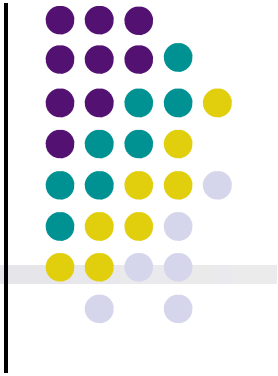
Machine Learning

Lecture 10

Yang Yang
Department of Computer Science & Engineering
Shanghai Jiao Tong University

K-Means



Algorithm

1. Decide on a value for k.
2. Initialize the k cluster centers randomly if necessary.
3. Decide the class memberships of the N objects by assigning them to the nearest cluster centroids

$$\vec{\mu}_k = \frac{1}{C_k} \sum_{i \in C_k} \vec{x}_i$$

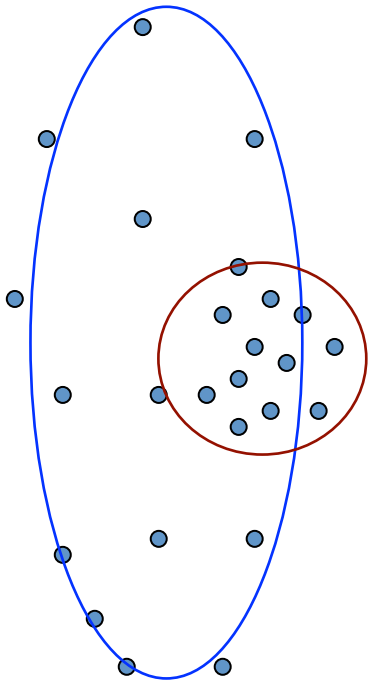
4. Re-estimate the k cluster centers, by assuming the memberships found above are correct.
5. If none of the N objects changed membership in the last iteration, exit. Otherwise go to 3.

Clustering And Partially Observable Probabilistic Models

Unobserved Variables

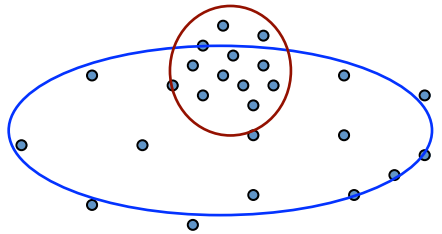
- A variable can be unobserved (latent) because:
  - it is an imaginary quantity meant to provide some simplified and abstractive view of the data generation process
    - e.g., speech recognition models, mixture models …
  - it is a real-world object and/or phenomena, but difficult or impossible to measure
    - e.g., the temperature of a star, causes of a disease, evolutionary ancestors …
  - it is a real-world object and/or phenomena, but sometimes wasn't measured, because of faulty sensors; or was measured with a noisy channel, etc.
- Discrete latent variables can be used to partition/cluster data into sub-groups
- Continuous latent variables (factors) can be used for dimensionality reduction (factor analysis, etc., later lectures).

The Evils of "**Hard Assignments**"?



- Clusters may overlap
- Some clusters may be "wider" than others
- Distances can be deceiving!

Probabilistic Clustering

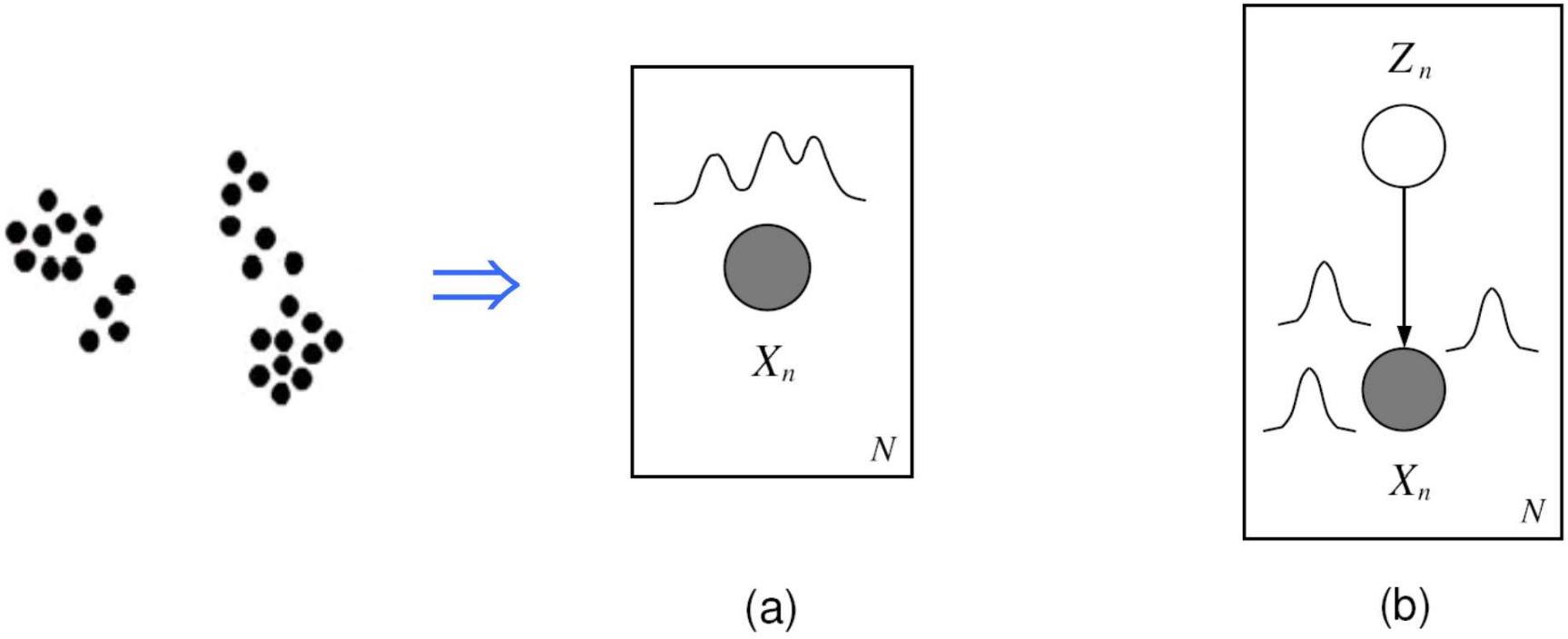


- Try a probabilistic model!
  - allows overlaps, clusters of different size, etc.
- Can tell a *generative story* for data
  - P(X|Y) P(Y)
- Challenge: we need to estimate model parameters without labeled Ys

Y	X_1	X_2
??	0.1	2.1
??	0.5	-1.1
??	0.0	3.0
??	-0.1	-2.0
??	0.2	1.5
…	…	…

Mixture Models

- A density model p(x) may be multi-modal.
- We may be able to model it as a mixture of uni-modal distributions (e.g., Gaussians).
- Each mode may correspond to a different sub-population (e.g., male and female).



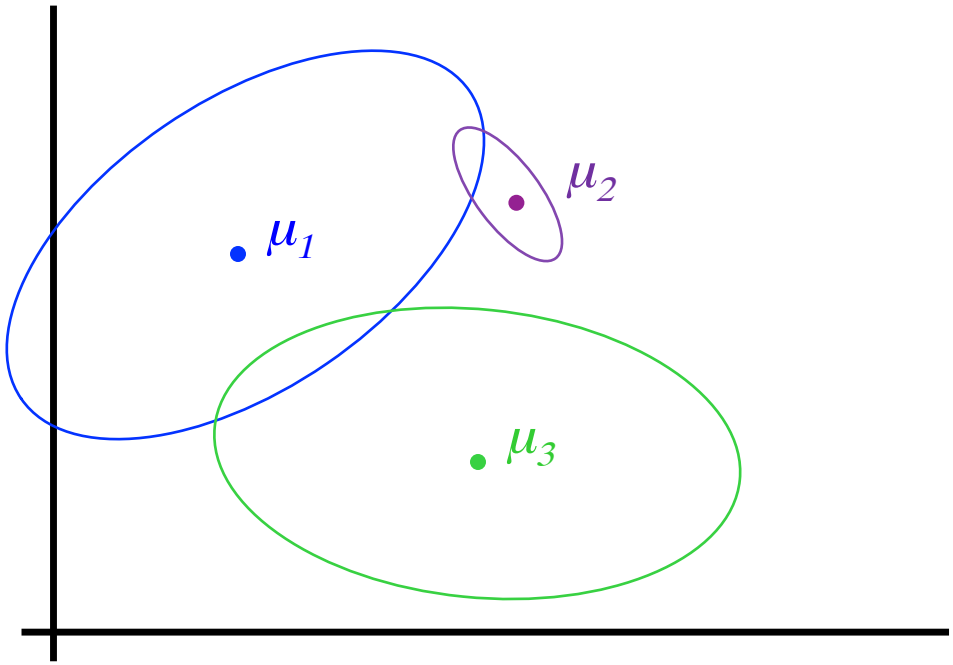
The General GMM assumption

- P(Y): There are k components
- P(X|Y): Each component generates data from a **multivariate Gaussian** with mean μ_i and covariance matrix Σ_i

Each data point is sampled from a ***generative process*:**

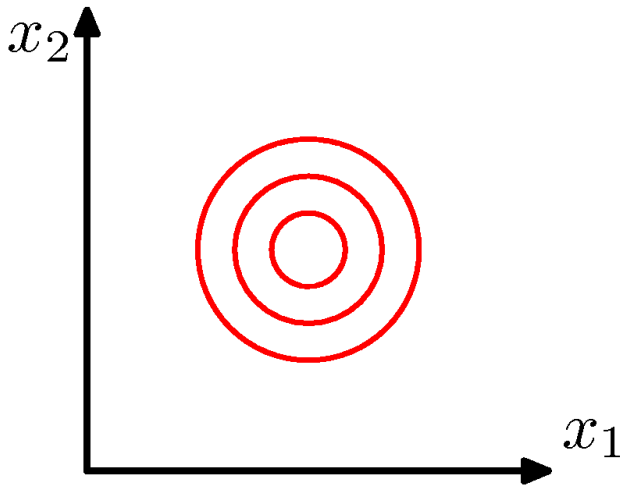
1. Choose component i with probability *P(y=i)*
2. Generate datapoint ~ N(m_i, Σ_i)

Gaussian mixture model (GMM)



Multivariate Gaussians

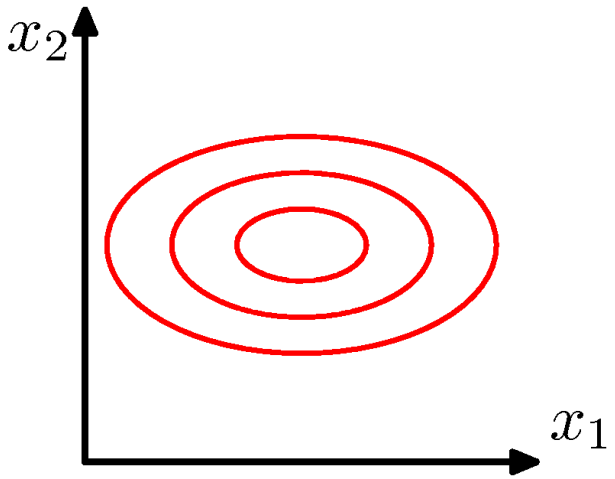
$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2}\,\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}_j-\mu\right)^T \Sigma^{-1}\left(\mathbf{x}_j-\mu\right)\right]$$



$\Sigma \propto$ identity matrix

Multivariate Gaussians

$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2}\,\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}_j-\mu\right)^T \Sigma^{-1}\left(\mathbf{x}_j-\mu\right)\right]$$

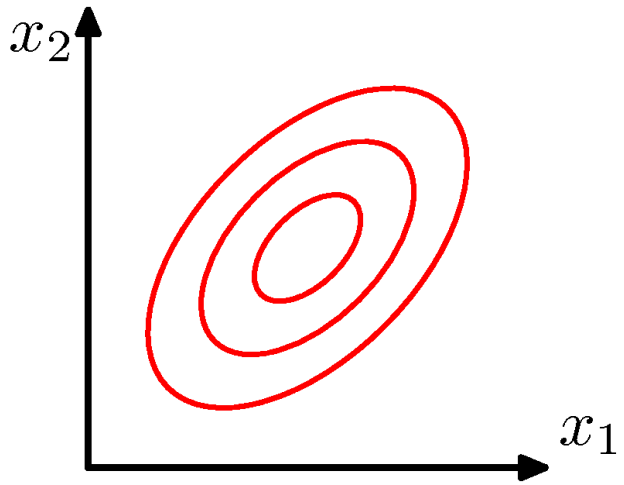


Σ = diagonal matrix

X_i are independent *ala* Gaussian NB

Multivariate Gaussians

$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2}\,\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}_j-\mu\right)^T \Sigma^{-1}\left(\mathbf{x}_j-\mu\right)\right]$$

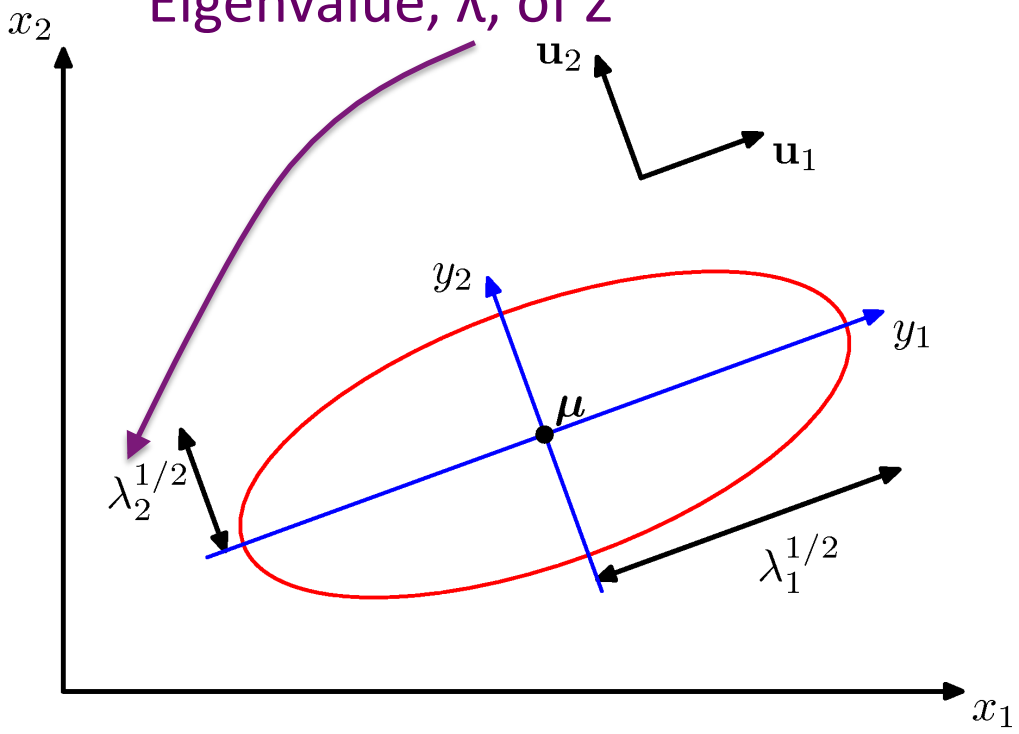


Σ = arbitrary (semidefinite) matrix:

- specifies rotation (change of basis)
- eigenvalues specify relative elongation

Multivariate Gaussians

Eigenvalue, λ, of Σ

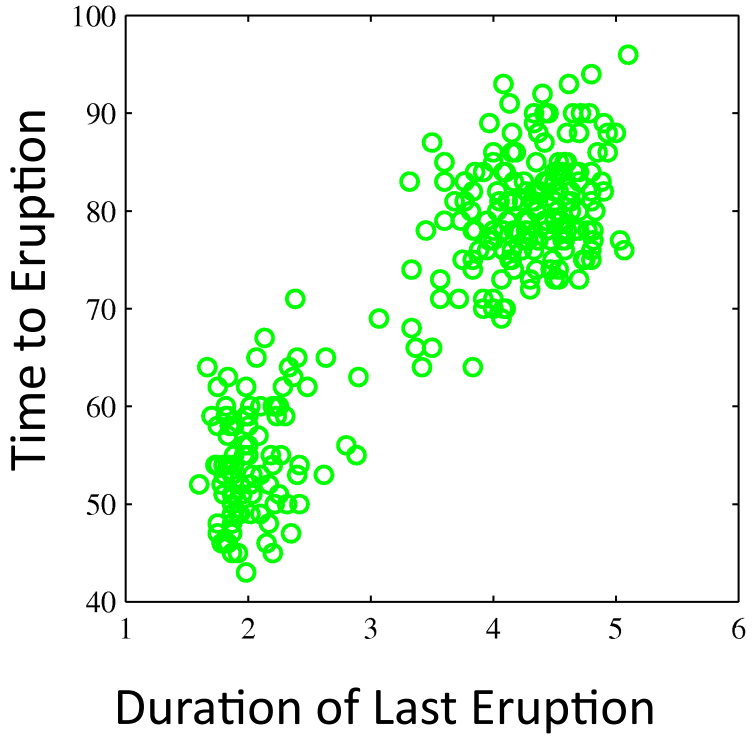


Covariance matrix, Σ, = degree to which x_i vary together

$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2} \| \Sigma \|^{1/2}} \exp\left[-\frac{1}{2} \left(\mathbf{x}_j - \mu \right)^T \Sigma^{-1} \left(\mathbf{x}_j - \mu \right) \right]$$

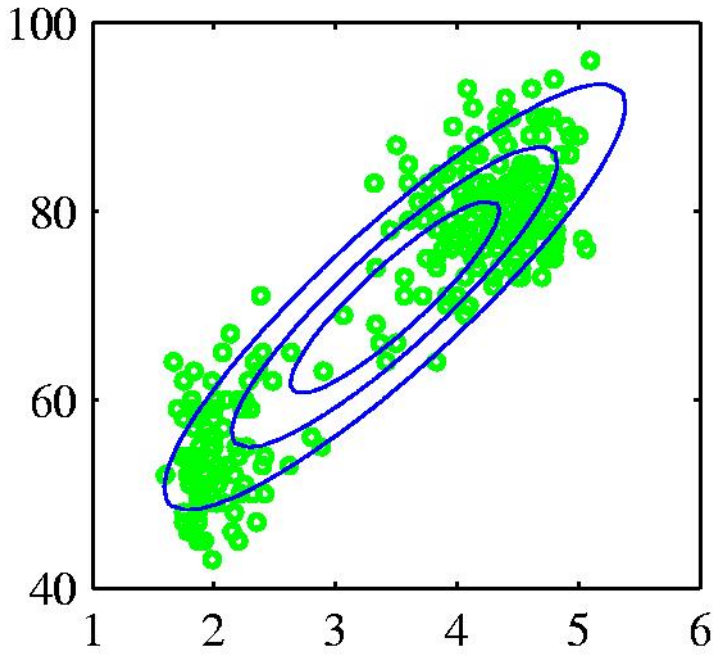
Mixtures of Gaussians (1)

Old Faithful Data Set

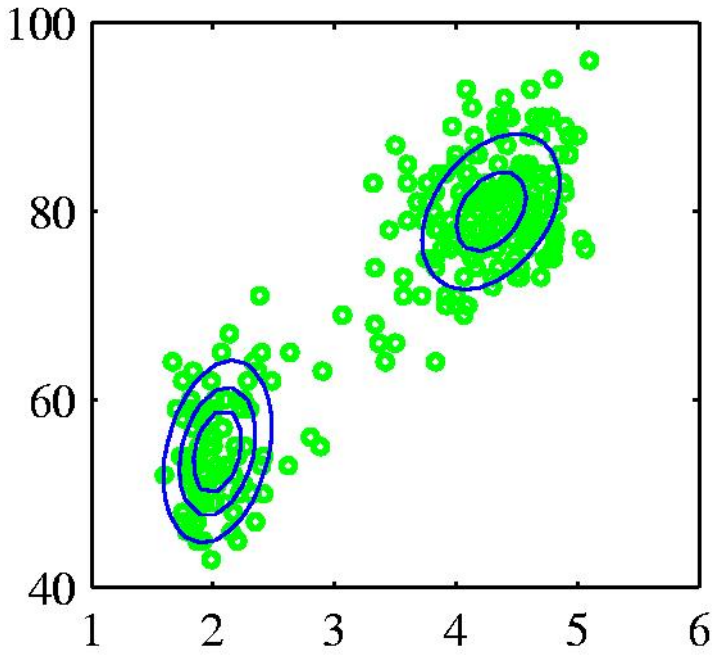


Mixtures of Gaussians (1)

Old Faithful Data Set



Single Gaussian



Mixture of two Gaussians

Mixtures of Gaussians (2)

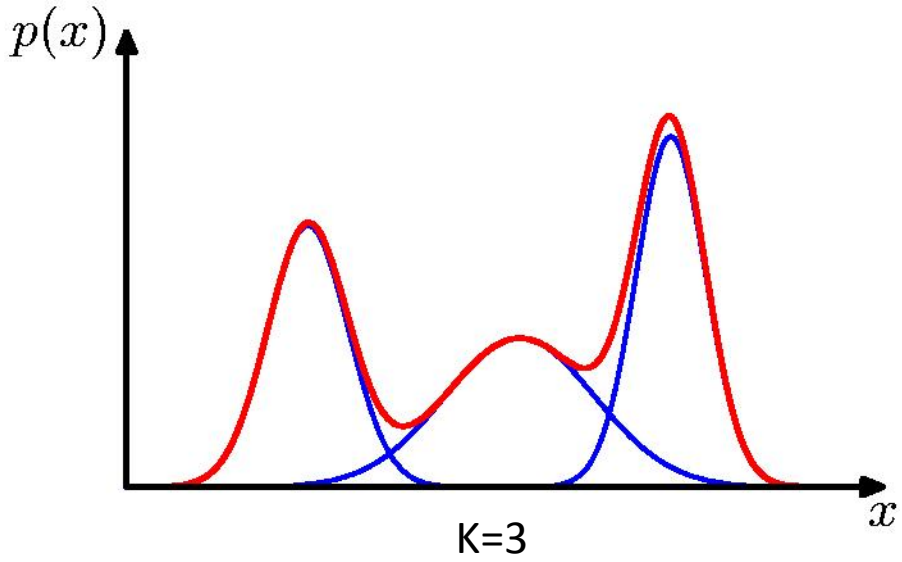
Combine simple models into a complex model:

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

Component

Mixing coefficient

$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$



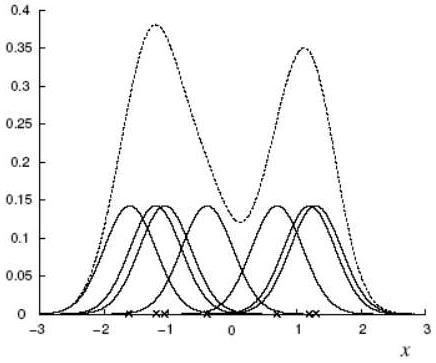
K=3

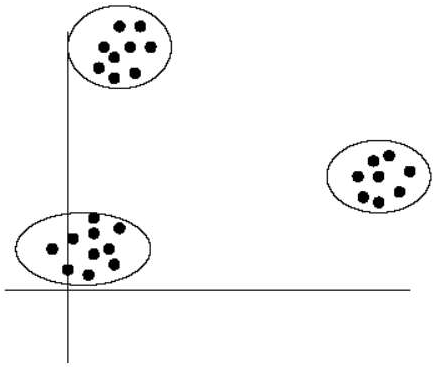
Gaussian Mixture Models (GMMs)

- Consider a mixture of K Gaussian components:

$$p(x_n|\mu,\Sigma) = \sum_k \pi_k N(x,|\ \mu_k,\Sigma_k)$$

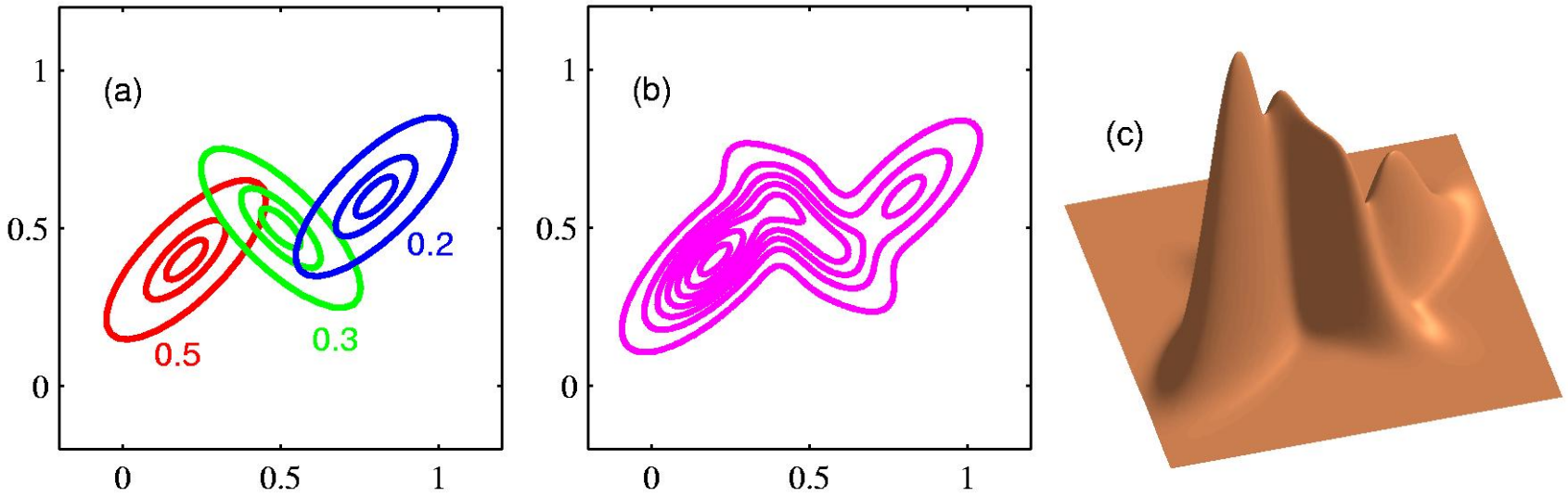
mixture proportion mixture component





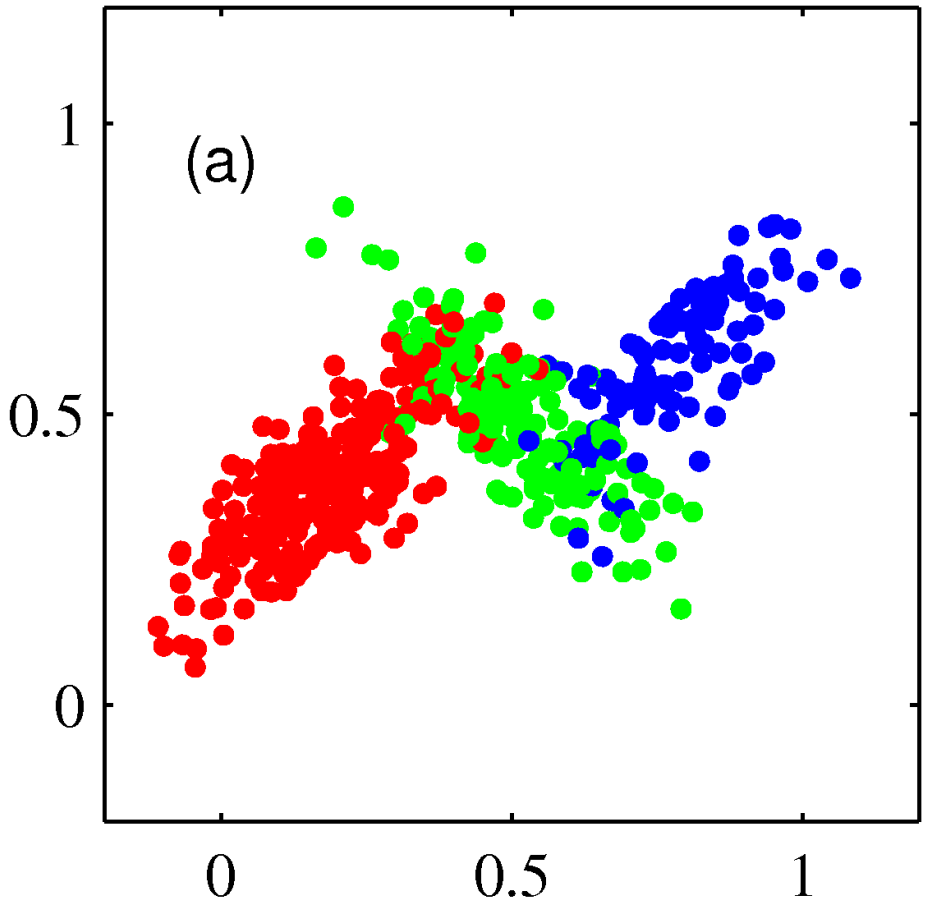
- This model can be used for unsupervised clustering.

Mixtures of Gaussians (3)



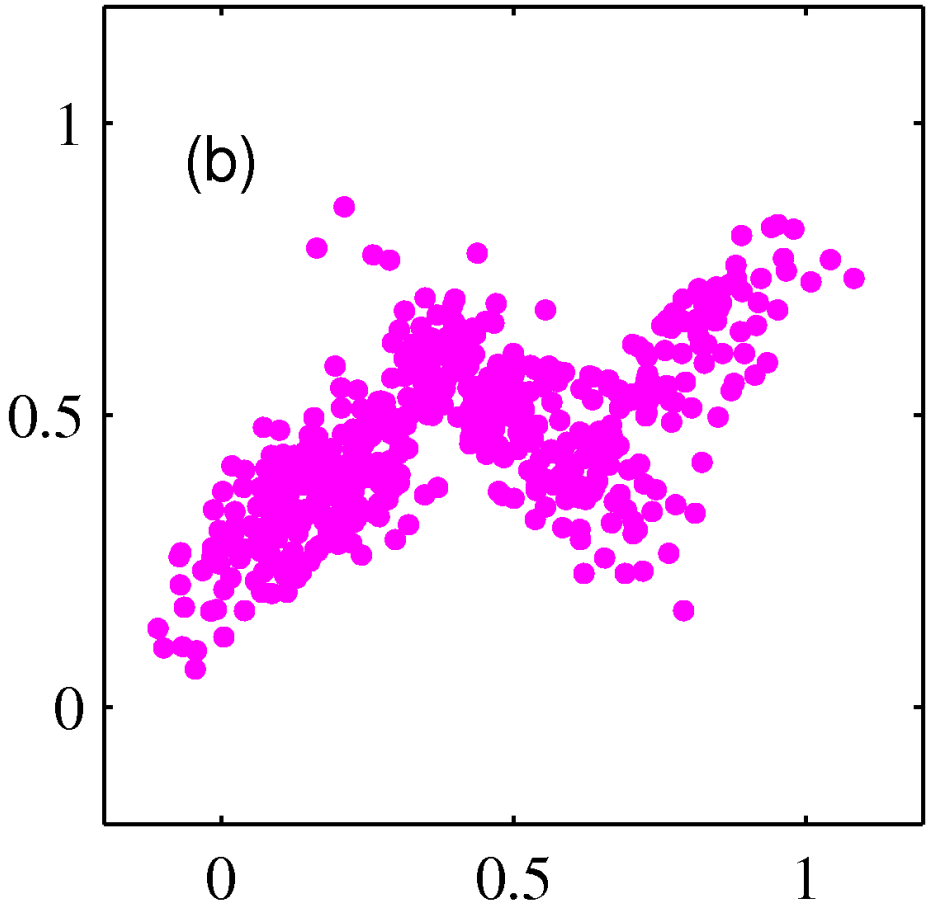
Eliminating Hard Assignments to Clusters

