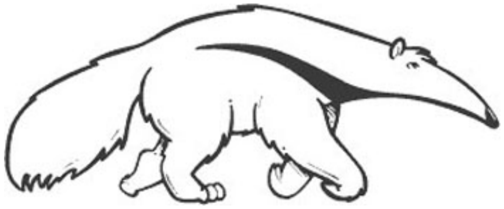
Machine Learning and Data Mining

Clustering

Prof. Alexander Ihler

Fall 2012

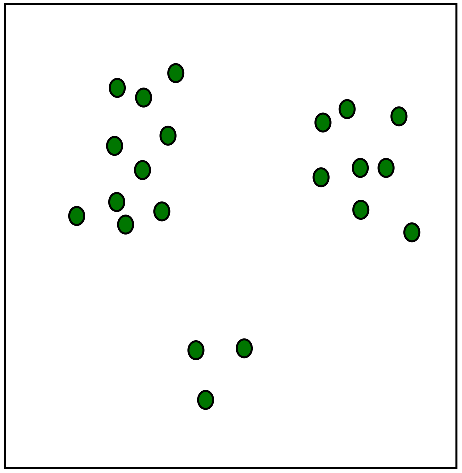


BREN:ICS INFORMATION AND COMPUTER SCIENCES

UNIVERSITY of CALIFORNIA IRVINE

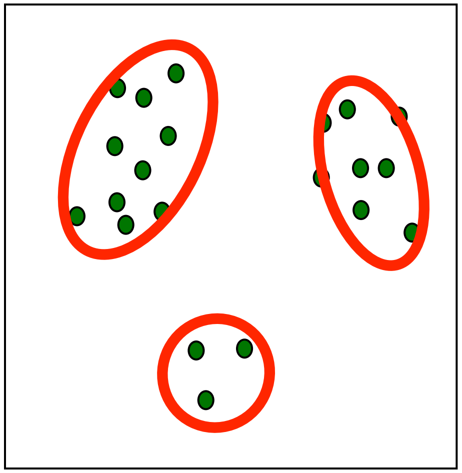
Unsupervised learning

- Supervised learning
  - Predict target value (“y”) given features (“x”)
- Unsupervised learning
  - Understand patterns of data (just “x”)
  - Useful for many reasons
    - Data mining (“explain”)
    - Missing data values (“impute”)
    - Representation (feature generation or selection)



Unsupervised learning

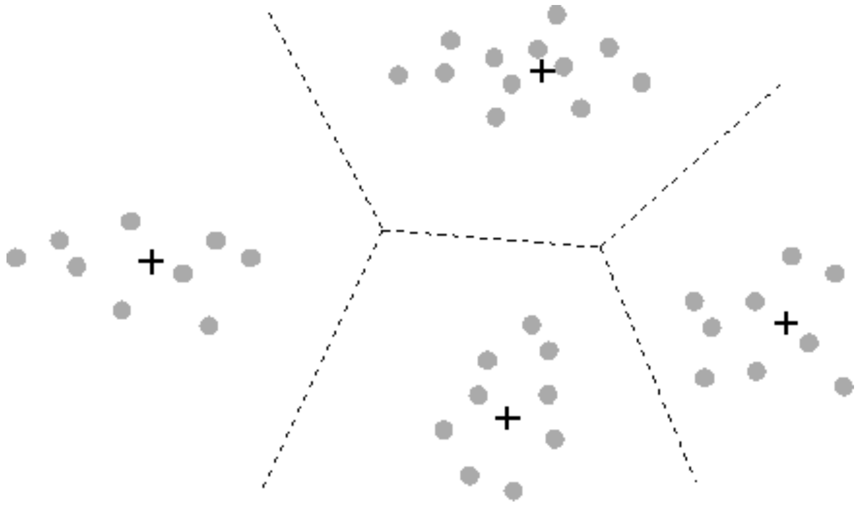
- Supervised learning
  - Predict target value (“y”) given features (“x”)
- Unsupervised learning
  - Understand patterns of data (just “x”)
  - Useful for many reasons
    - Data mining (“explain”)
    - Missing data values (“impute”)
    - Representation (feature generation or selection)

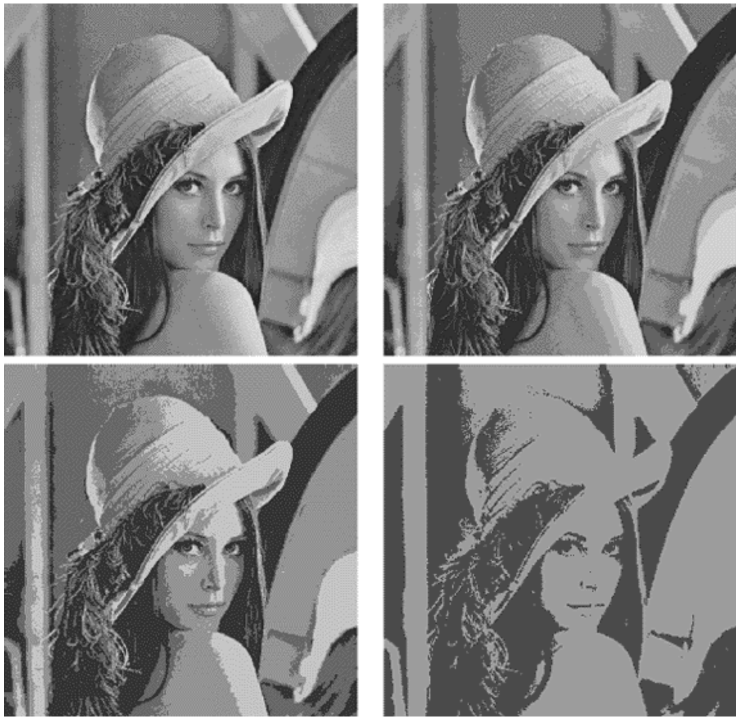


- One example: *clustering*

Clustering and Data Compression

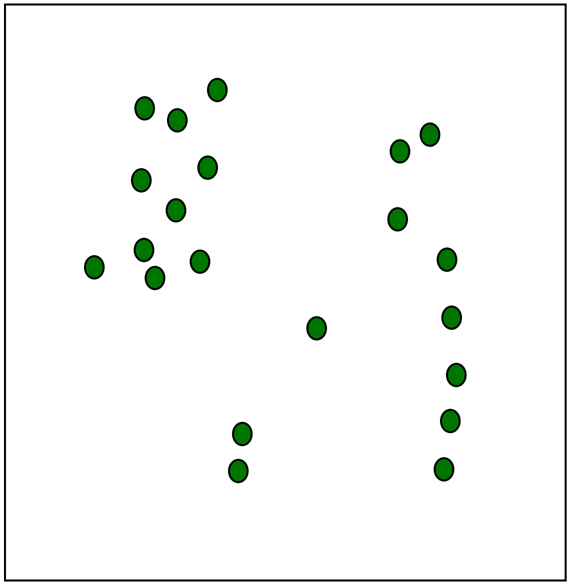
- Clustering is related to vector quantization
  - Dictionary of vectors (the cluster centers)
  - Each original value represented using a dictionary index
  - Each center “claims” a nearby region (Voronoi region)





