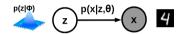
Generative Models for Clustering, GMM, and Intro to EM

Piyush Rai

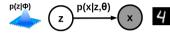
Machine Learning (CS771A)

Sept 26, 2016

• A probabilistic way to think about the data generation process

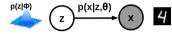


A probabilistic way to think about the data generation process



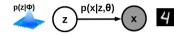
• Idea: First generate a random latent variable z from a prior distr. $p(z|\phi)$ and then generate x conditioned on z from the data distr. $p(x|z,\theta)$

• A probabilistic way to think about the data generation process

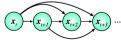


- Idea: First generate a random latent variable z from a prior distr. $p(z|\phi)$ and then generate x conditioned on z from the data distr. $p(x|z,\theta)$
- Some exceptions to this general view/definition of a generative model:

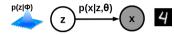
A probabilistic way to think about the data generation process



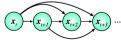
- Idea: First generate a random latent variable z from a prior distr. $p(z|\phi)$ and then generate x conditioned on z from the data distr. $p(x|z,\theta)$
- Some exceptions to this general view/definition of a generative model:
 - Not all generative models have latent variables (e.g., for each observation, the other observations may play the role of latent variables)



A probabilistic way to think about the data generation process

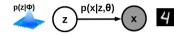


- Idea: First generate a random latent variable z from a prior distr. $p(z|\phi)$ and then generate x conditioned on z from the data distr. $p(x|z,\theta)$
- Some exceptions to this general view/definition of a generative model:
 - Not all generative models have latent variables (e.g., for each observation, the other observations may play the role of latent variables)

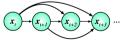


• The z to x map may be a deterministic fn. (e.g., a neural or deep neural net)

A probabilistic way to think about the data generation process



- Idea: First generate a random latent variable z from a prior distr. $p(z|\phi)$ and then generate x conditioned on z from the data distr. $p(x|z,\theta)$
- Some exceptions to this general view/definition of a generative model:
 - Not all generative models have latent variables (e.g., for each observation, the other observations may play the role of latent variables)



- The z to x map may be a deterministic fn. (e.g., a neural or deep neural net)
- We will focus on probabilistic generative models with latent variables



Generative Models for Clustering

- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- Let $0 \le \pi_k \le 1$ denote the "mixing weight" of the *k*-th Gaussian. It means:
 - π_k is the fraction of points generated from the k-th Gaussian

- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- Let $0 \le \pi_k \le 1$ denote the "mixing weight" of the *k*-th Gaussian. It means:
 - π_k is the fraction of points generated from the k-th Gaussian
 - $\pi_k = p(z_n = k)$ is the prior prob. of x_n belonging to the k-th Gaussian

- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- Let $0 \le \pi_k \le 1$ denote the "mixing weight" of the *k*-th Gaussian. It means:
 - π_k is the fraction of points generated from the k-th Gaussian
 - $\pi_k = p(z_n = k)$ is the prior prob. of x_n belonging to the k-th Gaussian
- Let $\pi = (\pi_1, \pi_2, \dots, \pi_K)$ denote the vector of mixing wts of K Gaussians. This is a probability vector and sums to 1, i.e., $\sum_{k=1}^K \pi_k = 1$

- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- Let $0 \le \pi_k \le 1$ denote the "mixing weight" of the *k*-th Gaussian. It means:
 - π_k is the fraction of points generated from the k-th Gaussian
 - $\pi_k = p(z_n = k)$ is the prior prob. of x_n belonging to the k-th Gaussian
- Let $\pi = (\pi_1, \pi_2, \dots, \pi_K)$ denote the vector of mixing wts of K Gaussians. This is a probability vector and sums to 1, i.e., $\sum_{k=1}^{K} \pi_k = 1$
- Notation $z_n = k$ is equivalent to a size K one-hot vector z_n

$$z_n = \underbrace{\begin{bmatrix} 0 & 0 & \dots & 1 & 0 & 0 \end{bmatrix}}_{\text{output}}$$

all zeros except the k-th bit, i.e., $z_{nk}\,=\,1$

- A generative model for data clustering
- Data assumed generated from a mixture of K Gaussians



