Clustering
K-means

Machine Learning – CSE546
Sham Kakade
University of Washington

November 15, 2016

Announcements:

- Project Milestones due date passed.
- HW3 due on Monday
  - It'll be collaborative
- HW2 grades posted today
  - Out of 82 points

- Today:
  - Review: PCA
  - Start: unsupervised learning

Dimensionality Reduction PCA

Machine Learning – CSE4546
Sham Kakade
University of Washington

November 8, 2016

©Sham Kakade 2016

Linear projections, a review

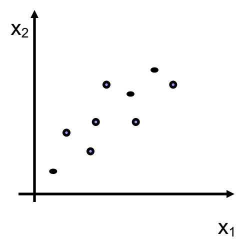
- Project a point into a (lower dimensional) space:
  - **point**: $\mathbf{x} = (x_1,\ldots,x_d)$
  - **select a basis** – set of basis vectors – $(\mathbf{u}_1,\ldots,\mathbf{u}_k)$
    - we consider orthonormal basis:
      - $\mathbf{u}_i \bullet \mathbf{u}_i = 1$, and $\mathbf{u}_i \bullet \mathbf{u}_j = 0$ for $i \neq j$
  - **select a center** – $\bar{\mathbf{x}}$, defines offset of space
  - **best coordinates** in lower dimensional space defined by dot-products: $(z_1,\ldots,z_k)$, $z_i = (\mathbf{x}-\bar{\mathbf{x}}) \bullet \mathbf{u}_i$

©Sham Kakade 2016

PCA finds projection that minimizes reconstruction error

- Given N data points: $\mathbf{x}^i = (x_1^i, \ldots, x_d^i)$, i=1…N
- Will represent each point as a projection:
  - $\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$ where: $\bar{\mathbf{x}} = \frac{1}{\text{N}} \sum_{i=1}^{\text{N}} \mathbf{x}^i$ and $z_j^i = (\mathbf{x}^i - \bar{\mathbf{x}}) \cdot \mathbf{u}_j$
- PCA:
  - Given k<<d, find ($\mathbf{u}_1, \ldots, \mathbf{u}_k$) minimizing reconstruction error:

$$error_k = \sum_{i=1}^{\text{N}} (\mathbf{x}^i - \hat{\mathbf{x}}^i)^2$$



Understanding the reconstruction error

$$\hat{\mathbf{x}}^i = \bar{\mathbf{x}} + \sum_{j=1}^{k} z_j^i \mathbf{u}_j$$

$$z_j^i = (\mathbf{x}^i - \bar{\mathbf{x}}) \cdot \mathbf{u}_j$$

- Given k<<d, find ($\mathbf{u}_1, \ldots, \mathbf{u}_k$) minimizing reconstruction error:

$$error_k = \sum_{i=1}^{\text{N}} (\mathbf{x}^i - \hat{\mathbf{x}}^i)^2$$

- Note that $\mathbf{x}^i$ can be represented exactly by d-dimensional projection:

$$\mathbf{x}^i = \bar{\mathbf{x}} + \sum_{j=1}^{\text{d}} z_j^i \mathbf{u}_j$$

- Rewriting error:

Reconstruction error and covariance matrix

$$error_k = \sum_{i=1}^{N} \sum_{j=k+1}^{d} [\mathbf{u}_j \cdot (\mathbf{x}^i - \bar{\mathbf{x}})]^2$$

$$\Sigma = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}^i - \bar{\mathbf{x}})(\mathbf{x}^i - \bar{\mathbf{x}})^T$$

Minimizing reconstruction error and eigen vectors

- Minimizing reconstruction error equivalent to picking orthonormal basis ($\mathbf{u}_1,\ldots,\mathbf{u}_d$) minimizing:

$$error_k = N \sum_{j=k+1}^{d} \mathbf{u}_j^T \Sigma \mathbf{u}_j$$

- Eigen vector definition:

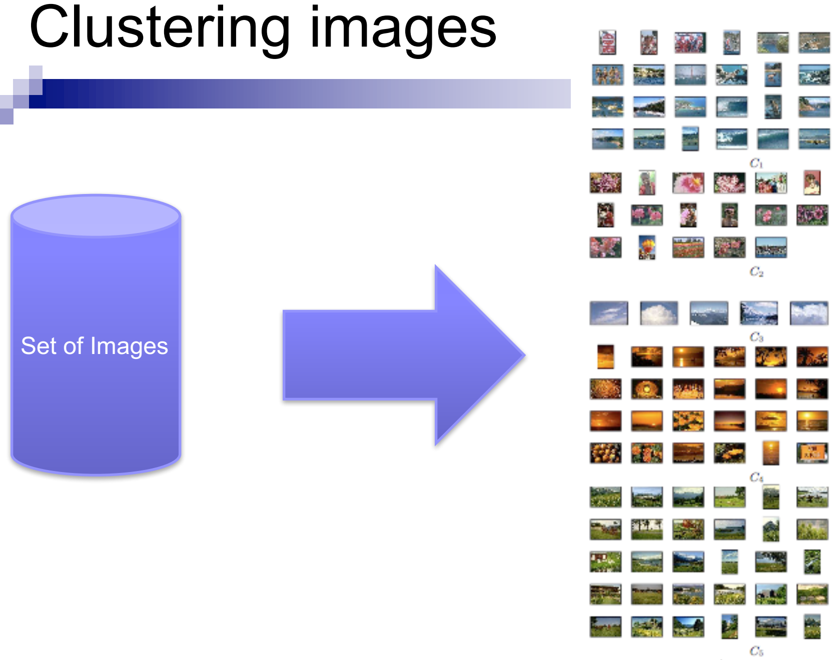
- Solution: use the eigenvectors from the SVD

