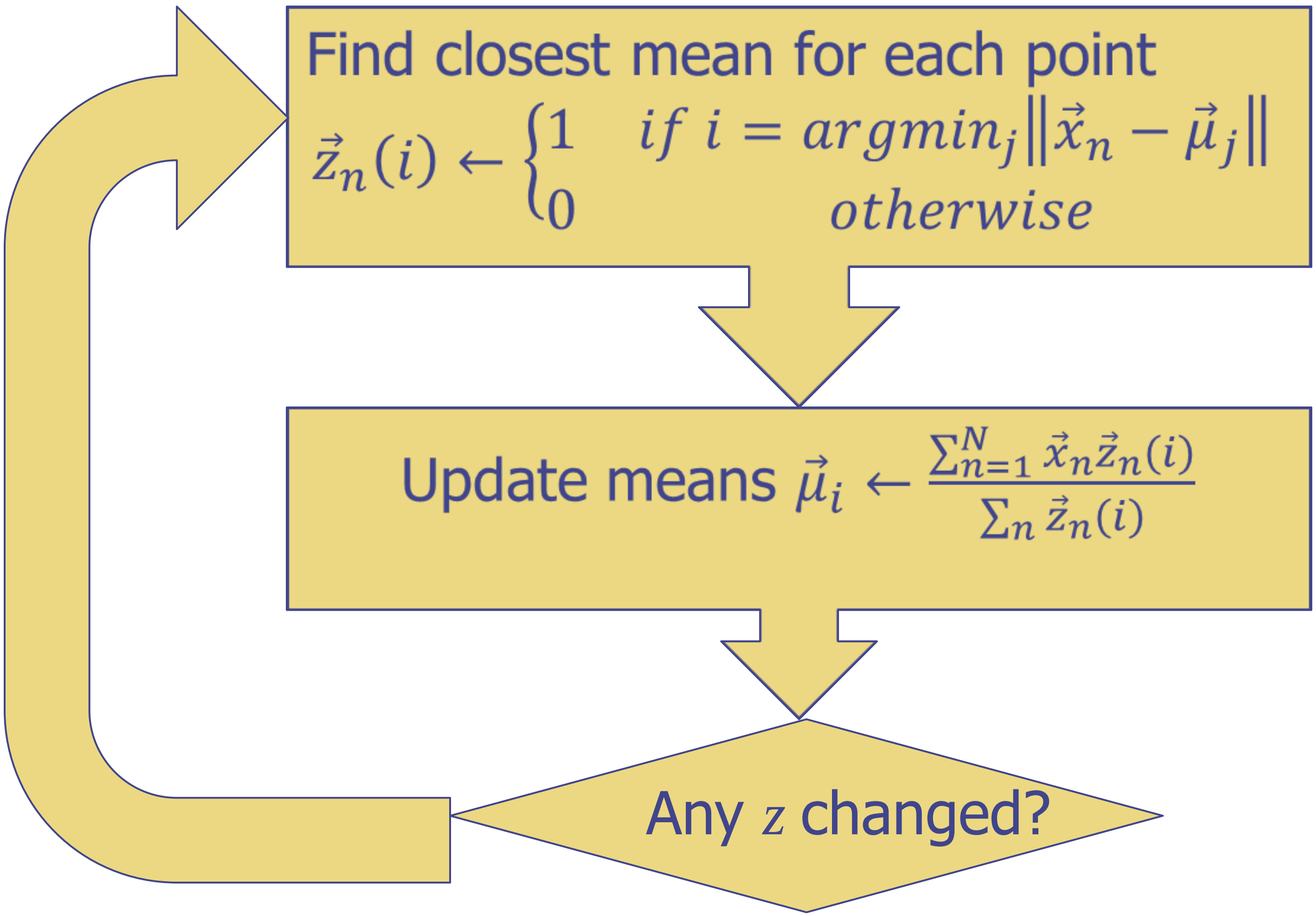
# The K-Means Algorithm

Find closest mean for each point

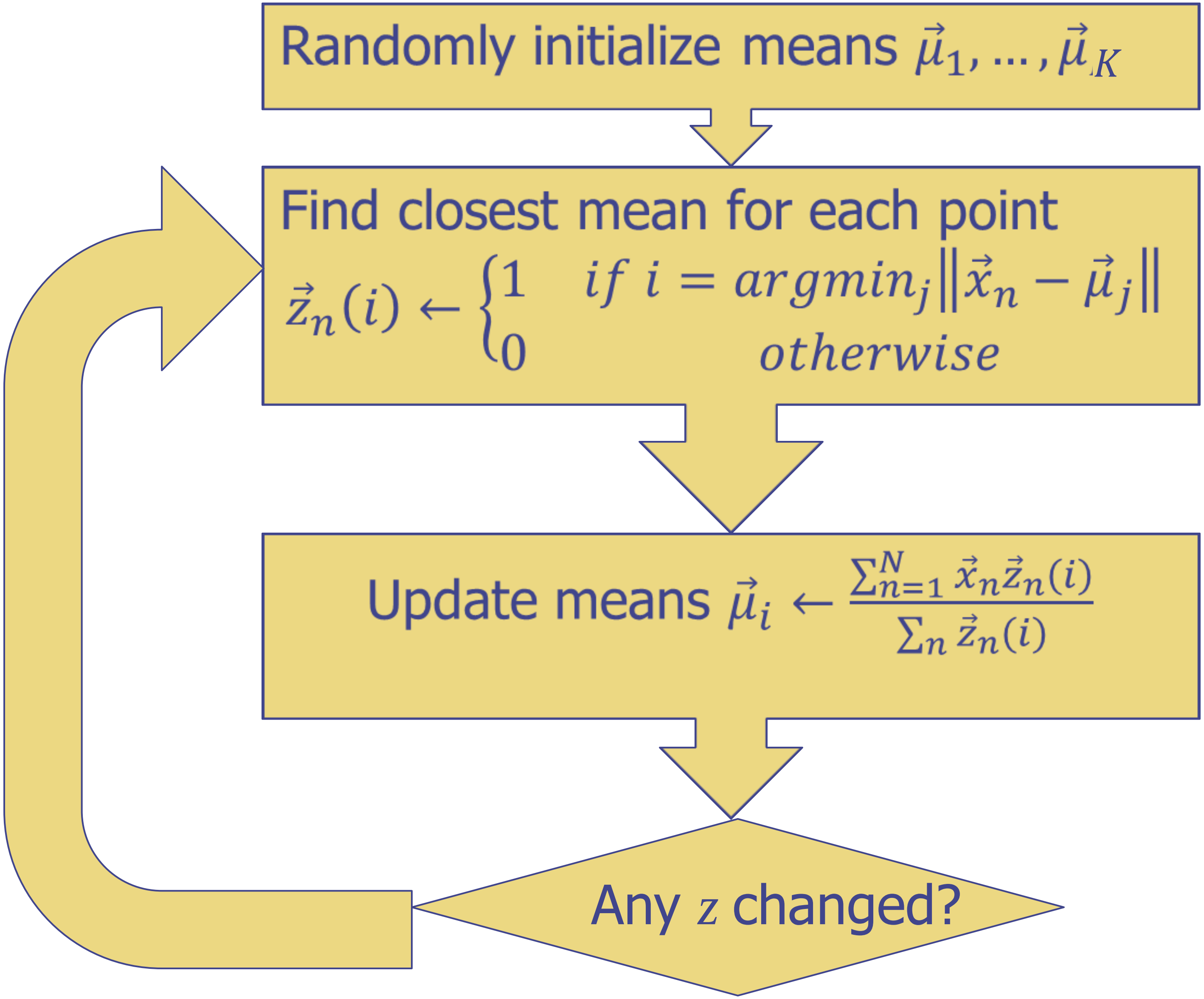
$$\vec{z}_n(i) \leftarrow \begin{cases} 1 & if\ i = argmin_j \left\| \vec{x}_n - \vec{\mu}_j \right\| \\ 0 & otherwise \end{cases}$$