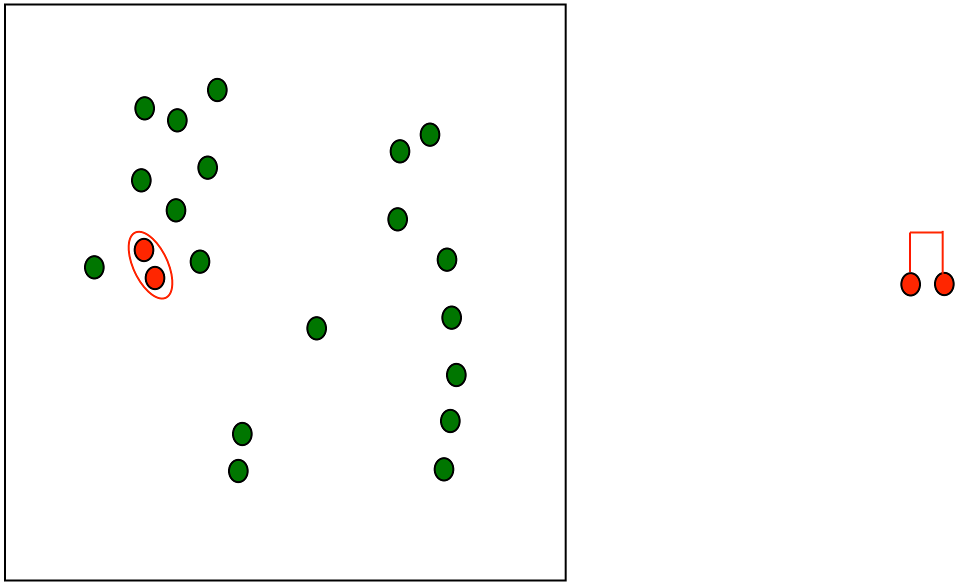
Hierarchical Agglomerative Clustering

Initially, every datum is a cluster

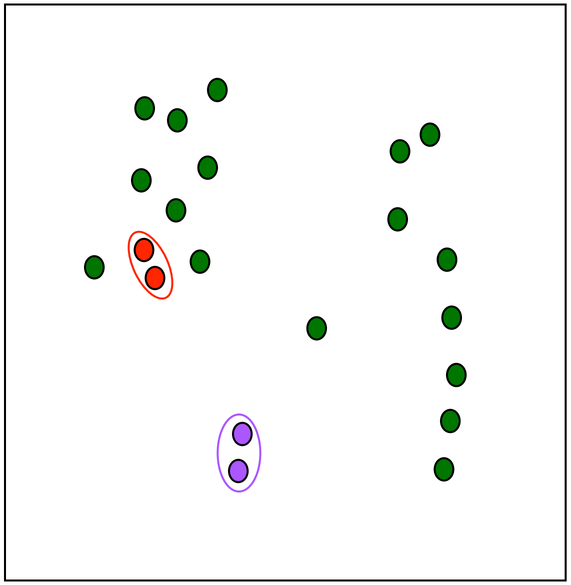


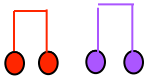
- Another simple clustering alg
- Define a distance between clusters (return to this)
- Initialize: every example is a cluster
- Iterate:
  - Compute distances between all clusters
    (store for efficiency)
  - Merge two closest clusters
- Save both clustering and *sequence* of cluster ops
- “Dendrogram”

Iteration 1



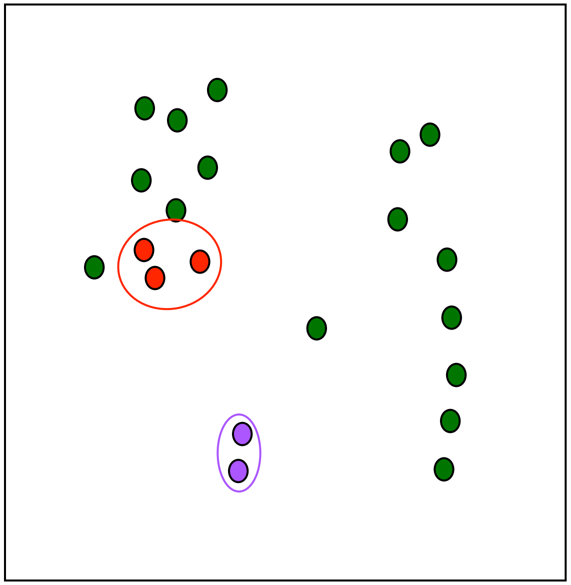
Iteration 2

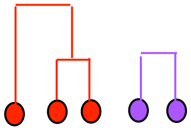




Iteration 3

- Builds up a sequence of clusters ("hierarchical")

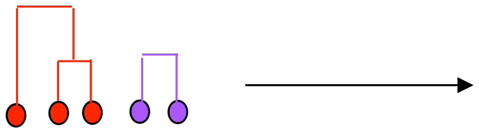


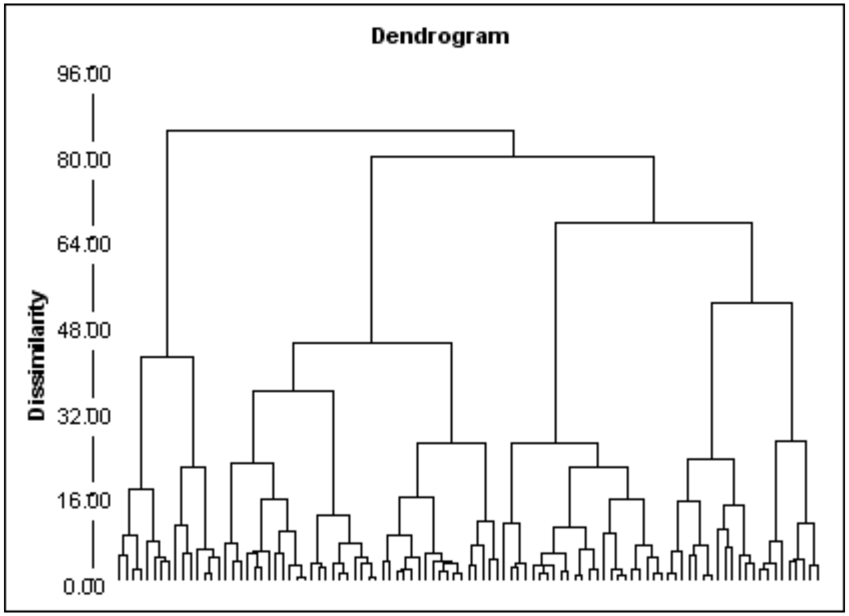


- Algorithm complexity $O(N^2)$ (Why?)

In matlab: "linkage" function (stats toolbox)

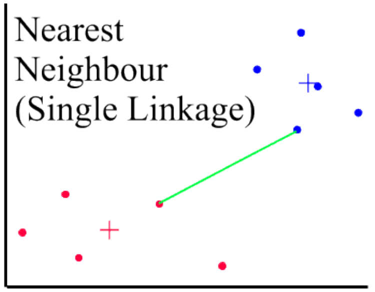
Dendrogram




Dendrogram
96.00
80.00
64.00
48.00
32.00
16.00
0.00
Dissimilarity

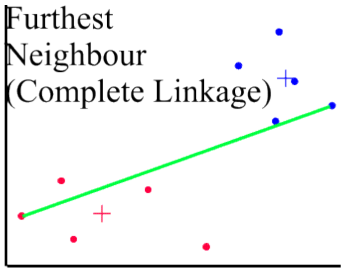
Cluster Distances

$$D_{\min}(C_i, C_j) = \min_{x \in C_i,\ y \in C_j} \|x - y\|^2$$



produces minimal spanning tree.