Clustering
K-means

Machine Learning – CSE546
Sham Kakade
University of Washington

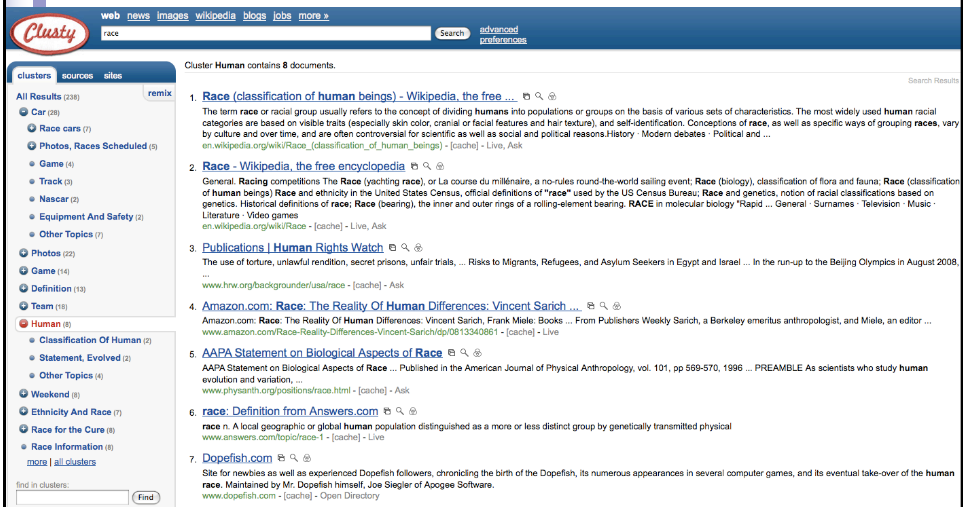
November 15, 2016

Clustering images

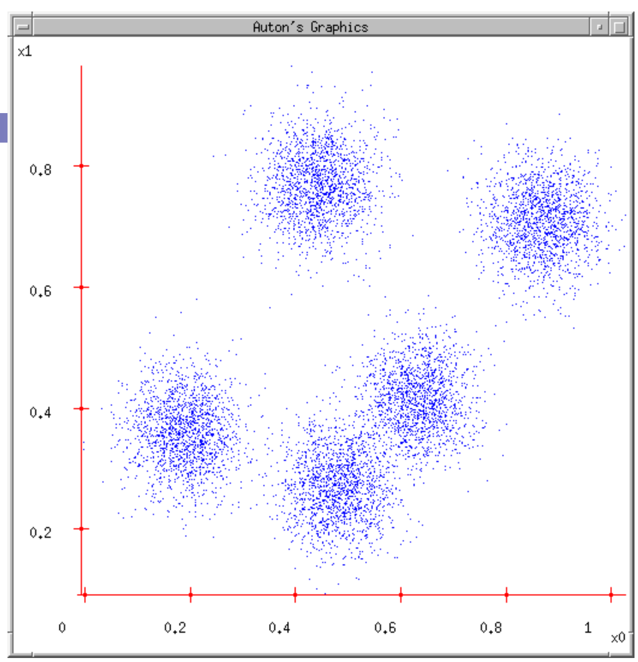


[Goldberger et al.]

Clustering web search results

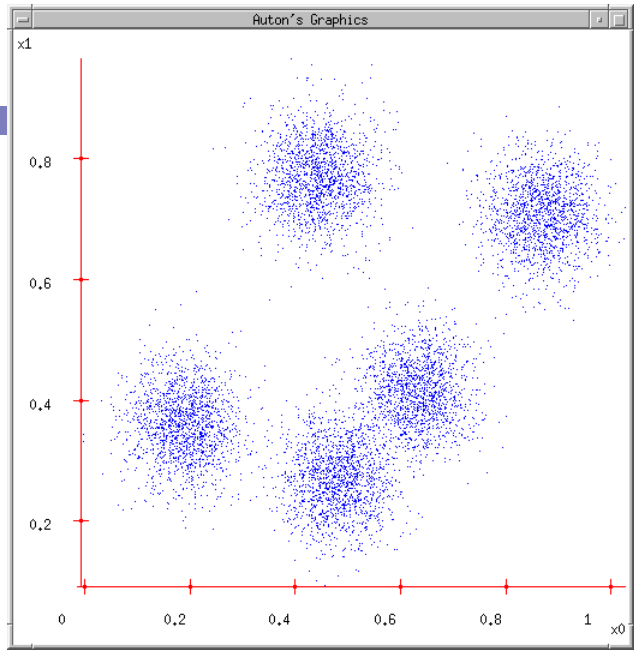


Some Data



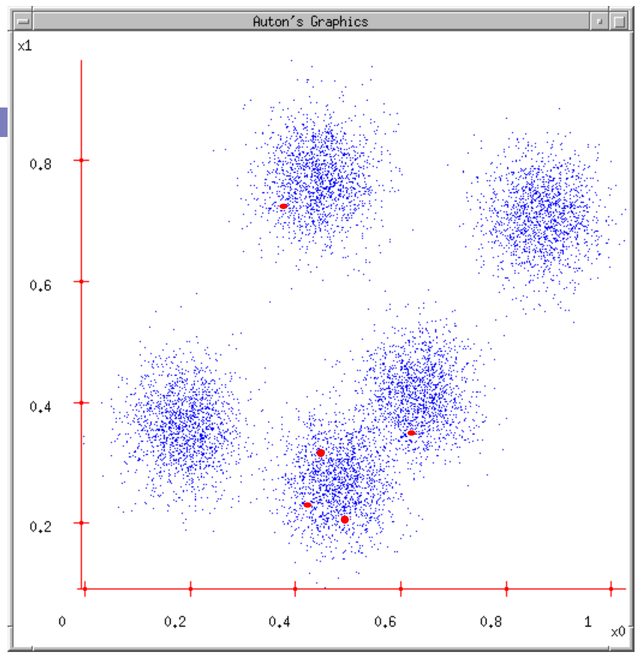
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



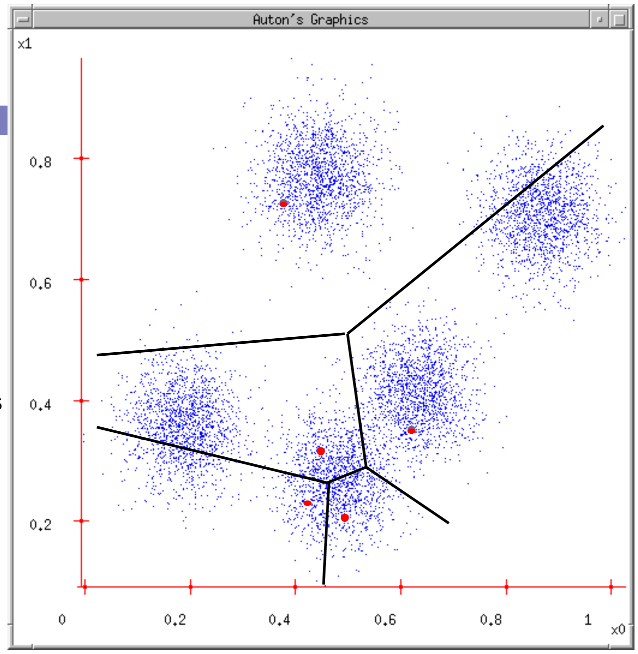
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