Update means $\vec{\mu}_i \leftarrow \frac{\sum_{n=1}^{N} \vec{x}_n \vec{z}_n(i)}{\sum_n \vec{z}_n(i)}$

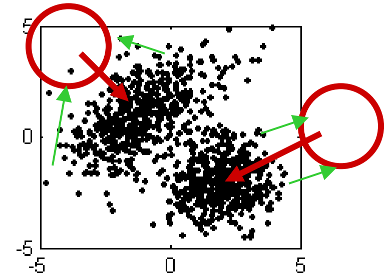
# The K-Means Algorithm



# The K-Means Algorithm



# K-Means Clustering

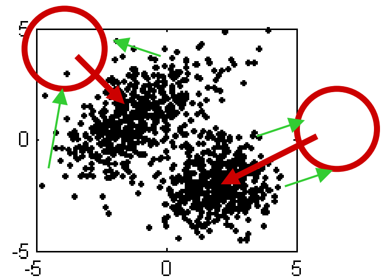



- Geometric, each point goes to closest Gaussian
- Recompute the means by their assigned points
- Minimizing $\min_{\mu} \min_{z} J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \vec{z}_1, \ldots, \vec{z}_N)$ cost function:

$$J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \vec{z}_1, \ldots, \vec{z}_N) = \sum_{n=1}^{N} \sum_{i=1}^{K} \vec{z}_n(i) \|\vec{x}_n - \vec{\mu}_i\|^2$$

$$\vec{z}_n(i) = \begin{cases} 1 & if\ i = argmin_j \left\|\vec{x}_n - \vec{\mu}_j\right\| \\ 0 & otherwise \end{cases}$$ and $$\vec{\mu}_i = \frac{\sum_{n=1}^{N} \vec{x}_n \vec{z}_n(i)}{\sum_n \vec{z}_n(i)}$$

# K-Means Clustering

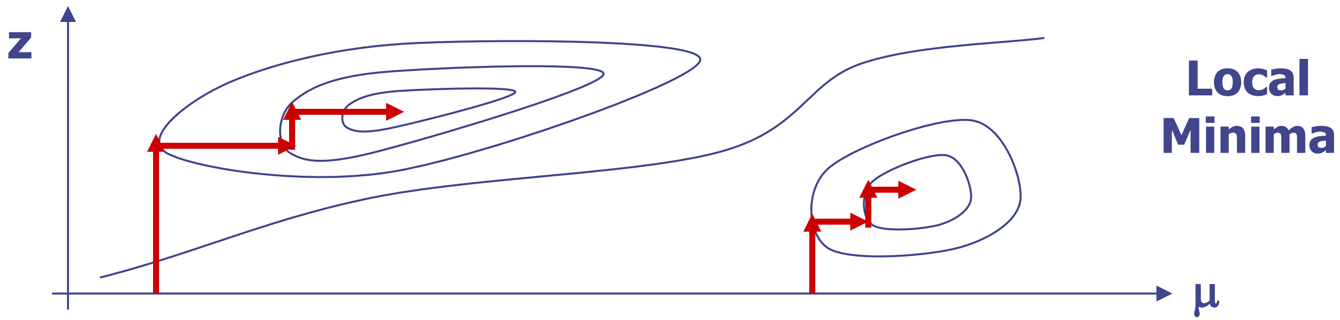



- Geometric, each point goes to closest Gaussian
- Recompute the means by their assigned points
- Minimizing $\min_{\mu} \min_{z} J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \vec{z}_1, \ldots, \vec{z}_N)$ cost function:

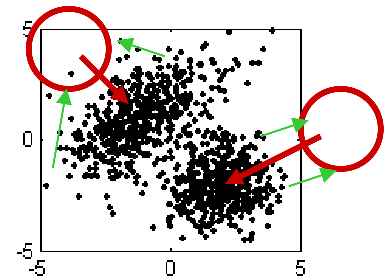
$$J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \vec{z}_1, \ldots, \vec{z}_N) = \sum_{n=1}^{N} \sum_{i=1}^{K} \vec{z}_n(i) \|\vec{x}_n - \vec{\mu}_i\|^2$$

$$\vec{z}_n(i) = \begin{cases} 1 & if\ i = argmin_j \|\vec{x}_n - \vec{\mu}_j\| \\ 0 & otherwise \end{cases} \quad \text{and } \vec{\mu}_i = \frac{\sum_{n=1}^{N} \vec{x}_n \vec{z}_n(i)}{\sum_n \vec{z}_n(i)}$$

- Guaranteed to improve per iteration and converge
- Like Coordinate Descent (lock one var, maximize the other)
- A.k.a. Axis-Parallel Optimization or Alternating Minimization