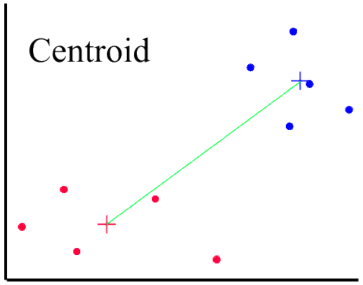
$$D_{\max}(C_i, C_j) = \max_{x \in C_i,\ y \in C_j} \|x - y\|^2$$



avoids elongated clusters.

$$D_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i,\ y \in C_j} \|x - y\|^2$$

$$D_{\text{means}}(C_i, C_j) = \|\mu_i - \mu_j\|^2$$



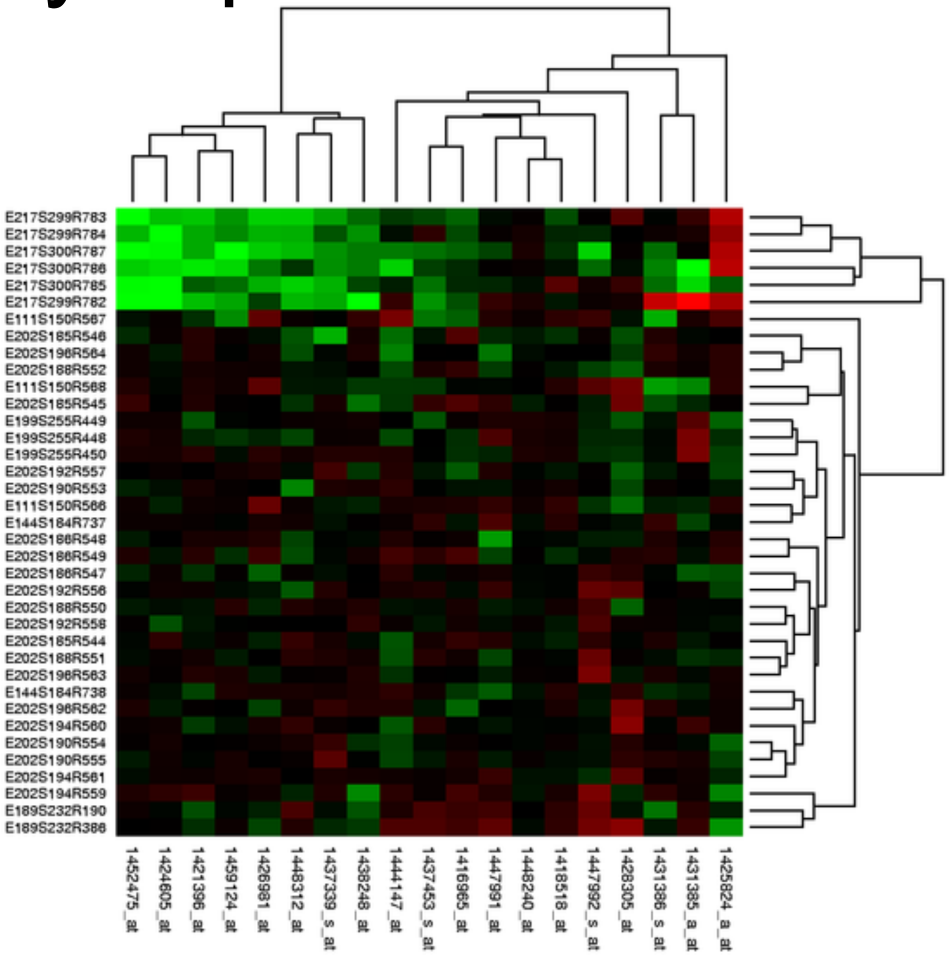
Example: microarray expression

- Measure gene expression

- Various experimental conditions
  - Cancer, normal
  - Time
  - Subjects

- Explore similarities
  - What genes change together?
  - What conditions are similar?

- Cluster on both genes and conditions



K-Means Clustering

- A simple clustering algorithm
- Iterate between
  - Updating the assignment of data to clusters
  - Updating the cluster’ s summarization
- Suppose we have K clusters, c=1..K
  - Represent clusters by locations μ_c
  - Example i has features x_i
  - Represent assignment of i^{th} example $z_i \in 1..K$
- Iterate until convergence:
  - For each datum, find the closest cluster

$$z_i = \arg\min_c \|x_i - \mu_c\|^2 \qquad \forall i$$

  - Set each cluster to the mean of all assigned data:

$$\forall c, \qquad \mu_c = \frac{1}{N_c} \sum_{i \in S_c} x_i \qquad\qquad S_c = \{i : z_i = c\},\ N_c = |S_c|$$

K-Means as optimization

- Optimizing the cost function

$$C(\underline{z}, \underline{\mu}) = \sum_i \|x_i - \mu_{z_i}\|^2$$

- Greedy descent technique
  - Steps
    - Choose closest cluster
    - Choose mean of assigned data
  - Each step only decreases the cost (why?)

- As with any descent method, beware of local minima
  - Algorithm behavior depends significantly on initalization
  - Many heuristics
    - Random (not bad); Farthest (sensitive); some mix maybe?

Choosing the number of clusters

- With cost function

$$C(\underline{z}, \underline{\mu}) = \sum_i \|x_i - \mu_{z_i}\|^2$$

  what is the optimal value of k?
- Can increasing k ever increase the cost?

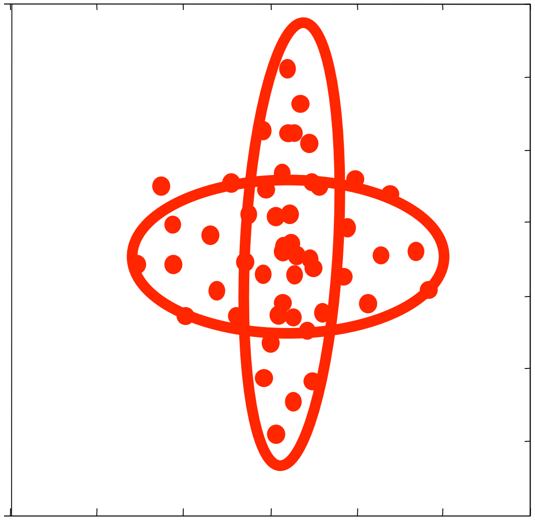
- This is a model complexity issue
  - Much like choosing lots of features – they only (seem to) help
  - But we want our clustering to *generalize* to new data

- One solution is to penalize for complexity
  - Bayesian information criterion (BIC)
  - Add (# parameters) * log(N) to the cost
  - Now more clusters can increase cost, if they don't help "enough"

Mixtures of Gaussians

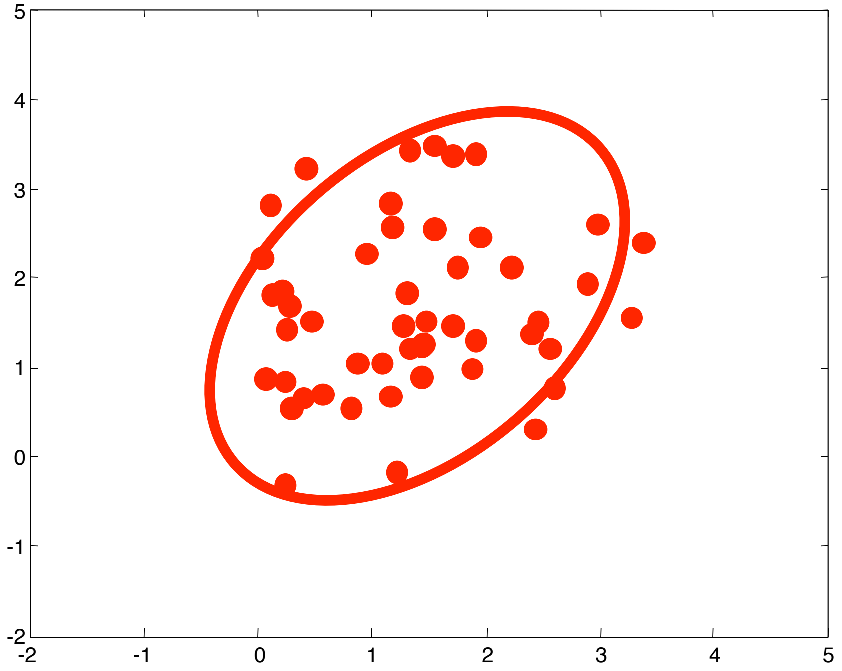
- K-means algorithm
  - Assigned each example to exactly one cluster
  - What if clusters are overlapping?
    - Hard to tell which cluster is right
    - Maybe we should try to remain uncertain
  - Used Euclidean distance
  - What if cluster has a non-circular shape?