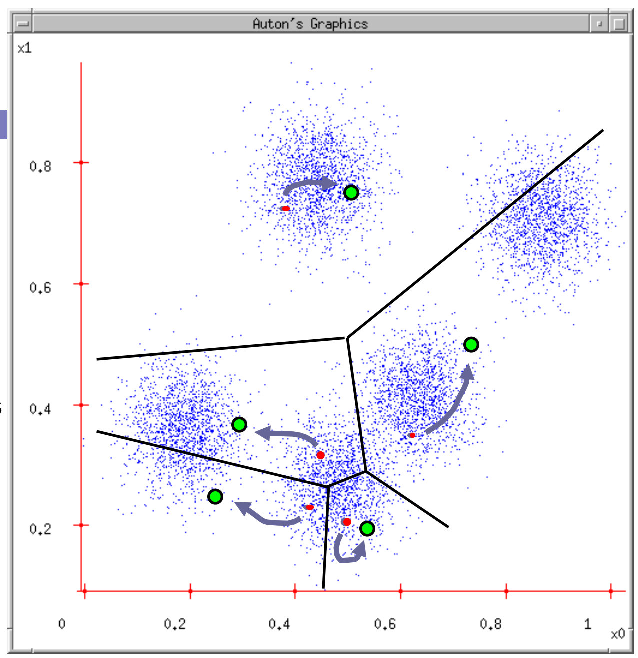
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



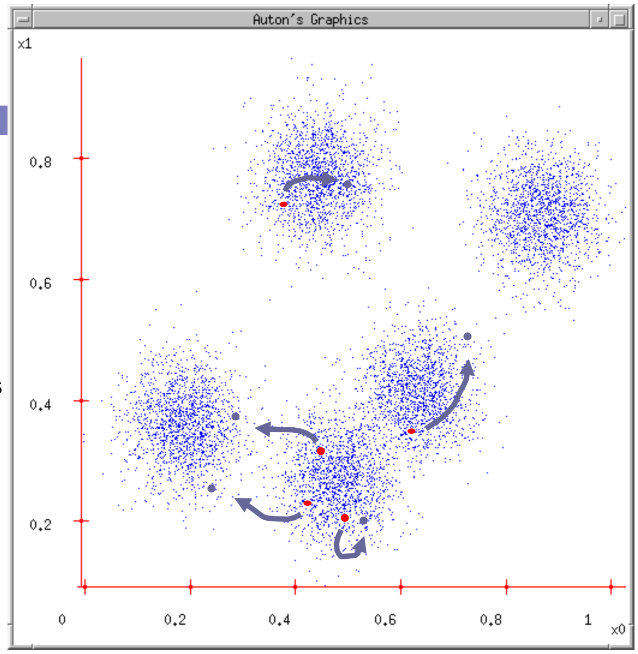
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



K-means

- Randomly initialize *k* centers
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$
- **Classify**: Assign each point $j \in \{1, \ldots N\}$ to nearest center:
  - $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$
- **Recenter**: μ_i becomes centroid of its point:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2$
  - Equivalent to $\mu_i \leftarrow$ average of its points!

What is K-means optimizing?

- Potential function F(μ,C) of centers μ and point allocations C:
  - $F(\mu, C) = \sum_{j=1}^{N} ||\mu_{C(j)} - x_j||^2$
- Optimal K-means:
  - $\min_\mu \min_C F(\mu,C)$

Does K-means converge??? Part 1

- Optimize potential function:

$$\min_\mu \min_C F(\mu, C) = \min_\mu \min_C \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix μ, optimize C

Does K-means converge??? Part 2

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix C, optimize μ

Coordinate descent algorithms

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

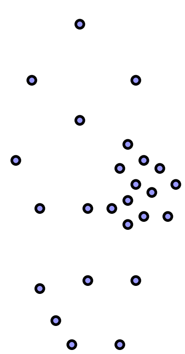
- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good??) local optimum
    - (For LASSO it converged to the global optimum, because of convexity)
  - Some theory of quality of local opt…
- K-means is a coordinate descent algorithm!

Mixtures of Gaussians

Machine Learning – CSE546
Sham Kakade
University of Washington

November 8, 2016

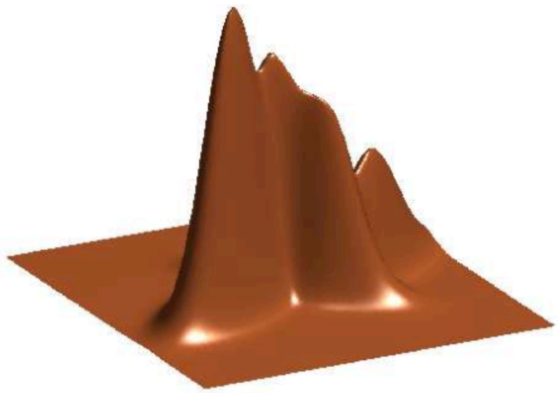
(One) bad case for k-means



- Clusters may overlap
- Some clusters may be "wider" than others

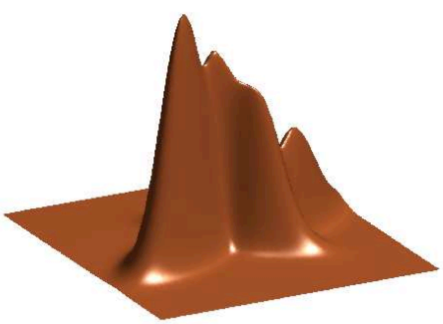
Density Estimation

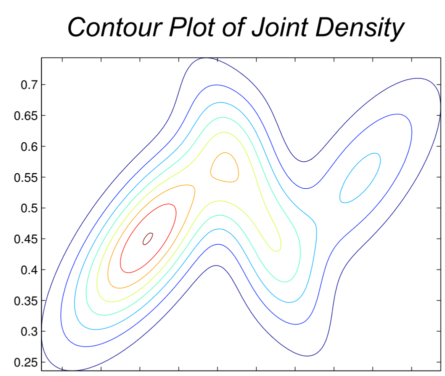
- Estimate a density based on $x^1, \ldots, x^N$