# K-Means Clustering

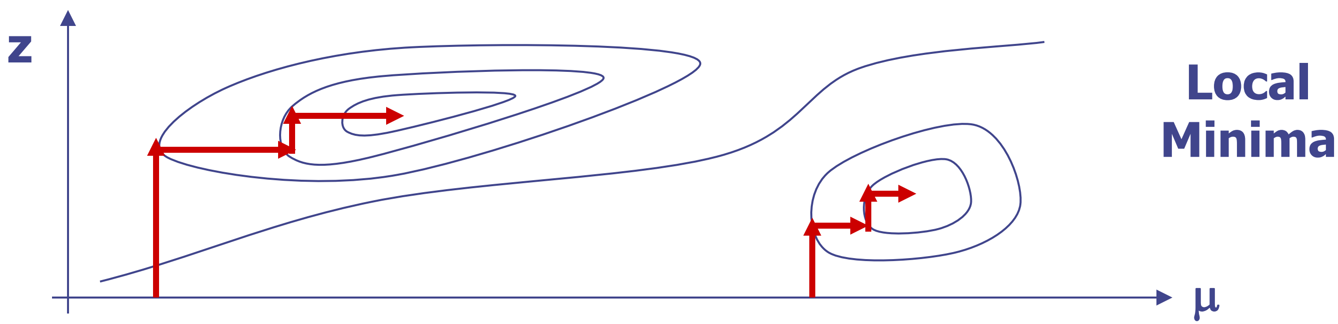



- Geometric, each point goes to closest Gaussian
- Recompute the means by their assigned points
- Minimizing $\min_{\mu,\Sigma} \min_{z} J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \Sigma_1, \ldots, \Sigma_K, \vec{z}_1, \ldots, \vec{z}_N)$ cost:

$$J(\vec{\mu}_1, \ldots, \vec{\mu}_K, \vec{z}_1, \ldots, \vec{z}_N) = \sum_{n=1}^{N} \sum_{i=1}^{K} \vec{z}_n(i)(\vec{x}_n - \vec{\mu}_i)^T \Sigma^{-1} (\vec{x}_n - \vec{\mu}_i)$$

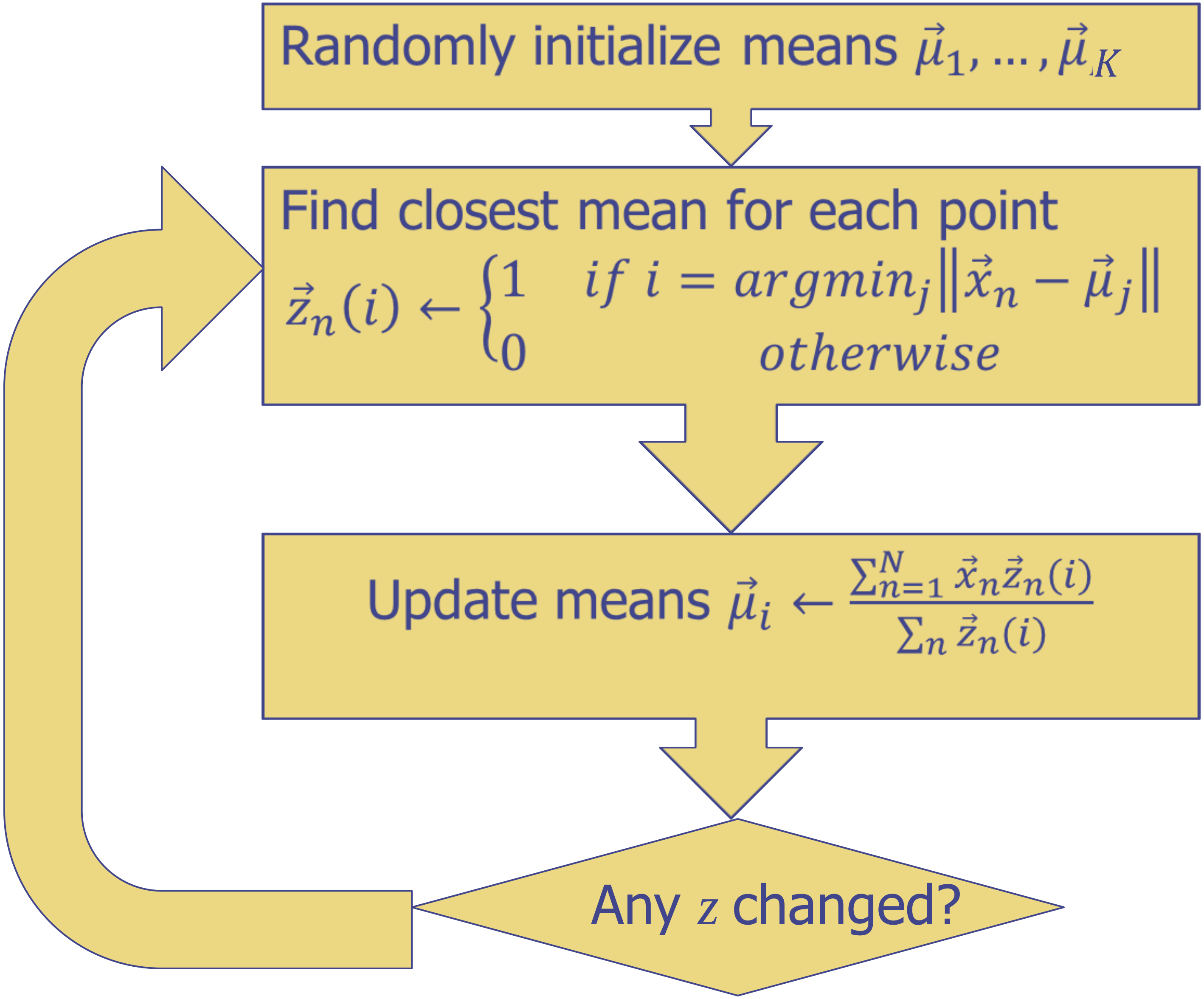
$$\vec{z}_n(i) = \begin{cases} 1 & if\ i = argmin_j (\vec{x}_n - \vec{\mu}_i)^T \Sigma^{-1} (\vec{x}_n - \vec{\mu}_i) \\ 0 & otherwise \end{cases}$$