- Let $0 \le \pi_k \le 1$ denote the "mixing weight" of the k-th Gaussian. It means:
 - π_k is the fraction of points generated from the k-th Gaussian
 - $\pi_k = p(z_n = k)$ is the prior prob. of x_n belonging to the k-th Gaussian
- Let $\pi = (\pi_1, \pi_2, \dots, \pi_K)$ denote the vector of mixing wts of K Gaussians. This is a probability vector and sums to 1, i.e., $\sum_{k=1}^{K} \pi_k = 1$
- Notation $z_n = k$ is equivalent to a size K one-hot vector z_n

$$\mathbf{z}_n = [0 \ 0 \dots 1 \ 0 \ 0]$$

all zeros except the k-th bit, i.e., $z_{nk}=1$

• Note: The prior $p(\boldsymbol{z}|\pi)$ on each \boldsymbol{z}_n is a multinomial, i.e., $p(\boldsymbol{z}_n|\pi) = \prod_{k=1}^K \pi_k^{\boldsymbol{z}_{nk}}$

• The generative story for each x_n , n = 1, 2, ..., N

- The generative story for each x_n , n = 1, 2, ..., N
 - First choose one of the K mixture components as

 $z_n \sim \text{Multinomial}(z_n|\pi)$ (from the prior p(z) over z)

- The generative story for each x_n , n = 1, 2, ..., N
 - First choose one of the K mixture components as

$$z_n \sim \text{Multinomial}(z_n|\pi)$$
 (from the prior $p(z)$ over z)

• Suppose $z_n = k$. Now generate x_n from the k-th Gaussian as

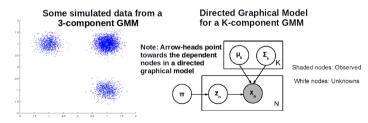
$$\mathbf{x}_n \sim \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (from the data distr. $p(\mathbf{x} | \mathbf{z})$)

- The generative story for each x_n , n = 1, 2, ..., N
 - First choose one of the K mixture components as

$$z_n \sim \text{Multinomial}(z_n|\pi)$$
 (from the prior $p(z)$ over z)

• Suppose $z_n = k$. Now generate x_n from the k-th Gaussian as

$$\mathbf{x}_n \sim \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (from the data distr. $p(\mathbf{x} | \mathbf{z})$)



• Multivariate Gaussian in D dimensions

$$p(\pmb{x}|\mu,\pmb{\Sigma}) = rac{1}{(2\pi)^{D/2}|\pmb{\Sigma}|^{1/2}} \exp\left(-rac{1}{2}(\pmb{x}-\mu)^{ op}\pmb{\Sigma}^{-1}(\pmb{x}-\mu)
ight)$$

• Multivariate Gaussian in D dimensions

$$p(\pmb{x}|\mu,\pmb{\Sigma}) = rac{1}{(2\pi)^{D/2}|\pmb{\Sigma}|^{1/2}} \exp\left(-rac{1}{2}(\pmb{x}-\mu)^{ op}\pmb{\Sigma}^{-1}(\pmb{x}-\mu)
ight)$$

• Given N i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian

• Multivariate Gaussian in D dimensions

$$p(oldsymbol{x}|\mu,oldsymbol{\Sigma}) = rac{1}{(2\pi)^{D/2}|oldsymbol{\Sigma}|^{1/2}} \exp\left(-rac{1}{2}(oldsymbol{x}-\mu)^{ op}oldsymbol{\Sigma}^{-1}(oldsymbol{x}-\mu)
ight)$$

- Given N i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian
 - ullet MLE for the D imes 1 mean $\mu \in \mathbb{R}^D$

$$\hat{\mu} = rac{1}{N} \sum_{n=1}^{N} x_n$$

• Multivariate Gaussian in D dimensions

$$p(oldsymbol{x}|\mu,oldsymbol{\Sigma}) = rac{1}{(2\pi)^{D/2}|oldsymbol{\Sigma}|^{1/2}} \exp\left(-rac{1}{2}(oldsymbol{x}-\mu)^{ op}oldsymbol{\Sigma}^{-1}(oldsymbol{x}-\mu)
ight)$$

- Given N i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian
 - ullet MLE for the D imes 1 mean $\mu\in\mathbb{R}^D$

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

• MLE for the $D \times D$ p.s.d. covariance matrix Σ

$$\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \hat{\mu}) (\mathbf{x}_n - \hat{\mu})^{\top}$$

• Multivariate Gaussian in D dimensions

$$p(oldsymbol{x}|\mu,oldsymbol{\Sigma}) = rac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left(-rac{1}{2}(oldsymbol{x}-\mu)^{ op}oldsymbol{\Sigma}^{-1}(oldsymbol{x}-\mu)
ight)$$

- Given N i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian
 - \bullet MLE for the $D\times 1$ mean $\mu\in\mathbb{R}^D$

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n$$

• MLE for the $D \times D$ p.s.d. covariance matrix Σ

$$\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \hat{\mu}) (\mathbf{x}_n - \hat{\mu})^{\top}$$

• Note: The "trace trick" simplifies the derivative calculations

$$\underbrace{\boldsymbol{x}^{\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}}_{\text{a scalar}} = \operatorname{trace}(\boldsymbol{x}^{\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}) = \operatorname{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\boldsymbol{x}^{\top})$$



ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$

- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K

- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K
- The conditional p.d.f. of a data point x if it comes from Gaussian k

$$p(x|z=k) = \mathcal{N}(x|\mu_k, \Sigma_k)$$

- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K
- The conditional p.d.f. of a data point x if it comes from Gaussian k

$$p(x|z=k) = \mathcal{N}(x|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Since z is NOT known, we need to look at the marginal p.d.f. of x

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{x}, \mathbf{z} = k) = \sum_{k=1}^{K} p(\mathbf{z} = k) p(\mathbf{x} | \mathbf{z} = k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K
- The conditional p.d.f. of a data point x if it comes from Gaussian k

$$p(\mathbf{x}|\mathbf{z}=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Since z is NOT known, we need to look at the marginal p.d.f. of x

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{x}, \mathbf{z} = k) = \sum_{k=1}^{K} p(\mathbf{z} = k) p(\mathbf{x} | \mathbf{z} = k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Note: Here $p(\mathbf{x})$ means $p(\mathbf{x}|\Theta)$ where $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$



- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K
- The conditional p.d.f. of a data point x if it comes from Gaussian k

$$p(\mathbf{x}|\mathbf{z}=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Since z is NOT known, we need to look at the marginal p.d.f. of x

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{x}, \mathbf{z} = k) = \sum_{k=1}^{K} p(\mathbf{z} = k) p(\mathbf{x} | \mathbf{z} = k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Note: Here $p(\mathbf{x})$ means $p(\mathbf{x}|\Theta)$ where $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
- To learn the GMM parameters Θ , we have to do MLE on p(x)



- ullet Let's do MLE for estimating GMM parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$
- We basically have to estimate K multivariate Gaussians with wts π_1, \ldots, π_K
- The conditional p.d.f. of a data point x if it comes from Gaussian k

$$p(x|z=k) = \mathcal{N}(x|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Since z is NOT known, we need to look at the marginal p.d.f. of x

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) = \sum_{k=1}^{K} p(\mathbf{x}, \mathbf{z} = k) = \sum_{k=1}^{K} p(\mathbf{z} = k) p(\mathbf{x} | \mathbf{z} = k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Note: Here p(x) means $p(x|\Theta)$ where $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$
- To learn the GMM parameters Θ , we have to do MLE on p(x)
- In general, it is not an easy problem due to the difficult form of p(x) (for "why", see the next slide)

$$\mathcal{L} = \log \prod_{n=1}^{N} p(\mathbf{x}_n) = \sum_{n=1}^{N} \log p(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

• Given N observations x_1, \ldots, x_N , the log-likelihood will be

$$\mathcal{L} = \log \prod_{n=1}^{N} \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
params get coupled!

• Due to the coupling of parameters, MLE by simply taking derivatives of \mathcal{L} and setting to zero won't give a closed form solution of $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$

$$\mathcal{L} = \log \prod_{n=1}^{N} \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
params get coupled!

- Due to the coupling of parameters, MLE by simply taking derivatives of \mathcal{L} and setting to zero won't give a closed form solution of $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
 - Gradient based iterative methods can be used

$$\mathcal{L} = \log \prod_{n=1}^{N} \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \rho(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
params get coupled!

- Due to the coupling of parameters, MLE by simply taking derivatives of \mathcal{L} and setting to zero won't give a closed form solution of $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
 - Gradient based iterative methods can be used
 - However, we will use Expectation Maximization (EM) a more general way of solving such problems (i.e., parameter estimation with latent variables)

$$\mathcal{L} = \log \prod_{n=1}^{N} p(\mathbf{x}_n) = \sum_{n=1}^{N} \log p(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Due to the coupling of parameters, MLE by simply taking derivatives of \mathcal{L} and setting to zero won't give a closed form solution of $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
 - Gradient based iterative methods can be used
 - However, we will use Expectation Maximization (EM) a more general way of solving such problems (i.e., parameter estimation with latent variables)
- Note: For problems where z_n is continuous or comes from a combinatorially large space (e.g., it's a binary vector), doing MLE will be even harder!

$$\mathcal{L} = \sum_{n=1}^{N} \log \underbrace{\int_{z_n} p(x_n | z_n) p(z_n) dz_n}_{\text{Ouch! Intractable integral!!!}}$$



• Given N observations x_1, \ldots, x_N , the log-likelihood will be

$$\mathcal{L} = \log \prod_{n=1}^{N} p(\mathbf{x}_n) = \sum_{n=1}^{N} \log p(\mathbf{x}_n) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Due to the coupling of parameters, MLE by simply taking derivatives of \mathcal{L} and setting to zero won't give a closed form solution of $\Theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
 - Gradient based iterative methods can be used
 - However, we will use Expectation Maximization (EM) a more general way of solving such problems (i.e., parameter estimation with latent variables)
- Note: For problems where z_n is continuous or comes from a combinatorially large space (e.g., it's a binary vector), doing MLE will be even harder!