©2016 Sham Kakade ©Sham Kakade 2016 25

Density Estimation



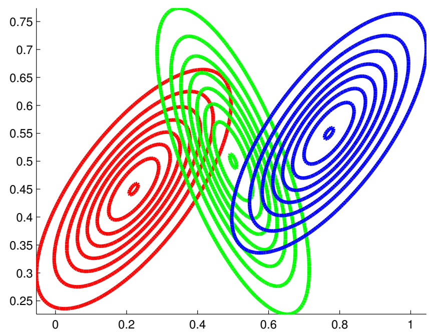


©2016 Sham Kakade ©Sham Kakade 2016 26

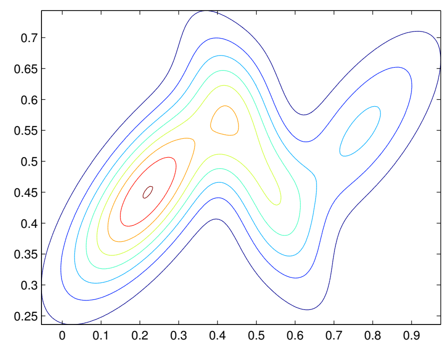
Density as Mixture of Gaussians

- Approximate density with a mixture of Gaussians

Mixture of 3 Gaussians



Contour Plot of Joint Density



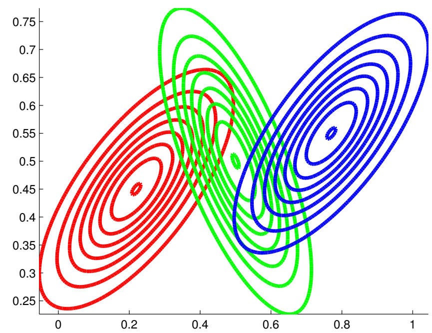
Gaussians in *d* Dimensions

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

Density as Mixture of Gaussians

- Approximate density with a mixture of Gaussians

Mixture of 3 Gaussians

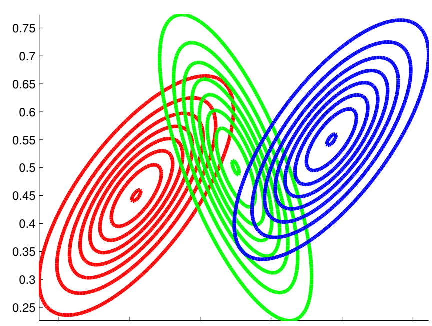


$$p(x^i|\pi, \mu, \Sigma) =$$

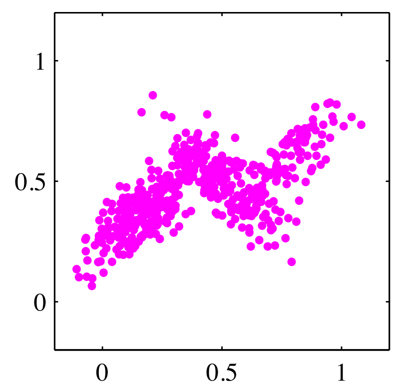
Density as Mixture of Gaussians

- Approximate with density with a mixture of Gaussians

Mixture of 3 Gaussians



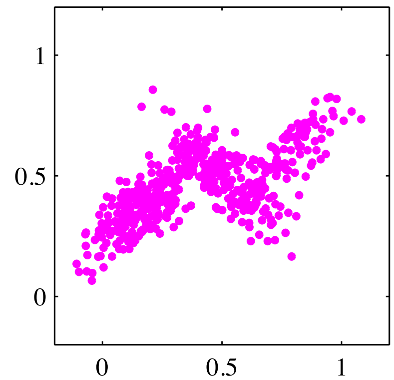
Our actual observations



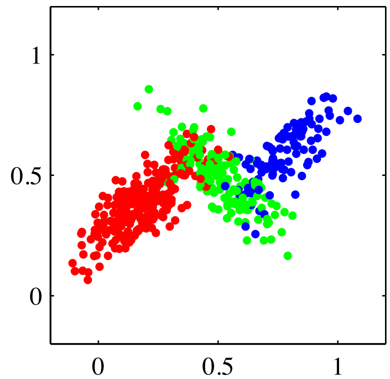
Clustering our Observations

- Imagine we have an assignment of each x^i to a Gaussian

Our actual observations

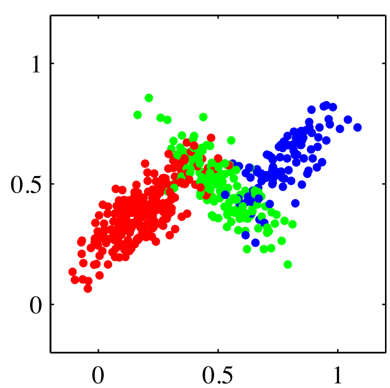


Complete data labeled by true cluster assignments



Clustering our Observations

- Imagine we have an assignment of each x^i to a Gaussian



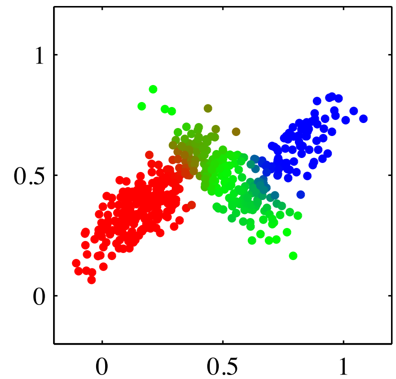
Complete data labeled by true cluster assignments