- Guaranteed to improve per iteration and converge
- Like Coordinate Descent (lock one var, maximize the other)
- A.k.a. Axis-Parallel Optimization or Alternating Minimization



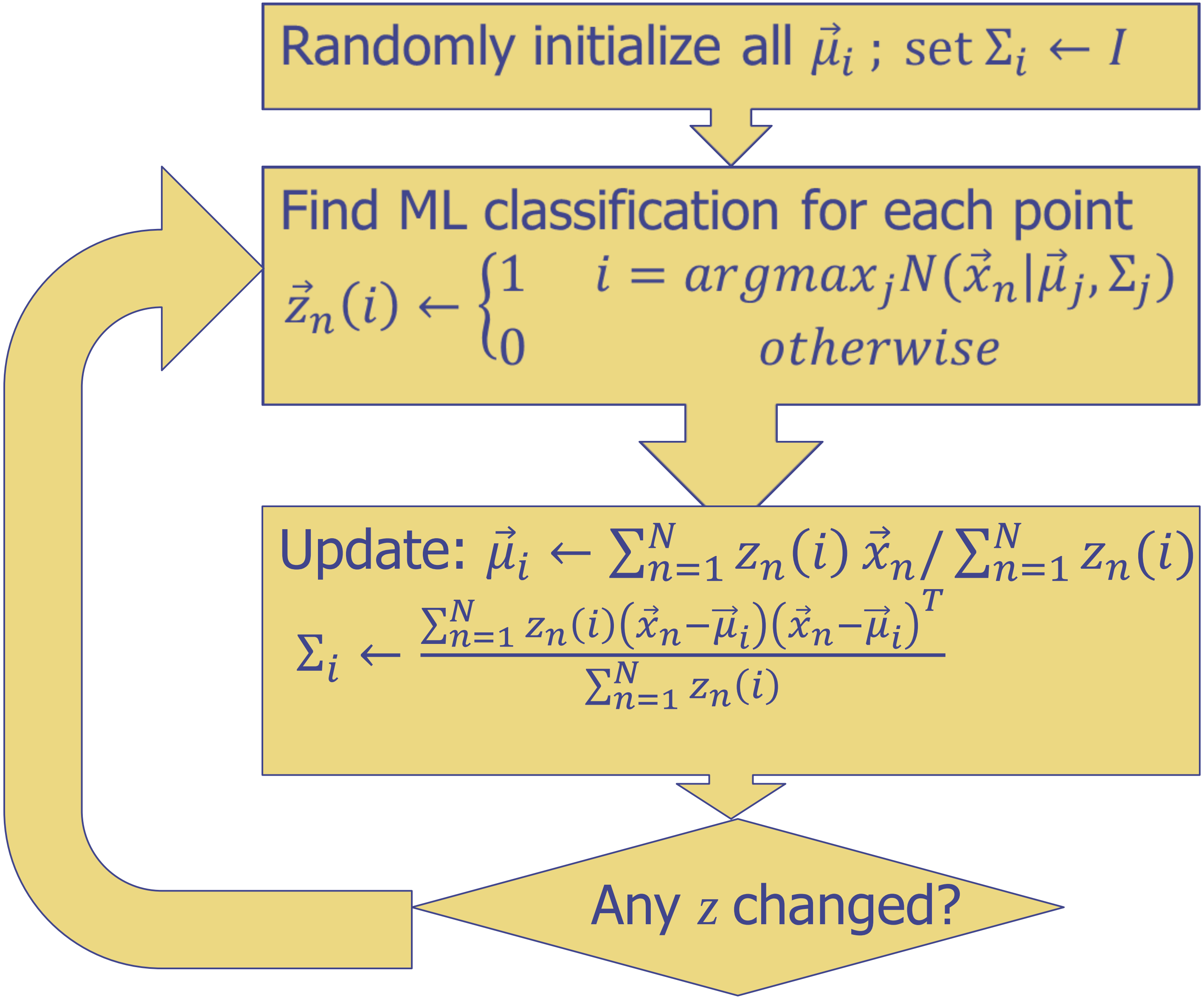
Itsik Pe'er, Columbia University

# The K-Means Algorithm

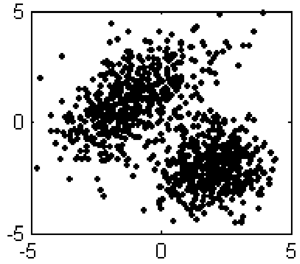


Itsik Pe'er, Columbia University

# The Gaussian K-Means Algorithm



# Kmeans→Expectation Maximization




At each stage of K-means, an element chooses the most likely (responsible) cluster

Hedging allows smoother optimization surface

Take expectation over responsible cluster

**mixing proportions (prior) =** $\boldsymbol{\pi} = p\left(\vec{z} = \vec{\delta}_i \middle| \pi\right)$

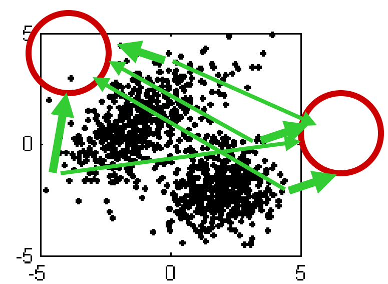
**mixture components =** $p\left(\vec{x} \middle| \vec{z} = \vec{\delta}_i, \theta\right)$

**posteriors (responsibilities) =** $\tau_{n,i} = p\left(\vec{z} = \vec{\delta}_i \middle| \vec{x}_n, \theta\right) = \frac{p\left(\vec{x}_n \middle| \vec{z}=\vec{\delta}_i, \theta\right) p\left(\vec{z}=\vec{\delta}_i \middle| \theta\right)}{p(\vec{x}_n|\theta)}$

# Expectation-Maximization (EM)

- EM is a soft/fuzzy version of K-Means (which does winner-takes-all, closest Gaussian Mean completely wins datapoint)

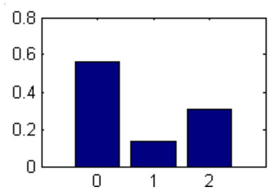
$$\vec{z}_n(i) = \begin{cases} 1 & i = argmax_j N(\vec{x}_n | \vec{\mu}_j, \Sigma_j) \\ 0 & otherwise \end{cases}$$




