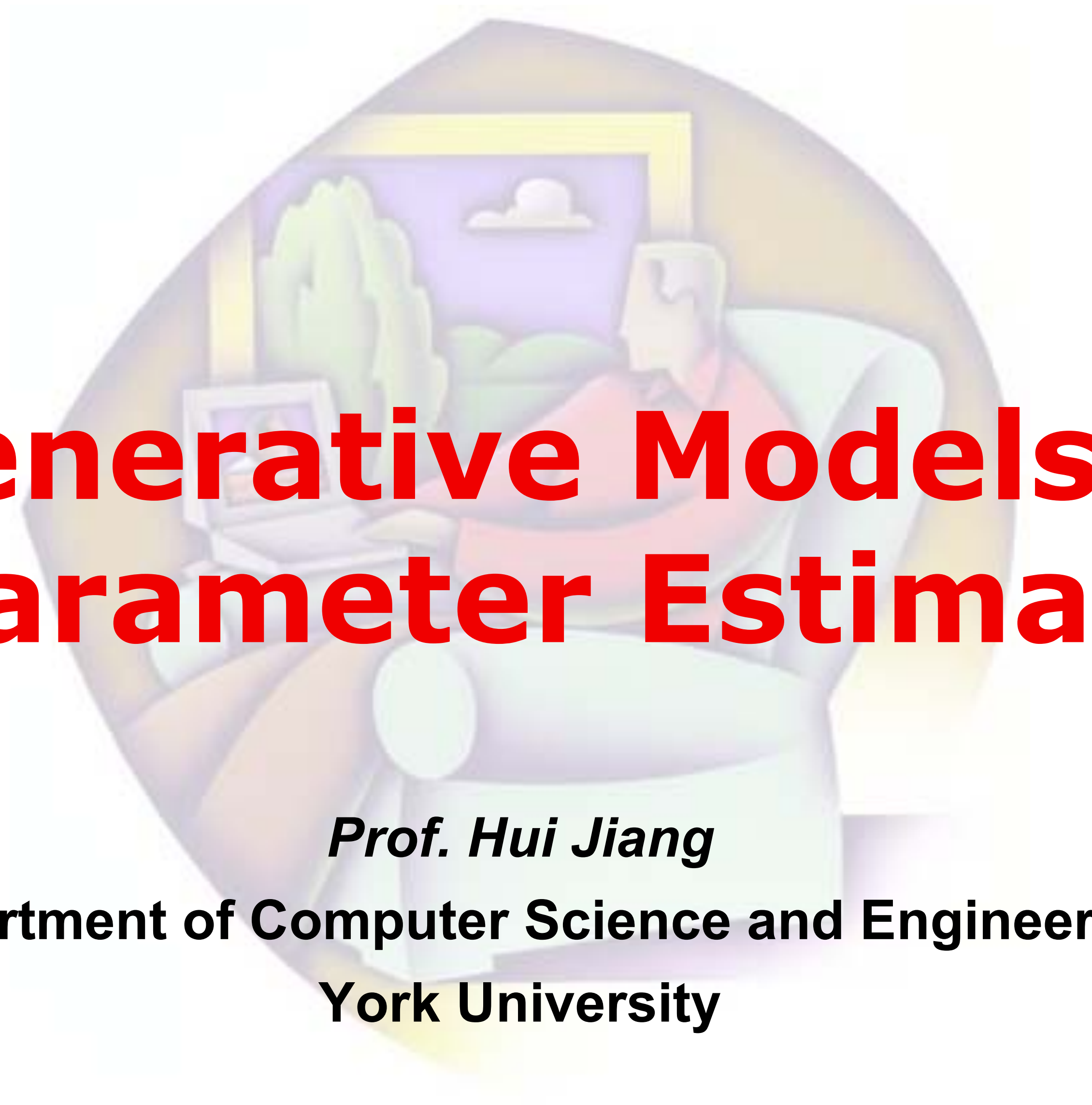
Probabilistic Models and Machine Learning

YORK UNIVERSITÉ UNIVERSITY redefine THE POSSIBLE.



No.7

Generative Models (II): Parameter Estimation

Prof. Hui Jiang

Department of Computer Science and Engineering

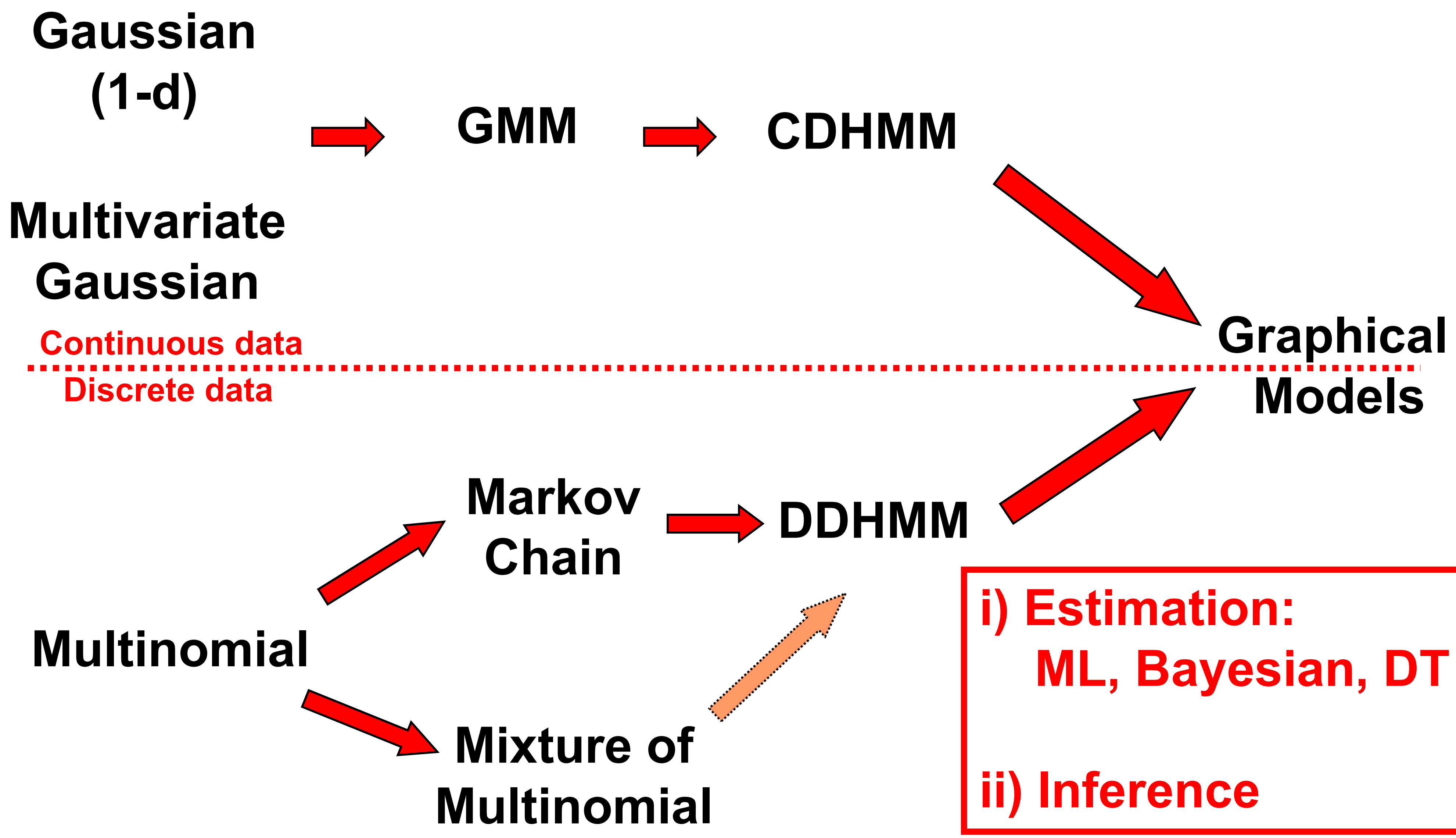
York University

Statistical Data Modeling

- For any real problem, the true p.d.f.'s are always unknown, neither the forms of the functions nor the parameters.
- Our approach – statistical data modeling : based on the available sample data set, choose a proper statistical model to fit into the available data set.
  - Data Modeling stage: once the statistical model is selected, its function form becomes known except the set of model parameters associated with the model are unknown to us.
  - Learning (training) stage: the unknown parameters can be estimated by fitting the model into the data set based on certain estimation criterion.
    - the estimated statistical model (assumed model format + estimated parameters) will give a parametric p.d.f. to approximate the real but unknown p.d.f. of each class.
  - Decision (test) stage: the estimated p.d.f.'s are plugged into the optimal Bayes decision rule in place of the real p.d.f.'s