- Introduce latent cluster indicator variable z^i
- Then we have

$$p(x^i|z^i, \pi, \mu, \Sigma) =$$

Clustering our Observations

- We must infer the cluster assignments from the observations

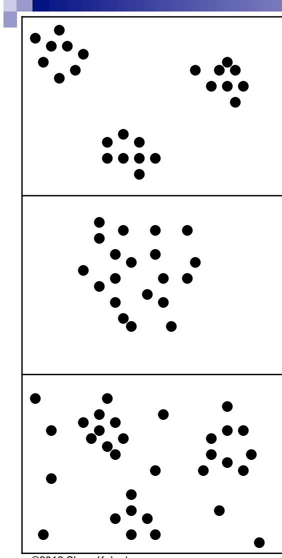


Soft assignments to clusters

- Posterior probabilities of assignments to each cluster *given* model parameters:

$$r_{ik} = p(z^i = k | x^i, \pi, \mu, \Sigma) =$$

Unsupervised Learning: not as hard as it looks



Sometimes easy

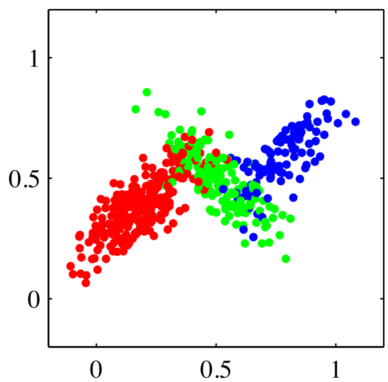
Sometimes impossible

and sometimes in between

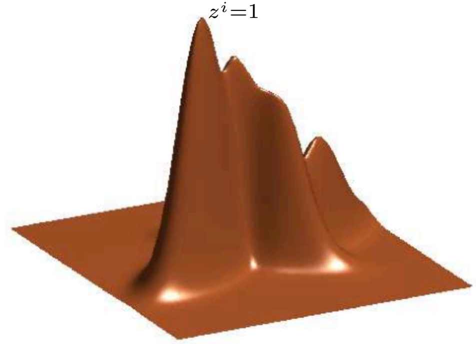
Summary of GMM Concept

- Estimate a density based on $x^1,\ldots,x^N$

$$p(x^i|\pi,\mu,\Sigma) = \sum_{z^i=1}^{K} \pi_{z^i}\mathcal{N}(x^i|\mu_{z^i},\Sigma_{z^i})$$



Complete data labeled by true cluster assignments



Surface Plot of Joint Density, Marginalizing Cluster Assignments

Summary of GMM Components

- Observations $x^i \in \mathbb{R}^d, \quad i = 1,2,\ldots,N$
- Hidden cluster labels $z_i \in \{1,2,\ldots,K\}, \quad i = 1,2,\ldots,N$
- Hidden mixture means $\mu_k \in \mathbb{R}^d, \quad k = 1,2,\ldots,K$
- Hidden mixture covariances $\Sigma_k \in \mathbb{R}^{d\times d}, \quad k = 1,2,\ldots,K$
- Hidden mixture probabilities $\pi_k, \quad \sum_{k=1}^{K} \pi_k = 1$

Gaussian mixture marginal and conditional likelihood :

$$p(x^i|\pi,\mu,\Sigma) = \sum_{z^i=1}^{K} \pi_{z^i}\; p(x^i|z^i,\mu,\Sigma)$$

$$p(x^i|z^i,\mu,\Sigma) = \mathcal{N}(x^i|\mu_{z^i},\Sigma_{z^i})$$

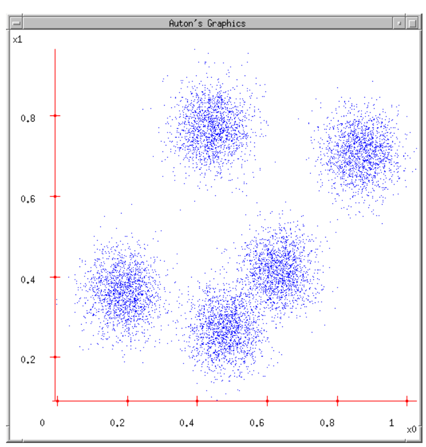
Expectation Maximization

Machine Learning – CSE546
Sham Kakade
University of Washington

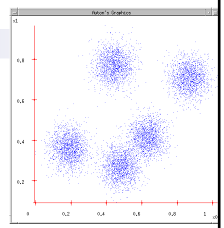
November 8, 2016

Next… back to Density Estimation

What if we want to do density estimation with multimodal or clumpy data?



But we don't see class labels!!!



- MLE:
  - argmax $\prod_i P(z^i, x^i)$
- But we don't know z^i
- Maximize marginal likelihood:
  - argmax $\prod_i P(x^i)$ = argmax $\prod_i \sum_{k=1}^{K} P(z^i=k, x^i)$

Special case: spherical Gaussians and hard assignments

$$P(z^i = k, \mathbf{x}^i) = \frac{1}{(2\pi)^{m/2} \| \Sigma_k \|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}^i - \mu_k\right)^T \Sigma_k^{-1} \left(\mathbf{x}^i - \mu_k\right)\right] P(z^i = k)$$

- If P(X|z=k) is spherical, with same σ for all classes:

$$P(\mathbf{x}^i \mid z^i = k) \propto \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}^i - \mu_k\right\|^2\right]$$

- If each x^i belongs to one class C(i) (hard assignment), marginal likelihood:

$$\prod_{i=1}^{N} \sum_{k=1}^{K} P(\mathbf{x}^i, z^i = k) \propto \prod_{i=1}^{N} \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}^i - \mu_{C(i)}\right\|^2\right]$$

- Same as K-means!!!

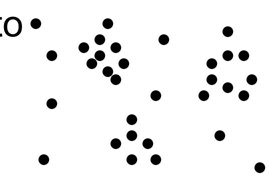
Supervised Learning of Mixtures of Gaussians

- Mixtures of Gaussians:
  - Prior class probabilities: P(z=k)
  - Likelihood function per class: P(**x**|z=k)

- Suppose, for each data point, we know location **x** and class z
  - Learning is easy… ☺
  - For prior P(z)
  - For likelihood function:

EM: "Reducing" Unsupervised Learning to Supervised Learning

- If we knew assignment of points to classes ➔ Supervised Learning!



- Expectation-Maximization (EM)
  - Guess assignment of points to classes
    - In standard ("soft") EM: each point associated with prob. of being in each class
  - Recompute model parameters
  - Iterate

Form of Likelihood

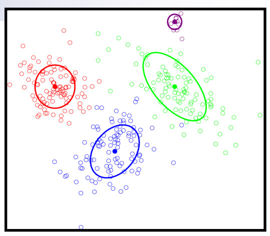
- Conditioned on class of point $\mathbf{x}^i$...