- Instead, consider soft percentage assignment of datapoint (responsibility)

$$\tau_{n,i} = p\left(\vec{z}_n = \vec{\delta}(j) | \vec{x}_n, \theta \right) \propto \pi_j \frac{\exp\left(-\frac{1}{2}(\vec{x}_n - \vec{\mu}_j)^T \Sigma^{-1} (\vec{x}_n - \vec{\mu}_j)\right)}{(2\pi)^{\frac{D}{2}} \sqrt{|\Sigma|}}$$

$$\tau_{n,1}, \ldots, \tau_{n,K} =$$

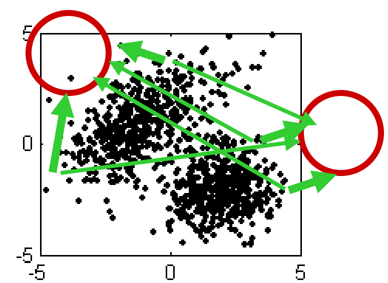



- Update for the means are then

# Expectation-Maximization (EM)

- EM is a soft/fuzzy version of K-Means (which does winner-takes-all, closest Gaussian Mean completely wins datapoint)

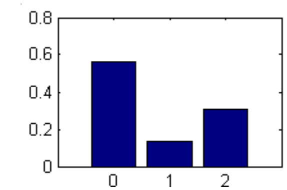
$$\vec{z}_n(i) = \begin{cases} 1 & i = argmax_j N(\vec{x}_n | \vec{\mu}_j, \Sigma_j) \\ 0 & otherwise \end{cases}$$




- Instead, consider soft percentage assignment of datapoint (responsibility)

$$\tau_{n,i} = p\left(\vec{z}_n = \vec{\delta}(j) | \vec{x}_n, \theta\right) \propto \pi_j \frac{\exp\left(-\frac{1}{2}(\vec{x}_n - \vec{\mu}_j)^T \Sigma^{-1} (\vec{x}_n - \vec{\mu}_j)\right)}{(2\pi)^{\frac{D}{2}} \sqrt{|\Sigma|}}$$

$$\tau_{n,1}, \dots, \tau_{n,K} =$$




- Update for the means are then 'weighted' by responsibilities: $\vec{\mu}_i = \frac{\sum_{n=1}^{N} \tau_{n,i} \vec{x}_n}{\sum_n \tau_{n,i}}$
- Same for other parameters

# Expectation-Maximization

- EM uses expected value of $\vec{z}_n(i)$ rather than max

$$\tau_{n,i} = E\{\vec{z}_n(i)|\vec{x}_n\} = p\left(\vec{z}_n = \vec{\delta}(j)|\vec{x}_n, \theta\right)$$

- EM updates covariances, mixing proportions AND means…
- The algorithm for Gaussian mixtures:

EXPECTATION: $\tau_{n,i}^{(t)} \leftarrow$

MAXIMIZATION: $\vec{\mu}_i^{(t+1)} \leftarrow$ $\pi_i^{(t+1)} \leftarrow$

$\Sigma_i^{(t+1)} \leftarrow$

# Expectation-Maximization

- EM uses expected value of $\vec{z}_n(i)$ rather than max

$$\tau_{n,i} = E\{\vec{z}_n(i)|\vec{x}_n\} = p\left(\vec{z}_n = \vec{\delta}(j)|\vec{x}_n, \theta\right)$$

- EM updates covariances, mixing proportions AND means…
- The algorithm for Gaussian mixtures:

EXPECTATION: $\tau_{n,i}^{(t)} \leftarrow \frac{\pi_i^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_j \pi_j^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_j^{(t)}, \Sigma_j^{(t)}\right)}$

MAXIMIZATION: $\vec{\mu}_i^{(t+1)} \leftarrow \frac{\sum_{n=1}^{N} \tau_{n,i}^{(t)} \vec{x}_n}{\sum_n \tau_{n.i}^{(t)}}$

$$\Sigma_i^{(t+1)} \leftarrow$$

# Expectation-Maximization

- EM uses expected value of $\vec{z}_n(i)$ rather than max

$$\tau_{n,i} = E\{\vec{z}_n(i)|\vec{x}_n\} = p\left(\vec{z}_n = \vec{\delta}(j)|\vec{x}_n, \theta\right)$$

- EM updates covariances, mixing proportions AND means…
- The algorithm for Gaussian mixtures:

EXPECTATION: $\tau_{n,i}^{(t)} \leftarrow \frac{\pi_i^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_j \pi_j^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_j^{(t)}, \Sigma_j^{(t)}\right)}$

MAXIMIZATION: $\vec{\mu}_i^{(t+1)} \leftarrow \frac{\sum_{n=1}^{N} \tau_{n,i}^{(t)} \vec{x}_n}{\sum_n \tau_{n,i}^{(t)}}$ $\quad \pi_i^{(t+1)} \leftarrow \frac{\sum_n \tau_{n,i}^{(t)}}{N}$

$$\Sigma_i^{(t+1)} \leftarrow \frac{\sum_{n=1}^{N} \tau_{n,i}^{(t)} \left(\vec{x}_n - \vec{\mu}_i^{(t+1)}\right)^T \left(\vec{x}_n - \vec{\mu}_i^{(t+1)}\right)}{\sum_n \tau_{n,i}^{(t)}}$$

# The EM Clustering Algorithm

Initialize: random $\vec{\mu}_i^{(1)}, \Sigma_i^{(1)} = I,\ \pi_i^{(1)} = \frac{1}{K}$

Find expected assignment to each $\vec{x}_n$

$$\tau_{n,i}^{(t)} \leftarrow \frac{\pi_i^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_j \pi_j^{(t)} N\left(\vec{x}_n \middle| \vec{\mu}_j^{(t)}, \Sigma_j^{(t)}\right)}$$

Expectation

Update $\vec{\mu}_i^{(t+1)} \leftarrow \sum_{n=1}^{N} \tau_{n,i}^{(t)}\, \vec{x}_n / \sum_{n=1}^{N} \tau_{n,i}^{(t)}$

$$\Sigma_i^{(t+1)} \leftarrow \frac{\sum_{n=1}^{N} \tau_{n,i}^{(t)} \left(\vec{x}_n - \vec{\mu}_i^{(t+1)}\right)\left(\vec{x}_n - \vec{\mu}_i^{(t+1)}\right)^T}{\sum_{n=1}^{N} \tau_{n,i}^{(t)}}$$

$$\pi_i^{(t+1)} \leftarrow \sum_{n=1}^{N} \tau_{n,i}^{(t)} / N$$

Maximization

Little changed?

