

3D Point Clouds

Lecture 3 – Clustering

主讲人 黎嘉信

Aptiv 自动驾驶
新加坡国立大学 博士
清华大学 本科





1、 Math Prerequisite



2、 K-Means



3、 Gaussian Mixture Models (GMM)



4、 Expectation Maximization (EM)



5、 Spectral Clustering



6、 Others – Mean Shift, DBSCAN



Introduction Clustering



Clustering is to group objects such that objects in the same group (called a cluster) are similar, while objects belong to different groups are dis-similar.

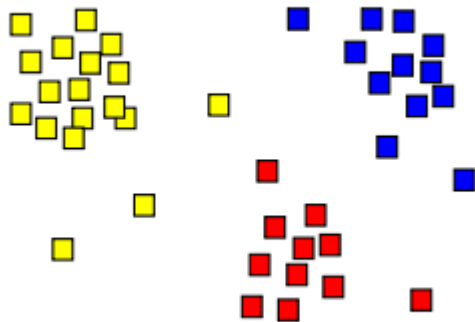


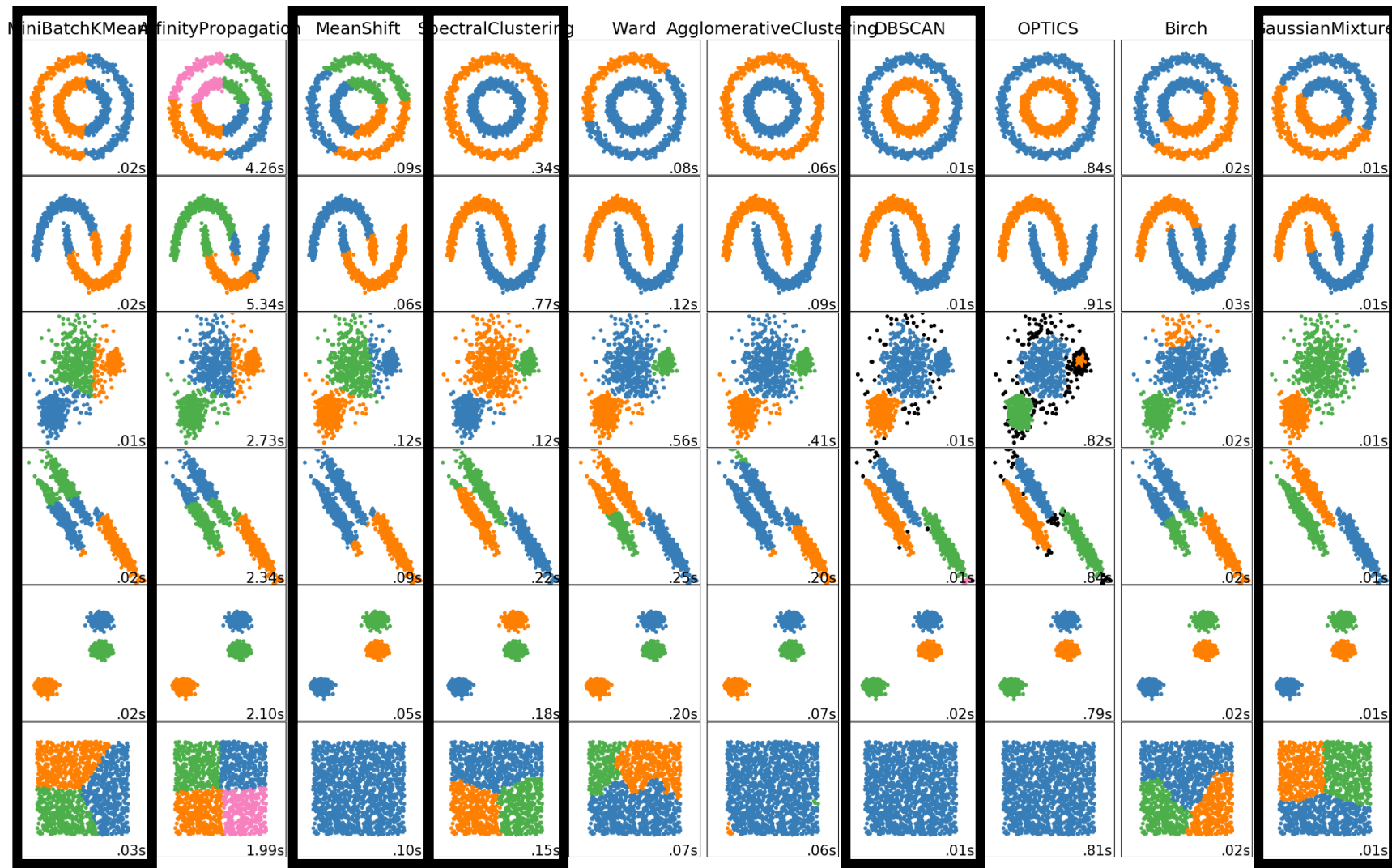
There are many clustering algorithms



We focus on two types:

- Theoretically profound
 - K-Means,
 - GMM (Gaussian Mixture Model)
 - EM (Expectation-Maximation)
 - Spectral Clustering
- Practically useful
 - Mean-Shift
 - DBSCAN (Density-Based Spatial Clustering of Applications with Noise)







Math Prerequisite



Need some background knowledge to understand K-Means, GMM, EM, Spectral Clustering

- Linear Algebra
 - SVD
 - Rayleigh Quotient
- Probability
 - Bayes rule
 - Conditional Probability
- Graphical Modeling
 - Directed graph
 - Undirected graph
- Lagrange Multiplier



Spectral Theorem

Let $A \in R^{n,n}$ be symmetric, and $\lambda_i \in R, i = 1, 2, \dots, n$ be the eigenvalues of A . There exists a set of orthonormal vectors $u_i \in R_n, i = 1, 2, \dots, n$, such that $Au_i = \lambda_i u_i$. Equivalently, there exists an orthogonal matrix $U = [u_1, \dots, u_n]$ (i.e., $UU^T = U^T U = I_n$), such that,

$$A = U\Lambda U^T = \sum_{i=1}^n \lambda_i u_i u_i^T, \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$$



Given a symmetric matrix $A \in S^n$,

Physical meaning of SVD!

$$\lambda_{\min}(A) \leq \frac{x^T A x}{x^T x} \leq \lambda_{\max}(A), \forall x \neq 0$$

$$\lambda_{\max}(A) = \max_{x: \|x\|_2=1} x^T A x$$

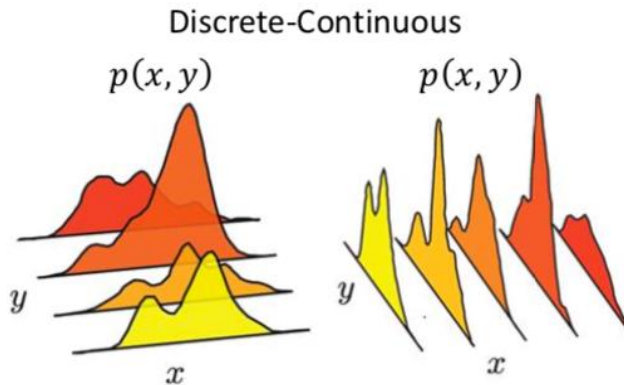
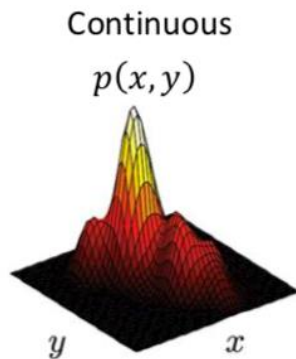
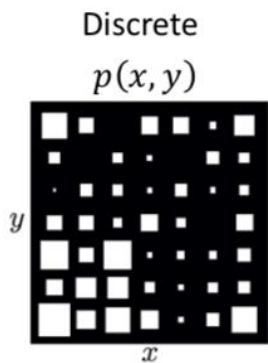
$$\lambda_{\min}(A) = \min_{x: \|x\|_2=1} x^T A x$$

The maximum and minimum are attained for $x = u_1$ and for $x = u_n$, respectively, where u_1 and u_n are the largest and smallest eigenvector of A , respectively.



Probability – Joint Probability

- Consider all combinations events of two random variables X and Y
- This is captured in the *joint probability* distribution $p(x, y)$
- Read as “*probability of X and Y* ”
- Can be more than two random variables, i.e., $p(a, b, c, \dots)$





Probability - Marginalization



Recover probability distribution of any variable in a joint distribution by integrating (or summing) over all other variables.