$$\mathcal{L} = \sum_{n=1}^{N} \log \underbrace{\int_{z_n} p(x_n | z_n) p(z_n) dz_n}_{\text{Ouch! Intractable integral!!!}}$$

In such cases, something like EM becomes even more important



• MLE for GMM will be simplified if we "knew" the z_n for each x_n

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\boldsymbol{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\mathbf{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

• With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\mathbf{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" $(z_n \text{ makes } x_n \text{ "complete"})$

- ullet MLE for GMM will be simplified if we "knew" the $oldsymbol{z}_n$ for each $oldsymbol{x}_n$
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\boldsymbol{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" $(z_n \text{ makes } x_n \text{ "complete"})$
 - $p(x_n)$ is known as "incomplete data likelihood"

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N}\log\left(\mathbf{x}_{n}|\boldsymbol{\mu}_{k},\boldsymbol{\Sigma}_{k}\right)$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" (z_n makes x_n "complete")
 - $p(x_n)$ is known as "incomplete data likelihood"
- Since we don't know the "true" z_n , we will have to rely on a "guess" for z_n

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N}\log\left(\sum_{k=1}^{K}\pi_{k}\mathcal{N}(\boldsymbol{x}_{n}|\boldsymbol{\mu}_{k},\boldsymbol{\Sigma}_{k})\right)$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" (z_n makes x_n "complete")
 - $p(x_n)$ is known as "incomplete data likelihood"
- Since we don't know the "true" z_n , we will have to rely on a "guess" for z_n
 - ullet This guess for $oldsymbol{z}_n$ will be based on the current values of params Θ

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\boldsymbol{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" $(z_n \text{ makes } x_n \text{ "complete"})$
 - $p(x_n)$ is known as "incomplete data likelihood"
- Since we don't know the "true" z_n , we will have to rely on a "guess" for z_n
 - ullet This guess for $oldsymbol{z}_n$ will be based on the current values of params Θ
 - Can do MLE on $p(x_n, z_n)$ to re-estimate Θ using these guesses, and repeat

- MLE for GMM will be simplified if we "knew" the z_n for each x_n
- Reason: If z_n is known, the summation over z_n isn't required

$$\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(\boldsymbol{x}_{n} | \boldsymbol{\mu}_{k}, \boldsymbol{\Sigma}_{k})$$

- With z_n "known", we can try doing MLE on $p(x_n, z_n)$, instead of on $p(x_n)$
 - $p(x_n, z_n)$ is known as "complete data likelihood" $(z_n \text{ makes } x_n \text{ "complete"})$
 - $p(x_n)$ is known as "incomplete data likelihood"
- Since we don't know the "true" z_n , we will have to rely on a "guess" for z_n
 - This guess for z_n will be based on the current values of params Θ
 - Can do MLE on $p(x_n, z_n)$ to re-estimate Θ using these guesses, and repeat
 - A more formal view of this iterative procedure is given by the Expectation Maximization (EM)
 algorithm (next lecture)

• The complete data log-likelihood over the N obs.

$$\sum_{n=1}^{N} \log p(x_n, z_n) = \sum_{n=1}^{N} \log p(x_n|z_n)p(z_n) = \sum_{n=1}^{N} \log \prod_{k=1}^{K} [p(x_n|z_n = k)p(z_n = k)]^{z_{nk}}$$
(note that, for each n, only one z_{nk} will be 1)

• The complete data log-likelihood over the *N* obs.

$$\sum_{n=1}^{N} \log p(\mathbf{x}_n, \mathbf{z}_n) = \sum_{n=1}^{N} \log p(\mathbf{x}_n | \mathbf{z}_n) p(\mathbf{z}_n) = \sum_{n=1}^{N} \log \underbrace{\prod_{k=1}^{K} [p(\mathbf{x}_n | \mathbf{z}_n = k) p(\mathbf{z}_n = k)]^{z_{nk}}}_{\text{(note that, for each } n, \text{ only one } z_{nk} \text{ will be 1)}}$$

The above gets further simplified to

$$\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \log p(z_n = k) p(x_n | z_n = k) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

• The complete data log-likelihood over the *N* obs.

$$\sum_{n=1}^{N} \log p(x_n, z_n) = \sum_{n=1}^{N} \log p(x_n|z_n)p(z_n) = \sum_{n=1}^{N} \log \underbrace{\prod_{k=1}^{K} [p(x_n|z_n = k)p(z_n = k)]^{z_{nk}}}_{\text{(note that, for each } n, \text{ only one } z_{nk} \text{ will be 1)}}$$