    ➔ plug-in MAP decision rule
    - Not optimal any more but performs reasonably well in practice

Statistical Models: roadmap



i) Estimation:
ML, Bayesian, DT

ii) Inference

Model Parameter Estimation

- Maximum Likelihood (ML) Estimation:
  - Objective function: likelihood function of all observed data
  - ML method: most popular model estimation; simplest
  - EM (Expected-Maximization) algorithm
  - Examples:
    - Univariate Gaussian distribution
    - Multivariate Gaussian distribution
    - Multinomial distribution
    - Gaussian Mixture model (GMM)
    - Markov chain model
    - Hidden Markov Model (HMM)
- Bayesian Model Estimation
  - The MAP (maximum a posteriori) estimation (point estimation)
  - General Bayesian theory for parameter estimation
  - Recursive Bayes Learning (Sequential Bayesian learning)

Maximum Likelihood (ML) Estimation

- Generative models for classification $\{\omega_1, \cdots, \omega_K\}$:
  - Prior probabilities: $\Pr(\omega_k) \quad (k = 1, \cdots, K)$
  - Class-dependent distribution: $p(\mathbf{x}|\omega_k) \quad (k = 1, \cdots, K)$
- Collect training data for each class: $\mathcal{D}_k \sim p(\mathbf{x}|\omega_k)$
- **Density estimation**: estimate the probability distribution from fine samples
- Select probabilistic models: $\hat{p}_{\theta_k}(\mathbf{x}) \approx p(\mathbf{x}|\omega_k)$
- **Maximum likelihood (ML) Estimation**: learn $\hat{p}_{\theta_k}(\mathbf{x})$ to maximize the probability of observing the training data $\mathcal{D}_k$

$$\theta_k^* = \arg\max_{\theta_k} \hat{p}_{\theta_k}(\mathcal{D}_k) \quad (k = 1, \cdots, K)$$

- ML estimation: fit data best; best interpret the observed data

Maximum Likelihood (ML) Estimation

- Drop index k and $\hat{p}(\cdot) \to p(\cdot)$, ML estimation turns to be:

$$\theta_{\mathsf{ML}} = \arg\max_{\theta} p_\theta(\mathcal{D}) = \arg\max_{\theta} p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N)$$

  where $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

- Assume all data are *i.i.d.* (independent and identically distributed), i.e., all samples are drawn independently from the same distribution:

$$p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) = \prod_{i=1}^{N} p_\theta(\mathbf{x}_i)$$

- Why called maximum *likelihood* (not probability)?
  - $p_\theta(\mathbf{x})$: data distribution of various $\mathbf{x}$ if θ is given (fixed)
  - $p_\theta(\mathbf{x})$: likelihood function of θ if $\mathbf{x}$ is given (fixed)

Maximum Likelihood (ML) Estimation

- In many cases, it is more convenient to work with the logarithm of the likelihood rather than the likelihood itself
- Denote the log-likelihood function $l(\theta) = \ln p_\theta(\mathcal{D})$, we have

$$\theta_{\mathsf{ML}} = \arg\max_\theta \; l(\theta) = \arg\max_\theta \; \sum_{i=1}^{N} \ln p_\theta(\mathbf{x}_i)$$

- Optimization methods for ML estimation:
  - Differential calculus for simple models, e.g., single univariate/multivariate Gaussian, ...
  - Lagrange optimization for models with constraints, e.g., multinomial, markov chain, ...
  - Expectation-Maximization (EM) method for mixture models, e.g., Gaussian mixture models (GMM), hidden Markov models (HMM), ...

Univariate Gaussian (with known variance)

- The training set: $\mathcal{D} = \{x_1, x_2, \cdots, x_N\}$ (a set of scalars)
- A univariate Gaussian (with known variance):

$$p_\theta(x) = \mathcal{N}(x|\mu, \sigma_0^2) = \frac{1}{\sqrt{2\pi\sigma_0^2}} e^{-\frac{(x-\mu)^2}{2\sigma_0^2}}$$

- The log-likelihood function:

$$l(\mu) = \sum_{i=1}^{N} \ln p_\theta(x_i) = \sum_{i=1}^{N} \Big[-\frac{\ln(2\pi\sigma_0^2)}{2} - \frac{(x_i - \mu)^2}{2\sigma_0^2} \Big]$$

- ML estimate of the unknown Gaussian mean is the sample mean:

$$\frac{dl(\mu)}{d\mu} = 0 \implies \mu_{\text{ML}} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

Multivariate Gaussian (I)

- The training set: $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ (each vector $\in \mathbb{R}^d$)
- Choose to model $\mathcal{D}$ with a multivariate Gaussian distribution:

$$p_{\boldsymbol{\mu},\Sigma}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{(\mathbf{x}-\boldsymbol{\mu})^\intercal \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}{2}}$$

- Assume mean vector $\boldsymbol{\mu}$ and covariance matrix Σ are unknown
- The log-likelihood function:

$$\begin{aligned} l(\boldsymbol{\mu}, \Sigma) &= \sum_{i=1}^{N} \ln p_{\boldsymbol{\mu},\Sigma}(\mathbf{x}_i) \\ &= C - \frac{N}{2} \ln |\Sigma| - \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^\intercal \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \end{aligned}$$

Multivariate Gaussian (II)

$$\frac{\partial l(\boldsymbol{\mu}, \Sigma)}{\partial \boldsymbol{\mu}} = 0 \implies \sum_{i=1}^{N} \Sigma^{-1}(\mathbf{x}_i - \boldsymbol{\mu}) = 0 \implies \boldsymbol{\mu}_{\mathsf{ML}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$$

$\frac{\partial l(\boldsymbol{\mu},\Sigma)}{\partial \Sigma} = 0 \implies$

$$-\frac{N}{2}(\Sigma^{\mathsf{T}})^{-1} + \frac{1}{2}(\Sigma^{\mathsf{T}})^{-1}\left[\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^{\mathsf{T}}\right](\Sigma^{\mathsf{T}})^{-1} = 0$$

$$\implies \Sigma_{\mathsf{ML}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu}_{\mathsf{ML}})(\mathbf{x}_i - \boldsymbol{\mu}_{\mathsf{ML}})^{\mathsf{T}}$$

Note that

$$\frac{\partial}{\partial A}\left(\mathbf{x}^{\mathsf{T}} A^{-1}\mathbf{y}\right) = -(A^{\mathsf{T}})^{-1}\mathbf{x}\mathbf{y}^{\mathsf{T}}(A^{\mathsf{T}})^{-1} \quad \text{(square } A\text{)}$$

$$\frac{\partial}{\partial A}\left(\ln|A|\right) = (A^{-1})^{\mathsf{T}} = (A^{\mathsf{T}})^{-1} \quad \text{(square } A\text{)}$$