$$p(x^i \mid z^i, \mu, \Sigma) =$$

- Marginalizing class assignment:

$$p(x^i \mid \pi, \mu, \Sigma) =$$

Gaussian Mixture Model

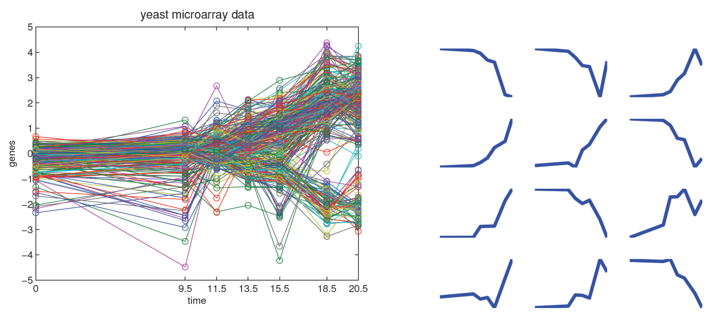


- Most commonly used mixture model
- Observations:
- Parameters:
- Likelihood:
- Ex. z^i = country of origin, x^i = height of i^{th} person
  - k^{th} mixture component = distribution of heights in country k

Example

(Taken from Kevin Murphy's ML textbook)

- Data: gene expression levels
- Goal: cluster genes with similar expression trajectories



Mixture models are useful for…

- Density estimation
  - Allows for multimodal density
- Clustering
  - Want membership information for each observation
    - e.g., topic of current document
  - Soft clustering:

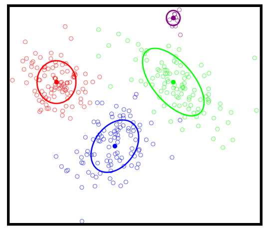
$$p(z^i = k \mid x^i, \theta) =$$

  - Hard clustering:

$$z^{i*} = \arg\max_k p(z^i = k \mid x^i, \theta) =$$

Issues

- Label switching
  - Color = label does not matter
  - Can switch labels and likelihood is unchanged



- Log likelihood is not convex in the parameters
  - Problem is simpler for “complete data likelihood”

ML Estimate of Mixture Model Params

- Log likelihood

$$L_x(\theta) \triangleq \log p(\{x^i\} \mid \theta) = \sum_i \log \sum_{z^i} p(x^i, z^i \mid \theta)$$

- Want ML estimate

$$\hat{\theta}^{ML} =$$

- Neither convex nor concave and local optima

If "complete" data were observed…

- Assume class labels z^i were observed in addition to x^i

$$L_{x,z}(\theta) = \sum_i \log p(x^i, z^i \mid \theta)$$

- Compute ML estimates
  - Separates over clusters *k*!
- Example: mixture of Gaussians (MoG) $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Iterative Algorithm

- Motivates a coordinate ascent-like algorithm:
  1. Infer missing values z^i given estimate of parameters $\hat{\theta}$
  2. Optimize parameters to produce new $\hat{\theta}$ given "filled in" data z^i
  3. Repeat
- Example: MoG (derivation soon… + HW)
  1. Infer "responsibilities"

$$r_{ik} = p(z^i = k \mid x^i, \hat{\theta}^{(t-1)}) =$$

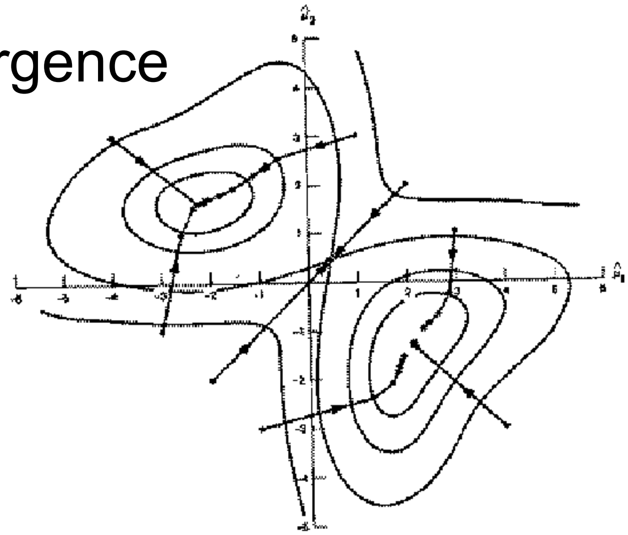
  2. Optimize parameters

$$\max \text{ w.r.t. } \pi_k :$$

$$\max \text{ w.r.t. } \mu_k, \Sigma_k :$$

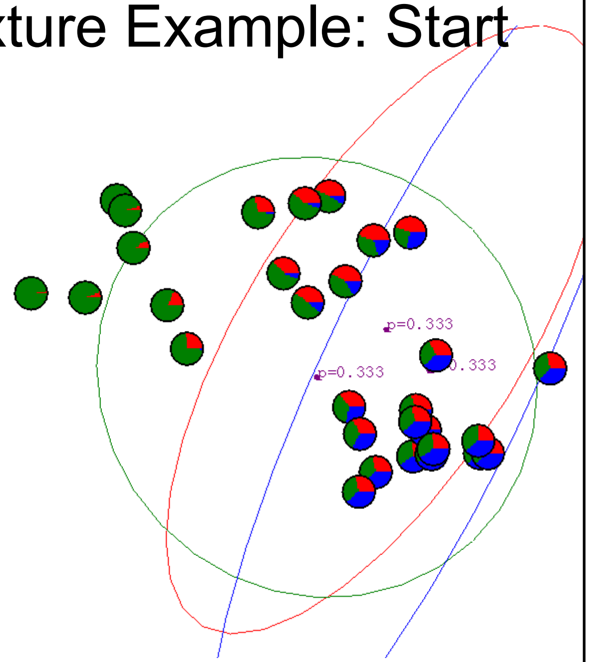
E.M. Convergence

- EM is coordinate ascent on an interesting potential function
- Coord. ascent for bounded pot. func. ➔ convergence to a local optimum guaranteed

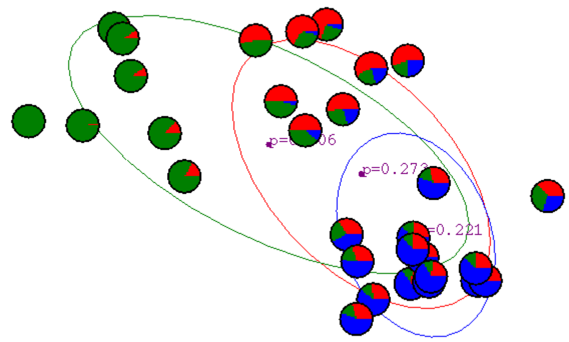


- This algorithm is REALLY USED. And in high dimensional state spaces, too. E.G. Vector Quantization for Speech Data