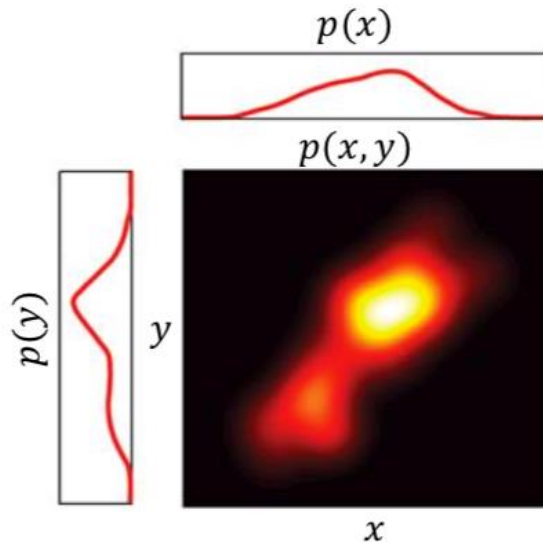


“sum rule” of probability

Continuous:

$$p(x) = \int p(x, y) dy$$

$$p(y) = \int p(x, y) dx$$



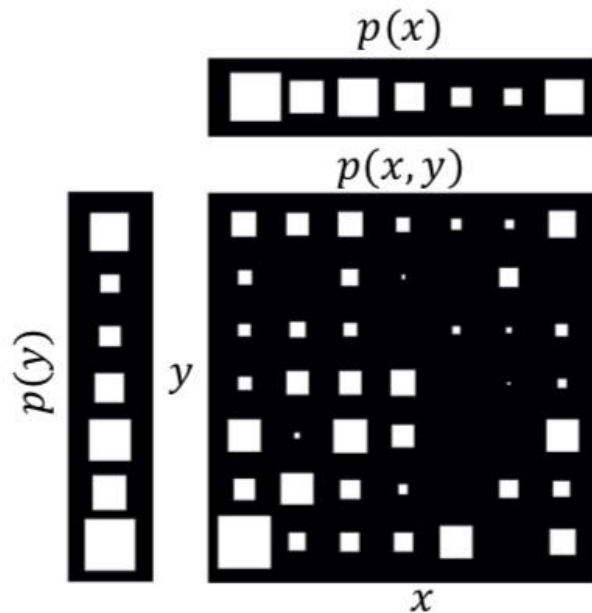


Probability - Marginalization

Discrete:

$$p(x) = \sum_y p(x, y)$$

$$p(y) = \sum_x p(x, y)$$

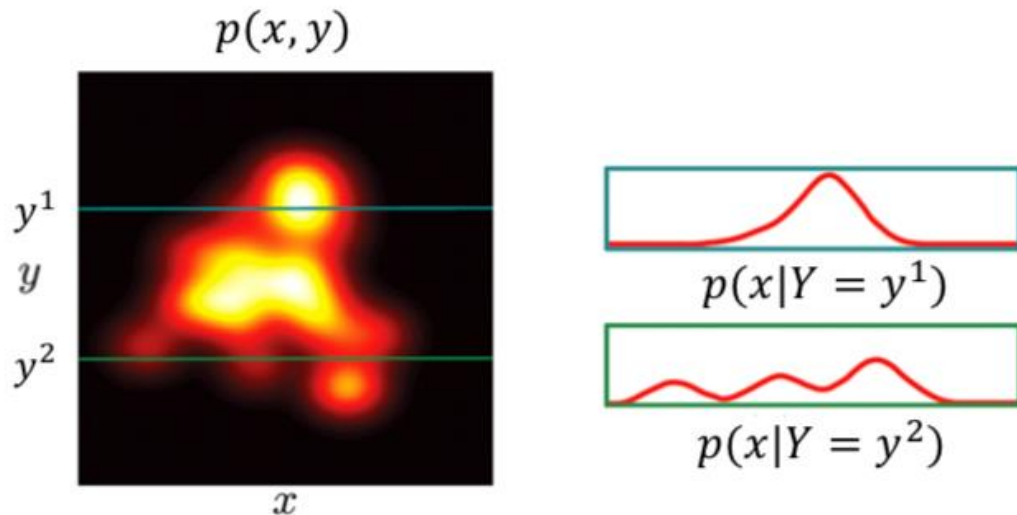




Probability – Conditional Probability



$p(x|Y = y^*)$: probability of X given $Y = y^*$





Probability – Conditional Probability

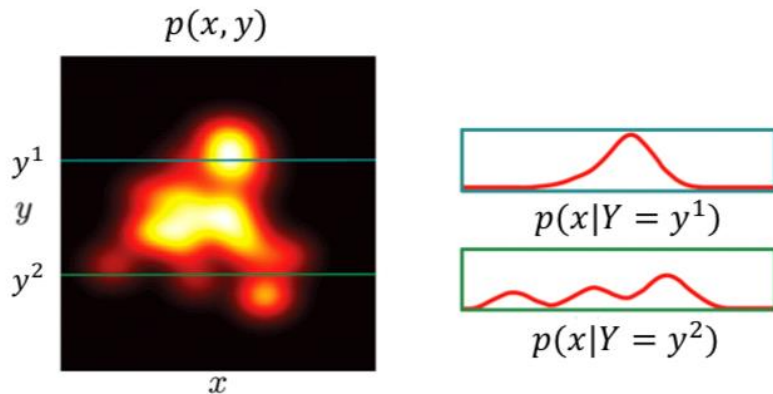


Conditional probability can be extracted from joint probability



Extract appropriate slice and **normalize** (sum to 1)

$$P(x|Y = y^*) = \frac{p(x, Y = y^*)}{\int p(x, Y = y^*)dx} = \frac{p(x, Y = y^*)}{p(Y = y^*)}$$





Probability – Conditional Probability



Conditional probability:

$$P(x|Y = y^*) = \frac{p(x, Y = y^*)}{\int p(x, Y = y^*)dx} = \frac{p(x, Y = y^*)}{p(Y = y^*)}$$




Can be written in compact form:

$$p(x|y) = \frac{p(x, y)}{p(y)}$$



Which leads to:

$$\begin{aligned} p(x, y) &= p(x|y)p(y) \\ p(x, y) &= p(y|x)p(x) \end{aligned}$$


Hence, the name “product rule”!



Probability – Conditional Probability



The product rule works for higher dimensions as well

$$\begin{aligned} p(w, x, y, z) &= p(w, x, y|z)p(z) \\ &= p(w, x|y, z)p(y|z)p(z) \\ &= p(w|x, y, z)p(x|y, z)p(y|z)p(z) \end{aligned}$$



Graphical Modeling - Directed Graphical Model (DGM)



DGM is used to represent conditional independence



DGM is represented by $G(V, E)$

V is a set of nodes. One node is one variable.

E is a set of **oriented** edges. One edge is one conditional relationship.





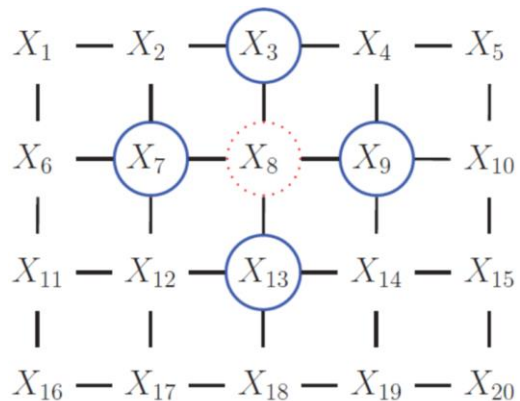
Markov assumption – a random variable X is **independent of its non-descendants given its parents**

- E.g., the joint distribution is $p(x, z) = p(z)p(x|z)$
- E.g., $z \rightarrow$ “age of a person”, $x \rightarrow$ “hair white or black”
 $p(x|z)$ means the probability of white/black hair given a person’s age
It is different to $p(x)$, which means the probability of white/black hair without other info



Graphical Modeling - Undirected Graphical Model (UGM)

-  It is called **Markov Random Field (MRF)** or **Markov Network** as well
-  UGM is represented by $G(V, E)$, where V is a set of nodes, E is a set of **undirected** edges
 - E.g., below can be a map, X_i are cities, the edges are roads.
 - E.g., with UGM, I know the shortest path from X_1 to X_8