Gaussian Models for K-class Pattern Classification

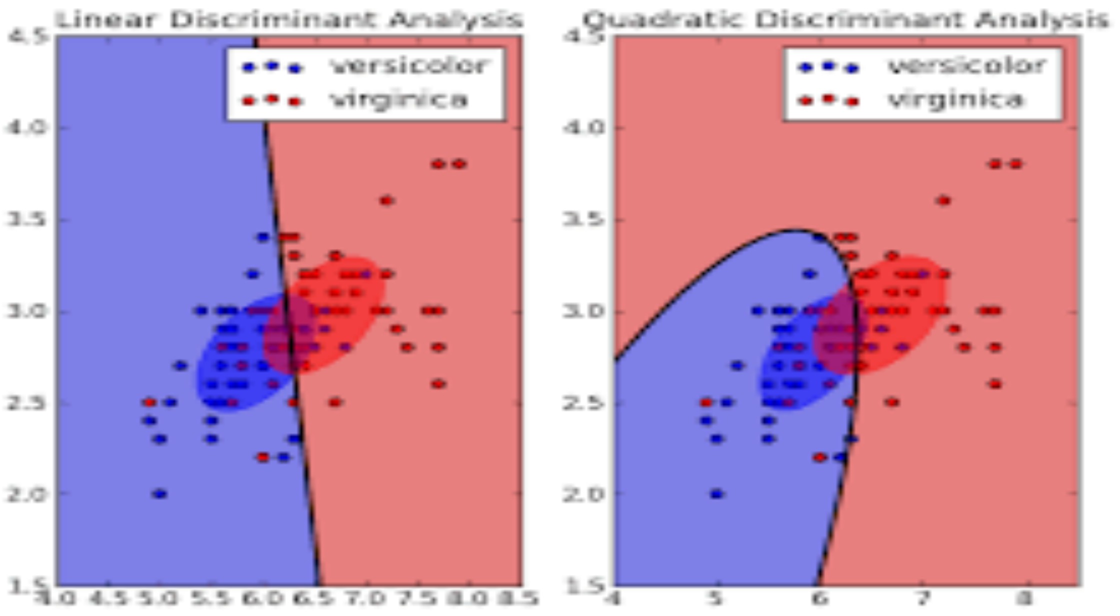
- Given K classes $\{\omega_1, \cdots, \omega_K\}$, we collect a training set $\mathcal{D}_k$ for each class ω_k
- If each feature vector is continuous ($\in \mathbb{R}^d$) and follows a unimodal distribution, we may choose a multivariate Gaussian for each class, $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(k)}, \Sigma^{(k)})$ $(k = 1, 2, \cdots, K)$
- ML estimation: $\mathcal{D}_k \implies \{\boldsymbol{\mu}^{(k)}_{\text{ML}}, \Sigma^{(k)}_{\text{ML}}\}$
- Classify any unknown $\mathbf{x}$ using the plug-in MAP decision rule:

$$g(\mathbf{x}) = \arg\max_k \Pr(\omega_k) p(\mathbf{x}|\omega_k) = \arg\max_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}^{(k)}_{\text{ML}}, \Sigma^{(k)}_{\text{ML}})$$

where we may assume all priors $\Pr(\omega_k)$ are equiprobable.

Linear and Quadratic Discriminant Analysis

- Classification: each class is modeled by a multivariate Gaussian
- Linear Discriminant Analysis
  - Two Gaussians share the same covariance matrix
  - The decision surface is a linear hyperplane
- Quadratic Discriminant Analysis
  - Two Gaussians have different covariance matrices
  - The decision surface is a quadratic parabola



Examples of ML estimation(4): multinomial distribution (I)

- A DNA sequence consists of a sequence of 4 different types of nucleotides (G, A, T, C). For example,

X= GAATTCTTCAAAGAGTTCCAGATATCCACAGGCAGATTCTACAAAAGAAGTGTTTCAATACTGCTCTATC
AAAAGATGTATTCCACTCAGTTACTTTCATGCACACATCTCAATGAAGTTCCTGAGAAAGCTTCTGTCTA
GTTTTTATGTGAAAATATTTCCTTTTCCATCATGGGCCTCAAAGCGCTCAAAATGAACCCTTGCAGATAC
TAGAGAAAGACTGTTTCAAAACTGCTCTATCCAAAGAACGGTTCCACTCTGTGAGGTGAATGCACACATC
ACAAAGCAGTTTCTGAGAACGCTTCTGTCTAGTTTGTAGGTGAAGATATTTCCTTTTCCTTCATAGGCCT
CTAATCGCTCCAAATATCCACAAGCAGATTCTTCAAAATGTGTGTTTCAACACTGCTCTATCAAAAGAAA
GGTTCAAGTCTGTGAGTTGAATGCACACATCACAAAGCAGTTTCTGAGAATGCCTCTGTCTAGTTTGTAT
GTGAAGATATTTCTTTTTCCGTCTTATGCCTCAAATCGCTCCAAATATCCACTTGCAGATACTTCAAAA

- If assume all nucleotides in a DNA sequence are independent, we can use multinomial distribution to model a DNA sequence,
- Use p_1 to denote probability to observe G in any one location, p_2 for A, p_3 for T, p_4 for C.
- Obviously, it meets $\sum_{i=1}^{4} p_i = 1$.
- Given a DNA sequence X, the probability to observe X is

$$\Pr(X) = C \cdot \prod_{i=1}^{4} p_i^{N_i}$$

Examples of ML estimation(4): multinomial distribution (II)

- Where N1 is frequency of G appearing in X, N2 frequency of A, N3 frequency of T, N3 frequency of C.
- Problem: estimate p1, p2, p3, p4 from a training sequence X based on the maximum likelihood criterion.
- The log-likelihood function:

$$l(p_1, p_2, p_3, p_4) = \sum_{i=1}^{4} N_i \cdot \ln p_i$$

- Where N1 is frequency of G in training sequence X, the similar for N2, N3 and N4.
- Maximization l(.) subject to the constraint $\sum_{i=1}^{4} p_i = 1$
- Use Lagrange optimization:

$$L(p_1, p_2, p_3, p_4, \lambda) = \sum_{i=1}^{4} N_i \cdot \ln p_i - \lambda(\sum_{i=1}^{4} p_i - 1)$$

$$\frac{\partial}{\partial p_i} L(p_1, p_2, p_3, p_4, \lambda) = 0 \quad \Rightarrow \quad N_i / p_i - \lambda = 0$$

Examples of ML estimation(4): multinomial distribution (III)

- Finally, we get the ML estimation for the multinomial distribution as:

$$p_i = \frac{N_i}{\sum_{i=1}^{4} N_i} \qquad (i = 1,2,3,4)$$

- We only need count the occurrence times (frequency) of each nucleotides in all training sequences, then the ML estimate can be easily calculated as above.

- Similar derivation also holds for Markov chain model.
  - It has an important application in language modeling, the so-called n-gram model.

Examples of ML estimation(5): Markov Chain Model (I)

- Markov assumption: a discrete-time Markov chain is a random sequence x[n] whose n-th conditional probability function satisfy:

$$p(x[n] \mid x[n-1]x[n-2]\ldots x[n-N]) = p(x[n] \mid x[n-1])$$

- In other words, probability of observing x[n] only depends on its previous one x[n-1] (for 1st order Markov chain) or the most recent history (for higher order Markov chain).

- Parameters in Markov Chain model are a set of conditional probability functions.

Examples of ML estimation(5): Markov Chain Model (II)

- Stationary assumption:

  p(x[n] | x[n-1]) = p(x[n'] | x[n' -1]) for all n and n' .