The above gets further simplified to

$$\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \log p(z_n = k) p(x_n | z_n = k) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

• If we know value of each z_{nk} deterministically, we can plug these in and do MLE on the above objective (which has a simple and separable structure)

• The complete data log-likelihood over the *N* obs.

$$\sum_{n=1}^{N} \log p(x_n, z_n) = \sum_{n=1}^{N} \log p(x_n|z_n)p(z_n) = \sum_{n=1}^{N} \log \underbrace{\prod_{k=1}^{K} [p(x_n|z_n = k)p(z_n = k)]^{z_{nk}}}_{\text{(note that, for each } n, \text{ only one } z_{nk} \text{ will be 1)}}$$

The above gets further simplified to

$$\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \log p(z_n = k) p(x_n | z_n = k) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

- If we know value of each z_{nk} deterministically, we can plug these in and do MLE on the above objective (which has a simple and separable structure)
- What if we don't have a *deterministic* guess for z_{nk} ?

• The complete data log-likelihood over the *N* obs.

$$\sum_{n=1}^{N} \log p(x_n, z_n) = \sum_{n=1}^{N} \log p(x_n|z_n)p(z_n) = \sum_{n=1}^{N} \log \underbrace{\prod_{k=1}^{K} [p(x_n|z_n = k)p(z_n = k)]^{z_{nk}}}_{\text{(note that, for each } n, \text{ only one } z_{nk} \text{ will be 1)}}$$

The above gets further simplified to

$$\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \log p(z_n = k) p(x_n | z_n = k) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

- If we know value of each z_{nk} deterministically, we can plug these in and do MLE on the above objective (which has a simple and separable structure)
- What if we don't have a deterministic guess for z_{nk} ?
 - In such cases, we can use the posterior expectations of the z_{nk} 's (which are basically posterior probabilities of cluster assignments of points to clusters)

$$\mathbb{E}[z_{nk}] = \mathbf{0} \times p(z_{nk} = 0 | \mathbf{x}_n) + 1 \times p(z_{nk} = 1 | \mathbf{x}_n)$$

$$\mathbb{E}[z_{nk}] = 0 \times \rho(z_{nk} = 0 | \mathbf{x}_n) + 1 \times \rho(z_{nk} = 1 | \mathbf{x}_n)$$
$$= \rho(z_{nk} = 1 | \mathbf{x}_n)$$

$$\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0 | x_n) + 1 \times p(z_{nk} = 1 | x_n)$$

$$= p(z_{nk} = 1 | x_n)$$

$$\propto p(z_{nk} = 1) p(x_n | z_{nk} = 1)$$
 (Bayes Rule)

```
\begin{split} \mathbb{E}[\mathbf{z}_{nk}] &= \mathbf{0} \times p(\mathbf{z}_{nk} = \mathbf{0} | \mathbf{x}_n) + 1 \times p(\mathbf{z}_{nk} = 1 | \mathbf{x}_n) \\ &= p(\mathbf{z}_{nk} = 1 | \mathbf{x}_n) \\ &\propto p(\mathbf{z}_{nk} = 1) p(\mathbf{x}_n | \mathbf{z}_{nk} = 1) \quad \text{(Bayes Rule)} \end{split} Thus \mathbb{E}[\mathbf{z}_{nk}] \propto \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad \text{(Posterior prob. of } \mathbf{x}_n \text{ belonging to cluster } k)
```

• Posterior expectations $\mathbb{E}[z_{nk}]$ can be computed using current estimates of Θ

```
\begin{split} \mathbb{E}[z_{nk}] &= 0 \times \rho(z_{nk} = 0 | x_n) + 1 \times \rho(z_{nk} = 1 | x_n) \\ &= \rho(z_{nk} = 1 | x_n) \\ &\propto \rho(z_{nk} = 1) \rho(x_n | z_{nk} = 1) \quad \text{(Bayes Rule)} \end{split} Thus \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \quad \text{(Posterior prob. of } x_n \text{ belonging to cluster } k) \end{split}
```

• The final expression of $\mathbb{E}[z_{nk}]$ makes intuitive sense

$$\begin{array}{rcl} \mathbb{E}[z_{nk}] & = & \mathbf{0} \times p(z_{nk} = \mathbf{0} | \mathbf{x}_n) + 1 \times p(z_{nk} = 1 | x_n) \\ & = & p(z_{nk} = 1 | \mathbf{x}_n) \\ & \propto & p(z_{nk} = 1) p(\mathbf{x}_n | z_{nk} = 1) \qquad \text{(Bayes Rule)} \end{array}$$
 Thus $\mathbb{E}[z_{nk}] \propto & \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{(Posterior prob. of } \boldsymbol{x}_n \text{ belonging to cluster } k)$