Optimization – Lagrange multiplier



Consider the optimization problem

$$\max f(x, y), \text{ s.t. : } g(x, y) = 0$$

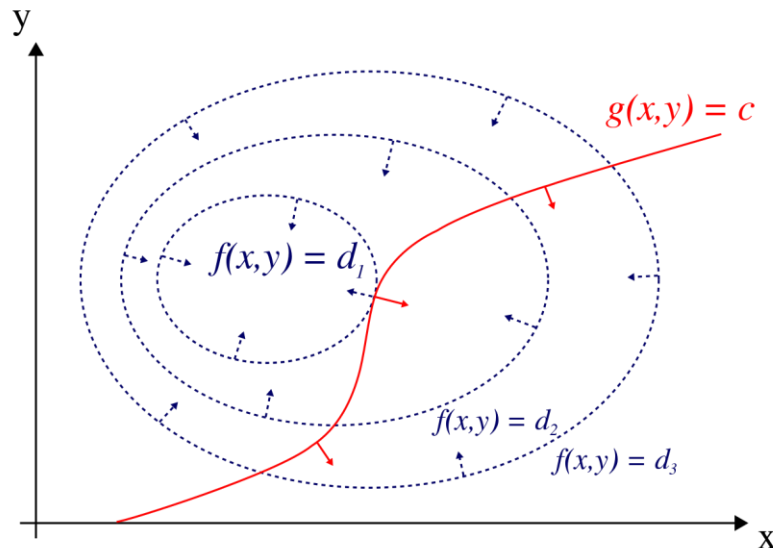


Assume that both f and g have continuous first partial derivatives

Intuition

- The circles are level sets of $f(x, y)$
- $g(x, y) = 0$ means x, y have to be on the red line
- At a (local) maximum, $f(x, y)$ can NOT be increasing along the direction (tangent) of $g(x, y) = 0$

$$\nabla_{x,y} f = \lambda \nabla_{x,y} g$$





Optimization – Lagrange multiplier



Introduce a new variable λ called *Lagrange multiplier*



Let's study the Lagrange function:

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda g(x, y)$$



And solve:

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = 0.$$



The above equation means

(1) $g(x, y) = 0$

(2) gradient of f and g are parallel

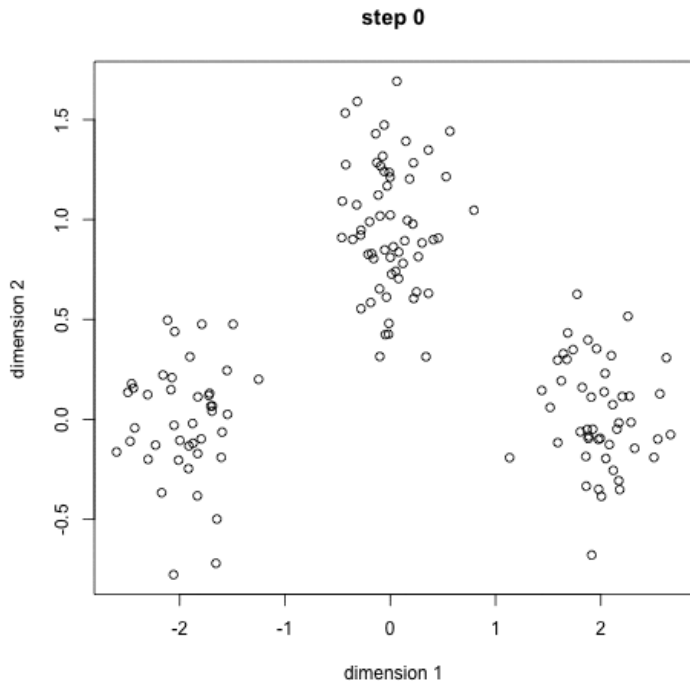
$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = 0 \iff \begin{cases} \nabla_{x,y} f(x, y) = \lambda \nabla_{x,y} g(x, y) \\ g(x, y) = 0 \end{cases}$$



K-Means

N input data points, find K clusters.

1. Randomly select K center points
2. Each data point is assigned to one of the K centers.
3. Re-compute the K centers by the mean of each group
4. Iterate step 2 & 3.





K-Means - Definition



Data set $\{x_1, \dots, x_N\}, x_n \in \mathbb{R}^D$



Cluster center $\mu_k, k = 1, \dots, K$

- μ_k is the center of k^{th} cluster



Binary variable $r_{nk} \in \{0, 1\}$

- x_n belong to which cluster



Objective of K-Means is to minimize **the distortion measure**

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



K-Means



Iteratively optimize r_{nk} and μ_k



For some fixed values for μ_k , minimize J with respect to r_{nk}

- This is the E (expectation) step of the EM algorithm



For some fixed values of r_{nk} , minimize J with respect to μ_k

- This is the M (maximization) step of the EM algorithm

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



K-Means E step

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



N data points are independent, so we can optimize for each n separately.



Simply assign the n^{th} data point to the closest cluster center, which will minimize $\|x_n - \mu_k\|^2$



Formally

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$



K-Means M step

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



With r_{nk} fixed, the objective function J is a quadratic function of μ_k



Compute its first order derivative and make it to 0



Consider each center μ_k separately

$$2 \sum_{n=1}^N r_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k) = 0$$

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$



K-Means



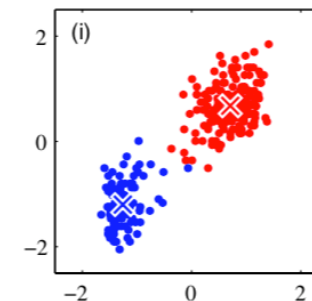
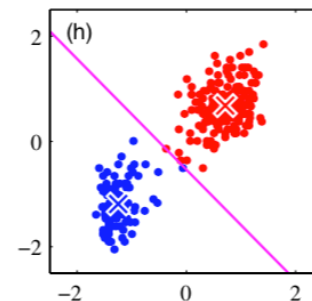
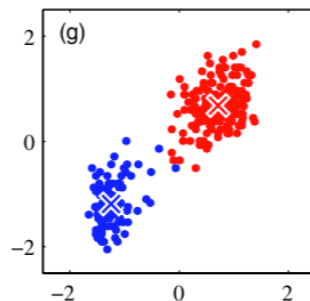
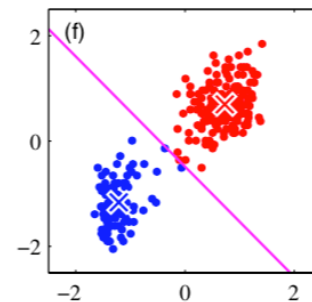
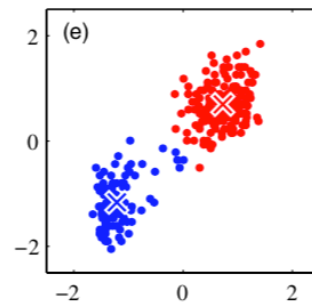
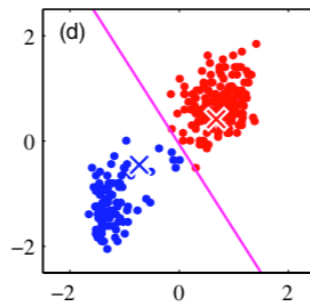
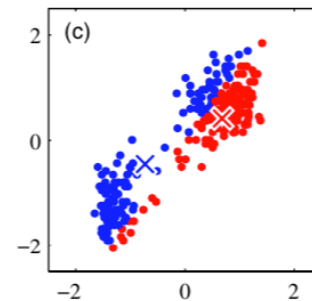
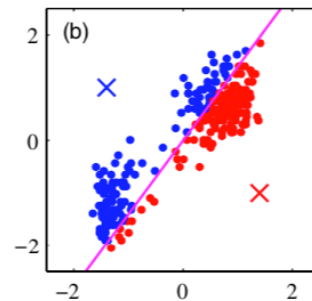
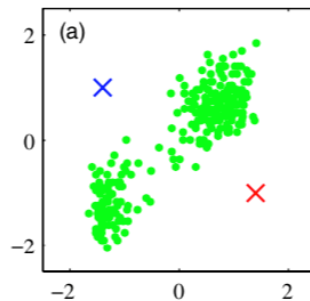
Illustration of the K=2

a) Initialization

b) Expectation – find r_{nk}

c) Maximization – find μ_k

Other figures are repeating (b)(c)





K-Means Practical Tricks



Randomly select K points as the initial μ_k



Run K-Means multiple times with random initialization, select the one with the lowest cost J



In E-step, it is necessary to compute distance between x_n and μ_k

- Nearest Neighbor algorithms in Lecture 2



Mini-batch K-Means: In each E&M iteration, select a subset of the data points.

- Reduce the compute time
- Converges faster
- The final result may be slightly worse.



K-Means: Sequential Update



The E / M step are performed for each data point x_n

1. For each data point x_n , find the nearest center μ_k^{old}
2. Update the center into

$$\mu_k^{new} = \mu_k^{old} + \eta_n(\mathbf{x}_n - \mu_k^{old})$$

where η_n is the learning rate



Mini-batch version of this is the mini-batch K-Means mentioned in previous slice



K-Medoids



Standard K-Means with Euclidean distance is problematic in some cases.