- For stationary discrete Markov Chain model:
  - Only one set of conditional probability function

- Discrete observation: in practice, the range of values taken on by each x[n] is finite, which is called state space. Each distinct one is a Markov state.
  - An observation of a discrete Markov chain model becomes a sequence of Markov states.
  - The set of conditional probs → transition matrix

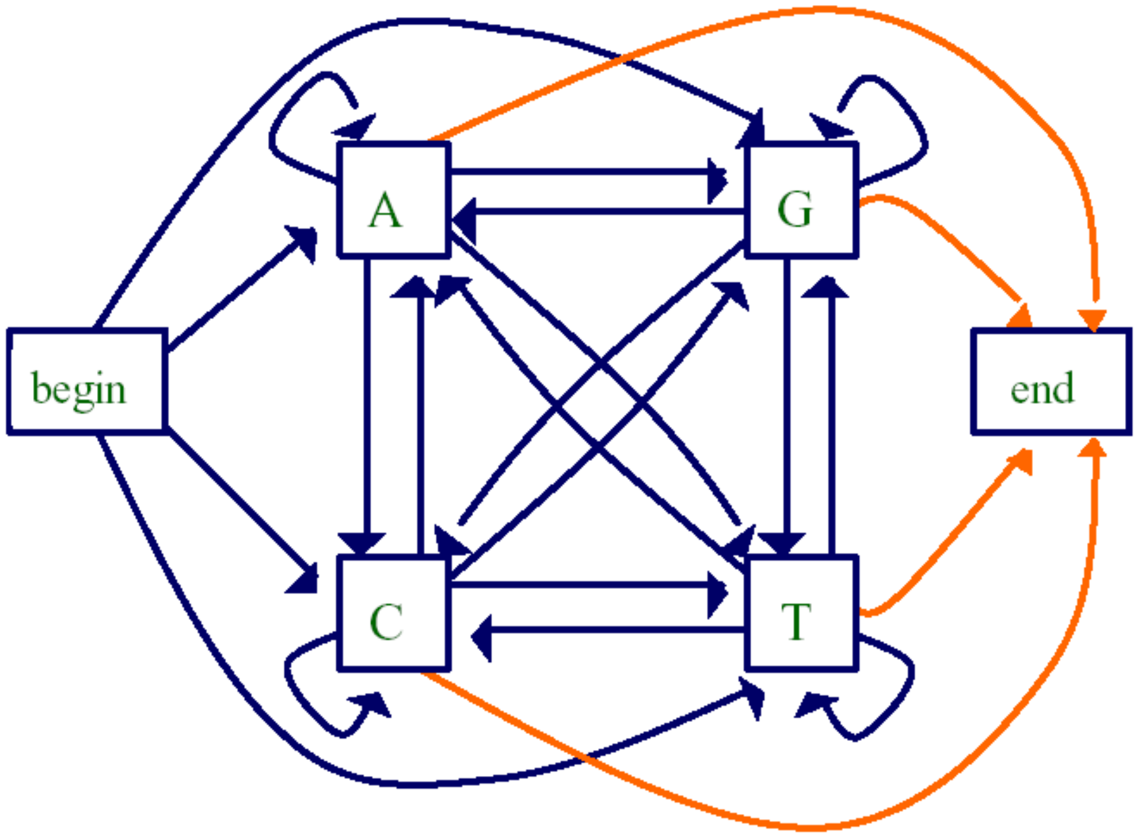
Examples of ML estimation(5): Markov Chain Model (III)

- Markov Chain Model (stationary & discrete):
  - A finite set of Markov states, to say M states.
  - A set of state conditional probabilities, i.e., ***transition matrix***

    In 1st order Markov chain model, $a_{ij} = p(j|i) \quad (i,j=1,2,\dots,M)$
- Markov Chain model can be represented by a directed graph.
  - Node → Markov state
  - Arc → state transition (each arc attached with a transition probability)
  - A Markov chain observation can be viewed as a path traversing a Markov chain model.
- Probability of observing a Markov chain can be calculated based on the path and the transition matrix.

Examples of ML estimation(5): Markov Chain Model (IV)

- First-order Markov Chain Model for DNA sequence



Full Transition matrix (6 by 6)

p(A|G)	=	0.16
p(C|G)	=	0.34
p(G|G)	=	0.38
p(T|G)	=	0.12

…

…

One transition probability is attached with each arc.

Pr(GAATTC) = p(begin)p(G|begin)p(A|G)p(A|A)p(T|A)p(T|T)p(C|T)p(end|C)

Examples of ML estimation(5): Markov Chain Model (V)

- Markov Chain Model for language modeling (***n-gram***)
  - Each word is a Markov state, total ***N*** words (vocabulary size)
  - A set of state (word) conditional probabilities
- Given any a sentence:

  S = I would like to fly from New York to Toronto this Friday
- 1st-order Markov chain model: ***N*N*** conditional probabilities

  $Pr(\boldsymbol{S}) = p(I|begin)\ p(would|I)\ p(like|would)\ p(to|like)\ p(fly|to)\ \ldots$
  - This is called bi-gram model
- 2nd-order Markov chain model: ***N*N*N***

  $Pr(\boldsymbol{S}) = p(I|begin)\ p(would|I,begin)\ p(like|would,I)\ p(to|like,would)\ p(fly|to,like)\ \ldots$
  - This is called tri-gram model
- Multinomial (0th-order Markov chain): N probabilities

  $Pr(\boldsymbol{S}) = p(I)\ p(would)\ p(like)\ p(to)\ p(fly)\ \ldots$
  - This is called uni-gram model

Examples of ML estimation(5): Markov Chain Model (VI)

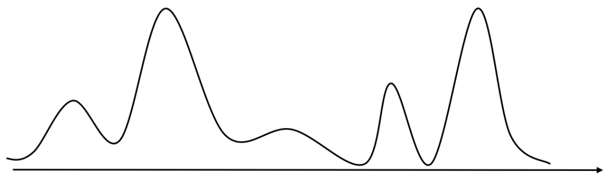
- How to estimate Markov Chain Model from training data
  - Similar to ML estimate of multinomial distribution
  - Maximization of log-likelihood function with constraints.
- Results:

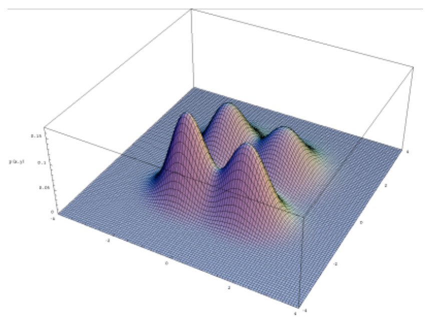
$$p(W_i \mid W_j) = \frac{\text{Frequency of } W_j W_i \text{ in training data}}{\text{Frequency of } W_j \text{ in training data}}$$

$$p(W_i \mid W_j, W_k) = \frac{\text{Frequency of } W_k W_j W_i \text{ in training data}}{\text{Frequency of } W_k W_j \text{ in training data}}$$

- Generally, N-gram model: a large number of probabilities to be estimated.

Gaussian mixture model (GMM)





To model **multi-modal** distributions of $\mathbf{x} \in \mathbb{R}^d$, we may consider a group of Gaussians:

$$p(\mathbf{x}) = \sum_{m=1}^{M} w_m \cdot \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \Sigma_m)$$

- Mixture weights w_m satisfy $\sum_{m=1}^{M} w_m = 1$,
- Mean vector and covariance matrix of m-th Gaussian component: $\boldsymbol{\mu}_m$ and Σ_m
- If M is large enough, GMMs can approximate any arbitrary distribution in $\mathbb{R}^d$

Mixture Models

- A mixture of any simpler component distributions:

$$p(\mathbf{x}) = \sum_{m=1}^{M} w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})$$

- Component distribution $f_{\boldsymbol{\theta}}(\mathbf{x})$: Gaussians, multinomial,...
- In general, $f_{\boldsymbol{\theta}}(\mathbf{x})$ is chosen from the **exponential family (e-family)**:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \exp\left\{ A(\bar{\mathbf{x}}) + \bar{\mathbf{x}}^{\intercal}\boldsymbol{\lambda} - \mathcal{K}(\boldsymbol{\lambda}) \right\}$$