Model data as mixture of multivariate Gaussians



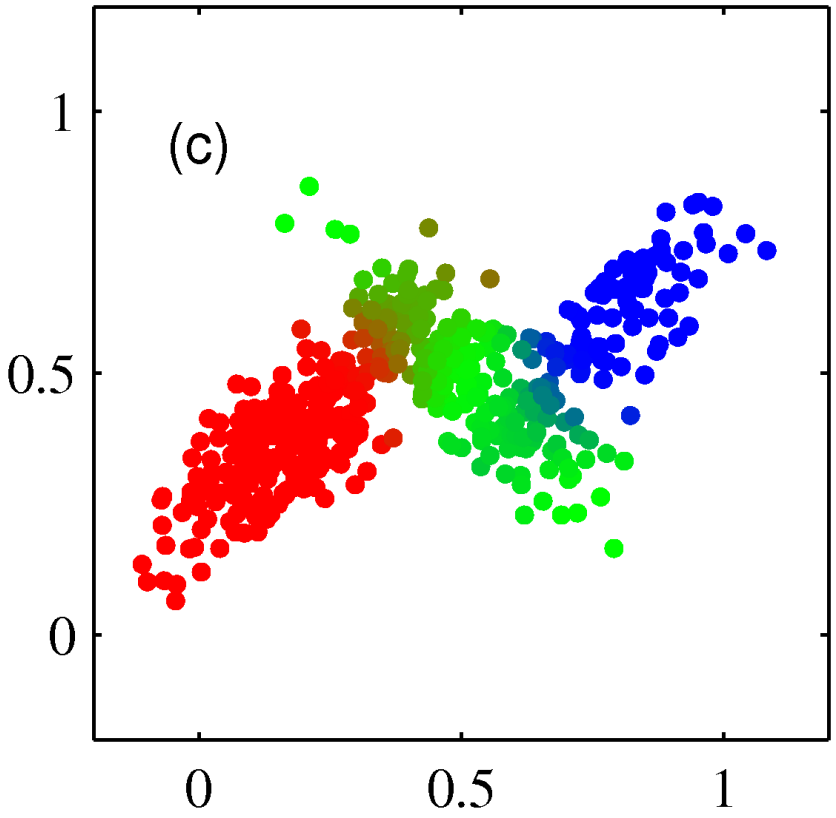
Eliminating Hard Assignments to Clusters

Model data as mixture of multivariate Gaussians



Eliminating Hard Assignments to Clusters

Model data as mixture of multivariate Gaussians



Shown is the *posterior probability* that a point was generated from ith Gaussian: $\Pr(Y = i \mid x)$

ML estimation in **supervised** setting

- Univariate Gaussian

$$\mu_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \sigma^2_{MLE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$$

- ***Mixture*** of ***Multi***variate Gaussians

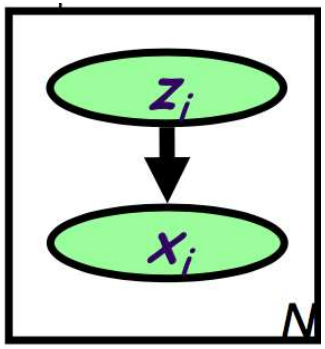
ML estimate for each of the Multivariate Gaussians is given by:

$$\mu^k_{ML} = \frac{1}{n}\sum_{j=1}^{n} x_n \qquad \Sigma^k_{ML} = \frac{1}{n}\sum_{j=1}^{n}\left(\mathbf{x}_j - \mu^k_{ML}\right)\left(\mathbf{x}_j - \mu^k_{ML}\right)^T$$

Just sums over *x* generated from the *k*'th Gaussian

Gaussian Discriminative Analysis

- Data log-likelihood



$$l(\theta; D) = log \prod_n p(z_n, x_n) = log \prod_n p(z_n|\pi) p(x_n|z_n, \mu, \sigma)$$

$$= \sum_n log \prod_k \pi_k^{z_n^k} + \sum_n log \prod_k N(x_n; \mu_k, \sigma)^{z_n^k}$$

$$= \sum_n \sum_k z_n^k log \pi_k - \sum_n \sum_k z_n^k \frac{1}{2\sigma^2}(x_n - \mu_k)^2 + C$$

- MLE

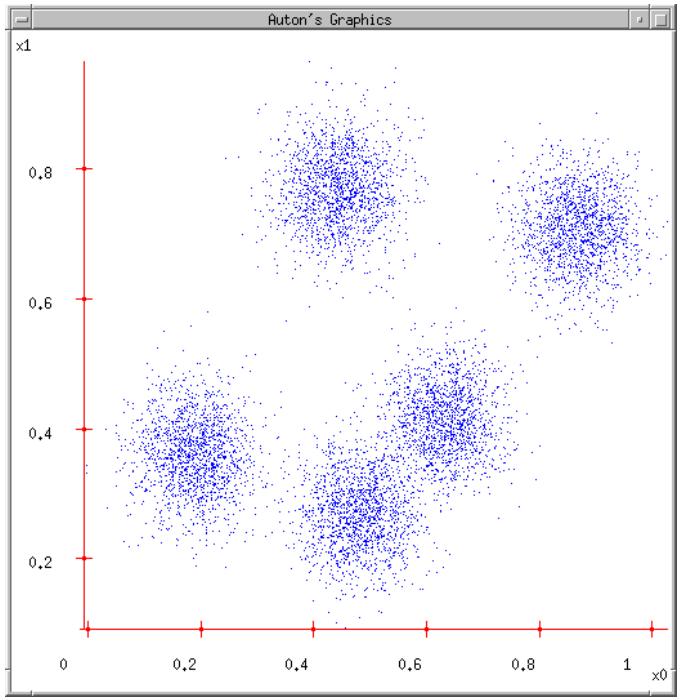
$$\hat{\pi}_{k,MLE} = argmax_\pi \; l(\theta; D)$$
$$\hat{\mu}_{k,MLE} = argmax_\mu \; l(\theta; D)$$
$$\hat{\sigma}_{k,MLE} = argmax_\sigma \; l(\theta; D)$$

$$\hat{\mu}_{k,MLE} = \frac{\sum_n z_n^k x_n}{\sum_n z_n^k}$$

$$p(y_n^k = 1|x_n, \mu, \sigma) = \frac{\pi_k \frac{1}{(2\pi\sigma^2)^{1/2}} exp\left\{-\frac{1}{2\sigma^2}(x_n - \mu_k)^2\right\}}{\sum_{k'} \pi_{k'} \frac{1}{(2\pi\sigma^2)^{1/2}} exp\left\{-\frac{1}{2\sigma^2}(x_n - \mu_{k'})^2\right\}}$$

- What if we do not know z_n ?

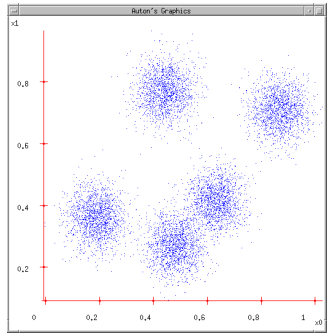
But what if ***unobserved data***?



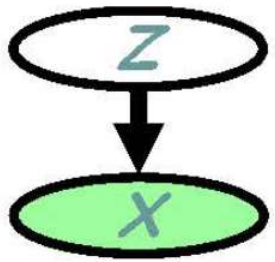
- MLE:
  - $\text{argmax}_\theta \prod_j P(y_j, x_j)$
  - θ: all model parameters
    - eg, class probs, means, and variances
- But we don't know y_j's!!!
- Maximize ***marginal likelihood***:
  - $\text{argmax}_\theta \prod_j P(x_j) = \text{argmax} \prod_j \sum_{k=1}^{K} P(Y_j=k, x_j)$

How do we optimize? Closed Form?

- Maximize ***marginal likelihood***:
  - $\text{argmax}_\theta \prod_j P(x_j) = \text{argmax} \prod_j \sum_{k=1}^{K} P(Y_j=k, x_j)$
- Almost always a hard problem!
  - Usually no closed form solution
  - Even when lgP(X,Y) is convex, lgP(X) generally isn't…
  - For all but the simplest P(X), we will have to do gradient ascent, in a big messy space with lots of local optimum…



GMM Derivations



- Consider a mixture of K Gaussian components:
  - Z is a latent class indicator vector:

$$p(z_n) = multi(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n|z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} exp\left\{-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

  - The likelihood of a sample:

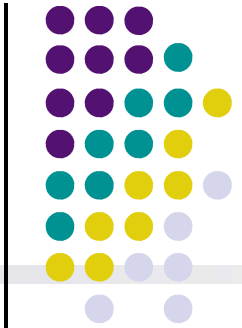
mixture component

mixture proportion

$$p(x_n|\mu, \Sigma) = \sum_k p(z^k = 1|\pi) p(x|z^k = 1, \mu, \Sigma)$$

$$= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n; \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x|\mu_k, \Sigma_k)$$

Learning Mixture Models



- Given data

$$D = \{x_n\}_{n=1}^{N}$$

- Likelihood:

$$L(\pi, \mu, \Sigma; D) = \prod_n p(x_n|\pi, \mu, \Sigma) = \prod_n \left(\sum_k \pi_k N(x|\mu_k, \Sigma_k) \right)$$

$$\{\pi^*, \mu^*, \Sigma^*\} = argmax\ L(\pi, \mu, \Sigma, D)$$

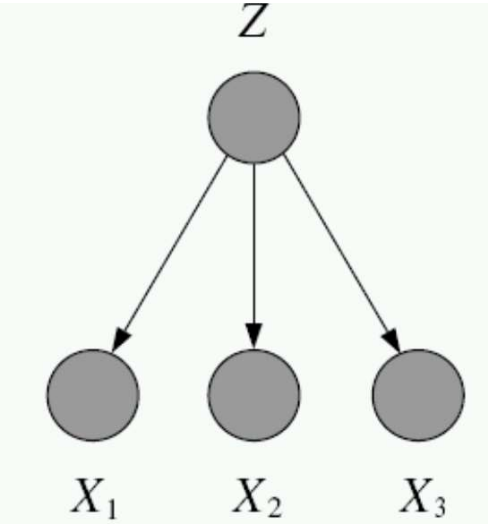
Why is Learning Harder ?

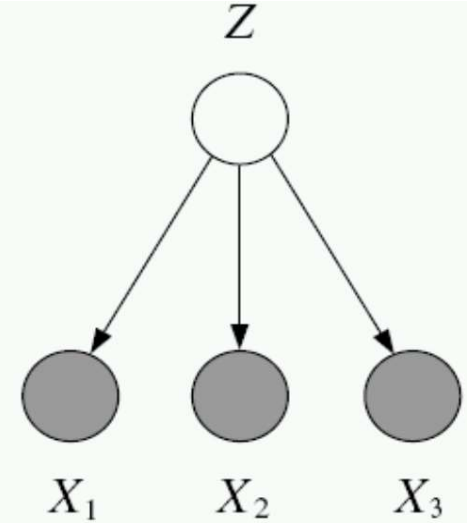
- In fully observed iid settings, the log likelihood decomposes into a sum of local terms.

$$l_c(\theta; D) = log\ p(x, z|\theta) = log\ p(z|\theta_z) + log\ p(x|z, \theta_x)$$

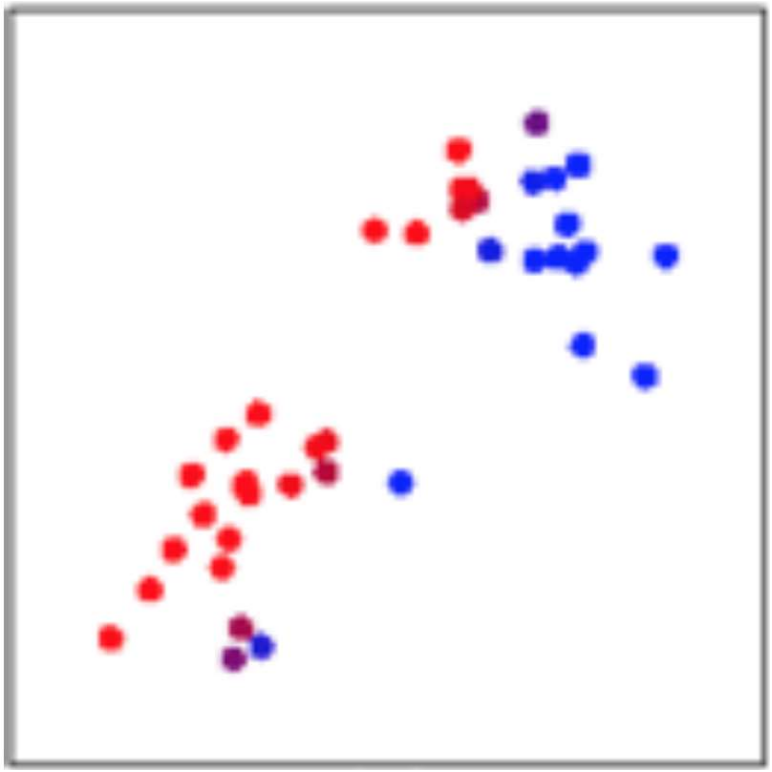
- With latent variables, all the parameters become coupled together via marginalization

$$l_c(\theta; D) = log \sum_z p(x, z|\theta) = log \sum_z p(z|\theta_z) p(x|z, \theta_x)$$





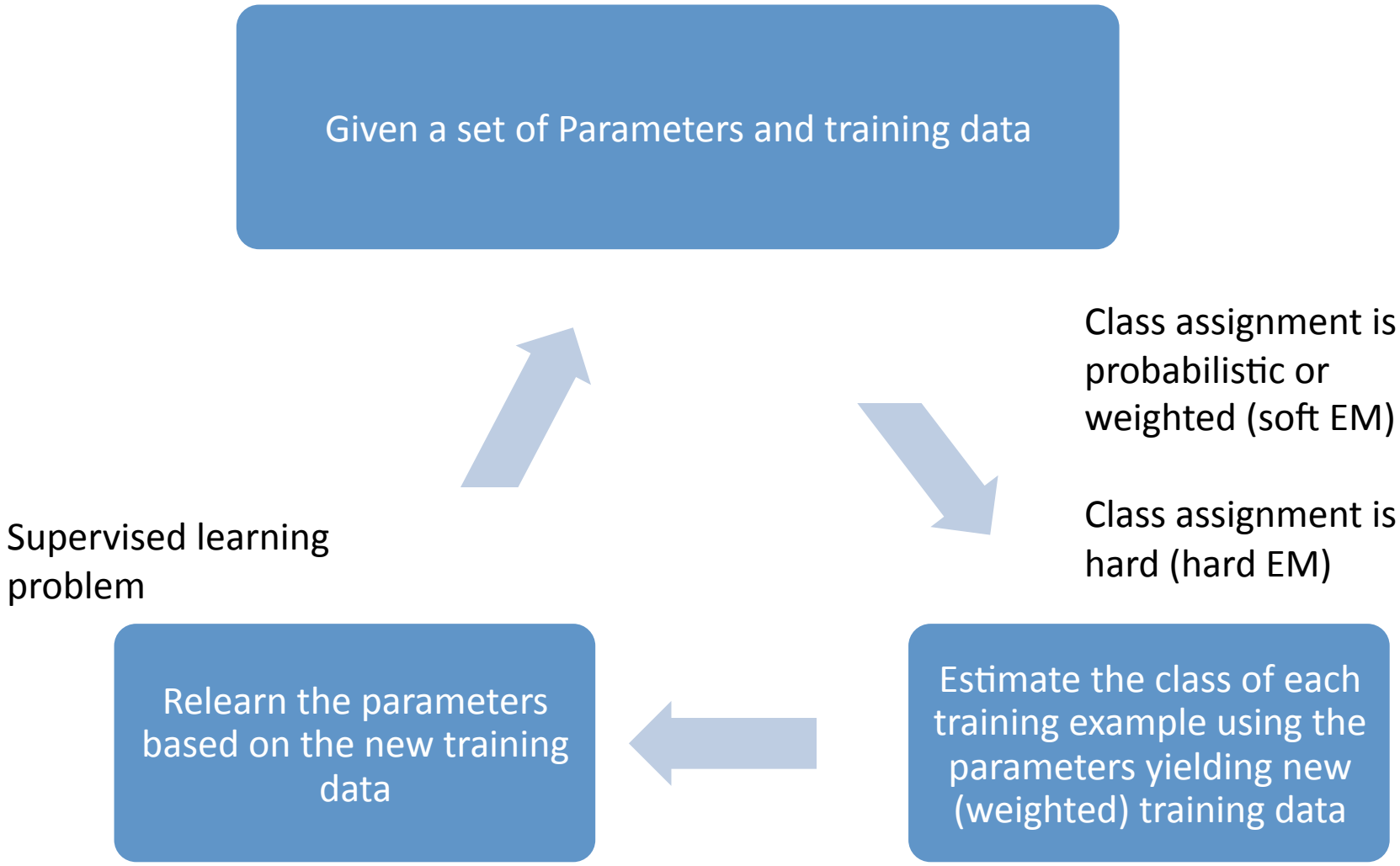
The Expectation-Maximization (EM) Algorithm



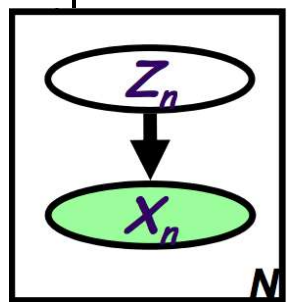
The EM Algorithm

- A clever method for maximizing marginal likelihood:
  - A type of gradient ascent that can be easy to implement (eg, no line search, learning rates, etc.)
- Alternate between two steps:
  - Compute an expectation
  - Compute a maximization
- Not magic: ***still optimizing a non-convex function with lots of local optima***
  - The computations are just easier (often, significantly so!)