- Outlier – outliers will drag the centers away
- Limits the type of data points – E.g., Euclidean distance is not for categorical labels



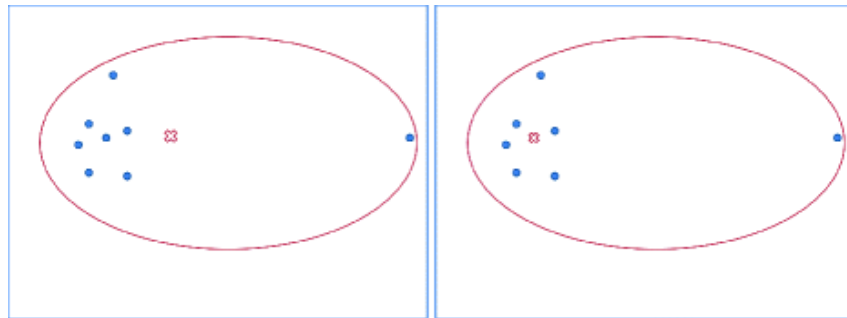
Apply general dissimilarity metric

K-Medoids

$$\tilde{J} = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \mathcal{V}(\mathbf{x}_n, \boldsymbol{\mu}_k)$$

K-Means

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



(a) Mean

(b) Medoid



K-Medoids



E step

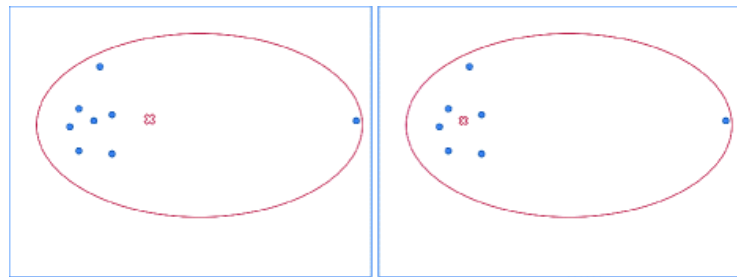
- Same, just need to evaluate the dissimilarity measure $V(\cdot, \cdot)$ instead of Euclidean distance
- $O(KN)$ complexity



M step

- First order derivative over $V(\cdot, \cdot)$ can be intractable
- Therefore, assign μ_k to be one of the data point that has smallest \tilde{J}
- Can be implemented as discrete search over the N_k points of that cluster
- $O(N_k^2)$ complexity

$$\tilde{J} = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \mathcal{V}(\mathbf{x}_n, \boldsymbol{\mu}_k)$$



(a) Mean

(b) Medoid



K-Means Application – Compression



Illustrated as image compression, can be easily applied to point cloud (XYZRGB) compression

1. A $H \times W \times 3$ image is represented as $N \times 3$ points
2. Cluster them into K clusters
3. Now the $N \times 3$ points can be represented by N labels.



Similarly, a [XYZRGB] point cloud can be compressed into [XYZL], where L is the cluster label

$K = 2$



$K = 3$



$K = 10$



Original image





K-Means Compression



Original Image have N pixels, 3 channels

- Each channel is 8bits
- In total $24N$ bits



Compressed Image have N labels + cluster centers

- N labels $\rightarrow \lceil N \log_2 K \rceil$ bits (Note that M bits can represent 2^M integers)
- Cluster centers $\rightarrow 24K$ bits
 - K clusters, each is [R,G,B]
- In total $\lceil N \log_2 K \rceil + 24K$ bits



$K \ll N$ therefore compression is effective



K-Means – Limitations

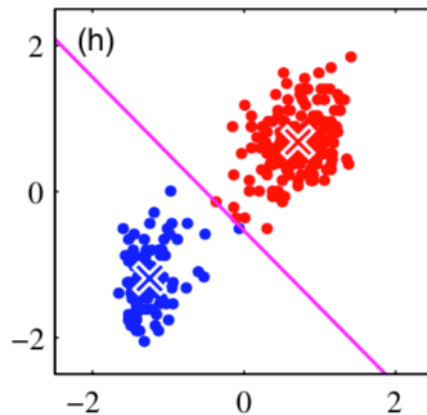


Hard assignment, i.e., $\mu_k \in \{0, 1\}$ is binary label.

- Sometimes a data point lies around the border line between centers
- Lack of uncertainty measure for the assignments



K is unknown





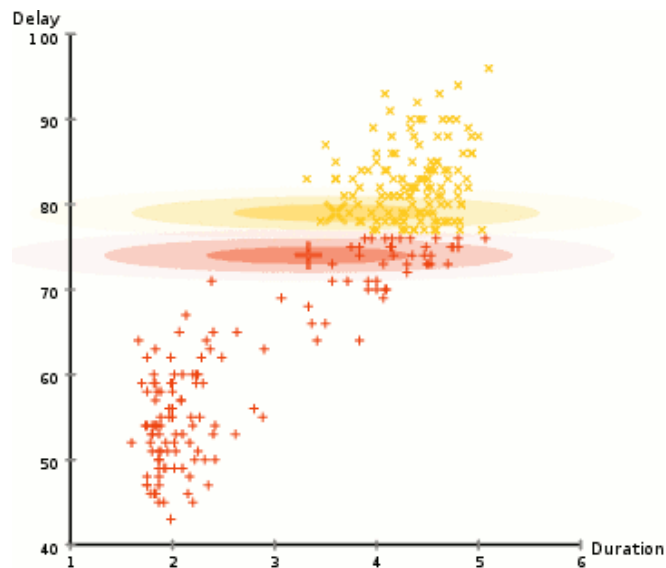
Gaussian Mixture Model (GMM)



Represent a cluster by a Gaussian Distribution $\mathcal{N}(\mu, \sigma)$



GMM tells the probability of a point belonging to each cluster





Gaussian Distribution



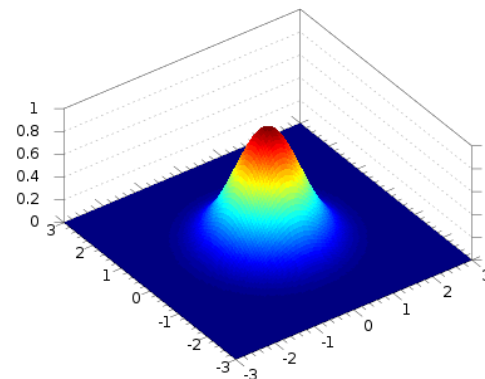
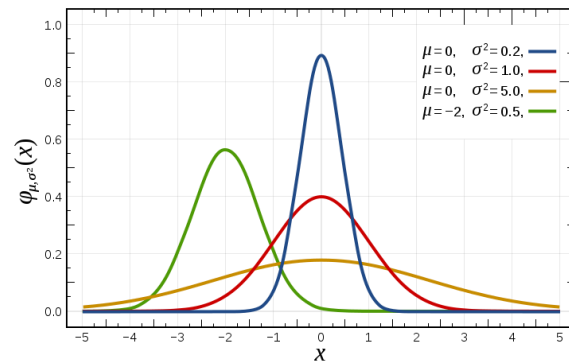
Single variable Gaussian

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp \left\{ -\frac{1}{2\sigma^2}(x - \mu)^2 \right\}$$



Multivariate Gaussian with D-dimensional vector

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) \right\}$$





GMM

- Formally, a Gaussian mixture distribution can be written as a **linear combination** of single Gaussians $\mathcal{N}(\mu_k, \Sigma_k)$ using weights π_k

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \Sigma_k)$$

- The $\{\mu_k, \Sigma_k, \pi_k\}, k = 1, \dots, K$ describes the GMM built from the data points. In another word, we can "generate" points using this $p(x)$
- But we want clustering! That is, the probability of each data point belongs to each single Gaussian $\mathcal{N}(\mu_k, \Sigma_k)$



GMM



Let's introduce a K-dimensional binary variable z having a *1-of-K* representation.

$$z_k \in \{0, 1\}, \quad \sum_k z_k = 1$$



This $p(z_k = 1)$ is the prior probability of Gaussian distribution $\mathcal{N}(\mu_k, \Sigma_k)$

$$p(z_k = 1) = \pi_k$$



The conditional **probability $p(z|x)$** is the clustering “label probability” we want, given a data point x

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}, \quad p(x) = \sum_z p(z)p(x|z)$$



GMM



Gaussian Mixture Model is represented by a graphical model, where the joint distribution $p(x, z) = p(z)p(x|z)$



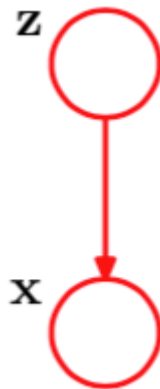
$p(z_k = 1) = \pi_k$, $z = [z_1, \dots, z_k, \dots, z_K]$, 1-of-K representation

Constraints: $0 \leq \pi_k \leq 1$, $\sum_{k=1}^K \pi_k = 1$



Alternatively,

$$p(z) = \prod_{k=1}^K \pi_k^{z_k}$$





GMM



$p(x|z_k = 1)$ is a single Gaussian distribution

$$p(x|z_k = 1) = \mathcal{N}(x|\mu_k, \Sigma_k)$$



Alternatively,

$$p(x|z) = \prod_{k=1}^K \mathcal{N}(x|\mu_k, \Sigma_k)^{z_k}$$



The marginal distribution of x is given by $p(x) = \sum_z p(x, z)$

$$p(x) = \sum_z p(z)p(x|z) = \sum_{k=1}^K \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$





GMM



Now we can get the posterior $p(\mathbf{z}|\mathbf{x})$

- Given a data point \mathbf{x} , which cluster it belongs to?