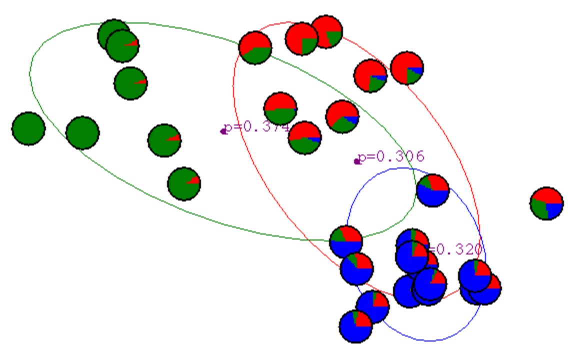
Gaussian Mixture Example: Start



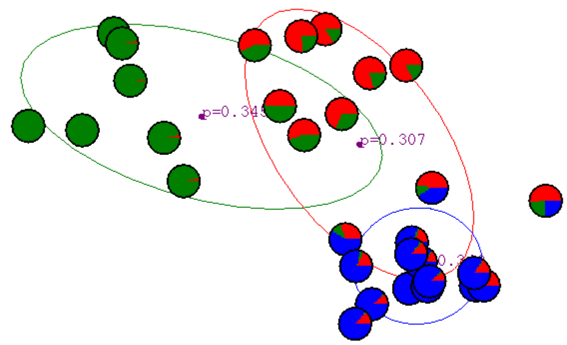
After first iteration



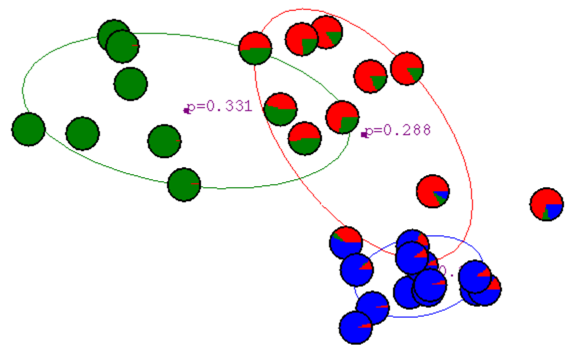
After 2nd iteration



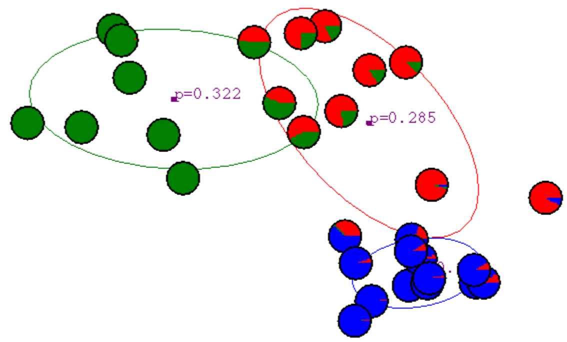
After 3rd iteration



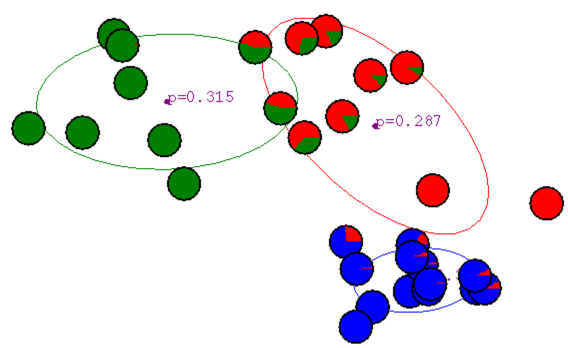
After 4th iteration



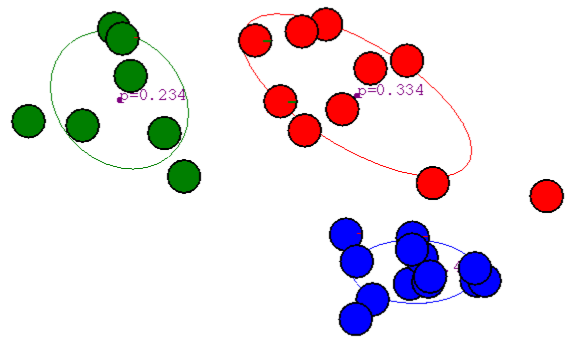
After 5th iteration



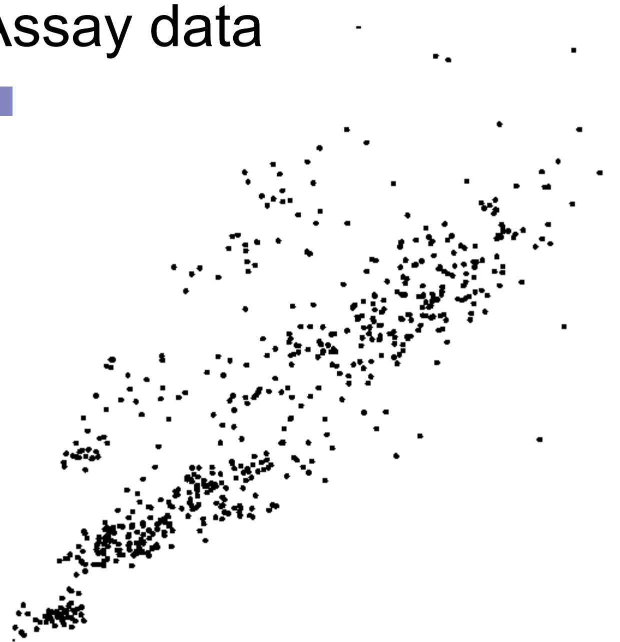
After 6th iteration



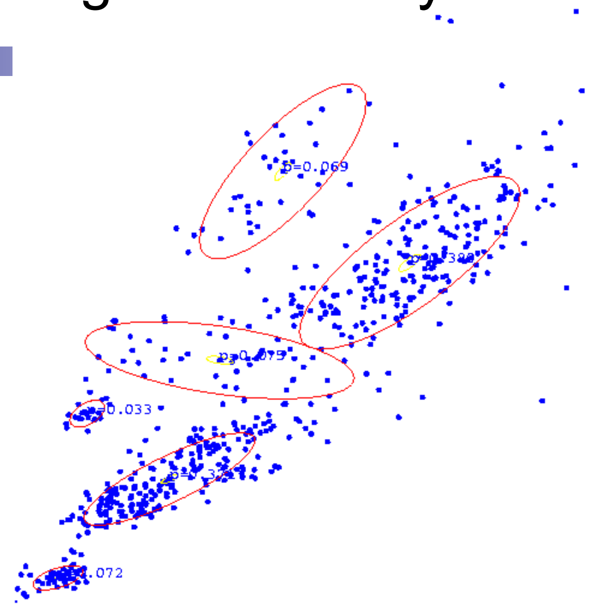
After 20th iteration



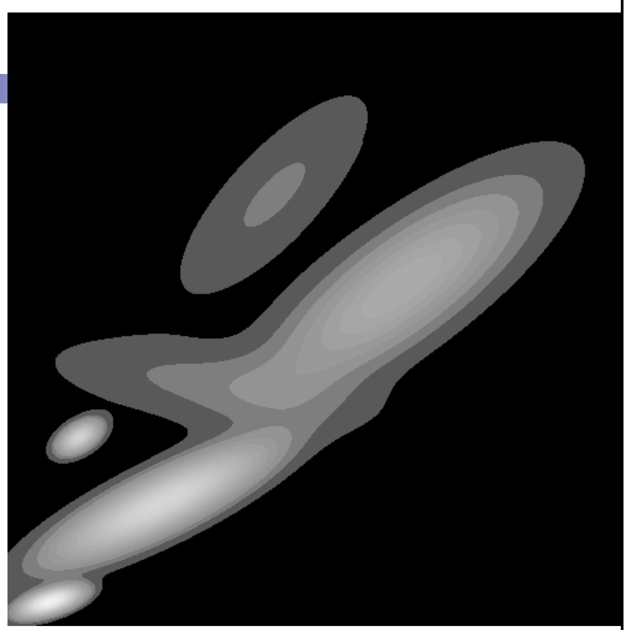
Some Bio Assay data



GMM clustering of the assay data



Resulting Density Estimator



Expectation Maximization (EM) – Setup

- More broadly applicable than just to mixture models considered so far
- Model: x observable – *"incomplete" data*
  y not (fully) observable – *"complete" data*
  θ parameters
- Interested in maximizing (wrt θ):

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

- Special case:

$$x = g(y)$$

Expectation Maximization (EM) – Derivation

- Step 1
  - Rewrite desired likelihood in terms of complete data terms

$$p(y \mid \theta) = p(y \mid x, \theta) p(x \mid \theta)$$

- Step 2
  - Assume estimate of parameters $\hat{\theta}$
  - Take expectation with respect to $p(y \mid x, \hat{\theta})$

Expectation Maximization (EM) – Derivation

- Step 3
  - Consider log likelihood of data at any θ relative to log likelihood at $\hat{\theta}$

  $L_x(\theta) - L_x(\hat{\theta})$

- **Aside: Gibbs Inequality** $E_p[\log p(x)] \geq E_p[\log q(x)]$

  Proof:

Motivates EM Algorithm

- Initial guess:
- Estimate at iteration *t*:
- **E-Step**

  Compute

- **M-Step**

  Compute

Expectation Maximization (EM) – Derivation

$$L_x(\theta) - L_x(\hat{\theta}) = [U(\theta,\hat{\theta}) - U(\hat{\theta},\hat{\theta})] - [V(\theta,\hat{\theta}) - V(\hat{\theta},\hat{\theta})]$$

- Step 4
  - Determine conditions under which log likelihood at θ exceeds that at $\hat{\theta}$

  Using Gibbs inequality:

  If

  Then

  $$L_x(\theta) \geq L_x(\hat{\theta})$$

Example – Mixture Models