# The EM Clustering Algorithm

Initialize: random $\vec{\mu}_i^{(1)}, \Sigma_i^{(1)} = I,\ \pi_i^{(1)} = \frac{1}{K}$

Find expected assignment to each $\vec{x}_n$

$$\tau_{n,i}^{(t)} \leftarrow E\{\vec{z}_n(i)\}$$

Expectation

Update

$$\vec{\mu}_i^{(t+1)} \leftarrow argmax_{\mu^*}\ E_{\vec{z}}\{l(\theta, z\ s.t.\ \vec{\mu}_i = \mu^*)\}$$

$$\Sigma_i^{(t+1)} \leftarrow argmax_{\Sigma^*} E_{\vec{z}}\{l(\theta, z\ s.t. \Sigma_i = \Sigma^*)\}$$

$$\pi_i^{(t+1)} \leftarrow argmax_{\pi^*} E_{\vec{z}}\{l(\theta, z\ s.t. \pi_i = \pi^*)\}$$

Maximization

Little changed?

# The General EM Algorithm

Initialize $\theta^{(1)}$

Find expected assignment to each $\vec{x}_n$

$$\tau_{n,i}^{(t)} \leftarrow E\{\vec{z}_n(i)\}$$

Expectation

Update

$$\vec{\theta}^{(t+1)} \leftarrow argmax_{\theta^*} E_{\vec{z}}\{l(\theta^*, z)\}$$

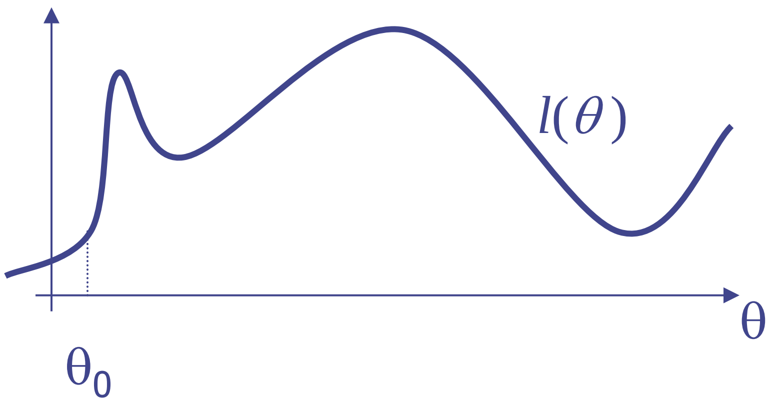
Maximization

Little changed?

# EM as Bound Maximization

- Let's now show that EM indeed maximizes likelihood

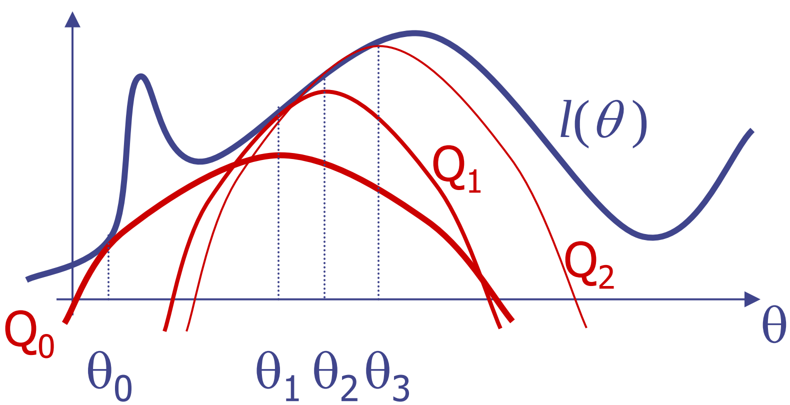
- Bound Maximization: optimize a lower bound on $l(\theta)$
- Since log-likelihood $l(\theta)$ not concave, can't max it directly
- Consider an auxiliary function $Q(\theta)$ which is concave
- $Q(\theta)$ kisses $l(\theta)$ at a point and is less than it elsewhere



# EM as Bound Maximization

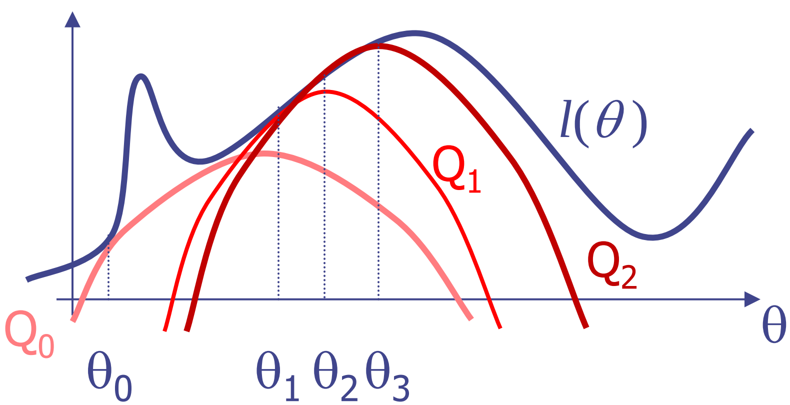
- Let's now show that EM indeed maximizes likelihood

- Bound Maximization: optimize a lower bound on $l(\theta)$
- Since log-likelihood $l(\theta)$ not concave, can't max it directly
- Consider an auxiliary function $Q(\theta)$ which is concave
- $Q(\theta)$ kisses $l(\theta)$ at a point and is less than it elsewhere