EM algorithm: Pictorial View



EM Algorithm for GMM



- E.g., A mixture of K Gaussians:
  - Z is a latent class indicator vector:

$$p(z_n) = multi(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

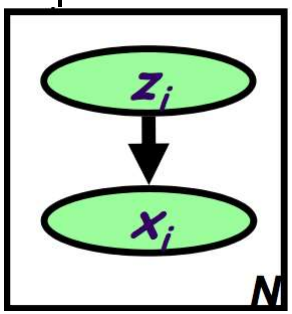
$$p(x_n|z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} exp\left\{-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

  - The likelihood of a sample:

$$p(x_n|\mu, \Sigma) = \sum_k p(z^k = 1|\pi) p(x|z^k = 1, \mu, \Sigma)$$

$$= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n; \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x|\mu_k, \Sigma_k)$$

EM Algorithm for GMM



- Recall MLE for completely observed data
- Data log-likelihood

$$l(\theta; D) = log \prod_n p(z_n, x_n) = log \prod_n p(z_n|\pi) p(x_n|z_n, \mu, \sigma)$$

$$= \sum_n log \prod_k \pi_k^{z_n^k} + \sum_n log \prod_k N(x_n; \mu_k, \sigma)^{z_n^k}$$

$$= \sum_n \sum_k z_n^k log \pi_k - \sum_n \sum_k z_n^k \frac{1}{2\sigma^2}(x_n - \mu_k)^2 + C$$

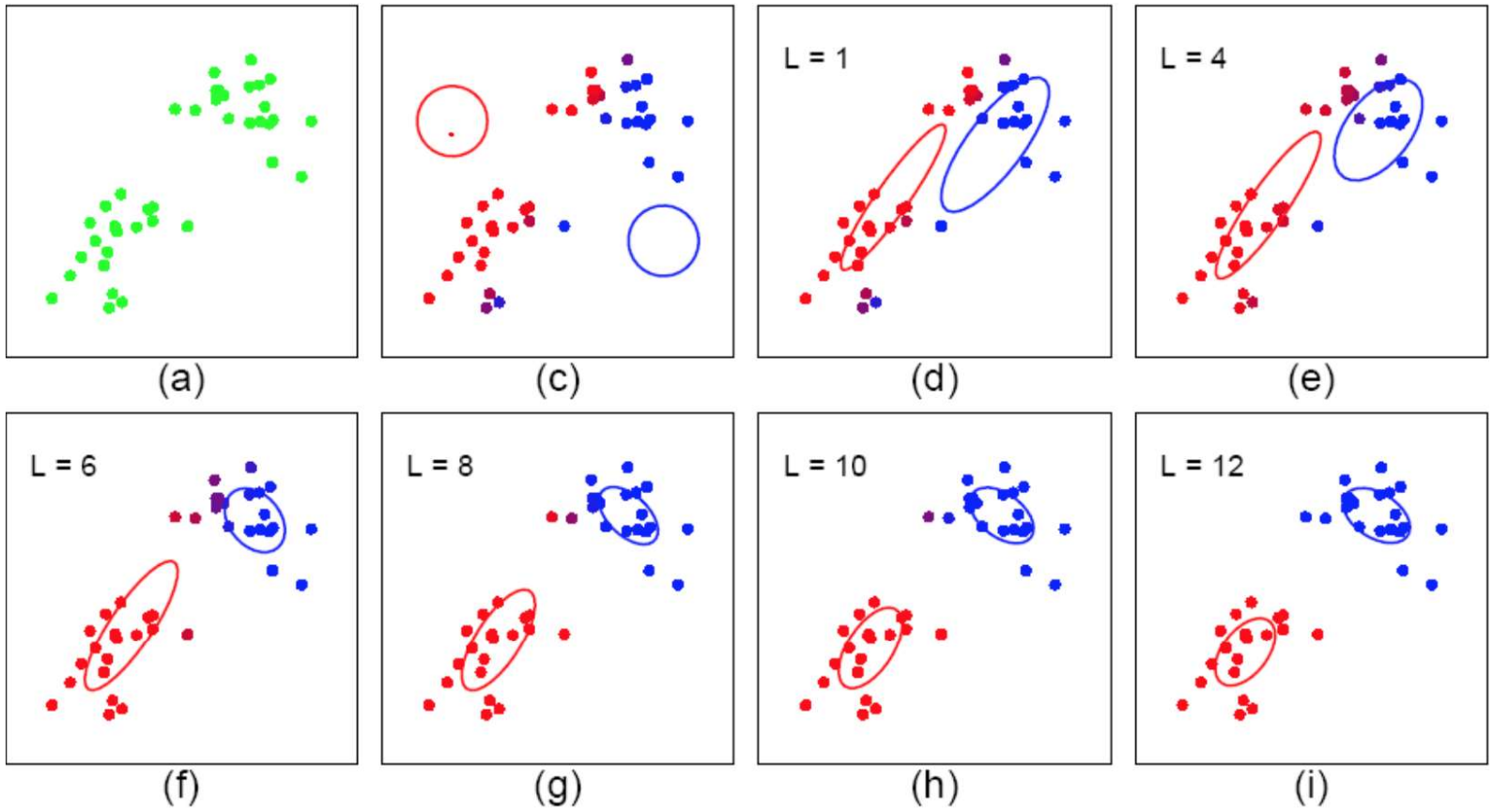
- MLE

$$\hat{\pi}_{k,MLE} = argmax_\pi \; l(\theta; D)$$
$$\hat{\mu}_{k,MLE} = argmax_\mu \; l(\theta; D) \quad \Rightarrow \quad \hat{\mu}_{k,MLE} = \frac{\sum_n z_n^k x_n}{\sum_n z_n^k}$$
$$\hat{\sigma}_{k,MLE} = argmax_\sigma \; l(\theta; D)$$

- What if we do not know z_n ?

EM Algorithm for GMM

- Start:
  - "Guess" the centroid μ_k and covariance Σ_k of each of the K clusters
- Loop



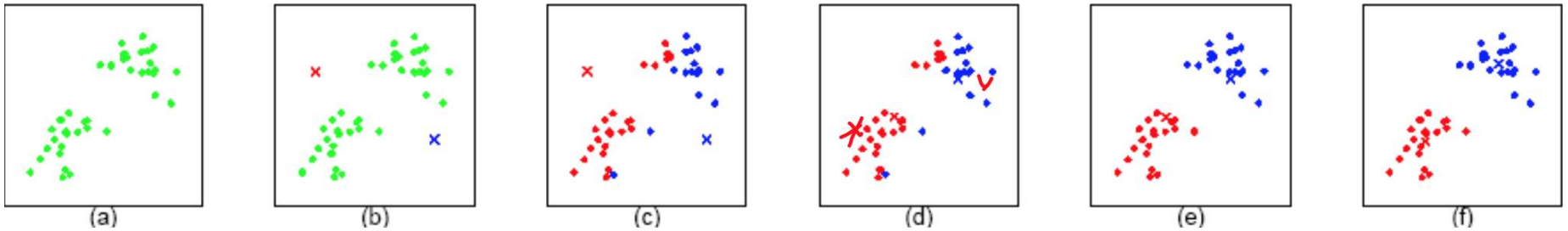
Comparing to K-means

- Start:
  - "Guess" the centroid μ_k and covariance Σ_k of each of the K clusters
- Loop
  - For each point n = 1 to N, computer its cluster label:

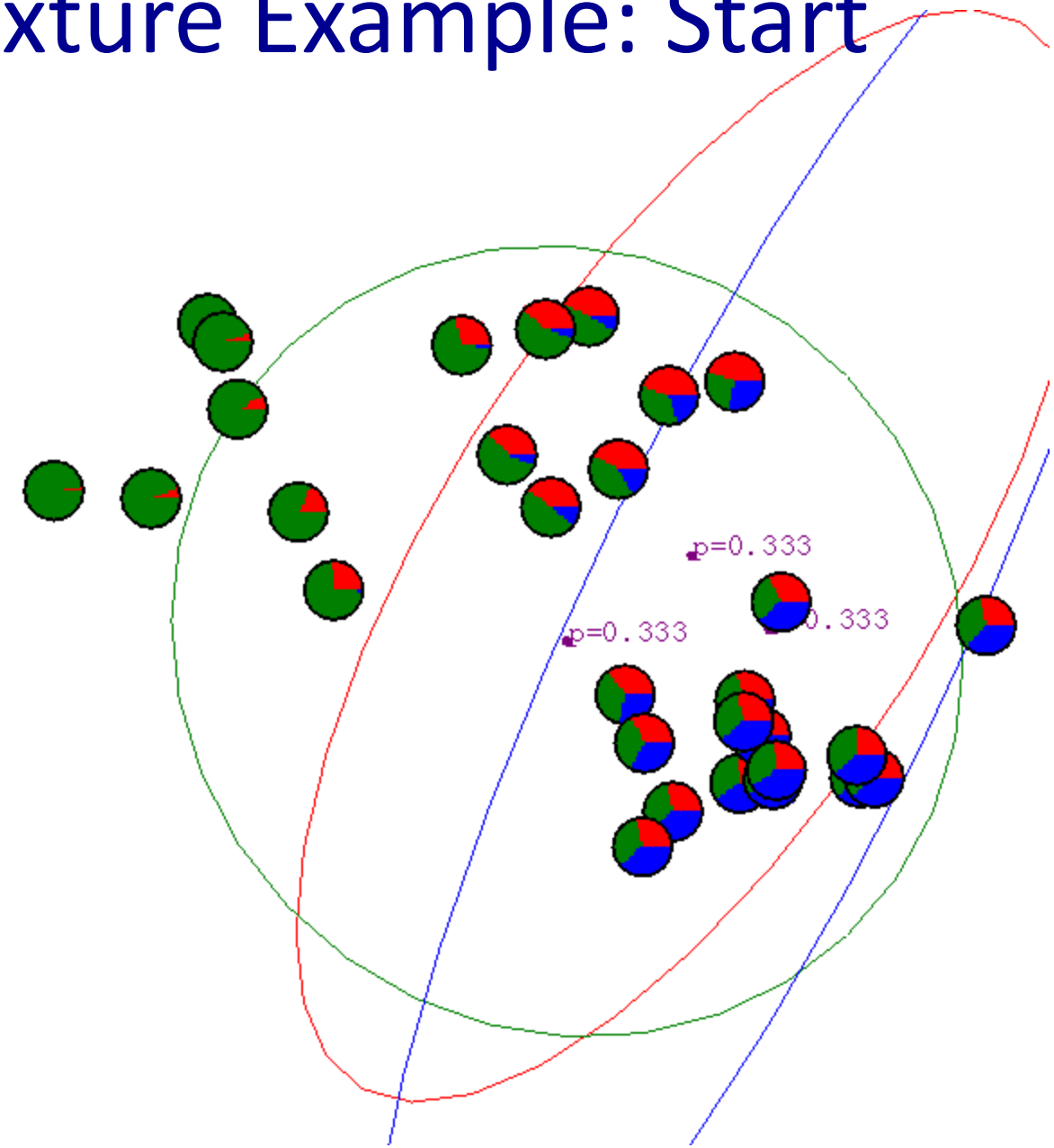
$$z_n^{(t)} = \underset{k}{argmax}(x_n - \mu_k^{(t)})^T \Sigma_k^{-1(t)} (x_n - \mu_k^{(t)})$$

  - For each cluster k = 1 : K

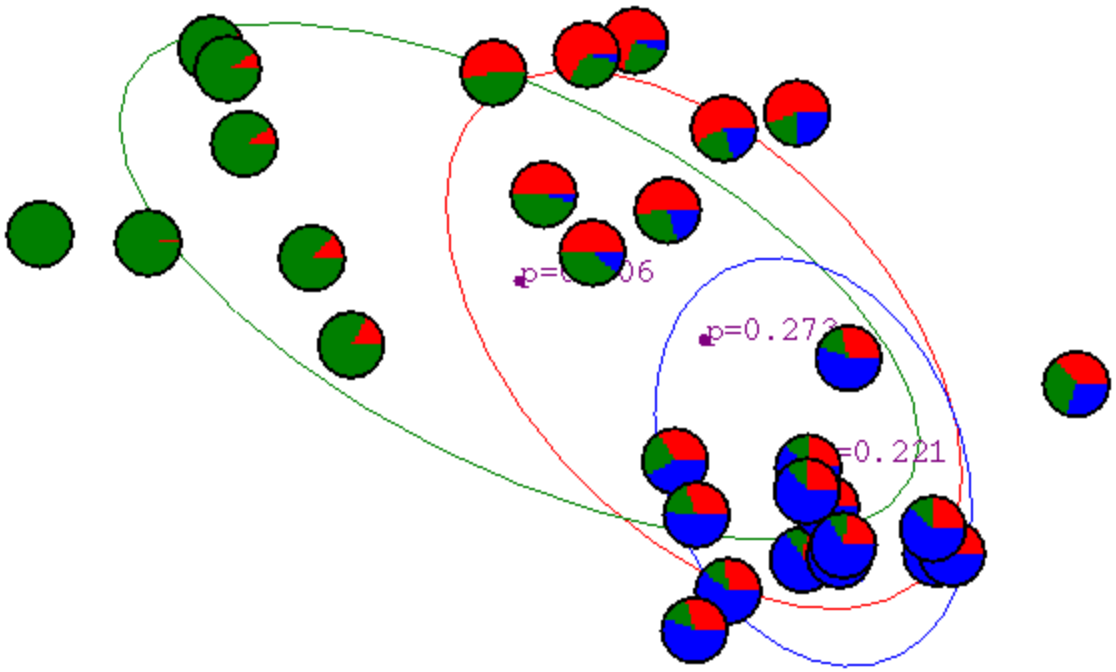
$$\mu_k^{(t+1)} = \frac{\sum_n \delta(z_n^{(t)}, k) x_n}{\sum_n \delta(z_n^{(t)}, k)} \qquad \Sigma_k^{(t+1)} = \ldots$$