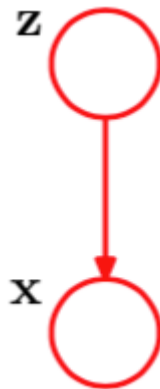
Denote $p(z_k = 1|\mathbf{x})$ as $\gamma(z_k)$



Use Bayes' rule and Total Probability Rule

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{z}|\mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{z})p(\mathbf{x}|\mathbf{z})}{\sum_{j=1}^K p(\mathbf{x}|z_j)p(z_j)}$$

$$\begin{aligned}\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) &= \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^K p(z_j = 1)p(\mathbf{x}|z_j = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}\end{aligned}$$





GMM – Maximum Likelihood (MLE)



Now we know how to get $\gamma(z_k)$, given GMM parameters $\{\pi_k, \mu_k, \Sigma_k\}$



But how do we estimate $\{\pi_k, \mu_k, \Sigma_k\}, \gamma(z_k)$ given data points $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$



Maximum likelihood!

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

$$\ln p(\mathbf{X} | \pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$



GMM – Maximum Likelihood (MLE)



Maximum likelihood formulation suffers from **singularity** problem



Assume that the GMM has the covariance matrix represented as $\Sigma_k = \sigma_k^2 I$, where I is identity matrix



Assume that there is a data point $x_n = \mu_j$, the likelihood contributed by x_n is:

$$\mathcal{N}(x_n | \mu_j, \sigma_j^2 \mathbf{I}) = \frac{1}{(2\pi)^{1/2}} \frac{1}{\sigma_j}$$



What if σ_j is very small?



So in GMM maximum likelihood formulation, we can **put a Gaussian over a single** point, and this leads to very large likelihood.



GMM – Maximum Likelihood (MLE)



There are methods to avoid singularity



Tricks

- If a Gaussian component is collapsing into very small Σ_k , reset its mean μ_k to randomly chosen values, and Σ_k to a large value



Systematic methods

- Maximum-a-Posterior (MAP) – Add prior constraints to the parameters
- Bayesian approach – Prior and proper uncertainty estimation of the parameters



Solve the MLE



Maximum Likelihood Formulation of GMM

$$\pi, \mu, \Sigma = \arg \max_{\pi, \mu, \Sigma} \ln p(\mathbf{X} | \pi, \mu, \Sigma) = \arg \max_{\pi, \mu, \Sigma} \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$



Solve π, μ, Σ iteratively, fixed the others, solve one



Let's look at μ_k first



Compute the first order derivatives with respect to μ_k

$$0 = - \sum_{n=1}^N \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}}_{\gamma(z_{nk})} \Sigma_k (\mathbf{x}_n - \mu_k)$$



Solve the MLE



Here N_k can be interpreted as **the effective number of points assigned to cluster k**



μ_k is the weighted average of all point in the data set



The weight is the posterior probability $\gamma(z_{nk})$



The denominator is the effective number N_k

$$0 = - \sum_{n=1}^N \gamma(z_{nk}) \Sigma_k (\mathbf{x}_n - \mu_k)$$

--

--



Solve the MLE



Compute the first order derivatives with respect to Σ_k , we can solve Σ_k similarly,

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T$$



Σ_k is the **weighted average** of all point's variance centered by μ_k



The weight is the **posterior probability** $\gamma(z_{nk})$



The denominator is the effective number N_k



Solve the MLE

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^N \ln \left\{ \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$



Now let's solve for π_k



However, there is a constraint $\sum_{k=1}^K \pi_k = 1$



Lagrange Multiplier λ

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) + \lambda \left(\sum_{k=1}^K \pi_k - 1 \right)$$



Compute first order derivate with respect to π_k and make it 0

$$0 = \sum_{n=1}^N \frac{\mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)} + \lambda$$



Solve the MLE



Solve λ first

$$0 = \sum_{n=1}^N \frac{\mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)} + \lambda$$

$$0 = \sum_{k=1}^K \sum_{n=1}^N \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)} + \sum_{k=1}^K \pi_k \lambda$$

$$0 = \left(\sum_{n=1}^N 1 \right) + \lambda$$

$$\lambda = -N$$

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}$$

$$N_k = \sum_{n=1}^N \gamma(z_{nk})$$



Solve the MLE



Solve π_k



Intuitively, π_k is the effective cluster member number over data set size.

$$0 = \sum_{n=1}^N \frac{\mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)} + \lambda$$

$$0 = \frac{1}{\pi_k} \sum_{n=1}^N \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)} - N$$

$$0 = \frac{1}{\pi_k} \sum_{n=1}^N \gamma_{nk} - N$$

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

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}$$

$$N_k = \sum_{n=1}^N \gamma(z_{nk})$$



Summary for GMM MLE

1. Initialize the means μ_k , covariances Σ_k and weights π_k
2. **E-step.** Evaluate the posterior $p(z_{nk} = 1|x_n)$, intuitively this is the probability of x_n being assigned to each of the K clusters.