- **E-Step** Compute $U(\theta,\hat{\theta}^{(t)}) = E[\log p(y \mid \theta) \mid x, \hat{\theta}^{(t)}]$
- **M-Step** Compute $\hat{\theta}^{(t+1)} = \arg\max_\theta U(\theta, \hat{\theta}^{(t)})$

- Consider $y^i = \{z^i, x^i\}$ i.i.d.

$$p(x^i, z^i \mid \theta) = \pi_{z^i} p(x^i \mid \phi_{z^i}) =$$

$$E_{q_t}[\log p(y \mid \theta)] = \sum_i E_{q_t}[\log p(x^i, z^i \mid \theta)] =$$

Coordinate Ascent Behavior

- Bound log likelihood:

$$L_x(\theta) = U(\theta, \hat{\theta}^{(t)}) + V(\theta, \hat{\theta}^{(t)})$$
$$\geq$$
$$L_x(\hat{\theta}^{(t)}) = U(\hat{\theta}^{(t)}, \hat{\theta}^{(t)}) + V(\hat{\theta}^{(t)}, \hat{\theta}^{(t)})$$

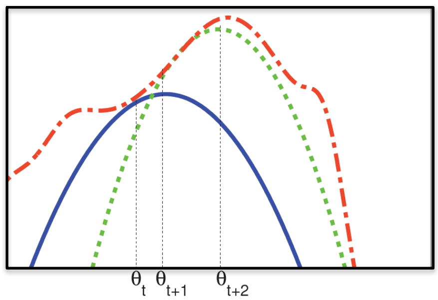


Figure from KM textbook

Comments on EM

- Since Gibbs inequality is satisfied with equality only if *p=q*, any step that changes θ should strictly **increase likelihood**
- In practice, can replace the **M-Step** with increasing *U* instead of maximizing it (**Generalized EM**)
- Under certain conditions (e.g., in exponential family), can show that EM **converges to a stationary point** of $L_x(\theta)$
- Often there is a **natural choice for *y*** … has physical meaning
- If you want to choose any *y*, not necessarily *x=g(y)*, replace $p(y \mid \theta)$ in *U* with $p(y, x \mid \theta)$

Initialization

- In mixture model case where $y^i = \{z^i, x^i\}$ there are many ways to initialize the EM algorithm
- Examples:
  - Choose K observations at random to define each cluster. Assign other observations to the nearest "centriod" to form initial parameter estimates
  - Pick the centers sequentially to provide good coverage of data
  - Grow mixture model by splitting (and sometimes removing) clusters until K clusters are formed
- Can be quite important to convergence rates in practice

What you should know

- K-means for clustering:
  - algorithm
  - converges because it's coordinate ascent
- EM for mixture of Gaussians:
  - How to "learn" maximum likelihood parameters (locally max. like.) in the case of unlabeled data
- Be happy with this kind of probabilistic analysis
- Remember, E.M. can get stuck in local minima, and empirically it <u>DOES</u>
- EM is coordinate ascent