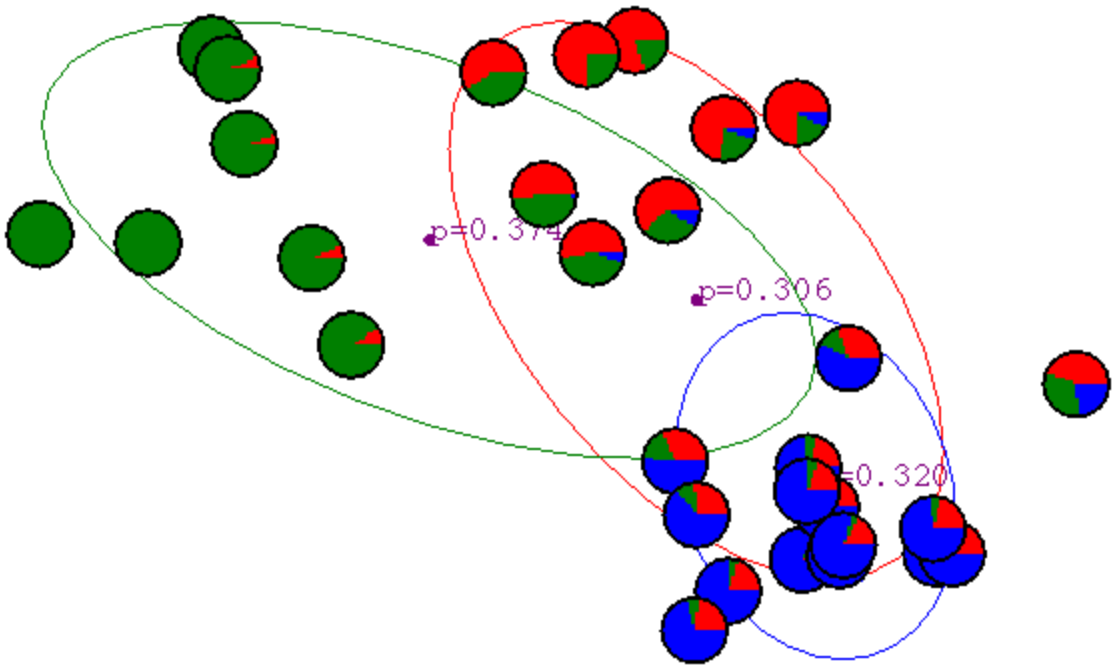
Gaussian Mixture Example: Start



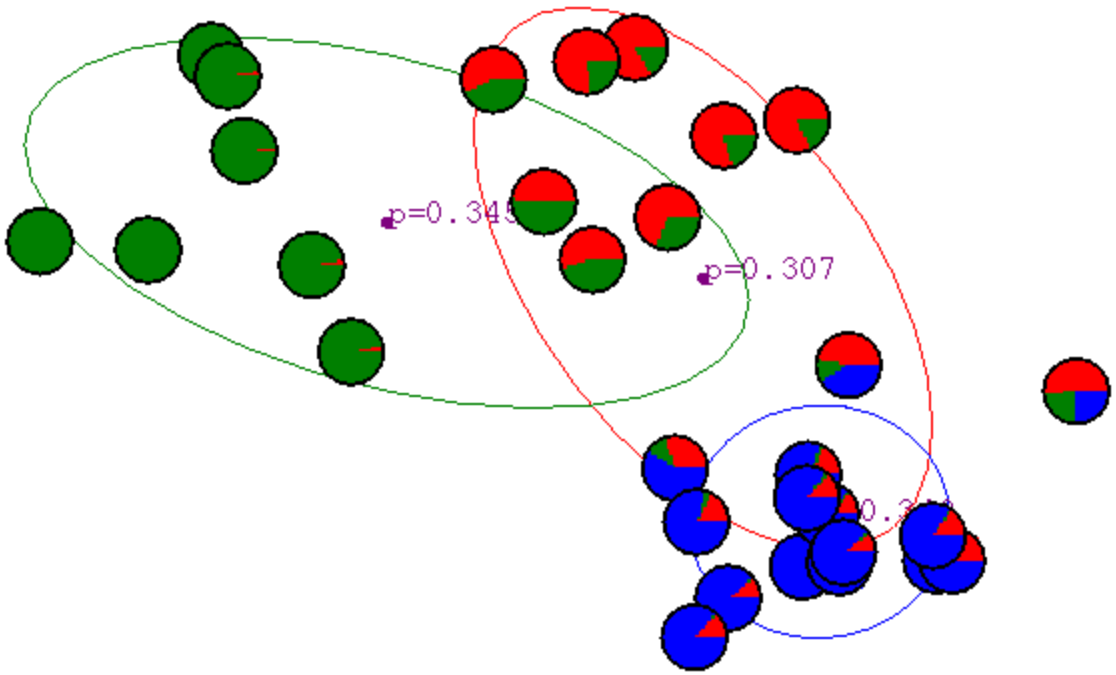
After first iteration



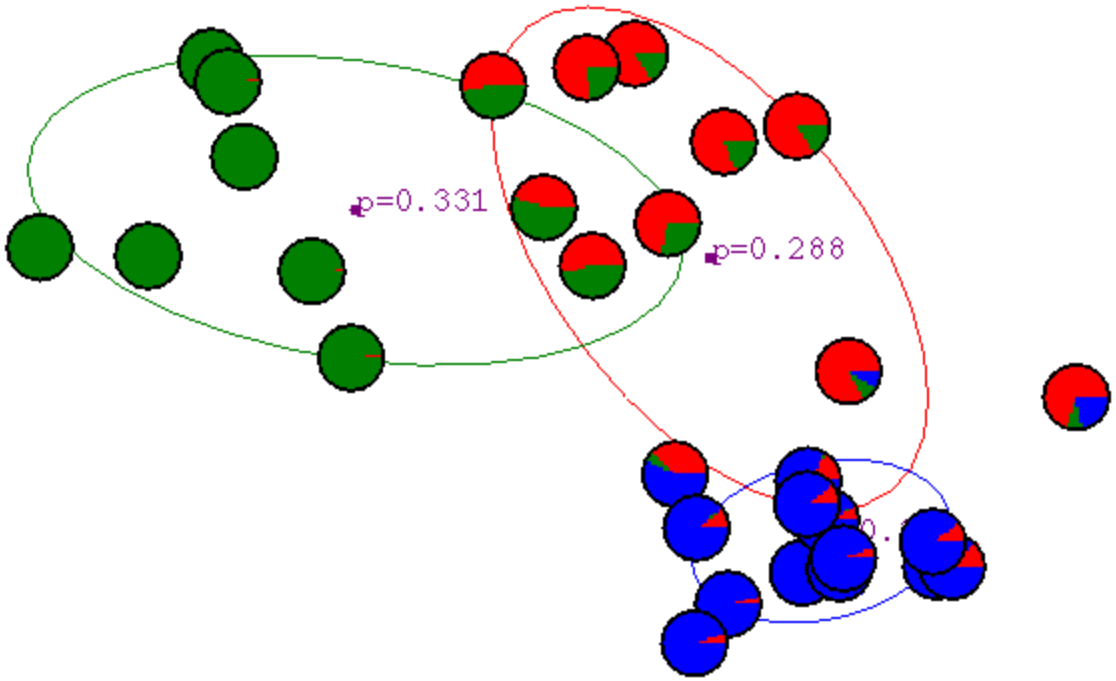
After 2nd iteration



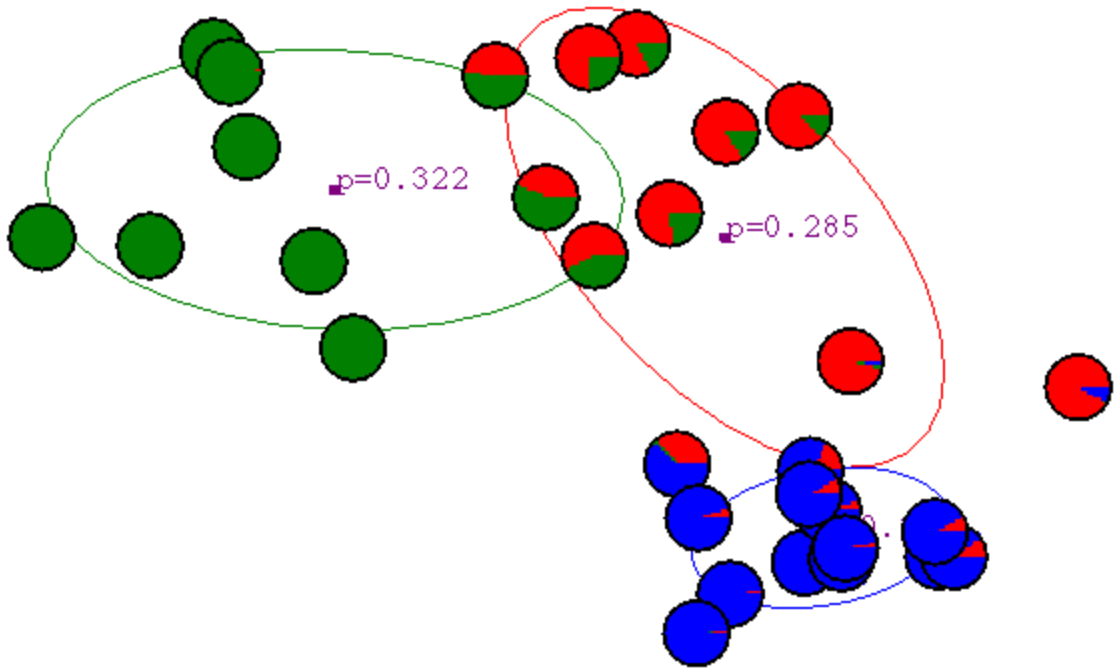
After 3rd iteration



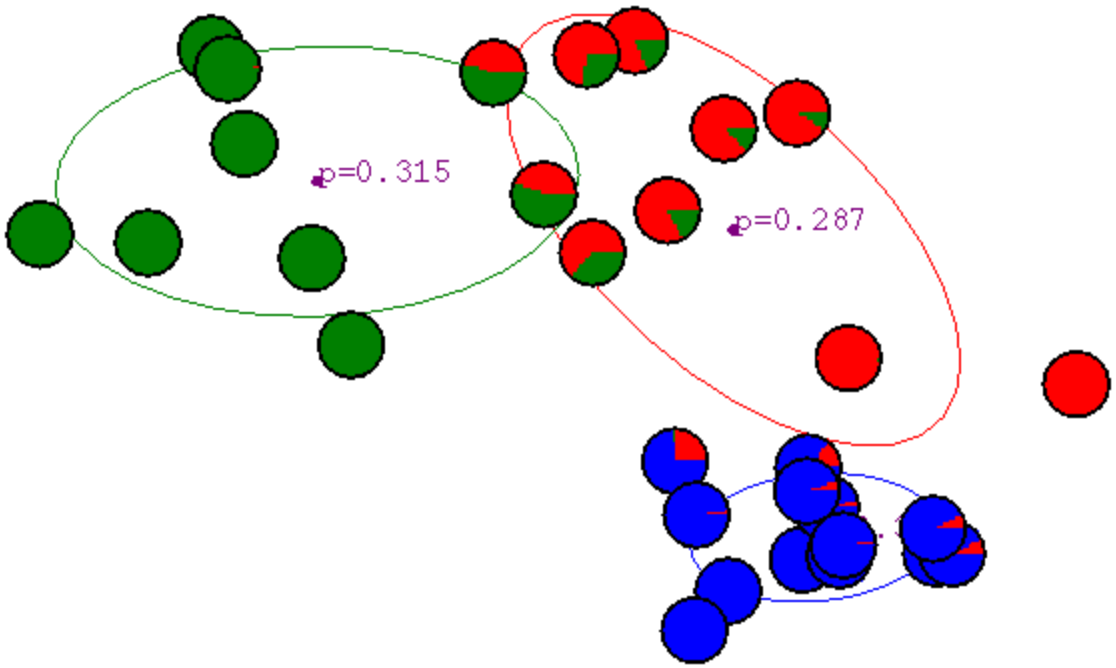
After 4th iteration



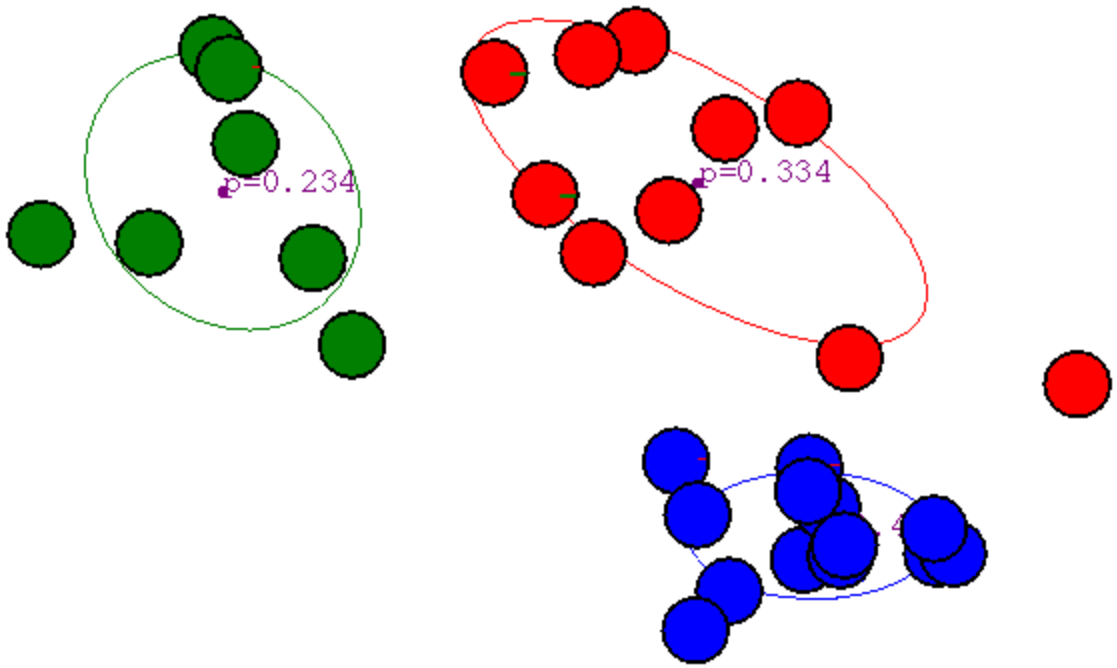
After 5th iteration



After 6th iteration



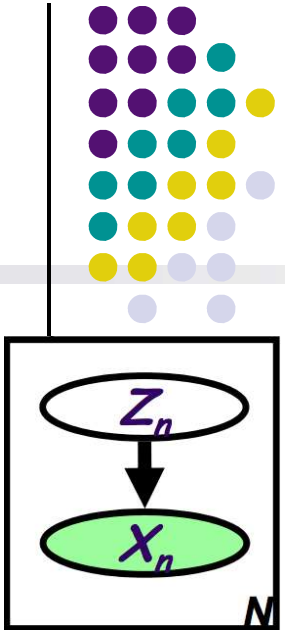
After 20th iteration



Notes on EM Algorithm

- EM is an optimization strategy for objective functions that can be interpreted as likelihoods in the presence of missing data.
- It is much simpler than gradient methods:
  - No need to choose step size.
  - Enforces constraints automatically.
  - Calls interence and fully observed learning as subroutines.
- EM is an iterative algorithm with two linked steps:
  - E-step: fill-in hidden values using inference, $p(z|x,\theta)$.
  - M-step: update parameters t+1 using standard MLE/MAP method applied to completed data
- We will prove that this procedure monotonically improves (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood

How Is EM Derived ?



- A mixture of K Gaussians:
  - Z is a latent class indicator vector: $p(z_n) = multi(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$
  - X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n|z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} exp\left\{-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

  - The likelihood of a sample:

$$p(x_n|\mu, \Sigma) = \sum_k p(z^k = 1|\pi) p(x|z^k = 1, \mu, \Sigma)$$

$$= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n; \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x|\mu_k, \Sigma_k)$$

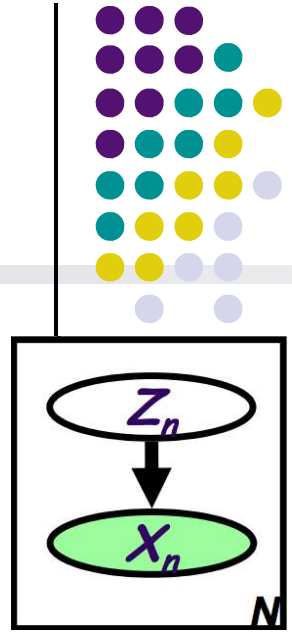
- The "complete" likelihood

$$p(x_n, z_n^k = 1|\mu, \Sigma) = p(z_n^k = 1|\pi) p(x|z_n^k = 1, \mu, \Sigma) = \pi_k N(x|\mu_k, \Sigma_k)$$

$$p(x_n, z_n|\mu, \Sigma) = \prod_k [\pi_k N(x|\mu_k, \Sigma_k)]^{z_n^k}$$

But this is itself a random variable ! Not good as objective function

How Is EM Derived ?