  - $\boldsymbol{\lambda} = g(\boldsymbol{\theta})$ is called *natural parameters*
  - $\bar{\mathbf{x}} = h(\mathbf{x})$ is called *sufficient statistics*
  - $\mathcal{K}(\boldsymbol{\lambda})$ is a normalization term:
    $\int_{\mathbf{x}} f_{\boldsymbol{\theta}}(\mathbf{x}) d\mathbf{x} = 1 \implies \mathcal{K}(\boldsymbol{\lambda}) = \ln\left[\int_{\mathbf{x}} \big(A(h(\mathbf{x})) + (h(\mathbf{x}))^{\intercal}\boldsymbol{\lambda} \big) d\mathbf{x} \right]$
- Taking logarithm $\implies \ln f_{\boldsymbol{\theta}}(\mathbf{x}) = A(\bar{\mathbf{x}}) + \bar{\mathbf{x}}^{\intercal}\boldsymbol{\lambda} - \mathcal{K}(\boldsymbol{\lambda})$

Exponential Family (e-family)

- Most basic probabilistic models belong to e-family, including *Gaussian, Binomial, Multinomial, Bernoulli, Dirichlet, Beta, Gamma, Von Mises, Wishart*, ...
- Some examples:

$f_{\boldsymbol{\theta}}(\mathbf{x})$	$\boldsymbol{\lambda} = g(\boldsymbol{\theta})$	$\bar{\mathbf{x}} = h(\mathbf{x})$	$\mathcal{K}(\boldsymbol{\lambda})$	$A(\bar{\mathbf{x}})$
Gausssian $\mathcal{N}(x \mid \mu, \sigma^2)$	$[\mu/\sigma^2, 1/\sigma^2]$	$[x, -x^2/2]$	$\frac{1}{2}\lambda_1^2/\lambda_2$ $-\frac{1}{2}\log(\lambda_2)$	$-\frac{1}{2}\ln(2\pi)$
Gaussian $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma_0)$	$\boldsymbol{\mu}$	$\Sigma_0^{-1}\mathbf{x}$	$\frac{1}{2}\boldsymbol{\lambda}^\intercal\Sigma_0^{-1}\boldsymbol{\lambda}$	$-\frac{d}{2}\ln(2\pi)$
Gaussian $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma)$	$[\Sigma^{-1}\boldsymbol{\mu}, \Sigma^{-1}]$	$[\mathbf{x}, -\frac{1}{2}\mathbf{x}\mathbf{x}^\intercal]$	$\frac{1}{2}\boldsymbol{\lambda}_1^\intercal\boldsymbol{\lambda}_2^{-1}\boldsymbol{\lambda}_1$ $-\frac{1}{2}\ln\lvert\boldsymbol{\lambda}_2\rvert$	$-\frac{d}{2}\ln(2\pi)$
Multinomial $C \cdot \prod_{d=1}^{D}\mu_d^{x_d}$	$[\ln\mu_1, \cdots,$ $\ln\mu_D]$	$\mathbf{x}$	0	$\ln(C)$

- Products of e-family distributions still belong to e-family

ML Estimation of GMMs

- It is not trivial to estimate GMMs and any mixture models
- Given training data $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$
- Log-likelihood function contains *log-sum*:

$$l\Big(\{w_m, \boldsymbol{\mu}_m, \Sigma_m\}\Big) = \sum_{i=1}^{N} \ln \left(\sum_{m=1}^{M} w_m \cdot \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_m, \Sigma_m) \right)$$

- Can we switch *log-sum* into *sum-log* ?

Expectation-Maximization (EM) algorithm

- Log-likelihood function of mixture models:

$$l(\boldsymbol{\theta}) = \ln \sum_{m=1}^{M} w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})$$

- Treat m as a **latent variable**: an unobserved random variable in $\{1, 2, \cdots, M\}$
- Given any model $\boldsymbol{\theta}^{(n)}$, we may compute a conditional probability distribution of m based on data $\mathbf{x}$: $\Pr(m \mid \mathbf{x}, \boldsymbol{\theta}^{(n)})$
- Define an auxiliary function of $\boldsymbol{\theta}$ as follows:

$$\begin{aligned} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) &= \mathbb{E}_m\Big[\ln\big(w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})\big) \mid \mathbf{x}, \boldsymbol{\theta}^{(n)}\Big] + const \\ &= \sum_{m=1}^{M} \ln\left[w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})\right] \cdot \Pr(m \mid \mathbf{x}, \boldsymbol{\theta}^{(n)}) + const \end{aligned}$$

where $const = -\sum_{m=1}^{M} \ln \Pr(m \mid \mathbf{x}, \boldsymbol{\theta}^{(n)}) \Pr(m \mid \mathbf{x}, \boldsymbol{\theta}^{(n)})$

Auxiliary Function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ (I)

Theorem

The auxiliary function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ satisfies the following three properties:

1. $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ *touches* $l(\boldsymbol{\theta})$ *at* $\boldsymbol{\theta}^{(n)}$*:*

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} = l(\boldsymbol{\theta})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}$$

2. $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ *and* $l(\boldsymbol{\theta})$ *make a tangent touch at* $\boldsymbol{\theta}^{(n)}$*:*

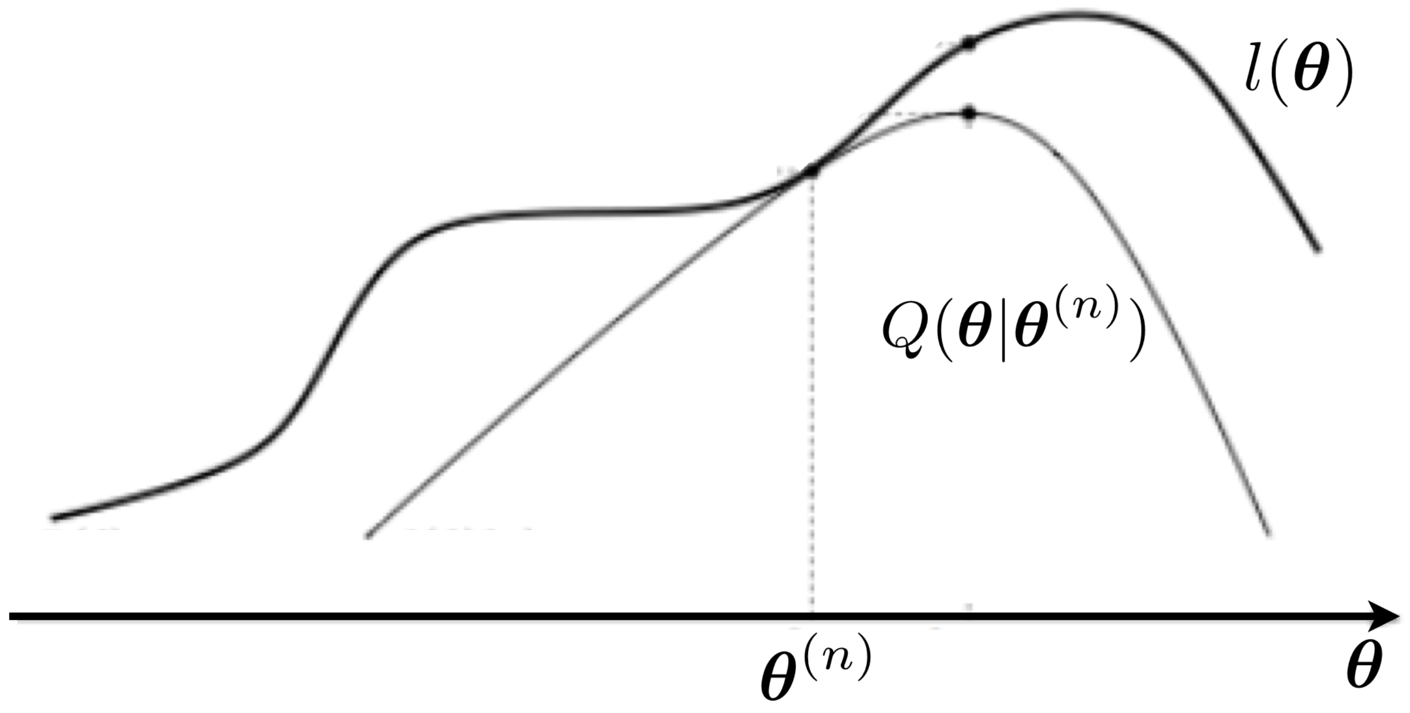
$$\frac{\partial Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} = \frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}$$