# EM as Bound Maximization

- Let's now show that EM indeed maximizes likelihood
- Bound Maximization: optimize a lower bound on $l(\theta)$
- Since log-likelihood $l(\theta)$ not concave, can't max it directly
- Consider an auxiliary function $Q(\theta)$ which is concave
- $Q(\theta)$ kisses $l(\theta)$ at a point and is less than it elsewhere




$$\forall\theta\forall t: \ l(\theta) \geq Q_t(\theta)$$

$$l(\theta_t) = Q_t(\theta_t)$$

$$Q_t(\theta_{t+1}) > Q_t(\theta_t)$$

because $\theta_{t+1} = \arg\max_{\theta} Q_t(\theta)$

$$l(\theta_{t+1}) \geq Q_t(\theta_{t+1}) > Q_t(\theta_t) = l(\theta_t)$$

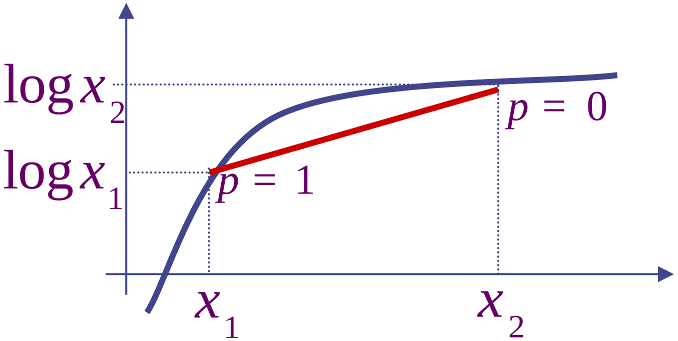
- Monotonically increases log-likelihood
- But how to find a bound and guarantee we max it?

# Jensen's Inequality

- Example: $f(x)=log(x)$=concave and $M$=2

$$\log(px_1 + (1-p)x_2) \geq p \log x_1 + (1-p) \log x_2$$

- Bound log(sum) with sum(log)

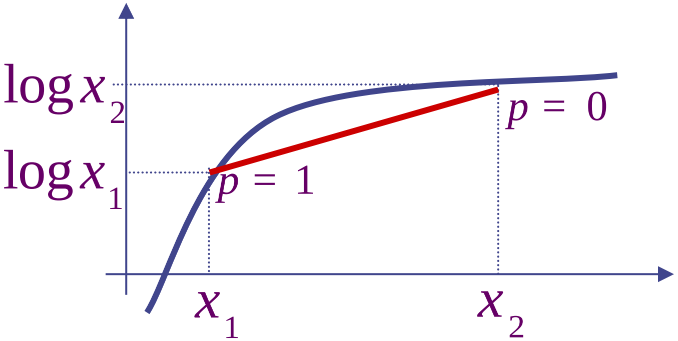


# Jensen's Inequality

- Expectation in discrete case is sum weight by probability
- For convex $f$: $f\left(\sum_{i=1}^{M} p_i x_i\right) \le \sum_{i=1}^{M} p_i f(x_i)$ when $\sum_{i=1}^{M} p_i = 1, p_i \ge 0$
- For concave $f$: $f\left(\sum_{i=1}^{M} p_i x_i\right) \ge \sum_{i=1}^{M} p_i f(x_i)$ when $\sum_{i=1}^{M} p_i = 1, p_i \ge 0$
- Example: $f(x)=log(x)$=concave and $M$=2

$$\log(p x_1 + (1-p) x_2) \ge p \log x_1 + (1-p) \log x_2$$

- Bound log(sum) with sum(log)

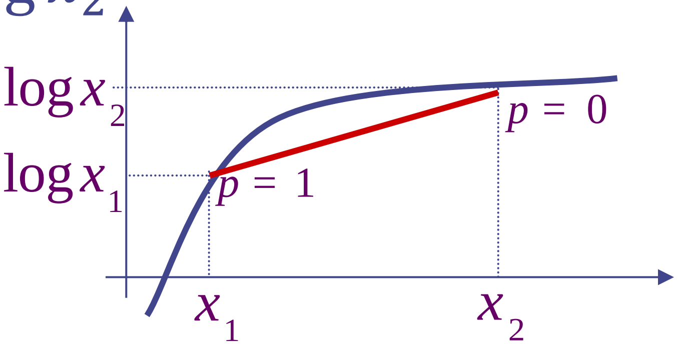


# Jensen's Inequality

- An important general bound from Jensen (1906)
- For convex $f$: $f(E\{x\}) \le E\{f(x)\}$
- For concave $f$: $f(E\{x\}) \ge E\{f(x)\}$
- Expectation in discrete case is sum weight by probability
- For convex $f$: $f\left(\sum_{i=1}^{M} p_i x_i\right) \le \sum_{i=1}^{M} p_i f(x_i)$ when $\sum_{i=1}^{M} p_i = 1, p_i \ge 0$
- For concave $f$: $f\left(\sum_{i=1}^{M} p_i x_i\right) \ge \sum_{i=1}^{M} p_i f(x_i)$ when $\sum_{i=1}^{M} p_i = 1, p_i \ge 0$
- Example: $f(x)=log(x)$=concave and $M=2$

$$\log(px_1 + (1-p)x_2) \ge p \log x_1 + (1-p) \log x_2$$

- Bound log(sum) with sum(log)
- How to apply this to mixture models?



# Expectation-Maximization

$l(\theta) =$

# Expectation-Maximization

$$l(\theta) = \sum_{n=1}^{N} \log p(x_n|\theta)$$

**Original Log-Likelihood**

$$= \sum_{n=1}^{N} \log \sum_{z} p(x_n, z|\theta)$$

**Has Hidden Variables (messy)**

$$= \sum_{n=1}^{N} \log \sum_z p(x_n, z|\theta) \frac{p(z|x_n,\theta_t)}{p(z|x_n,\theta_t)}$$

**Multiply by 1**
**Ratio of hidden posterior density**

$$= \sum_{n=1}^{N} \log \sum_z p(z|x_n, \theta_t) \frac{p(x_n,z|\theta)}{p(z|x_n,\theta_t)}$$

**Rearrange**

# Expectation-Maximization

$$l(\theta) = \sum_{n=1}^{N} \log p(x_n|\theta)$$

**Original Log-Likelihood**

$$= \sum_{n=1}^{N} \log \sum_{z} p(x_n, z|\theta)$$

**Has Hidden Variables (messy)**

$$= \sum_{n=1}^{N} \log \sum_z p(x_n, z|\theta) \frac{p(z|x_n,\theta_t)}{p(z|x_n,\theta_t)}$$

**Multiply by 1**
**Ratio of hidden posterior density**

$$= \sum_{n=1}^{N} \log \sum_z p(z|x_n, \theta_t) \frac{p(x_n,z|\theta)}{p(z|x_n,\theta_t)}$$