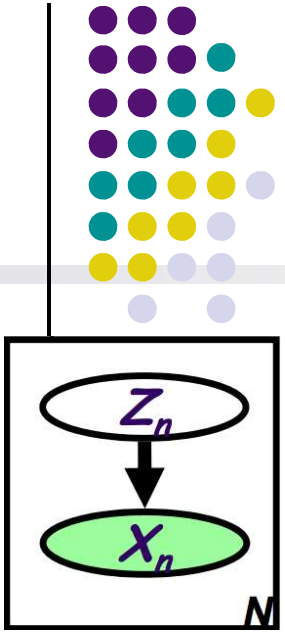
- The complete log likelihood:

$$l(\theta; D) = log \prod_n p(z_n, x_n) = log \prod_n p(z_n|\pi) p(x_n|z_n, \mu, \sigma)$$
$$= \sum_n log \prod_k \pi_k^{z_n^k} + \sum_n log \prod_k N(x_n; \mu_k, \sigma)^{z_n^k}$$
$$= \sum_n \sum_k z_n^k log \pi_k - \frac{1}{2} \sum_n \sum_k z_n^k \left((x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + log|\Sigma_k| + C \right)$$

- The expected complete log likelihood

$$< l_c(\theta; x, z) > = \sum_n < log\ p(z_n|\pi) >_{p(z|x)} + \sum_n < log\ p(x_n|z_n, \mu, \Sigma) >_{p(z|x)}$$
$$= \sum_n \sum_k < z_n^k > log \pi_k - \frac{1}{2} \sum_n \sum_k < z_n^k > \left((x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + log|\Sigma_k| + C \right)$$

E-step



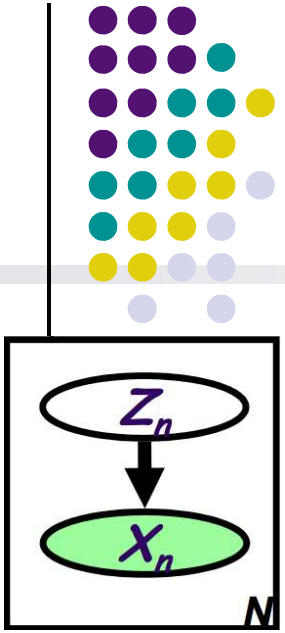
- We maximize $\langle l_c(\theta) \rangle$ iteratively using the following iterative procedure:

— Expectation step: computing the expected value of the sufficient statistics of the hidden variables (i.e., z) given current est. of the parameters (i.e., π and μ).

$$\tau_n^{k(t)} = \left\langle z_n^k \right\rangle_{q^{(t)}} = p(z_n^k = 1 \mid x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n, \mid \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n, \mid \mu_i^{(t)}, \Sigma_i^{(t)})}$$

- Here we are essentially doing **inference**

M-step



- We maximize $\langle l_c(\theta) \rangle$ iteratively using the following iterative procedure:

  — Maximization step: computing the parameters under current results of the expected value of the hidden variables

$$\pi_k^* = \arg\max \langle l_c(\theta) \rangle, \quad \Rightarrow \frac{\partial}{\partial \pi_k} \langle l_c(\theta) \rangle = 0, \forall k, \quad \text{s.t.} \sum_k \pi_k = 1$$

$$\Rightarrow \pi_k^* = \sum_n \langle z_n^k \rangle_{q^{(t)}} / N = \sum_n \tau_n^{k(t)} / N = \langle n_k \rangle / N$$

$$\mu_k^* = \arg\max \langle l(\theta) \rangle, \quad \Rightarrow \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}}$$

$$\Sigma_k^* = \arg\max \langle l(\theta) \rangle, \quad \Rightarrow \Sigma_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T}{\sum_n \tau_n^{k(t)}}$$

Fact :

$$\frac{\partial \log |A^{-1}|}{\partial A^{-1}} = A^T$$

$$\frac{\partial x^T A x}{\partial A} = x x^T$$

- This is isomorphic to **MLE** except that the variables that are hidden are replaced by their expectations (in general they will be replaced by their corresponding "**sufficient statistics**")

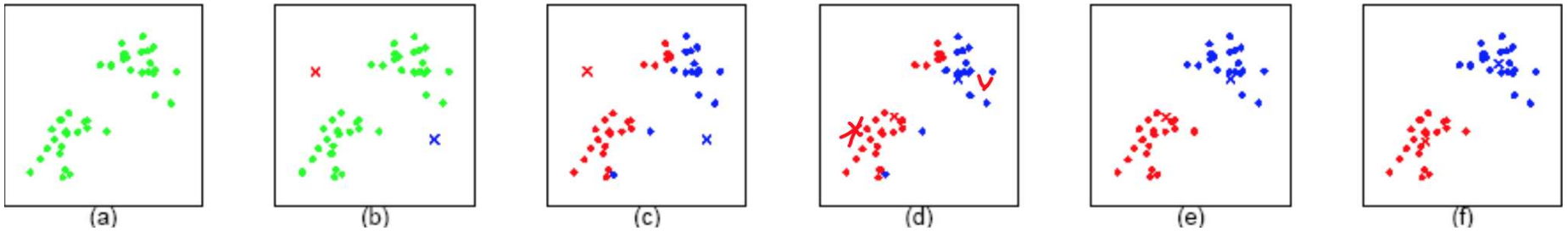
Compare: K-means

- The EM algorithm for mixtures of Gaussians is like a "soft version" of the K-means algorithm.
- In the K-means "E-step" we do hard assignment:

$$z_n^{(t)} = \arg\max_k (x_n - \mu_k^{(t)})^T \Sigma_k^{-1(t)} (x_n - \mu_k^{(t)})$$

- In the K-means "M-step" we update the means as the weighted sum of the data, but now the weights are 0 or 1:

$$\mu_k^{(t+1)} = \frac{\sum_n \delta(z_n^{(t)}, k) x_n}{\sum_n \delta(z_n^{(t)}, k)}$$



Theory Underlying EM

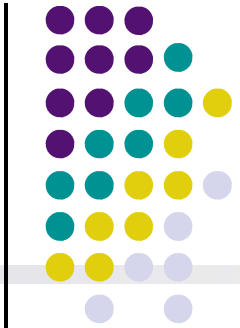
- What are we doing ?
- Recall that according to MLE, we intend to learn the model parameter that would have maximize the likelihood of the data.
- But we do not observe z, so computing

$$l_c(\theta; D) = log \sum_z p(x, z|\theta) = log \sum_z p(z|\theta_z) p(x|z, \theta_x)$$

  is difficult !

- What shall we do ?

Complete & Incomplete Log Likelihoods



- Complete log likelihood

  Let X denote the observable variable(s), and Z denote the latent variable(s). If Z could be observed, then

$$l_c(\theta; x, z) \stackrel{def}{=} log\ p(x, z|\theta)$$

  - Usually, optimizing $l_c()$ given both z and x is straightforward (c.f. MLE for fully observed models).
  - Recalled that in this case the objective for, e.g., MLE, decomposes into a sum of factors, the parameter for each factor can be estimated separately.
  - **But given that Z is not observed, lc() is a random quantity, cannot be maximized directly.**

- Incomplete log likelihood

  With z unobserved, our objective becomes the log of a marginal probability:

$$l_c(\theta; x) = log\ p(x|\theta) = log \sum_z p(x, z|\theta)$$

  - **This objective won't decouple**

Expected Complete Log Likelihood

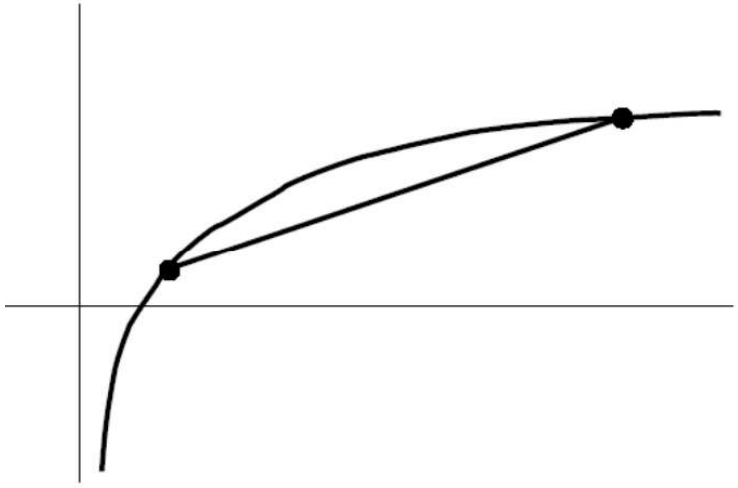
- For any distribution q(z), define expected complete log likelihood:

$$\left\langle \ell_c(\theta; x, z) \right\rangle_q \stackrel{\text{def}}{=} \sum_z q(z \mid x, \theta) \log p(x, z \mid \theta)$$

  - A deterministic function of θ
  - Linear in $l_c()$ — inherit its factorizability
  - Does maximizing this surrogate yield a maximizer of the likelihood ?

- Jensen's inequality

$$\begin{aligned}
\ell(\theta; x) &= \log p(x \mid \theta) \\
&= \log \sum_z p(x, z \mid \theta) \\
&= \log \sum_z q(z \mid x) \frac{p(x, z \mid \theta)}{q(z \mid x)} \\
&\geq \sum_z q(z \mid x) \log \frac{p(x, z \mid \theta)}{q(z \mid x)}
\end{aligned}$$



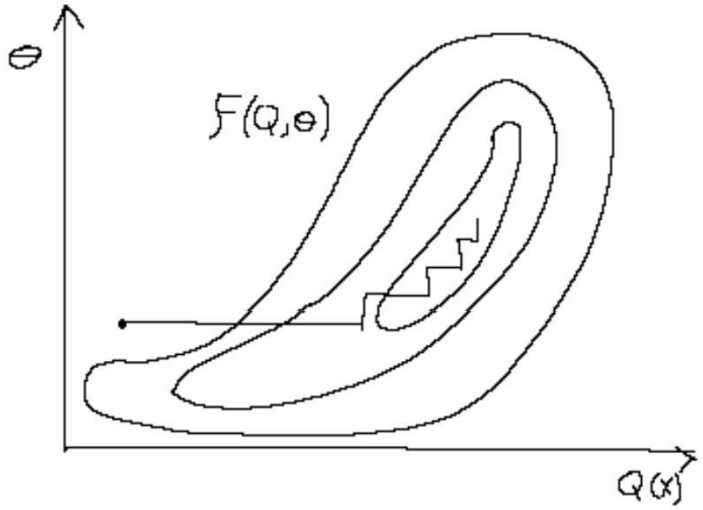
$$\Rightarrow \quad \ell(\theta; x) \geq \left\langle \ell_c(\theta; x, z) \right\rangle_q + H_q$$

Lower Bounds and Free Energy

- For fixed data x, define a functional called the free energy:

$$F(q,\theta) \overset{\text{def}}{=} \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)} \leq \ell(\theta; x)$$

- The EM algorithm is coordinate-ascent on F:
  - **E-step:** $q^{t+1} = \arg\max_q F(q, \theta^t)$
  - **M-step:** $\theta^{t+1} = \arg\max_\theta F(q^{t+1}, \theta^t)$



E-step: maximization of expected l_c w.r.t. q

- Claim:

$$q^{t+1} = \arg\max_{q} F(q, \theta^t) = p(z \mid x, \theta^t)$$

  - This is the posterior distribution over the latent variables given the data and the parameters. Often we need this at test time anyway (e.g., to perform classification).

- Proof (easy): this setting attains the bound $I(\theta;x) \geq F(q,\theta)$

$$\begin{aligned} F(p(z|x,\theta^t),\theta^t) &= \sum_z p(z|x,\theta^t) \log \frac{p(x,z \mid \theta^t)}{p(z|x,\theta^t)} \\ &= \sum_z p(z|x,\theta^t) \log p(x \mid \theta^t) \\ &= \log p(x \mid \theta^t) = \ell(\theta^t; x) \end{aligned}$$

- Can also show this result using variational calculus or the fact that $\ell(\theta;x) - F(q,\theta) = \mathrm{KL}(q \,\|\, p(z \mid x, \theta))$

M-step: maximization of expected l_c w.r.t. θ

- Note that the free energy breaks into two terms:

$$\begin{aligned} F(q,\theta) &= \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)} \\ &= \sum_z q(z \mid x) \log p(x,z \mid \theta) - \sum_z q(z \mid x) \log q(z \mid x) \\ &= \left\langle \ell_c(\theta; x, z) \right\rangle_q + H_q \end{aligned}$$

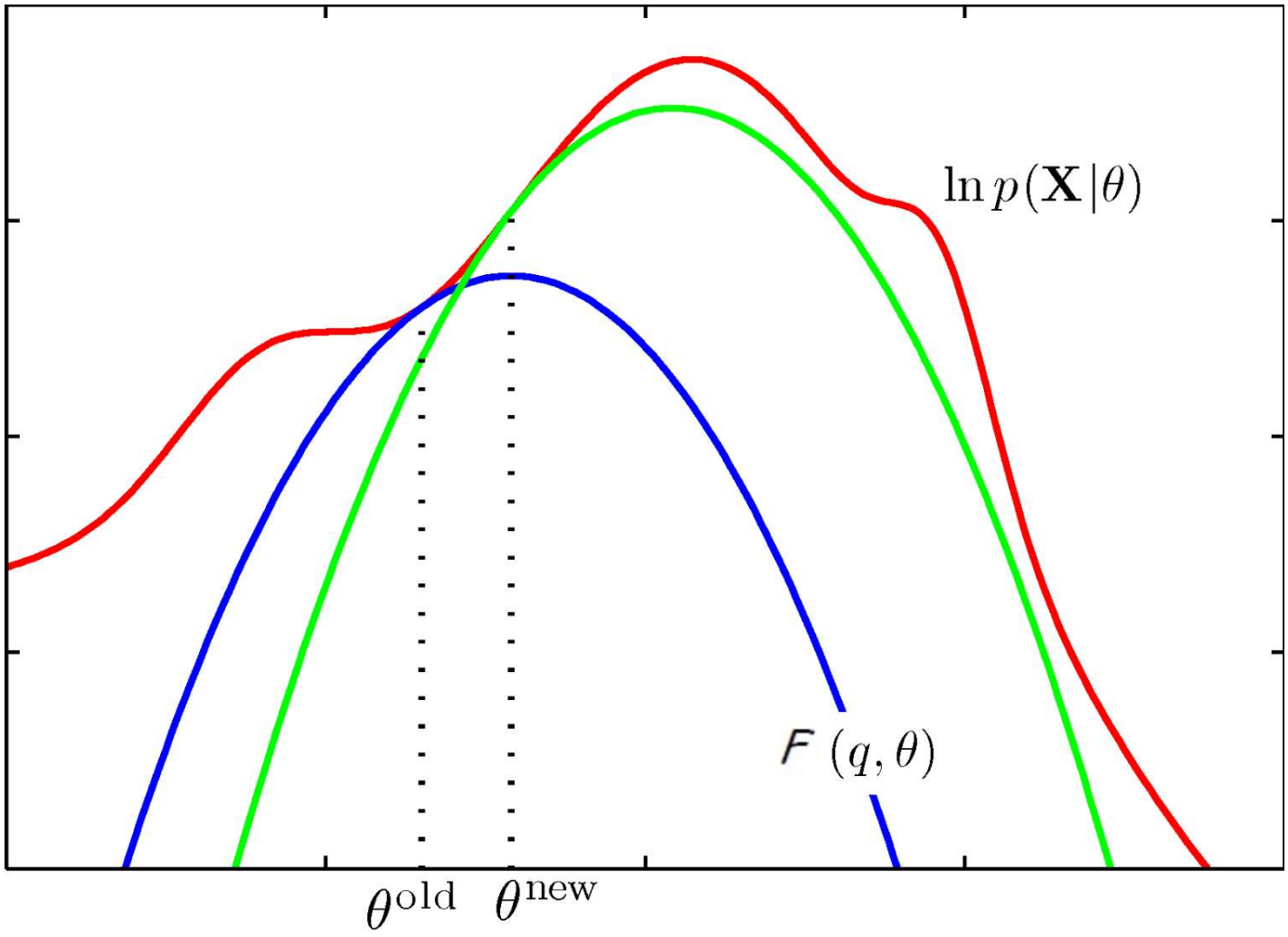
  - The first term is the expected complete log likelihood (energy) and the second term, which does not depend on θ, is the entropy.