3. *For all* $\boldsymbol{\theta} \neq \boldsymbol{\theta}^{(n)}$*,* $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ *locates strictly below* $l(\boldsymbol{\theta})$*:*

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) < l(\boldsymbol{\theta}) \quad (\forall \boldsymbol{\theta} \neq \boldsymbol{\theta}^{(n)})$$

Auxiliary Function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ (II)

The auxiliary function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ is related to $l(\boldsymbol{\theta})$ like this:



$l(\boldsymbol{\theta})$
$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$
$\boldsymbol{\theta}^{(n)}$
$\boldsymbol{\theta}$

Auxiliary Function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ (III)

Proof:

- Bayes theorem $Pr(y|x) = \frac{p(x,y)}{p(x)} \implies p(x) = \frac{p(x,y)}{Pr(y|x)}$
- Apply to the model $p_{\boldsymbol{\theta}}(m, \mathbf{x})$, we have

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{p_{\boldsymbol{\theta}}(m, \mathbf{x})}{\Pr(m|\mathbf{x}, \boldsymbol{\theta})} \implies \ln p_{\boldsymbol{\theta}}(\mathbf{x}) = \ln p_{\boldsymbol{\theta}}(m, \mathbf{x}) - \ln \Pr(m|\mathbf{x}, \boldsymbol{\theta})$$

- Multiply $\Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)})$ to both sides, and sum over all $m = \{1, 2, \cdots, M\}$:

$$\begin{aligned}\sum_{m=1}^{M} \ln p_{\boldsymbol{\theta}}(\mathbf{x}) \cdot \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) &= \sum_{m=1}^{M} \ln p_{\boldsymbol{\theta}}(m, \mathbf{x}) \cdot \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) \\ &- \sum_{m=1}^{M} \ln \Pr(m|\mathbf{x}, \boldsymbol{\theta}) \cdot \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)})\end{aligned}$$

Auxiliary Function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ (IV)

Proof (continued):

- Since $\sum_{m=1}^{M} \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) = 1$ and $l(\boldsymbol{\theta}) = \ln p_{\boldsymbol{\theta}}(\mathbf{x})$, and $p_{\boldsymbol{\theta}}(m, \mathbf{x}) = w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})$ we have

$$\begin{aligned}
l(\boldsymbol{\theta}) &= Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) + \left[\sum_{m=1}^{M} \ln \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) \right. \\
&- \left. \sum_{m=1}^{M} \ln \Pr(m|\mathbf{x}, \boldsymbol{\theta}) \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) \right] \\
&= Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) + \underbrace{\left[H(\boldsymbol{\theta}^{(n)}|\boldsymbol{\theta}^{(n)}) - H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) \right]}_{\mathsf{KL}\left(\Pr(m|\mathbf{x},\boldsymbol{\theta}^{(n)}) || \Pr(m|\mathbf{x},\boldsymbol{\theta})\right) \geq 0} \\
&\geq Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})
\end{aligned}$$

- Equality holds only when $\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}$, properties 1 and 3 are proved.

Auxiliary Function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$ (V)

Proof (continued):

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})}{\partial \boldsymbol{\theta}} - \frac{\partial H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})}{\partial \boldsymbol{\theta}}$$

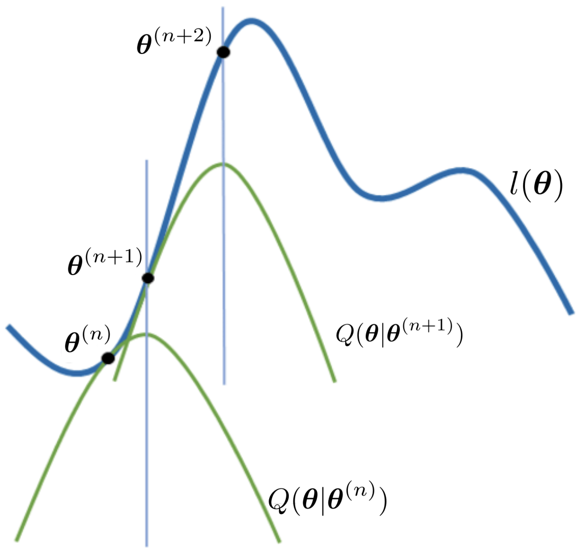
$$\begin{aligned}
-\frac{\partial H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} &= \left[\sum_{m=1}^{M} \frac{\Pr(m|\mathbf{x},\boldsymbol{\theta}^{(n)})}{\Pr(m|\mathbf{x},\boldsymbol{\theta})} \frac{\partial \Pr(m|\mathbf{x},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right]\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \\
&= \left[\sum_{m=1}^{M} \frac{\partial \Pr(m|\mathbf{x},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right]\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \\
&= \frac{\partial}{\partial \boldsymbol{\theta}}\left[\sum_{m=1}^{M} \Pr(m|\mathbf{x},\boldsymbol{\theta})\right]\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \\
&= \frac{\partial}{\partial \boldsymbol{\theta}}\left[1\right]\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} = 0
\end{aligned}$$

Property 3 is proved. ■

Expectation-Maximization (EM) algorithm

Algorithm 2 EM algorithm

```
initialize $\boldsymbol{\theta}^{(0)}$, set $n = 0$
while not converge do
    E-step: $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) = \mathbb{E}_m\left[\ln\left(w_m \cdot f_{\boldsymbol{\theta}_m}(\mathbf{x})\right)|\mathbf{x}, \boldsymbol{\theta}^{(n)}\right]$
    M-step: $\boldsymbol{\theta}^{(n+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$
    $n = n + 1$
end while
```



$\boldsymbol{\theta}^{(n+2)}$
$l(\boldsymbol{\theta})$
$\boldsymbol{\theta}^{(n+1)}$
$\boldsymbol{\theta}^{(n)}$
$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n+1)})$
$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})$

Convergence Analysis of EM algorithm

Theorem

Each EM iteration improves $l(\boldsymbol{\theta})$: $l(\boldsymbol{\theta}^{(n+1)}) \geq l(\boldsymbol{\theta}^{(n)})$.

Proof:

- Property 1 $\implies l(\boldsymbol{\theta}^{(n)}) = Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}$
- M-step $\implies Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n+1)}} \geq Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}$
- Property 3 $\implies l(\boldsymbol{\theta}^{(n+1)}) > Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n+1)}}$

$$l(\boldsymbol{\theta}^{(n+1)}) > Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n+1)}} \geq Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} = l(\boldsymbol{\theta}^{(n)})$$

- Therefore, we have $l(\boldsymbol{\theta}^{(n+1)}) \geq l(\boldsymbol{\theta}^{(n)})$ and
- $l(\boldsymbol{\theta}^{(n+1)}) - l(\boldsymbol{\theta}^{(n)}) > Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n+1)}} - Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}$

■

EM algorithm for GMMs (I)

- If $f_{\boldsymbol{\theta}_m}(\mathbf{x})$ belongs to e-family, $Q(\cdot)$ is concave and M-step can be solved in closed-form.
- For GMMs, $f_{\boldsymbol{\theta}_m}(\mathbf{x})$ is a Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_m, \Sigma_m)$
- Denote

$$\xi_m^{(n)}(\mathbf{x}) = \Pr(m|\mathbf{x}, \boldsymbol{\theta}^{(n)}) = \frac{w_m^{(n)} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_m^{(n)}, \Sigma_m^{(n)})}{\sum_{m=1}^{M} w_m^{(n)} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_m^{(n)}, \Sigma_m^{(n)})}$$