- Gaussian mixture models
  - Clusters modeled as Gaussians
    - Not just by their mean
  - EM algorithm: assign data to cluster with some *probability*



Multivariate Gaussian models

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



Maximum Likelihood estimates

$$\hat{\mu} = \frac{1}{N} \sum_i x^{(i)}$$

$$\hat{\Sigma} = \frac{1}{N} \sum_i (x^{(i)} - \hat{\mu})^T (x^{(i)} - \hat{\mu})$$

We' ll model each cluster using one of these Gaussian "bells"…

EM Algorithm: E-step

- Start with parameters describing each cluster
- Mean μ_c, Covariance Σ_c, "size" π_c

- E-step ("Expectation")
  - For each datum (example) x_i,
  - Compute "r_{ic}", the probability that it belongs to cluster c
    - Compute its probability under model c
    - Normalize to sum to one (over clusters c)

$$r_{ic} = \frac{\pi_c \mathcal{N}(x_i \; ; \; \mu_c, \Sigma_c)}{\sum_{c'} \pi_{c'} \mathcal{N}(x_i \; ; \; \mu_{c'}, \Sigma_{c'})}$$

  - If x_I is very likely under the cth Gaussian, it gets high weight
  - Denominator just makes r's sum to one

EM Algorithm: M-step

- Start with assignment probabilities r_{ic}
- Update parameters: mean μ_c, Covariance Σ_c, “size” π_c

- M-step (“Maximization”)
  - For each cluster (Gaussian) x_c,
  - Update its parameters using the (weighted) data points

$$N_c = \sum_i r_{ic}$$

Total responsibility allocated to cluster c

$$\pi_c = \frac{N_c}{N}$$

Fraction of total assigned to cluster c

$$\mu_c = \frac{1}{N_c} \sum_i r_{ic} x_i$$

Weighted mean of assigned data

$$\Sigma_c = \frac{1}{N_c} \sum_i r_{ic} (x_i - \mu_c)^T (x_i - \mu_c)$$

Weighted covariance of assigned data (use new weighted means here)

Expectation-Maximization

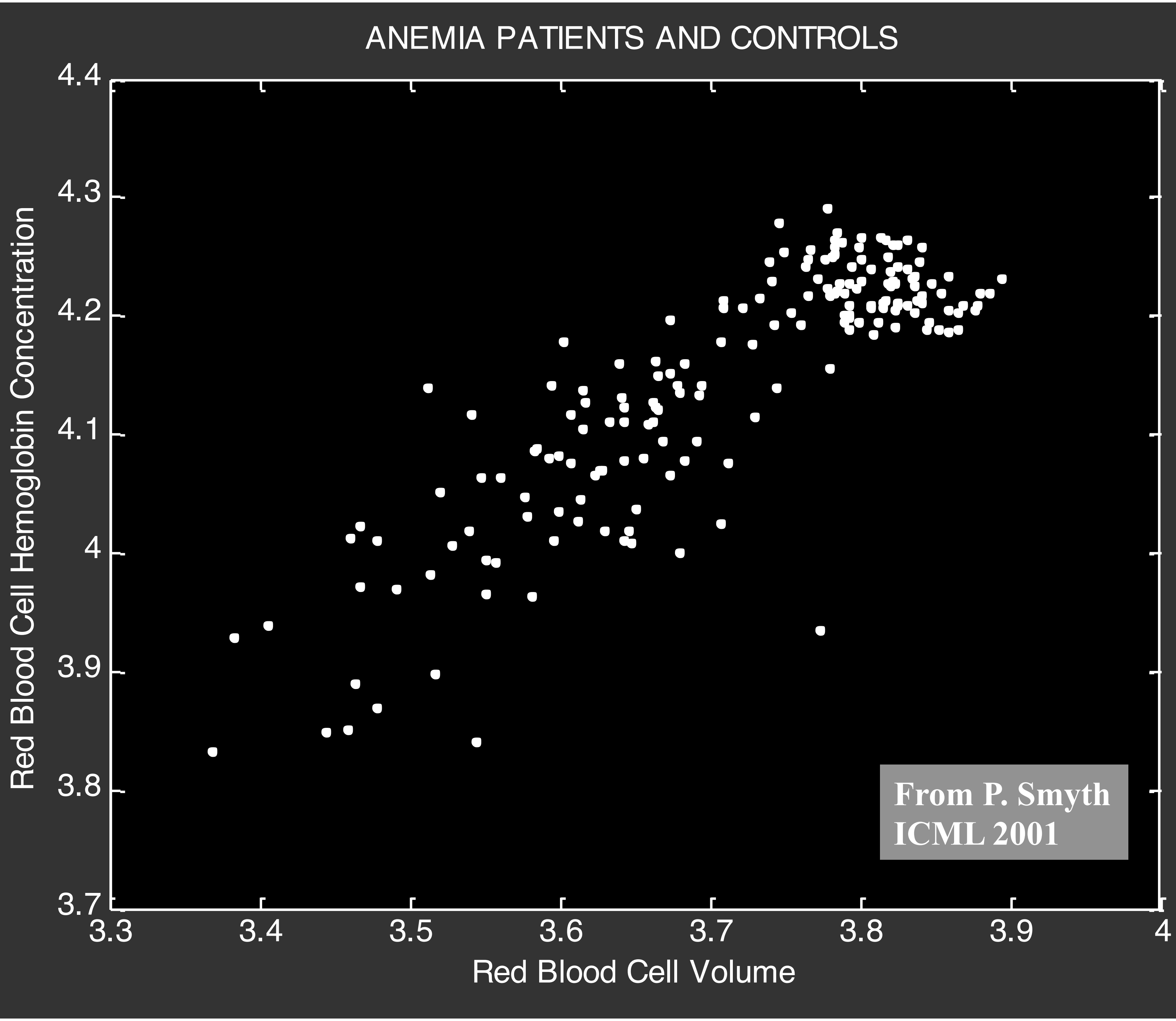
- Each step increases the log-likelihood of our model

$$\log p(\underline{X}) = \sum_i \log \left[\sum_c \pi_c \, \mathcal{N}(x_i \; ; \; \mu_c, \Sigma_c) \right]$$

(we won't derive this, though)