- Thus, in the M-step, maximizing with respect to θ for fixed q we only need to consider the first term:

$$\theta^{t+1} = \arg\max_\theta \left\langle \ell_c(\theta; x, z) \right\rangle_{q^{t+1}} = \arg\max_\theta \sum_z q(z \mid x) \log p(x,z \mid \theta)$$

  - Under optimal q^{t+1}, this is equivalent to solving a standard MLE of fully observed model $p(x,z|\theta)$, with the sufficient statistics involving z replaced by their expectations w.r.t. $p(z|x,\theta)$

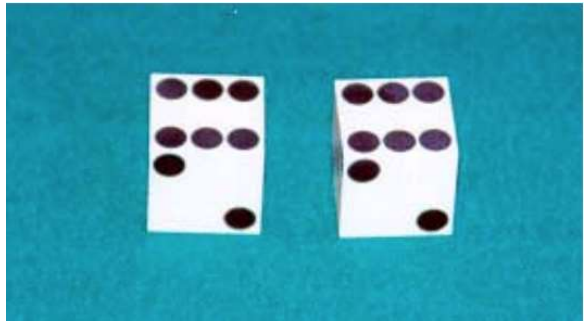
An Illustration



Summary: EM Algorithm

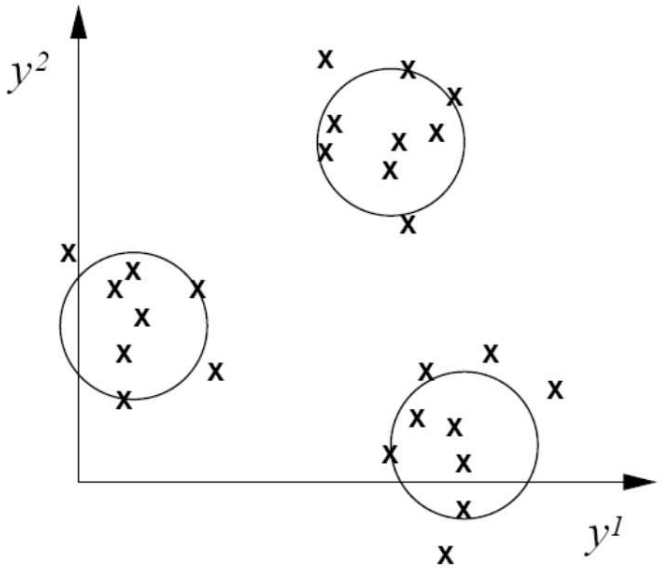
- A way of maximizing likelihood function for latent variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  1. Estimate some “missing” or “unobserved” data from observed data and current parameters.
  2. Using this “complete” data, find the maximum likelihood parameter estimates.
- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  - E-step: $q^{t+1} = \arg\max_{q} F(q, \theta^t)$
  - M-step: $\theta^{t+1} = \arg\max_{\theta} F(q^{t+1}, \theta^t)$
- In the M-step we optimize a lower bound on the likelihood. In the E-step we close the gap, making bound=likelihood.

Hidden Markov Model: dynamic clustering

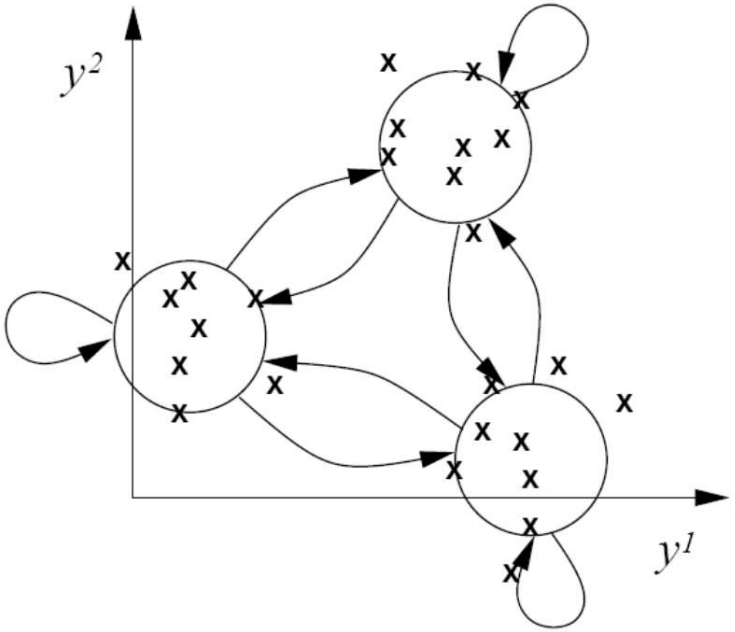


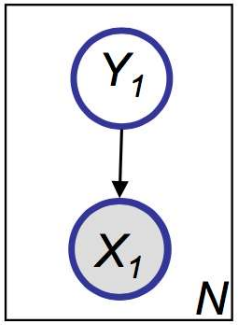
From Static to Dynamic Mixture Models

Static mixture



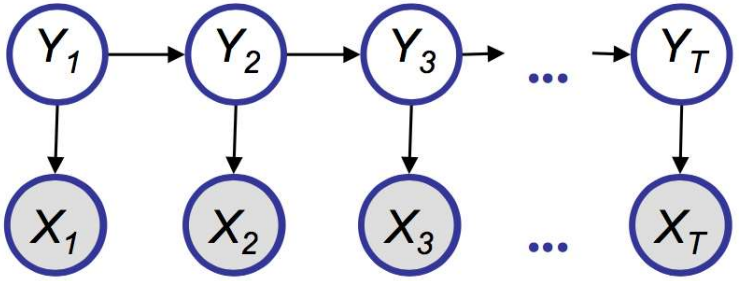
Dynamic mixture





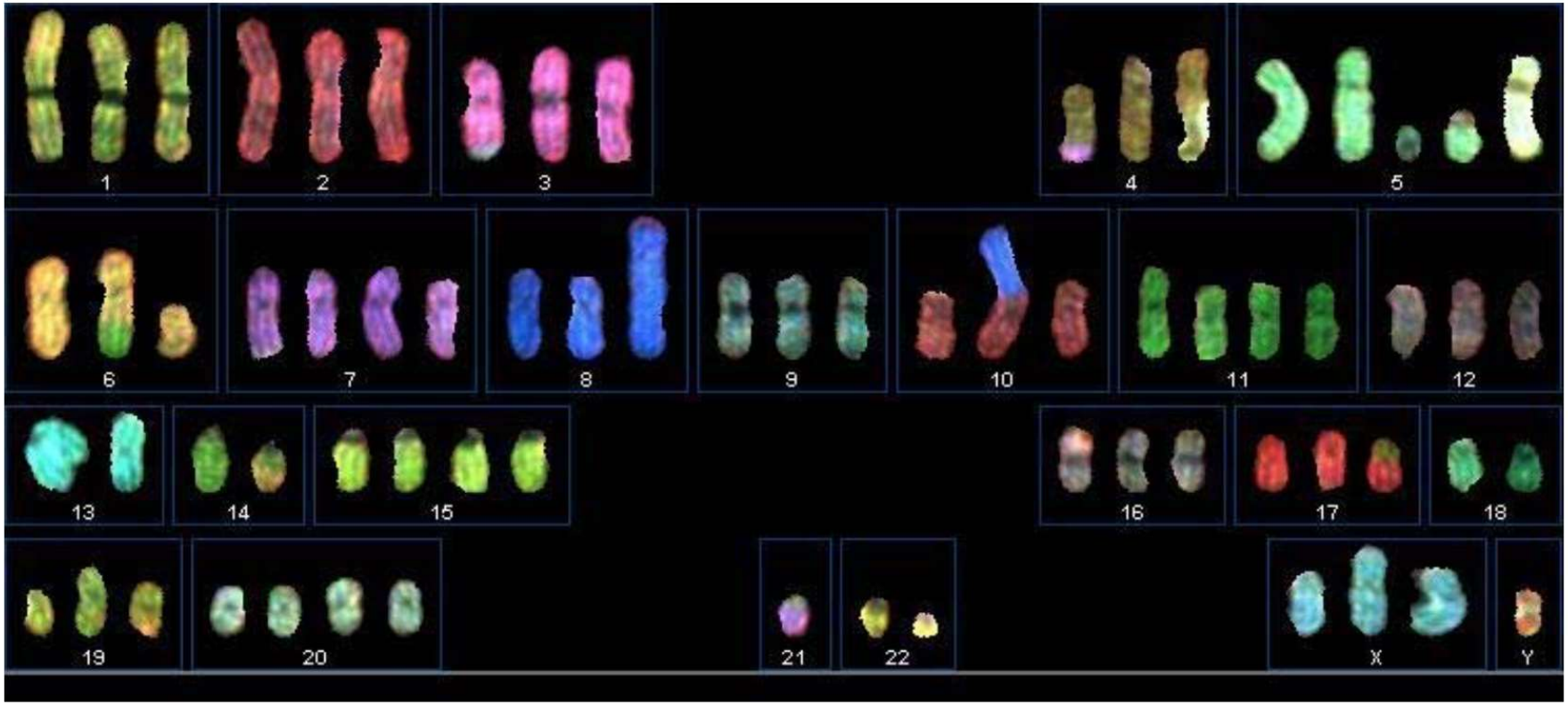
The underlying source:
Speech signal,
dice,

The sequence:
Phonemes,
sequence of rolls,

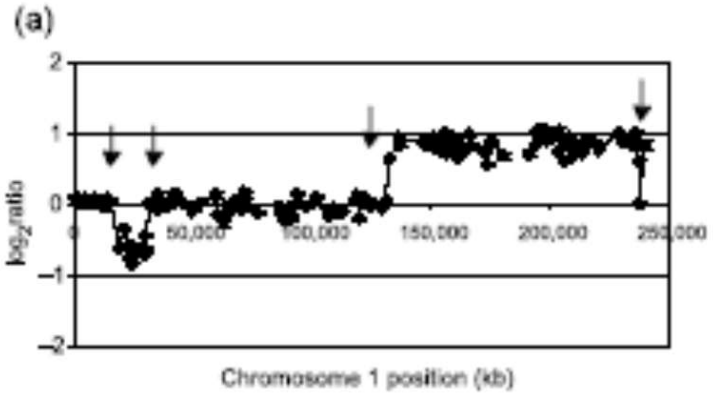


Predicting Tumor Cell States

Chromosomes of tumor cell:

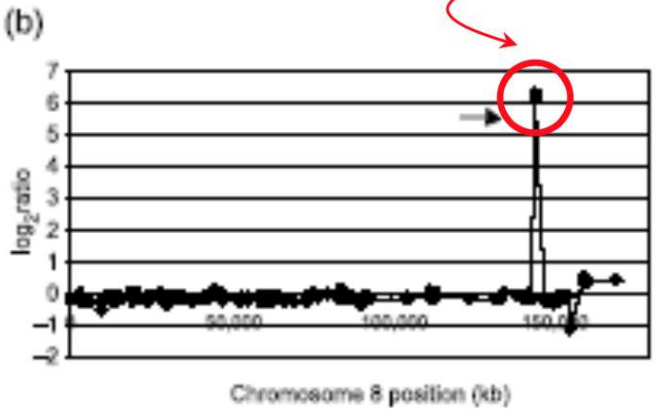


DNA Copy Number Aberration Types In Breast Cancer

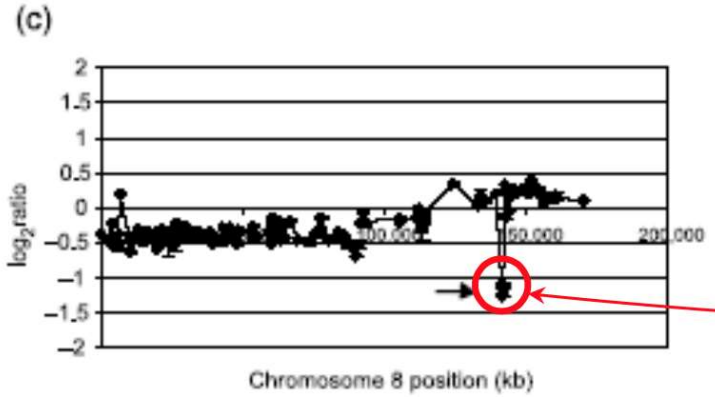


Copy number profile for chromosome 1 from 600 MPE cell line

60-70 fold amplification of CMYC region



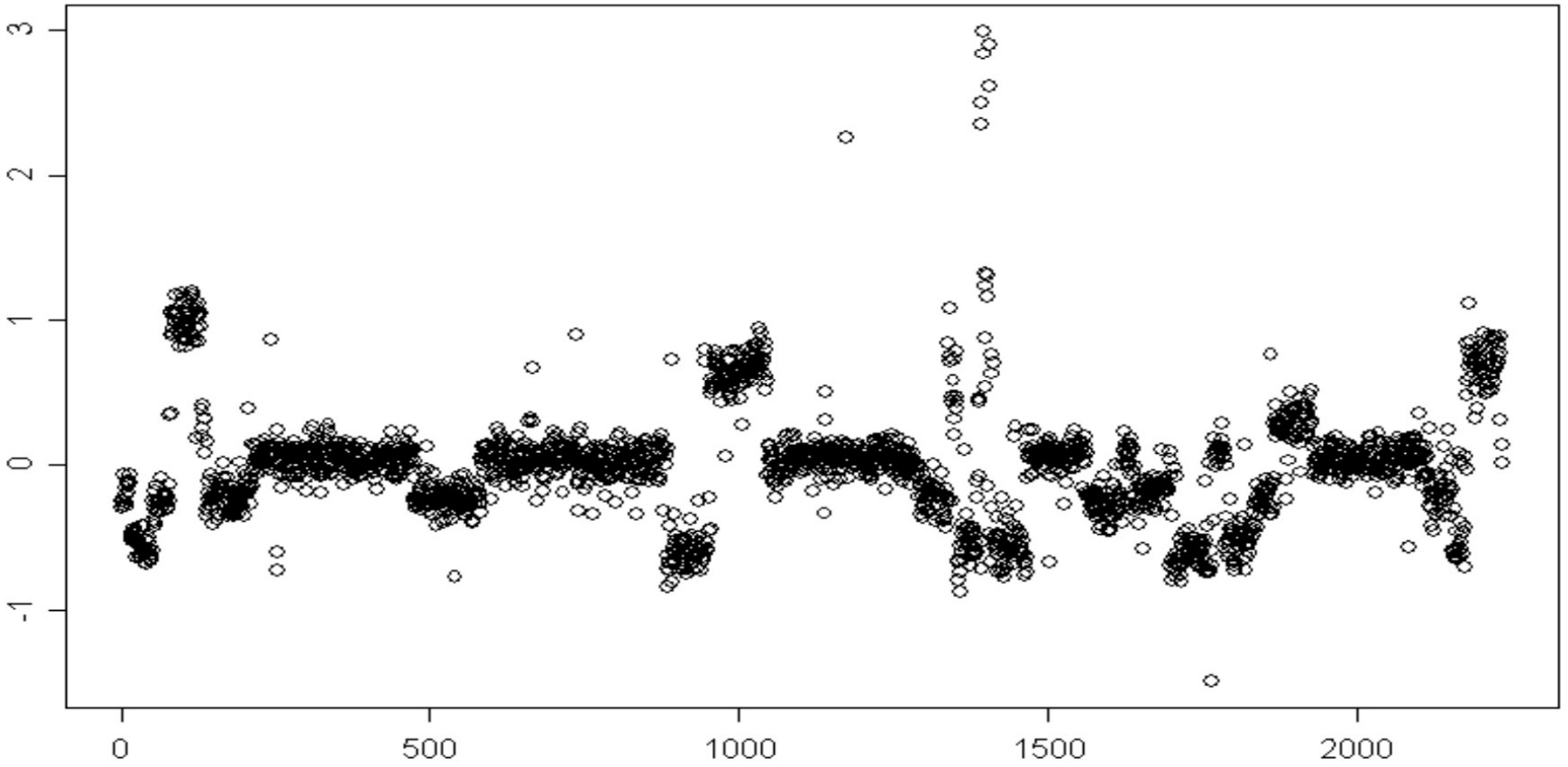
Copy number profile for chromosome 8 from COLO320 cell line



Copy number profile for chromosome 8 in MDA-MB-231 cell line

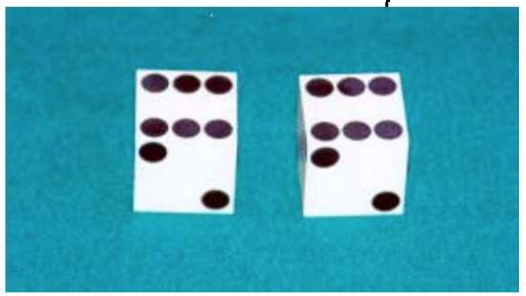
deletion

A Real CGH Run



Suppose you were told about the following story before heading to Vegas…

The Dishonest Casino !!