- Given a set of training data $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, the auxiliary function: $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(n)}) =$

$$\sum_{i=1}^{N} \sum_{m=1}^{M} \left[\ln w_m - \frac{\ln |\Sigma_m|}{2} - \frac{(\mathbf{x}_i - \boldsymbol{\mu}_m)^\intercal \Sigma_m^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_m)}{2} \right] \xi_m^{(n)}(\mathbf{x}_i)$$

EM algorithm for GMMs (II)

For all $m = 1, 2, \cdots M$,

$$\frac{\partial Q(\cdot)}{\partial \boldsymbol{\mu}_m} = 0 \implies \boldsymbol{\mu}_m^{(n+1)} = \frac{\sum_{i=1}^N \mathbf{x}_i \cdot \xi_m^{(n)}(\mathbf{x}_i)}{\sum_{i=1}^N \xi_m^{(n)}(\mathbf{x}_i)}$$

$$\frac{\partial Q(\cdot)}{\partial \Sigma_m} = 0 \implies \Sigma_m^{(n+1)} = \frac{\sum_{i=1}^N (\mathbf{x}_i - \boldsymbol{\mu}_m^{(n+1)})(\mathbf{x}_i - \boldsymbol{\mu}_m^{(n+1)})^\intercal \xi_m^{(n)}(\mathbf{x}_i)}{\sum_{i=1}^N \xi_m^{(n)}(\mathbf{x}_i)}$$

$$\frac{\partial}{w_m}\left[Q(\cdot) - \Big(\sum_{m=1}^M w_m - 1\Big)\right] = 0 \implies w_m^{(n+1)} = \frac{\sum_{i=1}^N \xi_m^{(n)}(\mathbf{x}_i)}{N}$$

ML Estimation of GMMs using EM

- Given a training set $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$
- Learn a multivariate GMM using $\mathcal{D}$:

$$p(\mathbf{x}) = \sum_{m=1}^{M} w_m \cdot \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_m, \Sigma_m)$$

  with $\sum_{m=1}^{M} w_m = 1$
- Iterative EM training algorithm:
  - Initialize $\{w_m^{(0)}, \boldsymbol{\mu}_m^{(0)}, \Sigma_m^{(0)}\}$, and $n = 0$
  - E-step: $\{w_m^{(n)}, \boldsymbol{\mu}_m^{(n)}, \Sigma_m^{(n)}\} \Longrightarrow \{\xi_m^{(n)}\}$
  - M-step: $\{\xi_m^{(n)}\} \Longrightarrow \{w_m^{(n+1)}, \boldsymbol{\mu}_m^{(n+1)}, \Sigma_m^{(n+1)}\}$
  - $n = n + 1$ until converged.

GMM Initialization: K-Means clustering

- K-Means Clustering: a.k.a. unsupervised learning
- Unsupervisedly cluster a data set into many homogeneous groups
- K-Means algorithm:
  - step 1: assign all data into one group; calculate centroid.
  - step 2: choose a group and split.
  - step 3: re-assign all data to groups.
  - step 4: calculate centroids for all groups.
  - step 5: go back to step 3 until convergence.
  - step 6: stop until K classes
- Basics for clustering:
  - distance measure
  - centroid calculation
  - choose a group and split

Bayesian Theory

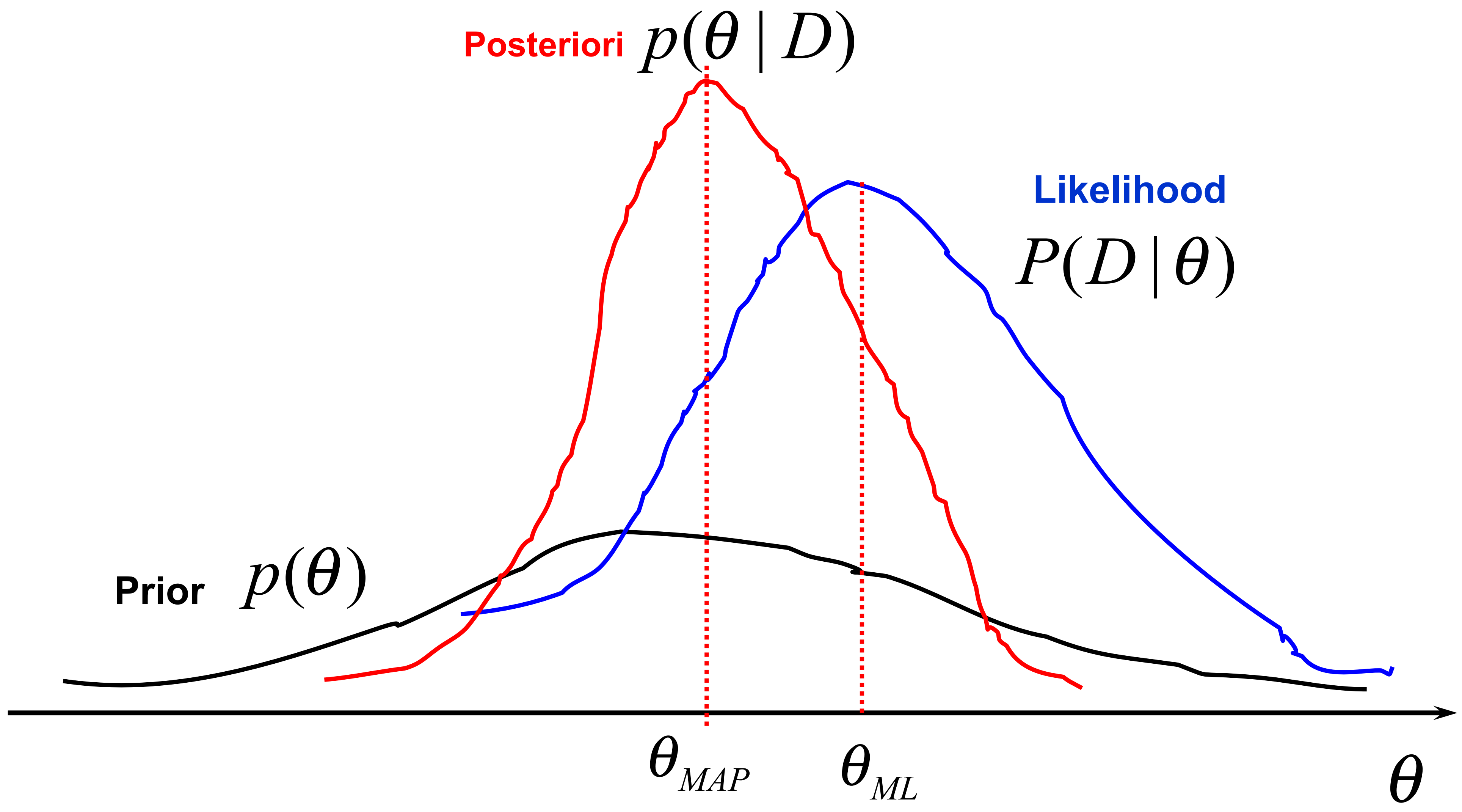
- Bayesian methods view model parameters as random variables having some known prior distribution. (Prior specification)
  - Specify prior distribution of model parameters θ as p(θ).

- Training data D allow us to convert the prior distribution into a posteriori distribution. (Bayesian learning)

$$p(\theta \mid D) = \frac{p(\theta) \cdot p(D \mid \theta)}{p(D)} \propto p(\theta) \cdot p(D \mid \theta)$$

- We infer or decide everything solely based on the posteriori distribution. (Bayesian inference)
  - Model estimation: the MAP (maximum a posteriori) estimation
  - Pattern Classification: Bayesian classification
  - Sequential (on-line, incremental) learning
  - Others: prediction, model selection, etc.

Bayesian Learning



The MAP estimation of model parameters

- Do a point estimate about θ based on the posteriori distribution

$$\theta_{MAP} = \arg\max_{\theta} p(\theta \mid D) = \arg\max_{\theta} p(\theta) \cdot p(D \mid \theta)$$

- Then θ_{MAP} is treated as estimate of model parameters (just like ML estimate). Sometimes need the EM algorithm to derive it.