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$



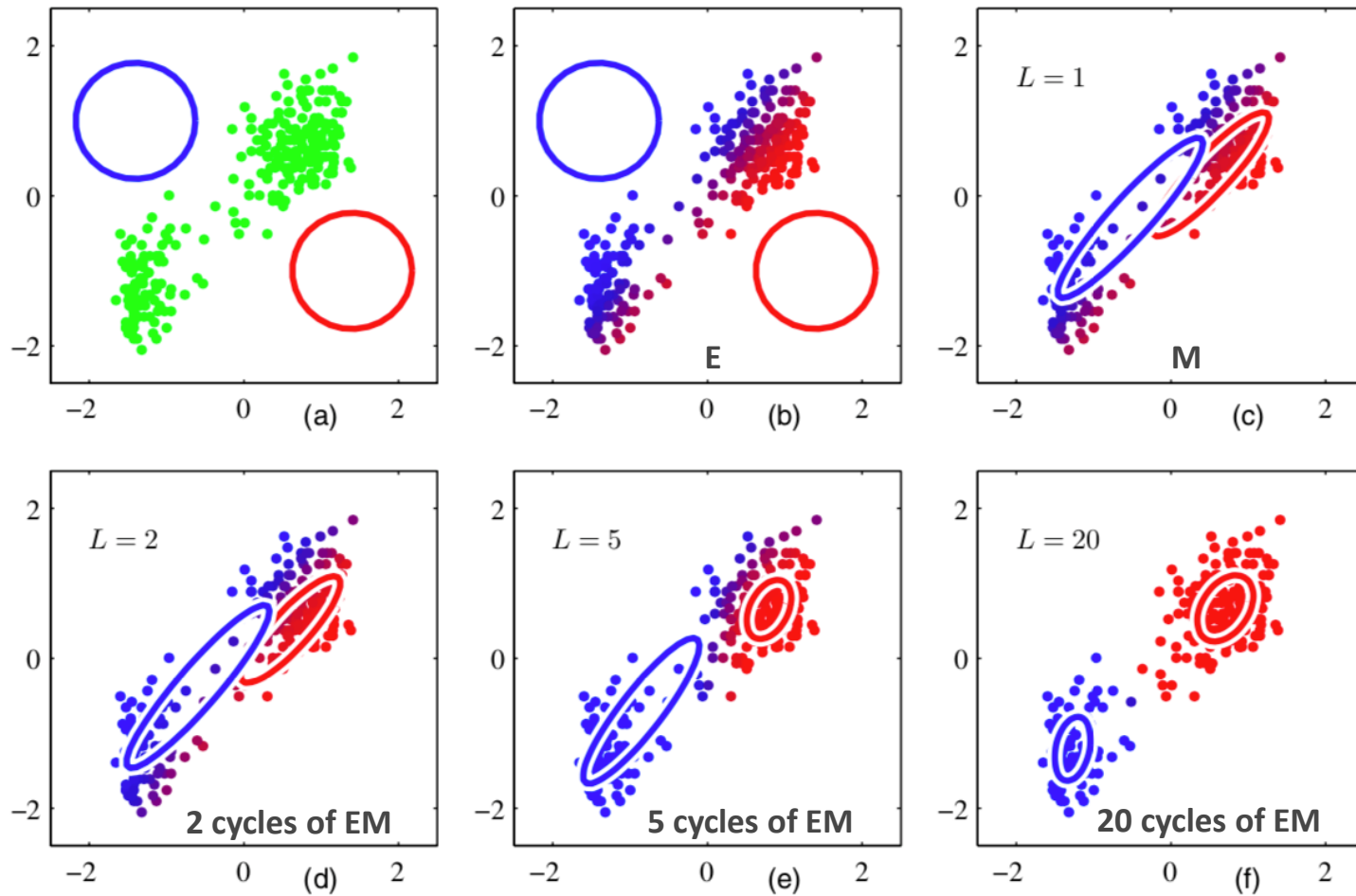
Summary for GMM MLE

3. **M-Step.** Estimate the parameters using MLE.

4. Evaluate the log likelihood, if converges, stop. Otherwise go back to E-step

$$\begin{aligned} N_k &= \sum_{n=1}^N \gamma(z_{nk}) \\ \boldsymbol{\mu}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n \\ \boldsymbol{\Sigma}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^T \\ \pi_k^{\text{new}} &= \frac{N_k}{N} \end{aligned}$$

$$\ln p(\mathbf{X} | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$





Expectation – Maximization (EM)



Given a joint distribution $p(X, Z|\theta)$ over observed variables X and latent variables Z , governed by parameters θ , the goal is to **maximize the likelihood function $p(X|\theta)$ with respect to θ**

1. Choose an initial parameter θ^{old}

2. **E-step.** Evaluate $p(Z|X, \theta^{old})$

3. **M-step.** Evaluate θ^{new} by solving the optimization problem

$$p(\mathbf{X}|\theta) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\theta)$$

$$\theta^{new} = \arg \max_{\theta} Q(\theta, \theta^{old})$$

$$Q(\theta, \theta^{old}) = \sum_{\mathbf{z}} p(\mathbf{Z}|\mathbf{X}, \theta^{old}) \ln p(\mathbf{X}, \mathbf{Z}|\theta) = \mathbb{E}_{\mathbf{z}}[\ln p(\mathbf{X}, \mathbf{Z}|\theta)]$$

4. Check for convergence, if not, go back to E-step

$$\theta^{old} \leftarrow \theta^{new}$$



GMM revisited



In GMM, we have the joint distribution $p(x, z) = p(z)p(x|z)$

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



Alternatively

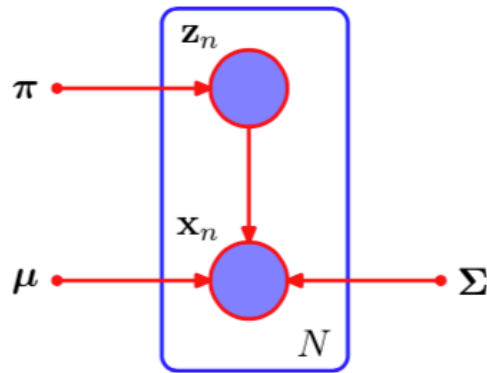
$$p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \prod_{n=1}^N \prod_{k=1}^K \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}}$$

where z_{nk} denotes the k^{th} component of \mathbf{z}_n



Take the logarithm, we have

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^N \sum_{k=1}^K z_{nk} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$$





GMM revisited



Let's recall the **M-step** of EM algorithm

$$\theta^{\text{new}} = \arg \max_{\theta} \mathcal{Q}(\theta, \theta^{\text{old}})$$

$$\mathcal{Q}(\theta, \theta^{\text{old}}) = \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{X}, \theta^{\text{old}}) \ln p(\mathbf{X}, \mathbf{z}|\theta) = \mathbb{E}_{\mathbf{z}}[\ln p(\mathbf{X}, \mathbf{z}|\theta)]$$



In the context of GMM, we have

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^N \sum_{k=1}^K z_{nk} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$$

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



GMM revisited



Recall the posterior in GMM

$$p(z_{nk} = 1 | \mathbf{x}_n, \mu_k, \Sigma_k, \pi_k) = \gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)}$$



So we have

$$\begin{aligned} \mathbb{E}[z_{nk}] &= \sum_{z_{nk}} z_{nk} p(z_{nk} | \mathbf{x}_n, \mu_k, \Sigma_k, \pi_k) \\ &= 0 \cdot p(z_{nk} = 0 | \mathbf{x}_n, \mu_k, \Sigma_k, \pi_k) + 1 \cdot p(z_{nk} = 1 | \mathbf{x}_n, \mu_k, \Sigma_k, \pi_k) \\ &= \gamma(z_{nk}) \end{aligned}$$



GMM revisited

$$\theta^{\text{new}} = \arg \max_{\theta} Q(\theta, \theta^{\text{old}})$$

M-step

$$Q(\theta, \theta^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \theta^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\theta) = \mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z}|\theta)]$$



Now we have Q function for the GMM problem

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})] = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \{ \ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \}$$



Solving for the maximum we have the same result as the previous GMM solution

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^{\text{T}}$$

$$\pi_k^{\text{new}} = \frac{N_k}{N}$$



GMM revisited



In summary, we derivate the GMM result using two methods

1. Standard MLE formulation

$$\pi, \mu, \Sigma = \arg \max_{\pi, \mu, \Sigma} \ln p(\mathbf{X} | \pi, \mu, \Sigma) = \arg \max_{\pi, \mu, \Sigma} \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

2. EM algorithm

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z} | \mu, \Sigma, \pi)] = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \{ \ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \}$$



K-Means revisited



K-Means is a special of the GMM

- The soft assignment becomes hard $\gamma(z_{nk}) \rightarrow r_{nk}$
- The variance is zero $\Sigma_k \rightarrow 0$



In the case that $\Sigma_k = \epsilon I$, the original $\gamma(z_{nk})$ becomes

$$\gamma(z_{nk}) = \frac{\pi_k \exp \{-\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2 / 2\epsilon\}}{\sum_j \pi_j \exp \{-\|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 / 2\epsilon\}}$$



Why? Let's look at the two equations below

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \Sigma_j)}$$

$$p(\mathbf{x} | \boldsymbol{\mu}_k, \Sigma_k) = \frac{1}{(2\pi\epsilon)^{1/2}} \exp \left\{ -\frac{1}{2\epsilon} \|\mathbf{x} - \boldsymbol{\mu}_k\|^2 \right\}$$



K-Means revisited



Let's consider the limit $\epsilon \rightarrow 0$



Look at the denominator



The j that $\|x_n - \mu_j\|^2$ is smallest goes to zero most slowly

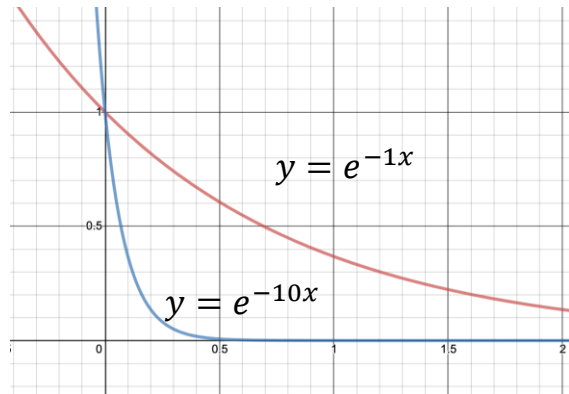


All other j goes to zero quickly



Therefore $\gamma(z_{nk}) \rightarrow r_{nk}$

$$\gamma(z_{nk}) = \frac{\pi_k \exp \{-\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2 / 2\epsilon\}}{\sum_j \pi_j \exp \{-\|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 / 2\epsilon\}}$$



$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$



K-Means revisited



The M-step of GMM/EM gives the following

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})] = \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \{ \ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \}$$



Given $\boldsymbol{\Sigma}_k = \epsilon I$, $\epsilon \rightarrow 0$, $\gamma(z_{nk}) \rightarrow r_{nk}$



It can be written as

$$\mathbb{E}_{\mathbf{Z}}[\ln p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})] \rightarrow -\frac{1}{2} \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2 + \text{const}$$