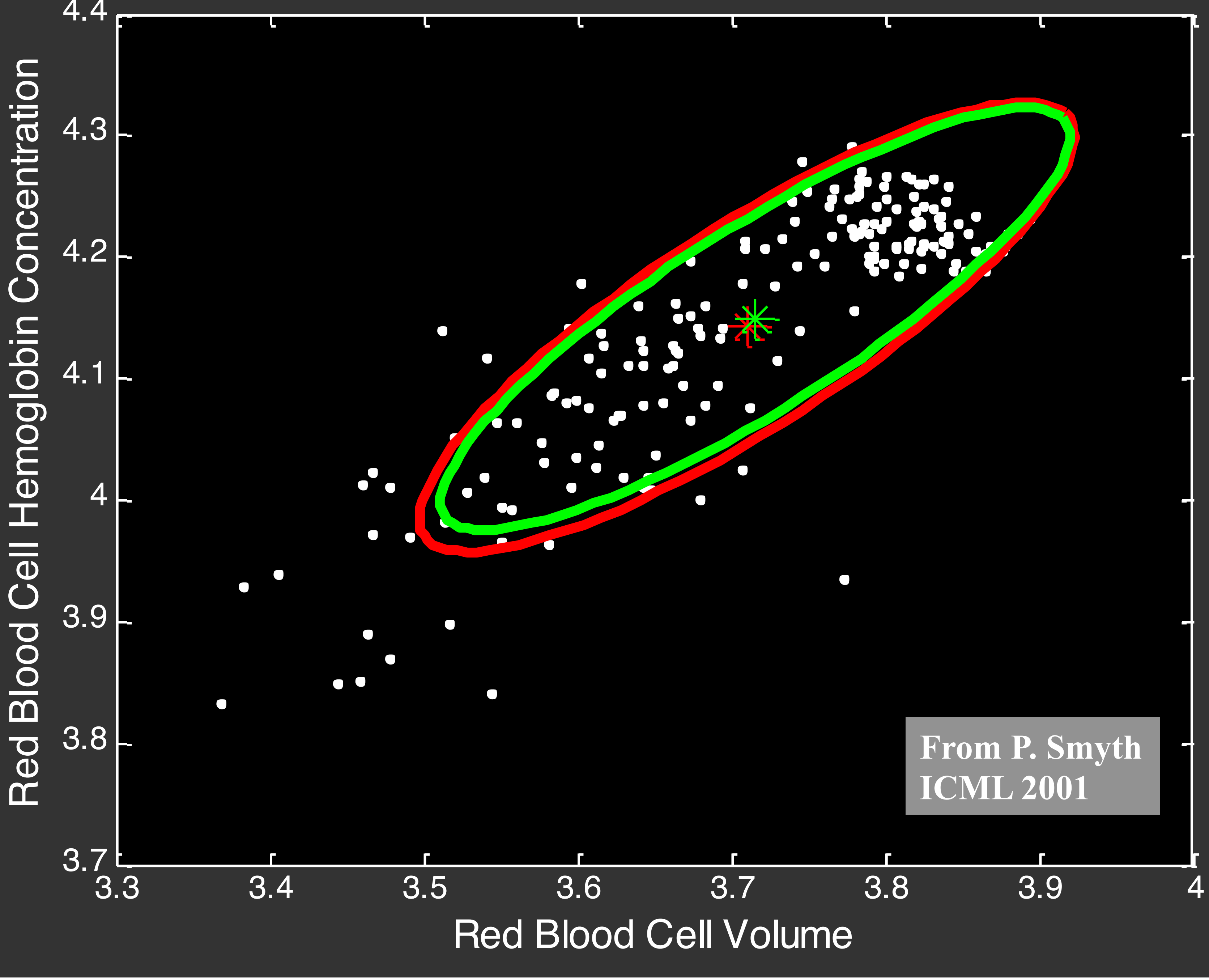
- Iterate until convergence
  - Convergence guaranteed – another ascent method

- What should we do
  - If we want to choose a single cluster for an "answer"?
  - With new data we didn't see during training?