- MAP estimation optimally combine prior knowledge with new information provided by data.

- MAP estimation is used in speech recognition to adapt speech models to a particular speaker to cope with various accents
  - From a generic speaker-independent speech model ➔ prior
  - Collect a small set of data from a particular speaker
  - The MAP estimate give a speaker-adaptive model which suit better to this particular speaker.

Bayesian Classification

- Assume we have N classes, ωi (i=1,2,...,N), each class has a class-conditional pdf p(X|ωi,θi) with parameters θi.
- The prior knowledge about θi is included in a prior p(θi).
- For each class ωi, we have a training data set Di.
- Problem: classify an unknown data Y into one of the classes.
- The Bayesian classification is done as:

$$\omega_Y = \arg\max_i p(Y \mid D_i) = \arg\max_i \int p(Y \mid \omega_i, \theta_i) \cdot p(\theta_i \mid D_i)\, \mathrm{d}\theta_\mathrm{i}$$

where

$$p(\theta_i \mid D_i) = \frac{p(\theta_i) \cdot p(D_i \mid \omega_i, \theta_i)}{p(D_i)} \propto p(\theta_i) \cdot p(D_i \mid \omega_i, \theta_i)$$

Recursive Bayes Learning (On-line Bayesian Learning)

- Bayesian theory provides a framework for on-line learning (a.k.a. incremental learning, adaptive learning).
- When we observe training data one by one, we can dynamically adjust the model to learn incrementally from data.
- Assume we observe training data set D={X1,X2,…,Xn} one by one,

$$p(\theta) \xrightarrow{X_1} p(\theta \mid X_1) \xrightarrow{X_2} p(\theta \mid X_1, X_2) \cdots\cdots p(\theta \mid D^{(n)})$$

Learning Rule: $posteriori \propto prior \times likelihood$

Knowledge about Model at this stage

Knowledge about Model at this stage

Knowledge about Model at this stage

Knowledge about Model at this stage

How to specify priors

- Noninformative priors
  - In case we don't have enough prior knowledge, just use a flat prior at the beginning.

- Conjugate priors: for computation convenience
  - For some models, if their probability functions are a reproducing density, we can choose the prior as a special form (called conjugate prior), so that after Bayesian leaning the posterior will have the exact same function form as the prior except the all parameters are updated.

  - Not every model has conjugate prior.

Conjugate Prior

- For a univariate Gaussian model with only unknown mean:

$$p(x|\omega_i) = N(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp[-\frac{(x-\mu)^2}{2\sigma^2}]$$

- If we choose the prior as a Gaussian distribution (Gaussian's conjugate prior is Gaussian)

$$p(\mu) = N(\mu|\mu_0,\sigma_0^2) = \frac{1}{\sqrt{2\pi\sigma_0^2}}\exp[-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}]$$

- After observing a new data x1, the posterior will still be Gaussian:

$$p(\mu|x_1) = N(\mu|\mu_1,\sigma_1^2) = \frac{1}{\sqrt{2\pi\sigma_1^2}}\exp[-\frac{(\mu-\mu_1)^2}{2\sigma_1^2}]$$

where $$\mu_1 = \frac{\sigma_0^2}{\sigma_0^2+\sigma^2}x_1 + \frac{\sigma^2}{\sigma_0^2+\sigma^2}\mu_0$$

$$\sigma_1^2 = \frac{\sigma_0^2\sigma^2}{\sigma_0^2+\sigma^2}$$

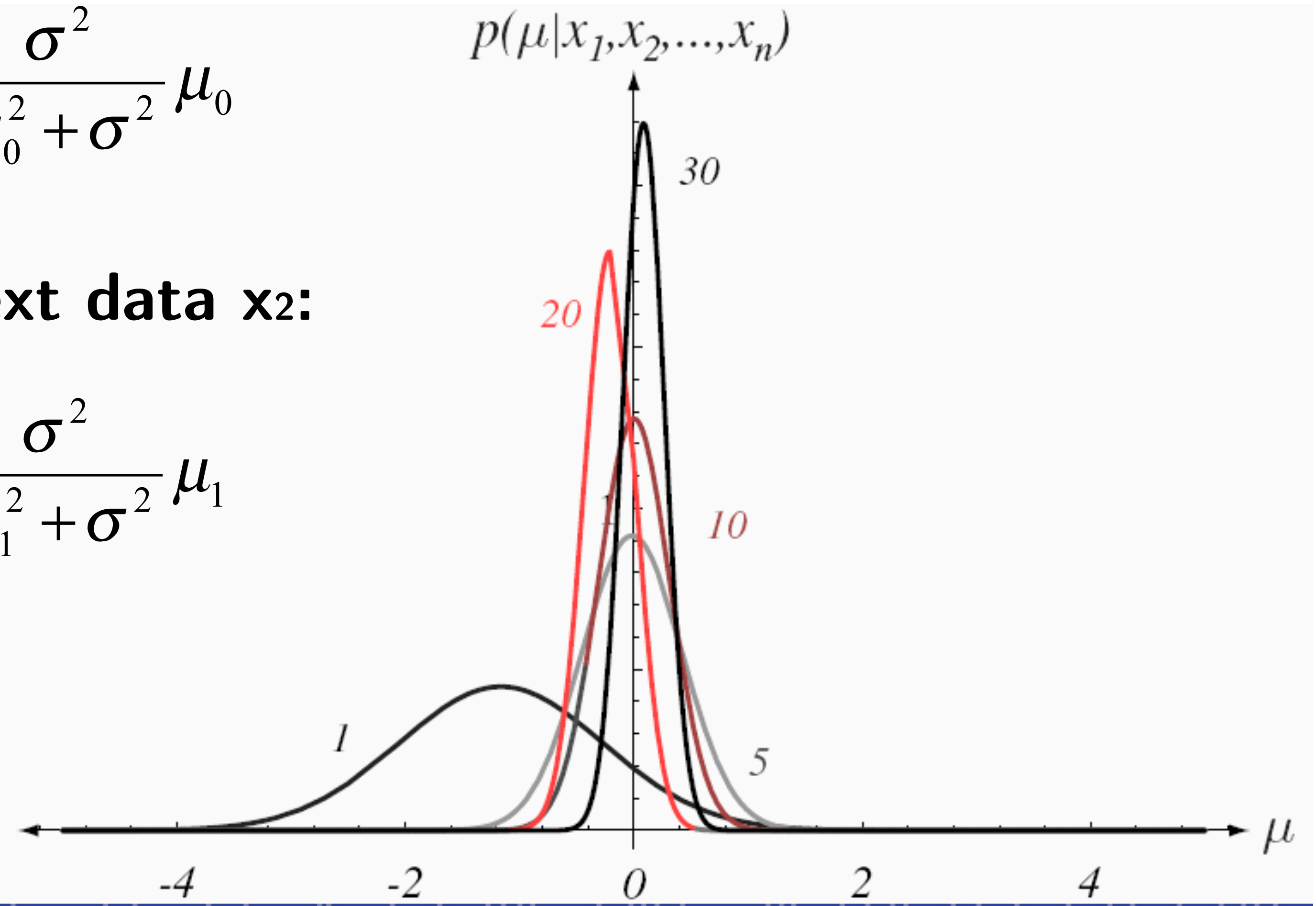
The sequential MAP Estimate of Gaussian

- For univariate Gaussian with unknown mean, the MAP estimate of its mean after observing x1:

$$\mu_1 = \frac{\sigma_0^2}{\sigma_0^2 + \sigma^2} x_1 + \frac{\sigma^2}{\sigma_0^2 + \sigma^2} \mu_0$$

- After observing next data x2:

$$\mu_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma^2} x_2 + \frac{\sigma^2}{\sigma_1^2 + \sigma^2} \mu_1$$



$p(\mu|x_1,x_2,\ldots,x_n)$
30
20
10
5
1
μ
-4
-2
0
2
4