Which is the same as the distortion measure J in the K-means algorithm

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$



Conclusion for K-Means, GMM, EM



GMM is a special application of EM

- The latent variable $\{z_{nk}\}$ and the model parameters $\{\mu_k, \Sigma_k, \pi_k\}$ are estimated in the E-step and M-step
- The clustering probability is given by $\gamma(z_{nk}) = p(z_{nk} = 1 | x_n, \mu_k, \Sigma_k, \pi_k)$



K-Means is a special case of GMM, where

- The soft assignment becomes hard $\gamma(z_{nk}) \rightarrow r_{nk}$
- The variance is zero $\Sigma_k \rightarrow 0$



K-Means Summary



Complexity $O(t \cdot k \cdot n \cdot d)$

- t - number of iteration, k - number of clusters, n - number of points, d - number of dimension



Advantage

- Simple
- Fast



Disadvantage

- Assumes circle/sphere-like data – same variance in all directions
- Need to provide number of clusters
- Sensitive to initialization
- Sensitive to outlier
 - Can be alleviated by K-Medoids



GMM Summary



Complexity $O(t \cdot k \cdot n \cdot d)$

- t - number of iteration, k - number of clusters, n - number of points, d - number of dimension



Advantage

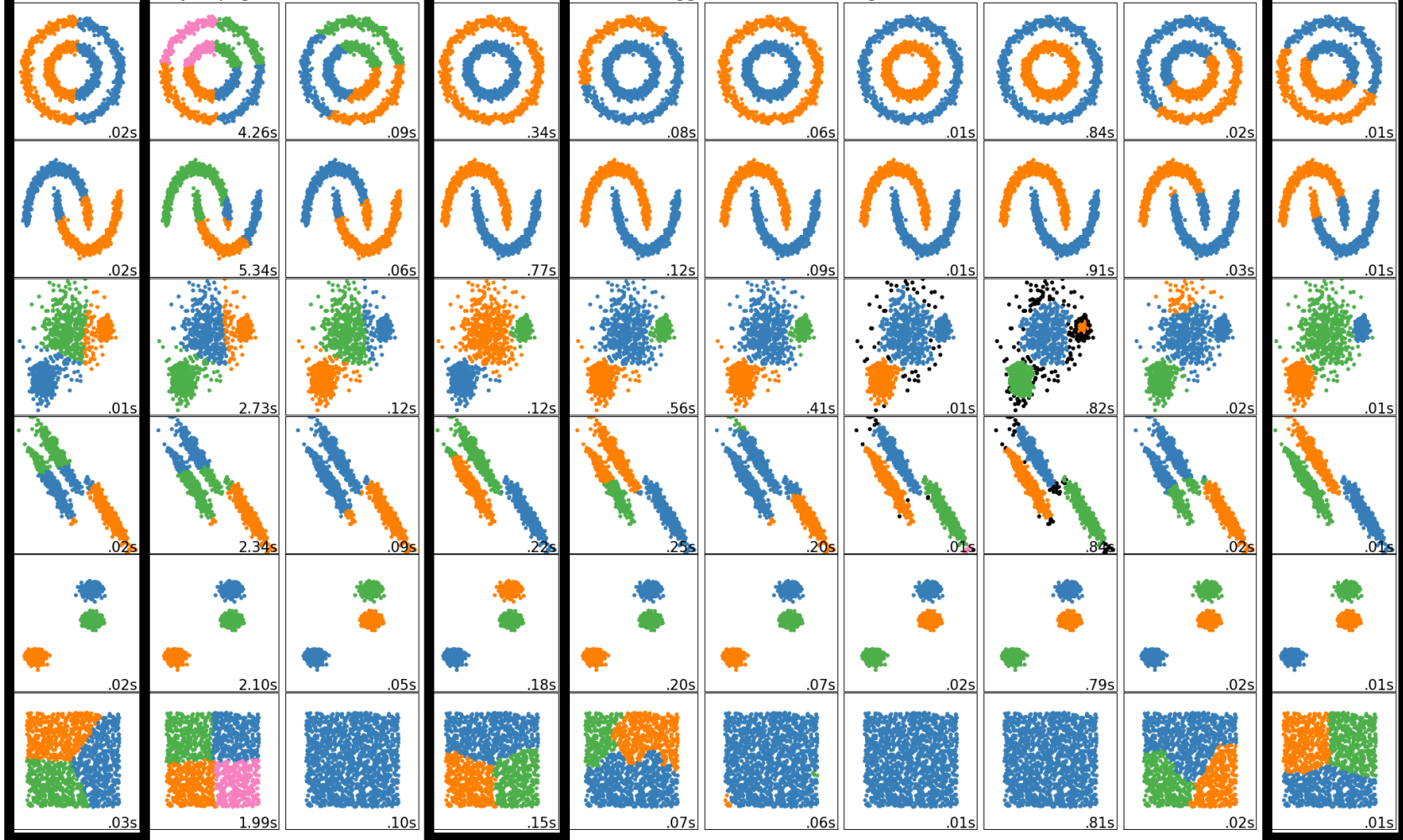
- Provides uncertainty estimation – soft assignment
- More robust to outlier



Disadvantage

- Need to provide number of clusters
- Mainly works with convex clusters
 - Gaussian distributions are in the shape of ellipsoid
- Singularity problem

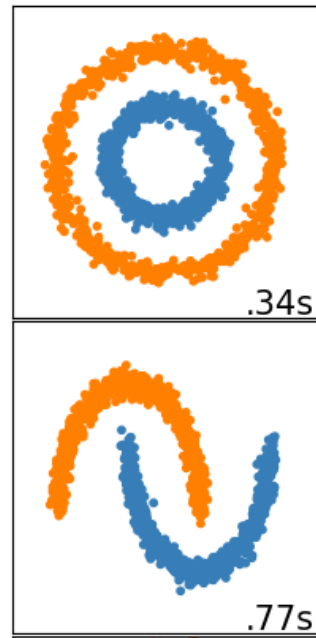
MiniBatchKMeans AffinityPropagation MeanShift SpectralClustering Ward AgglomerativeClusteringDBSCAN OPTICS Birch GaussianMixture





Spectral Clustering

- ⬡ K-Means fails row 1, 2, 3, 4
- ⬡ GMM fails row 1, 2
- ⬡ Why? They are based on **Euclidean** distance!
- ⬡ How about other metric? E.g., connectivity?
- ⬡ **Spectral Clustering works with connectivity!**





Spectral Clustering



Works for a (Weighted) Undirected Graph $G = (V, E)$, where $V = \{v_1, \dots, v_n\}$,

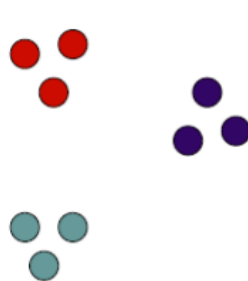


The edge between two vertices v_i, v_j carries weight $w_{ij} \geq 0$.

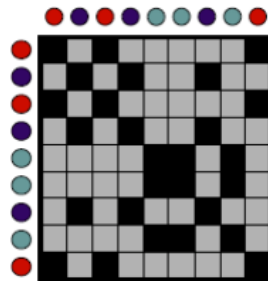
- Two vertices are not connected if $w_{ij} = 0$
- Specially, $w_{ii} = 0$



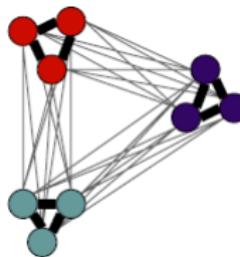
The weighted adjacency matrix / similarity matrix is represented by $W = (w_{ij})_{i,j=1,\dots,n}$



Data



Similarities



Similarity graph



Spectral Clustering – How to build the Graph



Every point is a vertex



There are 3 common graphs built for Spectral Clustering

1. ϵ -neighborhood graph. Radius nearest neighbor search. $w_{ij} = d(v_i, v_j)$
2. k-neighborhood graph. KNN search. $w_{ij} = d(v_i, v_j)$
 - a) Build an edge if v_i is one of the knn of v_j OR v_j is one of the knn of v_i
 - b) Build an edge if v_i is one of the knn of v_j AND v_j is one of the knn of v_i
3. Fully connected graph. Connect every point.