- The final expression of $\mathbb{E}[z_{nk}]$ makes intuitive sense
- Note: We can finally normalize $\mathbb{E}[z_{nk}]$ as $\mathbb{E}[z_{nk}] = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$ since $\sum_{k=1}^K \mathbb{E}[z_{nk}] = 1$

$$\begin{array}{rcl} \mathbb{E}[\mathsf{z}_{nk}] & = & \mathbf{0} \times p(\mathsf{z}_{nk} = \mathbf{0} | \mathbf{x}_n) + 1 \times p(\mathsf{z}_{nk} = 1 | \mathbf{x}_n) \\ & = & p(\mathsf{z}_{nk} = 1 | \mathbf{x}_n) \\ & \propto & p(\mathsf{z}_{nk} = 1) p(\mathsf{x}_n | \mathsf{z}_{nk} = 1) \qquad \text{(Bayes Rule)} \end{array}$$
 Thus $\mathbb{E}[\mathsf{z}_{nk}] \propto & \pi_k \mathcal{N}(\mathsf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{(Posterior prob. of } \boldsymbol{x}_n \text{ belonging to cluster } k)$

- The final expression of $\mathbb{E}[z_{nk}]$ makes intuitive sense
- Note: We can finally normalize $\mathbb{E}[z_{nk}]$ as $\mathbb{E}[z_{nk}] = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$ since $\sum_{k=1}^K \mathbb{E}[z_{nk}] = 1$
- Given $\mathbb{E}[z_{nk}]$, we can now define expected complete data log-lik.

ullet Posterior expectations $\mathbb{E}[z_{nk}]$ can be computed using current estimates of Θ

$$\begin{array}{rcl} \mathbb{E}[\mathsf{z}_{nk}] & = & \mathbf{0} \times p(\mathsf{z}_{nk} = \mathbf{0} | \mathbf{x}_n) + 1 \times p(\mathsf{z}_{nk} = 1 | \mathbf{x}_n) \\ & = & p(\mathsf{z}_{nk} = 1 | \mathbf{x}_n) \\ & \propto & p(\mathsf{z}_{nk} = 1) p(\mathsf{x}_n | \mathsf{z}_{nk} = 1) \qquad \text{(Bayes Rule)} \end{array}$$
 Thus $\mathbb{E}[\mathsf{z}_{nk}] \propto & \pi_k \mathcal{N}(\mathsf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{(Posterior prob. of } \boldsymbol{x}_n \text{ belonging to cluster } k)$

- The final expression of $\mathbb{E}[z_{nk}]$ makes intuitive sense
- Note: We can finally normalize $\mathbb{E}[z_{nk}]$ as $\mathbb{E}[z_{nk}] = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$ since $\sum_{k=1}^K \mathbb{E}[z_{nk}] = 1$
- Given $\mathbb{E}[z_{nk}]$, we can now define expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[\mathbf{z}_{nk}] [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

.. and do MLE for the parameters Θ using this as the objective function



• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$, the expected complete data log-lik.

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \frac{\partial}{\partial \boldsymbol{\mu}_k} \sum_{n=1}^N \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \boldsymbol{\mu}_k \text{ can be ignored)}$$

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \frac{\partial}{\partial \mu_k} \sum_{i=1}^{N} \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \mu_k, \mathbf{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \mu_k \text{ can be ignored)}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}_k} = \frac{\partial}{\partial \mathbf{\Sigma}_k} \sum_{n=1}^N \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \mathbf{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \mathbf{\Sigma}_k \text{ can be ignored)}$$

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

• Taking derivatives w.r.t. μ_k and Σ_k , $\forall k = 1, ..., K$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \frac{\partial}{\partial \boldsymbol{\mu}_k} \sum_{i=1}^N \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \boldsymbol{\mu}_k \text{ can be ignored)}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}_k} = \frac{\partial}{\partial \mathbf{\Sigma}_k} \sum_{i=1}^{N} \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \mathbf{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \mathbf{\Sigma}_k \text{ can be ignored)}$$

• For each k, it's a "weighted" version of the MLE problem for the multivariate Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, given observations $\{\boldsymbol{x}_n\}_{n=1}^N$ with weights $\{\gamma_{nk}\}_{n=1}^N$

• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_k} = \frac{\partial}{\partial \boldsymbol{\mu}_k} \sum_{i=1}^N \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \boldsymbol{\mu}_k \text{ can be ignored)}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}_k} = \frac{\partial}{\partial \mathbf{\Sigma}_k} \sum_{n=1}^N \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \mathbf{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \mathbf{\Sigma}_k \text{ can be ignored)}$$