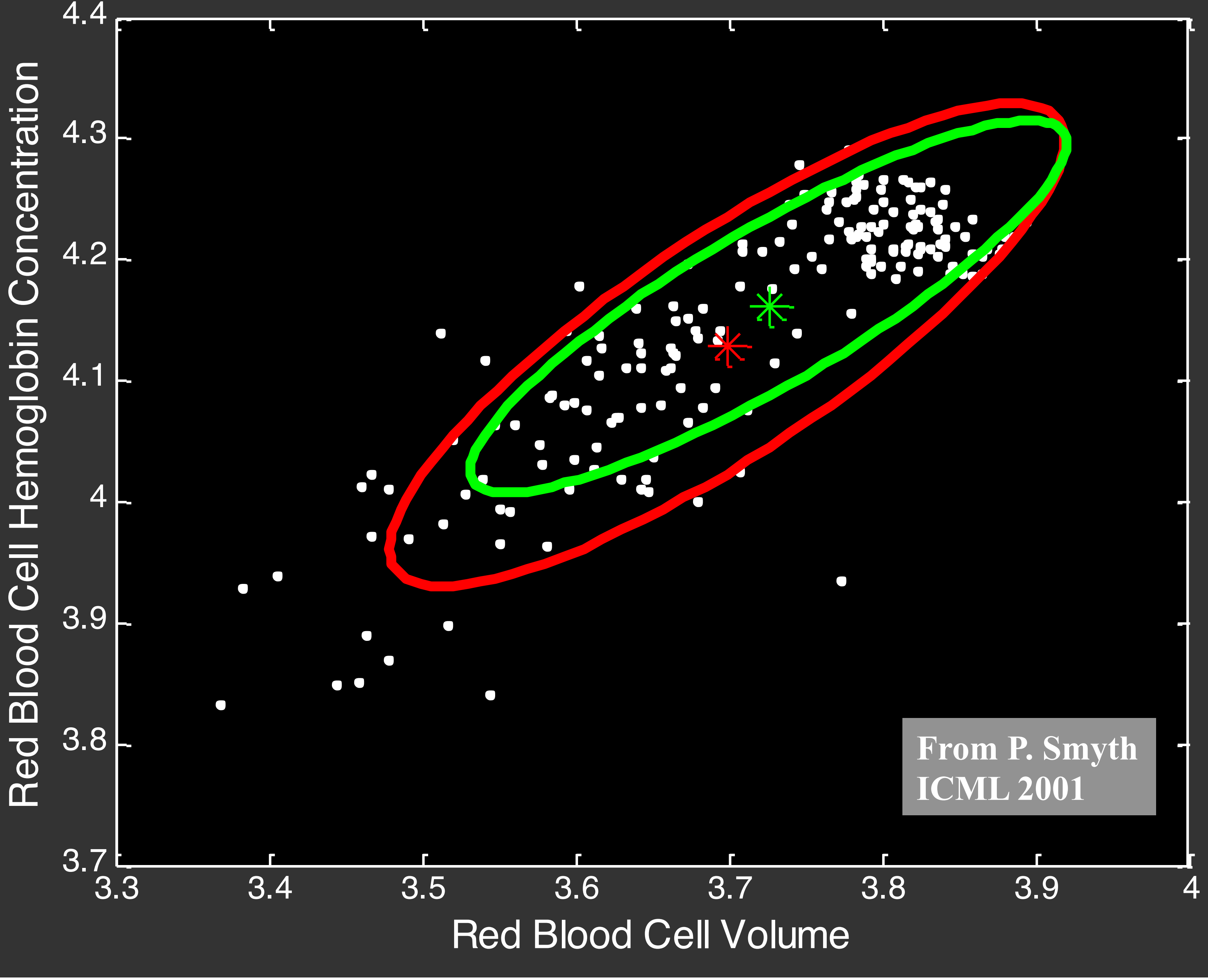
From P. Smyth
ICML 2001

EM ITERATION 1

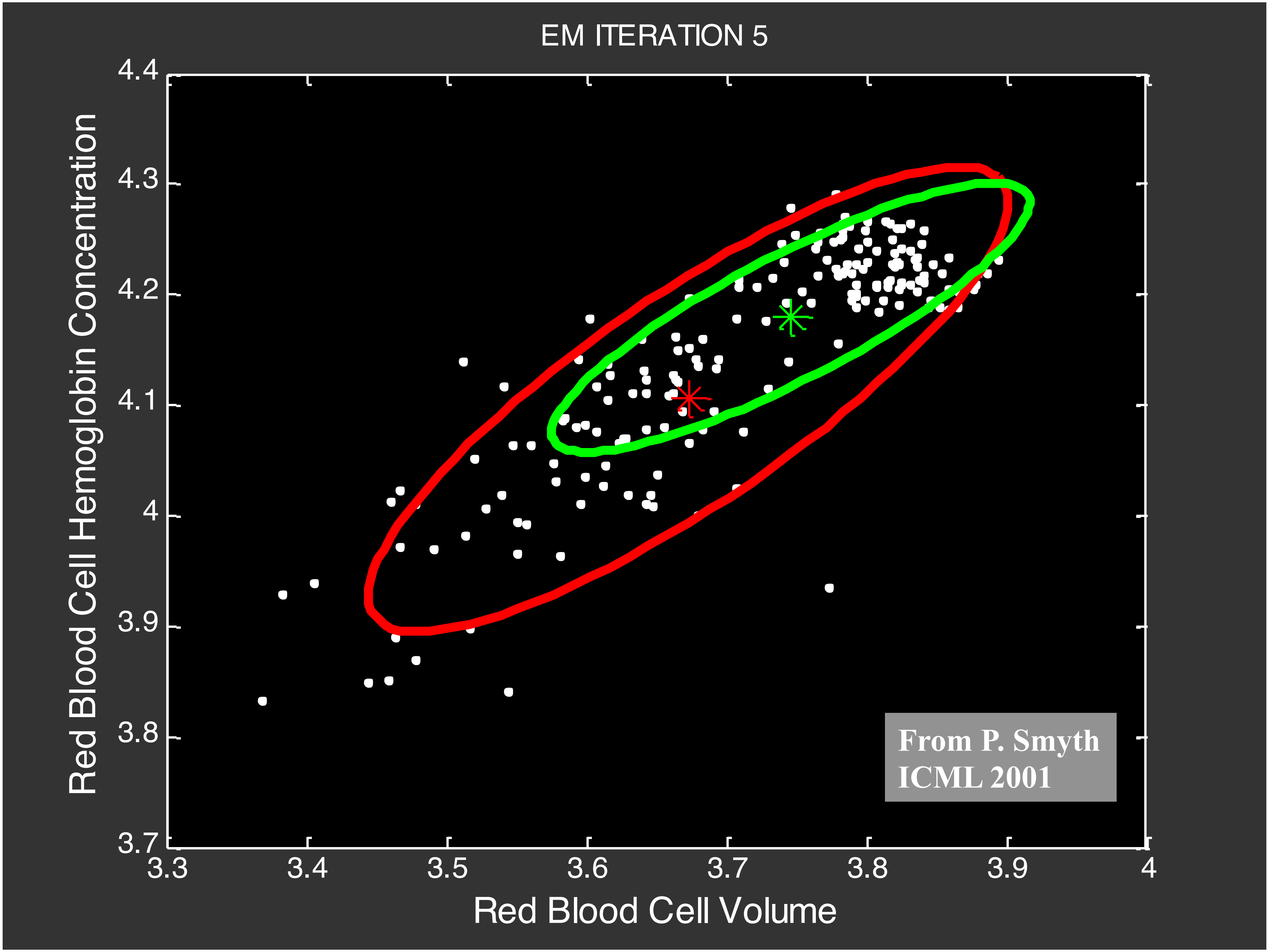


From P. Smyth
ICML 2001

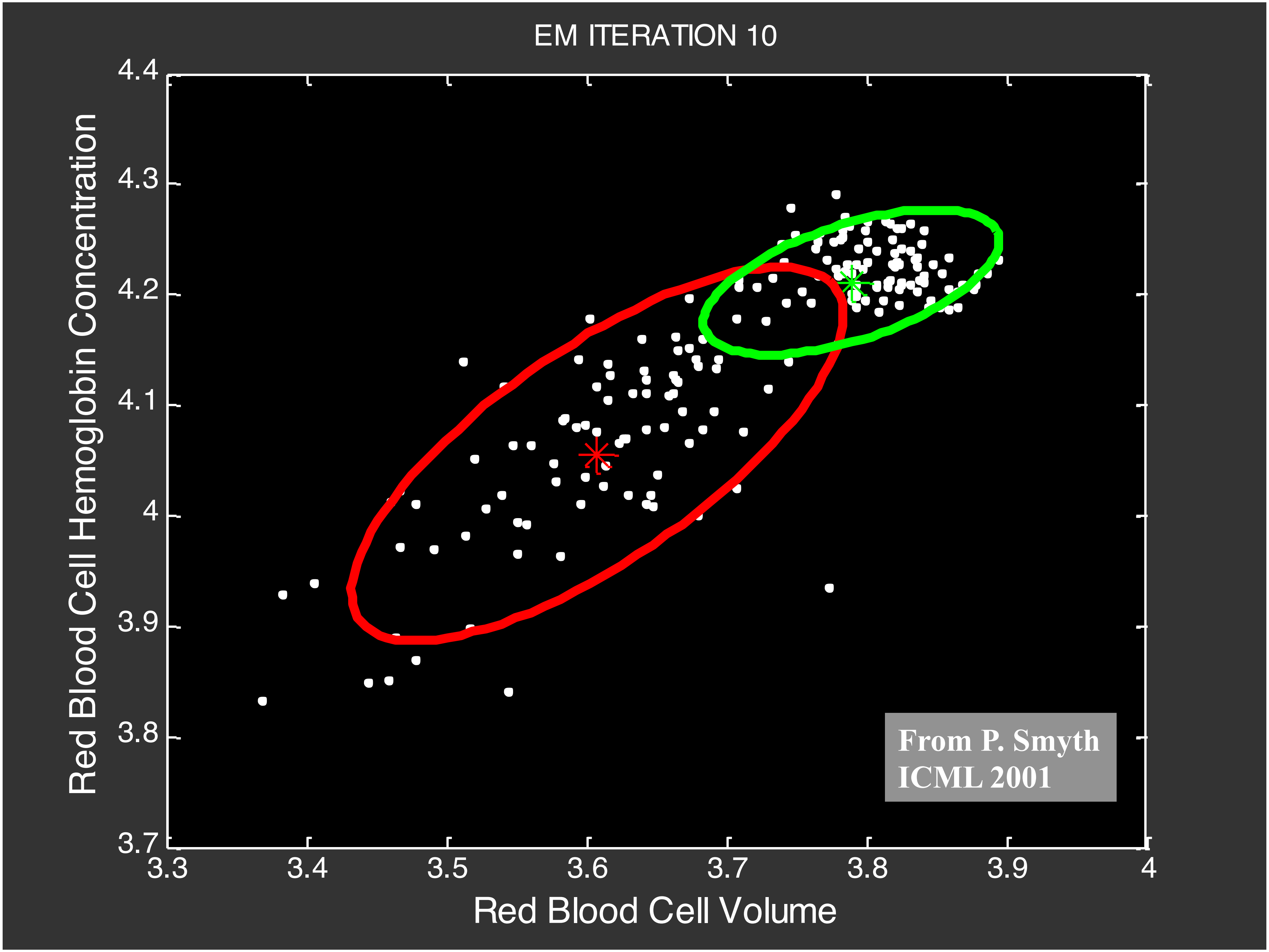
EM ITERATION 3



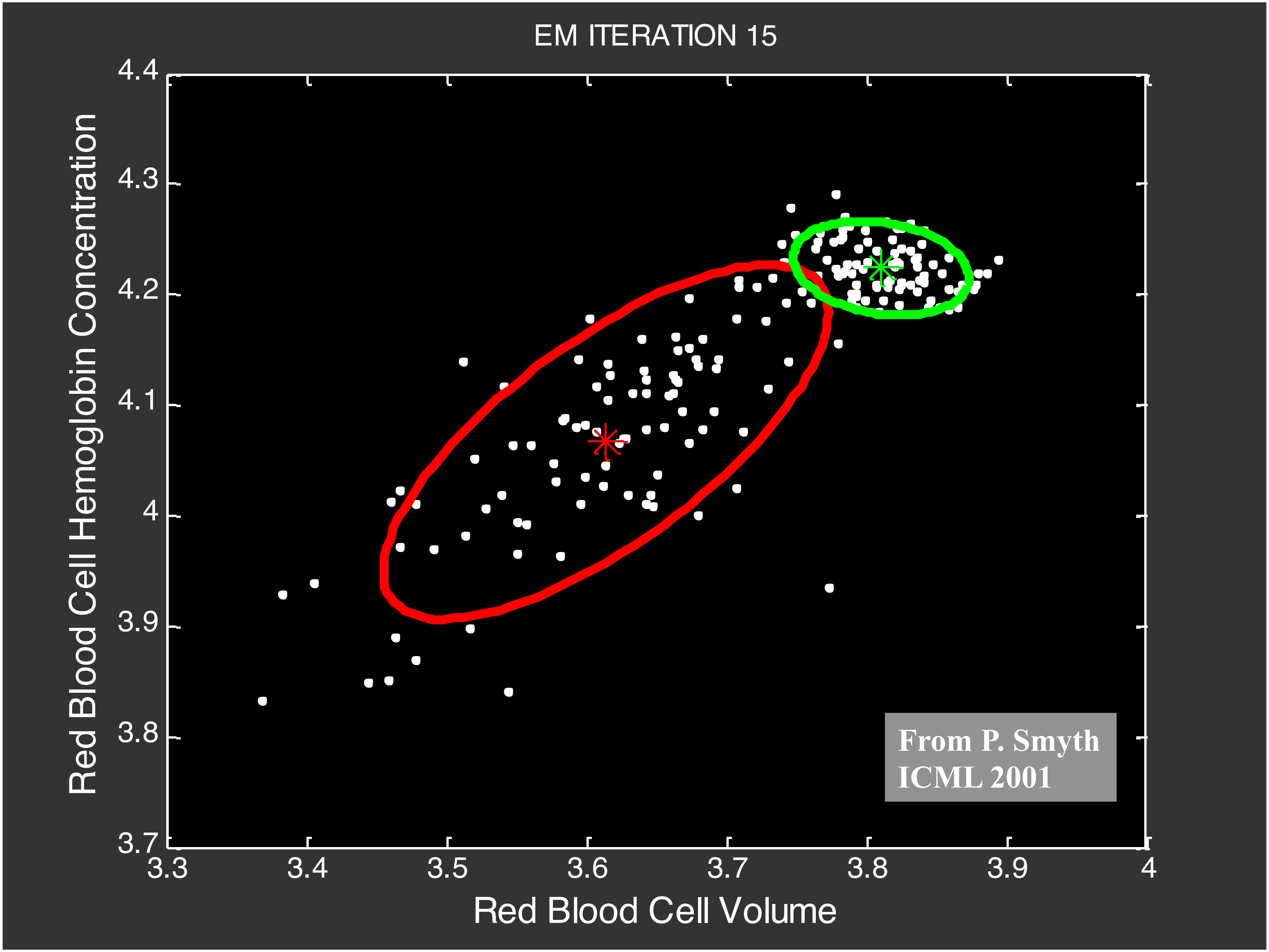
From P. Smyth
ICML 2001



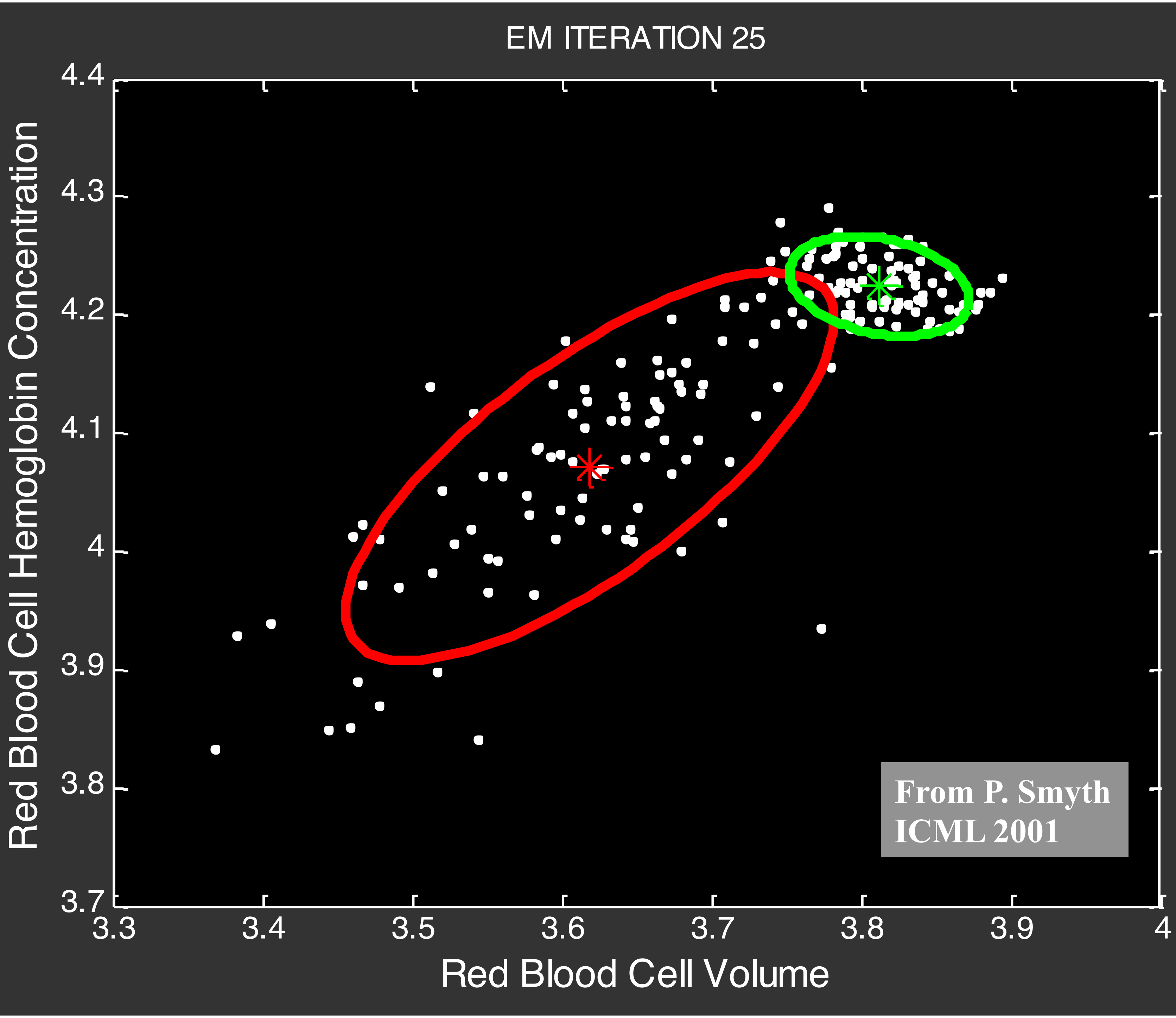
From P. Smyth
ICML 2001



From P. Smyth
ICML 2001

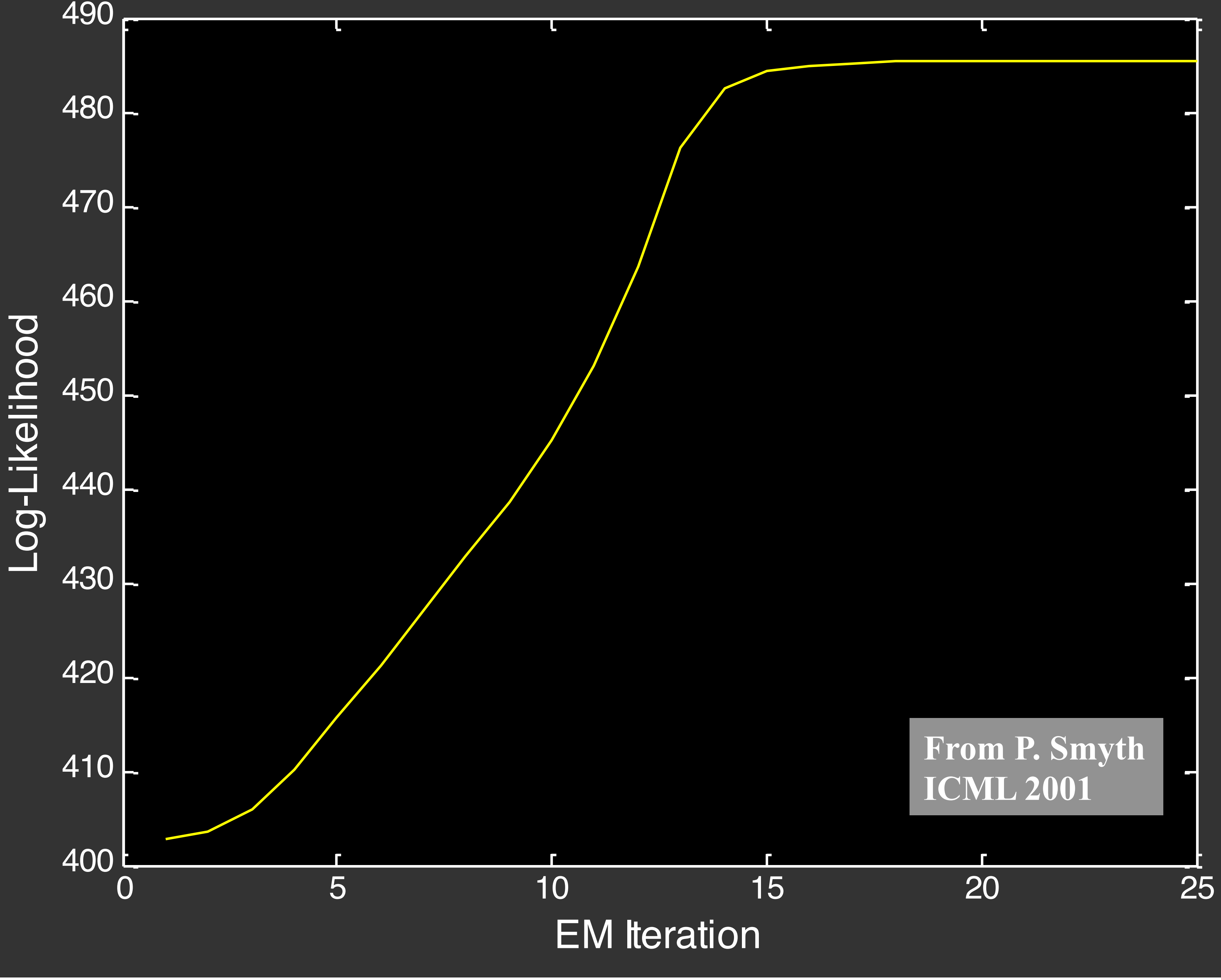


From P. Smyth
ICML 2001



From P. Smyth
ICML 2001

LOG-LIKELIHOOD AS A FUNCTION OF EM ITERATIONS



From P. Smyth
ICML 2001

EM and missing data

- EM is a general framework for partially observed data
  - “Complete data” xi, zi – features and assignments
  - Assignments zi are missing (unobserved)

- EM corresponds to
  - Computing the distribution over all zi given the parameters
  - Maximizing the “expected complete” log likelihood
  - GMMs = plug in “soft assignments”, but not always so easy

- Alternatives: Hard EM, Stochastic EM
  - Instead of expectations, just sample the zi or choose best (often easier)
  - Called “imputing” the values of zi
  - Hard EM: similar to EM, but less “smooth”
  - Stochastic EM: similar to EM, but with extra randomness