Spectral Clustering – Laplacian Matrix



Degree matrix D

- A diagonal matrix with degrees d_1, \dots, d_n on the diagonal
- $d_i = \sum_{j=1}^n w_{ij}$ is the row sum of adjacency matrix W



Unnormalized graph Laplacian matrix $L = D - W$



Normalized graph Laplacian matrix

- $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$
- $L_{rw} = D^{-1} L = I - D^{-1} W$



Unnormalized Spectral Clustering

1. Build the graph to get adjacency matrix $W \in \mathbb{R}^{n \times n}$
2. Compute **unnormalized Laplacian L**
3. Compute the first (smallest) k eigenvectors **v_1, \dots, v_k of L**
4. Let $V \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors v_1, \dots, v_k as columns
5. For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i -th row of V
6. Cluster the points $\{y_i \in \mathbb{R}^k\}$ with k-means algorithm into clusters C_1, \dots, C_k
7. The final output clusters are A_1, \dots, A_k where $A_i = \{j | y_j \in C_i\}$



Normalized Spectral Clustering

1. Build the graph to get adjacency matrix $W \in \mathbb{R}^{n \times n}$
2. Compute **normalized Laplacian** $L' = L_{rw}$
3. Compute the first (smallest) k eigenvectors **v_1, \dots, v_k of L'**
4. Let $V \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors v_1, \dots, v_k as columns
5. For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i -th row of V
6. Cluster the points $\{y_i \in \mathbb{R}^k\}$ with k-means algorithm into clusters C_1, \dots, C_k
7. The final output clusters are A_1, \dots, A_k where $A_i = \{j | y_j \in C_i\}$



Unnormalized vs. Normalized



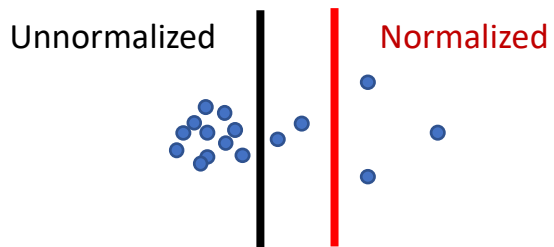
Unnormalized spectral clustering tends to make clusters have equal number of vertices



Normalized tends to group same density clusters



Why? Lecture-4





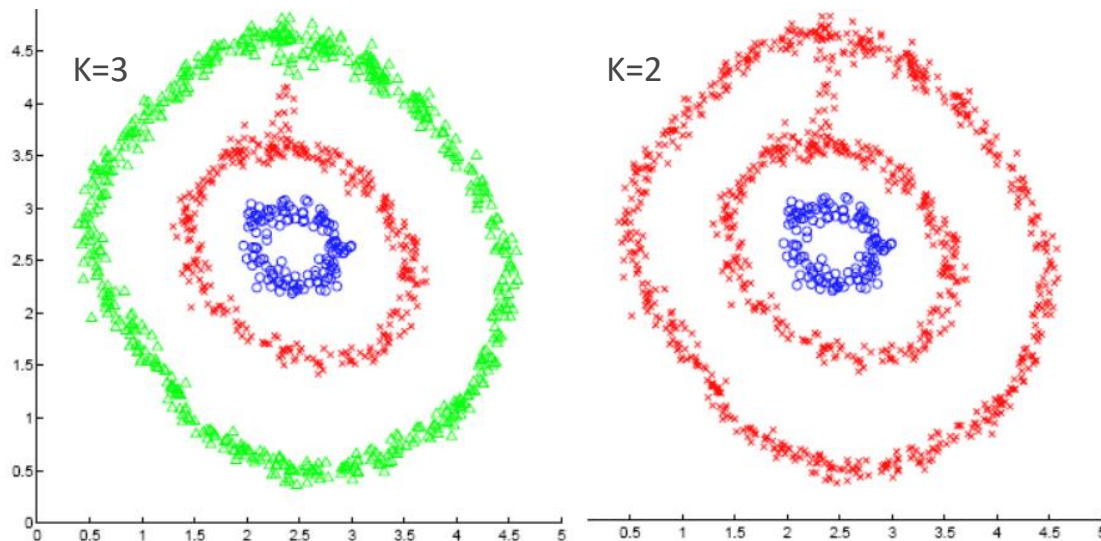
Spectral Clustering



Usually normalized spectral clustering works well with un-uniform point density.



Selection of number of clusters k





Spectral Clustering

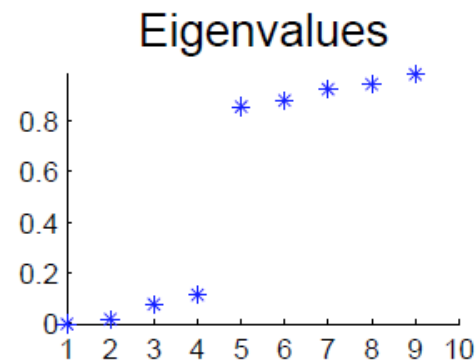
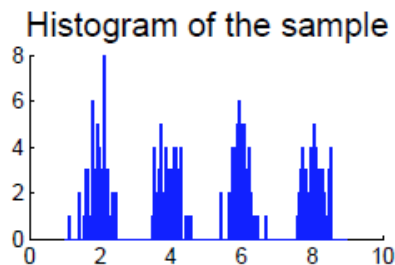


Selection of k can be done by eigenvalue analysis



Most stable clustering is given by the value of k that maximizes the eigen-gap

- Eigengap is the difference between consecutive eigenvalues
- $\Delta_k = |\lambda_k - \lambda_{k-1}|$





Spectral Clustering



Complexity: $O(n^3)$

- This is the complexity of eigen decomposition
- K-means complexity is $O(t \cdot k \cdot n \cdot d)$



Advantage

- No assumption on cluster shape
- Works with similarity, including Euclidean, connectivity
- Works with any dimensional data
- Able to estimate the number of clusters



Disadvantage

- Computational expensive
 - Can be alleviated using sparse similarity matrix and sparse eigen solver



Summary



K-Means

- Euclidean distance
- Hard assignment
- No modeling for a cluster
- Pre-defined cluster number k



GMM

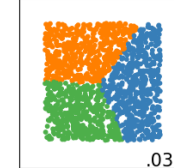
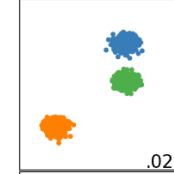
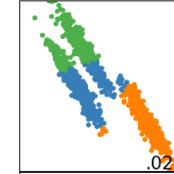
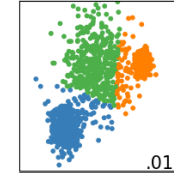
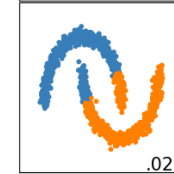
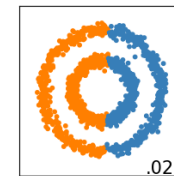
- Euclidean distance
- Probability formulation – soft clustering
- Mean and variance estimation for each cluster
- Pre-defined cluster number k



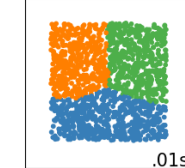
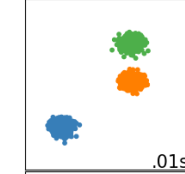
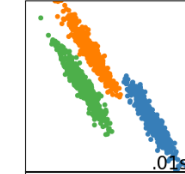
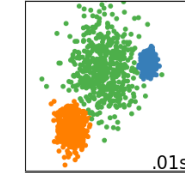
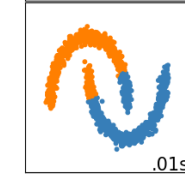
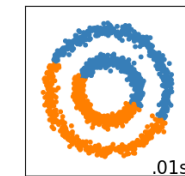
- Spectral Clustering

- Works with connectivity
- Heuristic to determine cluster number k

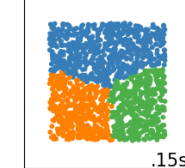
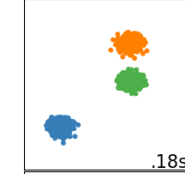
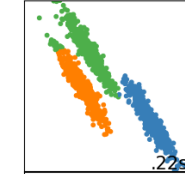
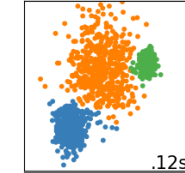
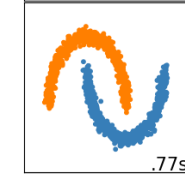
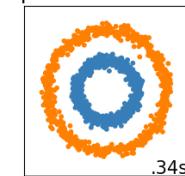
MiniBatchKMeans



GaussianMixture



SpectralClustering





Next Lecture



First half of next lecture covers other Clustering algorithms:

- Proof and intuition of Spectral Clustering
- DBSCAN
- Mean-Shift



Second half covers Model Fitting:

- Hough Transform
- Sample Consensus



Homework



Generate clustering dataset using sklearn

- https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html



Implement your own version of

- K-Means
- GMM
- Spectral Clustering



Visualize and compare the results with the standard results

- <https://scikit-learn.org/stable/modules/clustering.html#overview-of-clustering-methods>



结语

感谢各位聆听!

Thanks for Listening●