- For each k, it's a "weighted" version of the MLE problem for the multivariate Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, given observations $\{\boldsymbol{x}_n\}_{n=1}^N$ with weights $\{\gamma_{nk}\}_{n=1}^N$
- Can also solve for π_k likewise (subject to contraint $\sum_{k=1}^K \pi_k = 1$)



• Given $\mathbb{E}[z_{nk}] = \gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{\ell=1}^K \pi_\ell \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)}$, the expected complete data log-lik.

$$\mathcal{L} = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]$$

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \frac{\partial}{\partial \mu_k} \sum_{i=1}^{N} \gamma_{nk} \log \mathcal{N}(\mathbf{x}_n | \mu_k, \mathbf{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \mu_k \text{ can be ignored)}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}_k} = \frac{\partial}{\partial \boldsymbol{\Sigma}_k} \sum_{n=1}^N \gamma_{nk} \log \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = 0 \qquad \text{(note: constants w.r.t. } \boldsymbol{\Sigma}_k \text{ can be ignored)}$$

- For each k, it's a "weighted" version of the MLE problem for the multivariate Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, given observations $\{\boldsymbol{x}_n\}_{n=1}^N$ with weights $\{\gamma_{nk}\}_{n=1}^N$
- Can also solve for π_k likewise (subject to contraint $\sum_{k=1}^K \pi_k = 1$)
- Derivations are a bit tedious (but straightforward). I will provide a note.



GMM Parameter Update Equations

ullet The final expressions for updates of $\{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$

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

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

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

GMM Parameter Update Equations

ullet The final expressions for updates of $\{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$

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

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

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

- Note: $N_k = \sum_{n=1}^N \gamma_{nk}$ is the effective num. of pts. assigned to Gaussian k
- Update equations make intuitive sense

GMM Parameter Update Equations

ullet The final expressions for updates of $\{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$

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

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

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

- Note: $N_k = \sum_{n=1}^N \gamma_{nk}$ is the effective num. of pts. assigned to Gaussian k
- Update equations make intuitive sense
- Also note that each point \mathbf{x}_n contributes to each $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ but fractionally (based on the values of γ_{nk})

The Full Algorithm for Learning GMM

• Initialize the parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ randomly, or using K-means

The Full Algorithm for Learning GMM

- ullet Initialize the parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(x|\Theta)$ ceases to increase)

- ullet Initialize the parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(x|\Theta)$ ceases to increase)
 - Given Θ , compute each expectation z_{nk} (post. prob. of $z_{nk} = 1$), $\forall n, k$

- Initialize the parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(x|\Theta)$ ceases to increase)
 - Given Θ , compute each expectation z_{nk} (post. prob. of $z_{nk} = 1$), $\forall n, k$

$$\gamma_{nk} = \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (and re-normalize s.t. $\sum_{k=1}^K \gamma_{nk} = 1$)

- Initialize the parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(\mathbf{x}|\Theta)$ ceases to increase)
 - Given Θ , compute each expectation z_{nk} (post. prob. of $z_{nk} = 1$), $\forall n, k$

$$\gamma_{nk} = \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (and re-normalize s.t. $\sum_{k=1}^K \gamma_{nk} = 1$)

• Given $\gamma_{nk}=\mathbb{E}[z_{nk}]$ and $N_k=\sum_{n=1}^N\gamma_{nk}$, update $\Theta=\{\pi_k, \pmb{\mu}_k, \pmb{\Sigma}_k\}_{k=1}^K$ as

- Initialize the parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(\mathbf{x}|\Theta)$ ceases to increase)
 - Given Θ , compute each expectation z_{nk} (post. prob. of $z_{nk} = 1$), $\forall n, k$

$$\gamma_{nk} = \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (and re-normalize s.t. $\sum_{k=1}^K \gamma_{nk} = 1$)

• Given $\gamma_{nk}=\mathbb{E}[z_{nk}]$ and $N_k=\sum_{n=1}^N\gamma_{nk}$, update $\Theta=\{\pi_k, \mu_k, \mathbf{\Sigma}_k\}_{k=1}^K$ as

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} \mathbf{x}_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^{\top}$$

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

- ullet Initialize the parameters $\Theta = \{\pi_k, oldsymbol{\mu}_k, oldsymbol{\Sigma}_k\}_{k=1}^K$ randomly, or using K-means
- Iterate until convergence (e.g., when $\log p(x|\Theta)$ ceases to increase)
 - Given Θ , compute each expectation z_{nk} (post. prob. of $z_{nk} = 1$), $\forall n, k$

$$\gamma_{nk} = \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
 (and re-normalize s.t. $\sum_{k=1}^K \gamma_{nk} = 1$)