A casino has two dice:

- **Fair die**
  P(1) = P(2) = P(3) = P(5) = P(6) = 1/6
- **Loaded die**
  P(1) = P(2) = P(3) = P(5) = 1/10
  P(6) = 1/2

Casino player switches back-&-forth between fair and loaded die once every 20 turns

Puzzles Regarding the Dishonest Casino

GIVEN: A sequence of rolls by the casino player

1245526462146146136136661664661636616366163616515615115146123562344

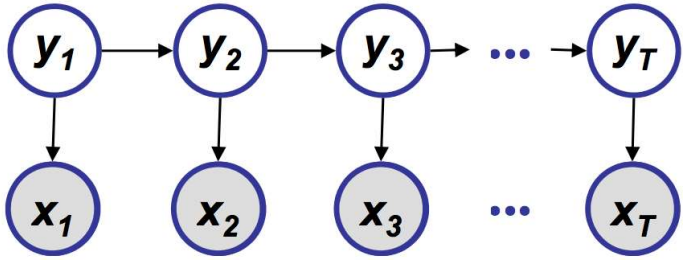
QUESTION

- How likely is this sequence, given our model of how the casino works?
  - This is the EVALUATION problem
- What portion of the sequence was generated with the fair die, and what portion with the loaded die ?
  - This is the DECODING question
- How “loaded” is the loaded die ? How “fair” is the fair die ? How often does the casino player change from fair to loaded, and back ?
  - This is the LEARNING question

Definition (of HMM)

- Observation space
  - Alphabetic set: $\mathbb{C} = \{c_1, c_2, \cdots, c_K\}$
  - Euclidean space: $\mathbb{R}^d$
- Index set of hidden states

$$\mathbb{I} = \{1, 2, \cdots, M\}$$

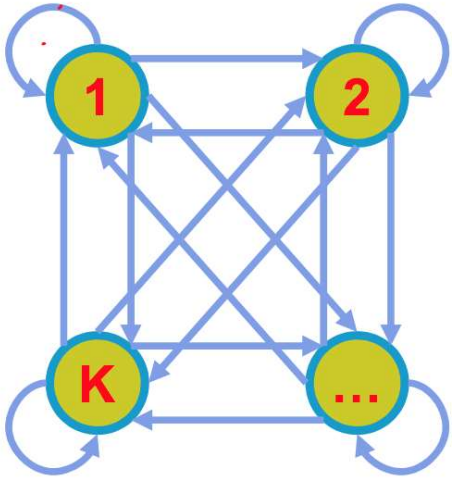


Graphical model

- Transition probabilities between any two states

$$p(y_t^j = 1 \mid y_{t-1}^i = 1) = a_{i,j},$$

or $$p(y_t \mid y_{t-1}^i = 1) \sim \text{Multinomial}(a_{i,1}, a_{i,2}, \ldots, a_{i,M}), \forall i \in \mathbb{I}.$$



State automata

- Start probabilities

$$p(y_1) \sim \text{Multinomial}(\pi_1, \pi_2, \ldots, \pi_M).$$

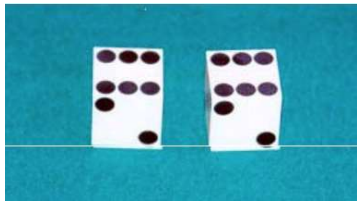
- Emission probabilities associated with each state

$$p(x_t \mid y_t^i = 1) \sim \text{Multinomial}(b_{i,1}, b_{i,2}, \ldots, b_{i,K}), \forall i \in \mathbb{I}.$$

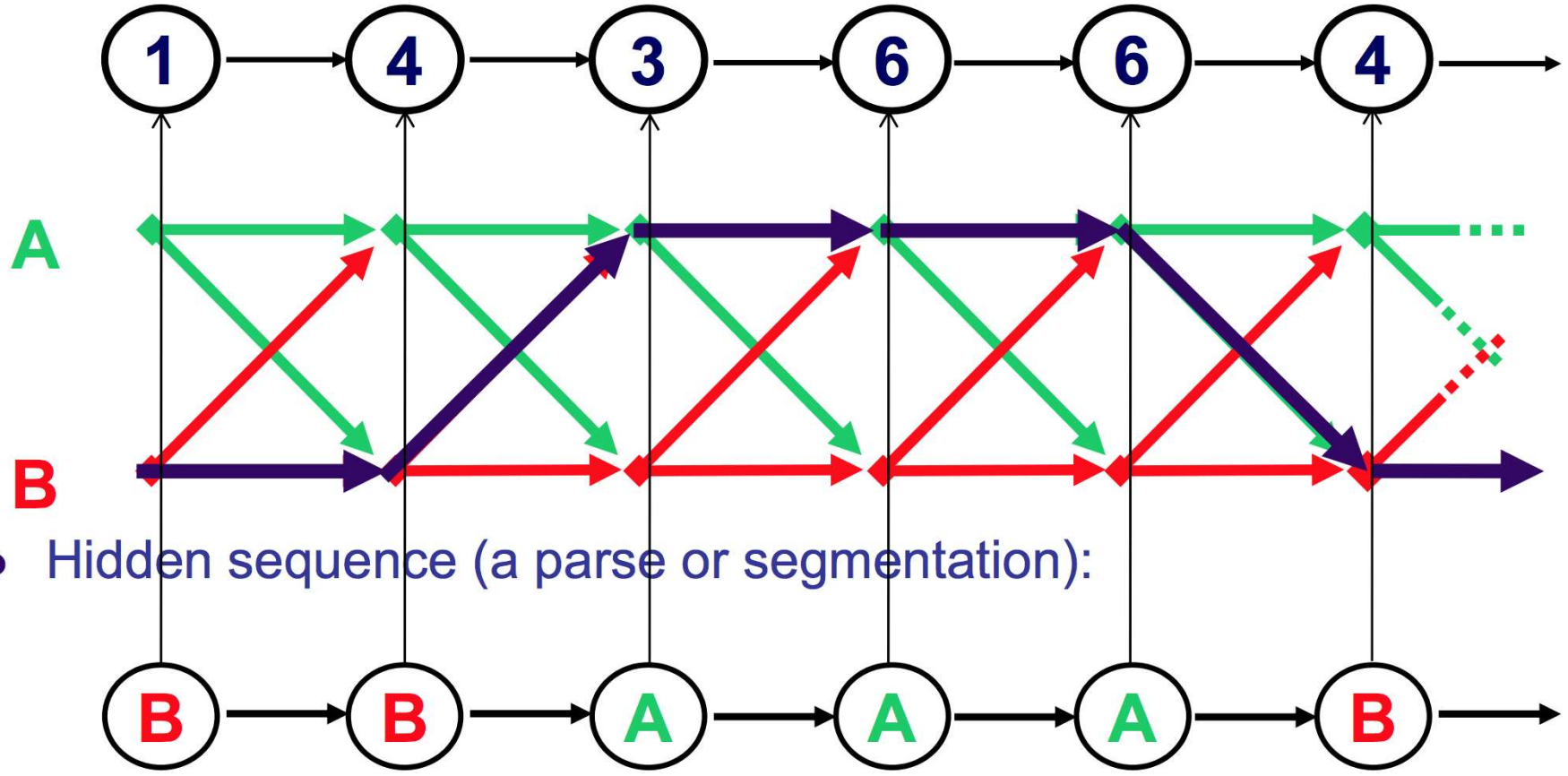
or in general:

$$p(x_t \mid y_t^i = 1) \sim \text{f}(\cdot \mid \theta_i), \forall i \in \mathbb{I}.$$

An HMM is a Stochastic Generative Model

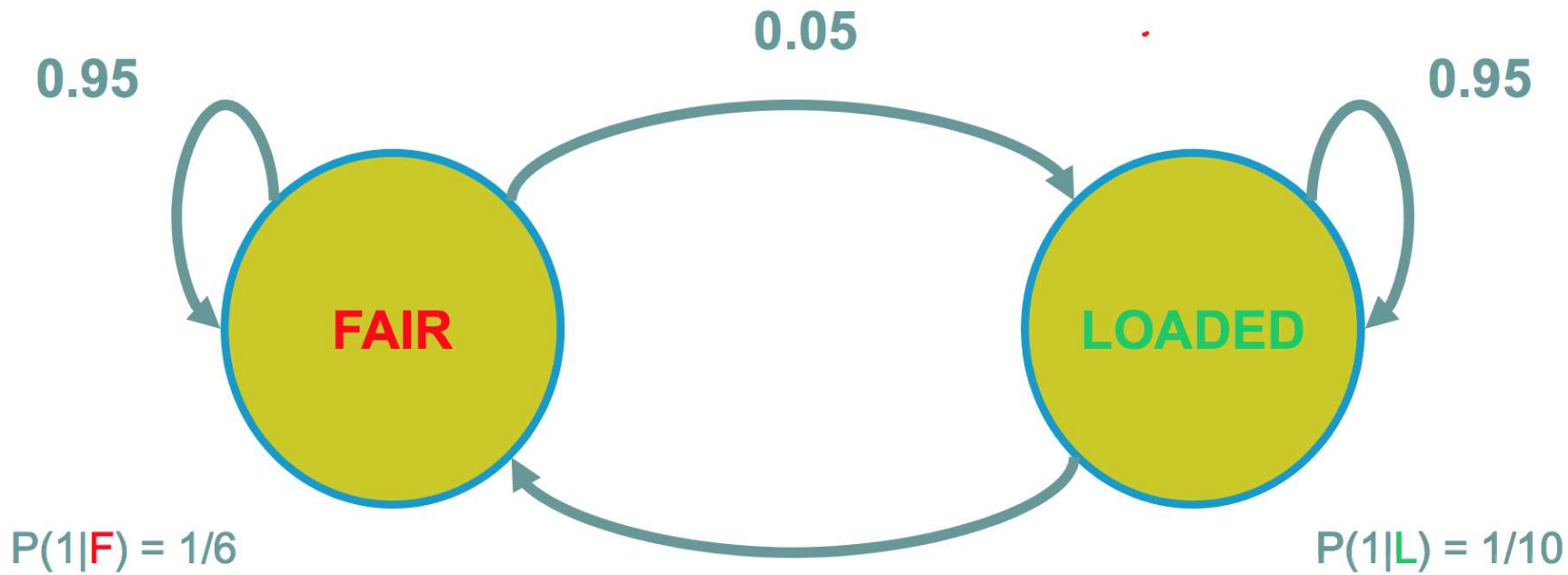


- Observed sequence:



- Hidden sequence (a parse or segmentation):

The Dishonest Casino Model



Three Main Questions on HMMs

1. Evaluation

   GIVEN an HMM M, and a sequence x,
   FIND Prob (x | M)
   ALGO. Forward

2. Decoding

   GIVEN an HMM M, and a sequence x,
   FIND the sequence y of states that maximizes, e.g., P(y|x,M),
   or the most probable subsequence of states
   ALGO. Viterbi, Forward-backward

2. Learning

   GIVEN an HMM M, with unspecified transition/emission probs.,
   and a sequence x,
   FIND parameters $\theta = (\pi_i, a_{ij}, \eta_{ik})$ that maximize $P(x| \theta)$
   ALGO. Baum-Welch (EM)

The Baum Welch Algorithm

- The complete log likelihood

$$\ell_c(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) = \log p(\mathbf{x},\mathbf{y}) = \log \prod_n \left(p(y_{n,1}) \prod_{t=2}^{T} p(y_{n,t} \mid y_{n,t-1}) \prod_{t=1}^{T} p(x_{n,t} \mid x_{n,t}) \right)$$

- The expected complete log likelihood

$$\left\langle \ell_c(\boldsymbol{\theta};\mathbf{x},\mathbf{y}) \right\rangle = \sum_n \left(\left\langle y_{n,1}^i \right\rangle_{p(y_{n,1}|\mathbf{x}_n)} \log \pi_i \right) + \sum_n \sum_{t=2}^{T} \left(\left\langle y_{n,t-1}^i y_{n,t}^j \right\rangle_{p(y_{n,t-1},y_{n,t}|\mathbf{x}_n)} \log a_{i,j} \right) + \sum_n \sum_{t=1}^{T} \left(x_{n,t}^k \left\langle y_{n,t}^i \right\rangle_{p(y_{n,t}|\mathbf{x}_n)} \log b_{i,k} \right)$$

- EM
  - The **E** step

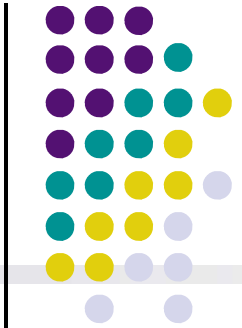
$$\gamma_{n,t}^i = \left\langle y_{n,t}^i \right\rangle = p(y_{n,t}^i = 1 \mid \mathbf{x}_n)$$

$$\xi_{n,t}^{i,j} = \left\langle y_{n,t-1}^i y_{n,t}^j \right\rangle = p(y_{n,t-1}^i = 1, y_{n,t}^j = 1 \mid \mathbf{x}_n)$$

  - The **M** step ("symbolically" identical to MLE)

$$\pi_i^{ML} = \frac{\sum_n \gamma_{n,1}^i}{N} \qquad a_{ij}^{ML} = \frac{\sum_n \sum_{t=2}^{T} \xi_{n,t}^{i,j}}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i} \qquad b_{ik}^{ML} = \frac{\sum_n \sum_{t=1}^{T} \gamma_{n,t}^i x_{n,t}^k}{\sum_n \sum_{t=1}^{T-1} \gamma_{n,t}^i}$$

The Baum-Welch Algorithm — Comments



Time Complexity:

iterations x $O(K^2N)$

- Guaranteed to increase the log likelihood of the model
- Not guaranteed to find globally best parameters
- Converges to local optimum, depending on initial conditions
- Too many parameters / too large model: Over-fitting

Summary: the HMM Algorithms

Questions:

- **Evaluation**: What is the probability of the observed sequence ? Forward
- **Decoding**: What is the probability that the state of the 3rd roll is loaded, given the observed sequence ? Forward-Backward
- **Decoding**: What is the most likely die sequence ? Viterbi
- **Learning**: Under what parameterization are the observed sequences most probable ? Baum-Welch (EM)