**Rearrange**

$$\geq \sum_{n=1}^{N} \sum_z p(z|x_n, \theta_t) \log \frac{p(x_n,z|\theta)}{p(z|x_n,\theta_t)}$$

**Jensen log($\Sigma_i p_i x_i$ )**

$$= \sum_{n=1}^{N} \sum_{z} p(z|x_n, \theta_t) \log p(x_n, z|\theta) - \sum_{n=1}^{N} \sum_{z} p(z|x_n, \theta_t) \log p(z|x_n, \theta_t)$$

$$= Q(\theta|\theta_t) - const$$

**New auxiliary function called Q (not messy)**

# EM as Bound Maximization

- Now have the following bound and maximize it:

$l(\theta) \geq$

# EM as Bound Maximization

- Now have the following bound and maximize it:

$$l(\theta) \geq Q(\theta|\theta_t) - \sum_{n=1}^{N} \sum_z p(z|x_n, \theta_t) \log p(z|x_n, \theta_t)$$

$$\theta_{t+1} = \arg\max Q(\theta|\theta_t) = \arg\max \sum_{n=1}^{N} \sum_{z} p(z|x_n, \theta_t) \log p(x_n, z|\theta_t)$$

$$= \arg\max \sum_{n=1}^{N} \sum_z \tau_{n,z} \log p(x_n, z|\theta_t)$$

- $Q(\theta|\theta_t)$ is called Auxiliary Function… take derivatives of it
- This is easy for many functions… just weighted max likelihood!
- For example, Gaussian mixture:

$$\frac{\partial Q(\theta)}{\partial \vec{\mu}_k} =$$

# EM as Bound Maximization

- Now have the following bound and maximize it:

$$l(\theta) \geq Q(\theta|\theta_t) - \sum_{n=1}^{N} \sum_z p(z|x_n, \theta_t) \log p(z|x_n, \theta_t)$$

$$\theta_{t+1} = \arg\max Q(\theta|\theta_t) = \arg\max \sum_{n=1}^{N} \sum_{z} p(z|x_n, \theta_t) \log p(x_n, z|\theta_t)$$

$$= \arg\max \sum_{n=1}^{N} \sum_z \tau_{n,z} \log p(x_n, z|\theta_t)$$

- $Q(\theta|\theta_t)$ is called Auxiliary Function... take derivatives of it
- This is easy for many functions... just weighted max likelihood!
- For example, Gaussian mixture:

$$\frac{\partial Q(\theta)}{\partial \vec{\mu}_k} = \frac{\partial}{\partial \vec{\mu}_k} \sum_{n=1}^{N} \sum_{j=1}^{K} \tau_{n,k} \log \pi_k N(\vec{x}_n | \vec{\mu}_k, \Sigma_k)$$

$$0 = \sum_{n=1}^{N} \tau_{n,k} \frac{\partial}{\partial \vec{\mu}_k} \left( -\frac{1}{2} (\vec{x}_n - \vec{\mu}_k)^T \Sigma^{-1} (\vec{x}_n - \vec{\mu}_k) \right)$$

$$\vec{\mu}_k = \frac{\sum_{n=1}^{N} \tau_{n,k} \vec{x}_n}{\sum_{n=1}^{N} \tau_{n,k}}$$

... similarly get $\pi_k$ and $\Sigma_k$

# EM as Expected Log-Likelihood

- Incomplete Log-Likelihood

- Complete Log-Likelihood

# EM as Expected Log-Likelihood

- Incomplete Log-Likelihood

$$l(\theta) = \log p(observed|\theta) = \sum_{n=1}^{N} \log \sum_{z} p(x_n, z_n|\theta)$$

- Complete Log-Likelihood

$$l(\theta) = \log p(observed, hidden|\theta) = \sum_{n=1}^{N} \log p(x_n, z_n|\theta)$$

- We don't know the hidden variables $z$
- EM computes expected values of hidden $z$ under current $\theta_t$
- EM chooses $Q$ to be the Expected Complete Log-Likelihood

# EM as Expected Log-Likelihood

- Incomplete Log-Likelihood

$l(\theta) = \log p(observed|\theta) = \sum_{n=1}^{N} \log \sum_z p(x_n, z_n|\theta)$

- Complete Log-Likelihood

$l(\theta) = \log p(observed, hidden|\theta) = \sum_{n=1}^{N} \log p(x_n, z_n|\theta)$

- We don't know the hidden variables z
- EM computes expected values of hidden z under current $\theta_t$
- EM chooses Q to be the Expected Complete Log-Likelihood

$E\{l^c(\theta)\} =$

# EM as Expected Log-Likelihood

- Incomplete Log-Likelihood

$l(\theta) = \log p(observed|\theta) = \sum_{n=1}^{N} \log \sum_z p(x_n, z_n|\theta)$

- Complete Log-Likelihood

$l(\theta) = \log p(observed, hidden|\theta) = \sum_{n=1}^{N} \log p(x_n, z_n|\theta)$

- We don't know the hidden variables z
- EM computes expected values of hidden z under current $\theta_t$
- EM chooses Q to be the Expected Complete Log-Likelihood

$E\{l^c(\theta)\} = \sum_{hidden} p(hidden|observed, \theta_t) l^c(\theta)$

$= \sum_{z_1} \sum_{z_2} \dots \sum_{z_N} p(z_1, \dots, z_N|x_1, \dots, x_N, \theta_t) l^c(\theta)$

$= \sum_{z_1} \sum_{z_2} \dots \sum_{z_N} \prod_n p(z_n|x_n, \theta_t) l^c(\theta)$

$= \sum_{z_1} \sum_{z_2} \dots \sum_{z_N} \prod_n p(z_n|x_n, \theta_t) \sum_n \log p(x_n, z_n|\theta)$

$= \sum_n \sum_{z_n} p(z_n|x_n, \theta_t) \log p(x_n, z_n|\theta) \sum_{z_1} \dots \sum_{z_{i\neq n}} \dots \sum_{z_N} \prod_{i\neq n} p(z_i|x_i, \theta_t)$

$= \sum_n \sum_n p(z_n|x_n, \theta_t) \log p(x_n, z_n|\theta) = Q(\theta|\theta_t)$

# Summary

- Mixture Models and Hidden Variables
- Clustering
- K-Means
- Expectation Maximization

- Happy Spring Break!