• Given $\gamma_{nk}=\mathbb{E}[z_{nk}]$ and $N_k=\sum_{n=1}^N\gamma_{nk}$, update $\Theta=\{\pi_k, \pmb{\mu}_k, \pmb{\Sigma}_k\}_{k=1}^K$ as

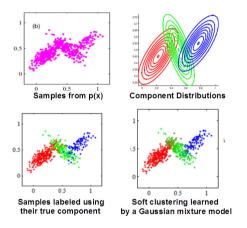
$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} \mathbf{x}_n$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k) (\mathbf{x}_n - \boldsymbol{\mu}_k)^{\top}$$

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

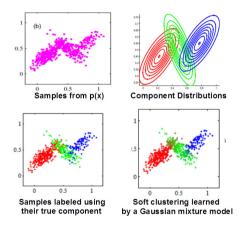
(It's basically an Expectation Maximization (EM) algorithm for learning GMM. We will look at EM more formally in the next class.)

Illustration of GMM Clustering



Notice the "mixed" colored points in the overlapping regions in the final clustering

Illustration of GMM Clustering

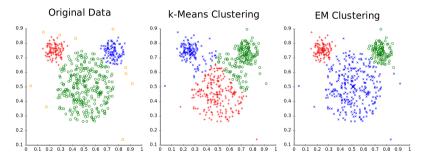


Notice the "mixed" colored points in the overlapping regions in the final clustering

Also check out this demo of GMM: https://www.youtube.com/watch?v=B36fzChfyGU

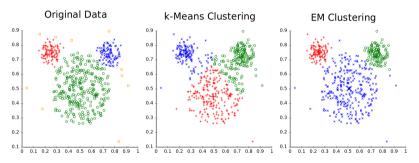
GMM vs *K*-means

For the GMM clustering (rightmost figure), the most probable cluster for each point has been labeled



GMM vs *K*-means

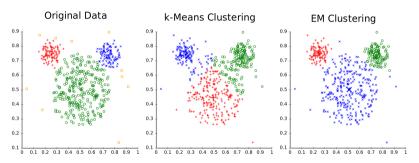
For the GMM clustering (rightmost figure), the most probable cluster for each point has been labeled



Note that K-means, unlike GMM, tends to learn equi-sized clusters.

GMM vs *K*-means

For the GMM clustering (rightmost figure), the most probable cluster for each point has been labeled



Note that K-means, unlike GMM, tends to learn equi-sized clusters.

GMM with $\Sigma_k = I$ and $\pi_k = 1/K$, and soft assignments converted to hard assign. (setting the largest prob. to 1, rest to 0), is equivalent to K-means.

• Other types of component distributions can be used

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

Also used in supervised learning problems (mixture of experts)

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$
 - Experts and points-to-experts assignments can be learned iteratively as in GMM

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$
 - Experts and points-to-experts assignments can be learned iteratively as in GMM
- Can be used for performing generative classification (e.g., naïve Bayes)

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$
 - Experts and points-to-experts assignments can be learned iteratively as in GMM
- Can be used for performing generative classification (e.g., naïve Bayes)

$$p(y_n = k | \mathbf{x}_n) \propto p(y_n = k) p(\mathbf{x}_n | y_n = k)$$
 (cluster ids are the known classes)

- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$
 - Experts and points-to-experts assignments can be learned iteratively as in GMM
- Can be used for performing generative classification (e.g., naïve Bayes)

$$p(y_n = k | \mathbf{x}_n) \propto p(y_n = k) p(\mathbf{x}_n | y_n = k)$$
 (cluster ids are the known classes)

p(y = k) and p(x|z = k) can be efficiently estimated using training data



- Other types of component distributions can be used
- Sequence data models, such as HMM, can also be seen as mixture models

$$p(\mathbf{z}_n|\phi) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$
 (Current cluster/state depends on previous)

- Also used in supervised learning problems (mixture of experts)
 - For data x_n , first choose one of K experts, and then use that expert's predictive model for $p(y_n|x_n)$
 - Experts and points-to-experts assignments can be learned iteratively as in GMM
- Can be used for performing generative classification (e.g., naïve Bayes)

$$p(y_n = k | \mathbf{x}_n) \propto p(y_n = k) p(\mathbf{x}_n | y_n = k)$$
 (cluster ids are the known classes)

p(y = k) and p(x|z = k) can be efficiently estimated using training data

• Number of clusters (K) in a mixture model can be learned from data using nonparametric Bayesian methods (e.g., "infinite" mixture models)

Next Class

- The general Expectation Maximization (EM) algorithm
- Generative models for dimensionality reduction
 - Factor Analysis and Probabilistic PCA (and extensions)
 - EM based parameter estimation for these models