    - Not obvious when it has converged

Gibbs sampling for clustering

- Another technique for inferring uncertain cluster assignments
  - K-means: take the best assignment
  - EM: assign “partially”
  - Stochastic EM: sample assignment
  - All: choose best cluster descriptions given assignments
- Gibbs sampling (“Markov chain Monte Carlo”)
  - Assign randomly, probability equal to EM’s weight
  - *Sample* a cluster description given assignment
  - Requires a probability model over cluster parameters

- This doesn’t really find the “best” clustering
  - It eventually samples almost all “good” clusterings
  - Converges “in probability”, randomness helps us explore configurations
  - Also tells us about uncertainty of clustering
  - Disadvantage: not obvious when “done”

“Infinite” mixture models

- How many clusters are there?

- Gibbs sampling has an interesting solution
  - Write a distribution over k, the # of clusters
  - Sample k also

- Can do our sampling sequentially
  - Draw each zi given all the others
  - Instead of sampling cluster parameters, marginalize them
  - Defines a distribution over groupings of data
- Now, for each zi, sample
  - Join an existing cluster? Or, join a new cluster?
- What are these probabilities?
  - “Dirichlet process” mixture models

Parametric and Nonparametric Models

- Every model has some parameters
  - “The stuff you have to store to make your prediction”
  - Logistic regression: weights
  - Decision tree: feature to split, value at each level
  - Gaussian mixture model: means, covariances, sizes
- Parametric vs Nonparametric models
  - Parametric: fixed # of parameters
  - Nonparametric: # of parameters grows with more data
- What type are
  - Logistic regression?
  - Nearest neighbor prediction?
  - Decision trees?
  - Decision trees of depth < 3?
  - Gaussian mixture model?

Summary

- Clustering algorithms
  - Agglomerative clustering
  - K-means
  - Expectation-Maximization
- Open questions for each application

- What does it mean to be “close” or “similar”?
  - Depends on your particular problem…

- “Local” versus “global” notions of simliarity
  - Former is easy, but we usually want the latter…

- Is it better to “understand” the data itself (unsupervised learning), to focus just on the final task (supervised learning), or both?