COMPUTER SCIENCE
INDIANA UNIVERSITY
School of Informatics and Computing
Bloomington

Mixture models

Reminders/Comments

- Thought questions due today
- Assignment 3 due on Wednesday
- After Thanksgiving, we will switch to review of the course material
- Let’s take a poll for concepts/sections that are particularly confusing to you

Feedback form Q2

- Assuming that p(y I x) is Bernoulli would be a reasonable choice

- Implementation/meta-parameter choices:
  - initialization of parameters
  - number of random restarts, or other optimization improvements to escape from local minima
  - number of hidden nodes
  - number of hidden layers
  - transfers on the layers
  - step-size selection and/or decay schedule

Hidden variable models

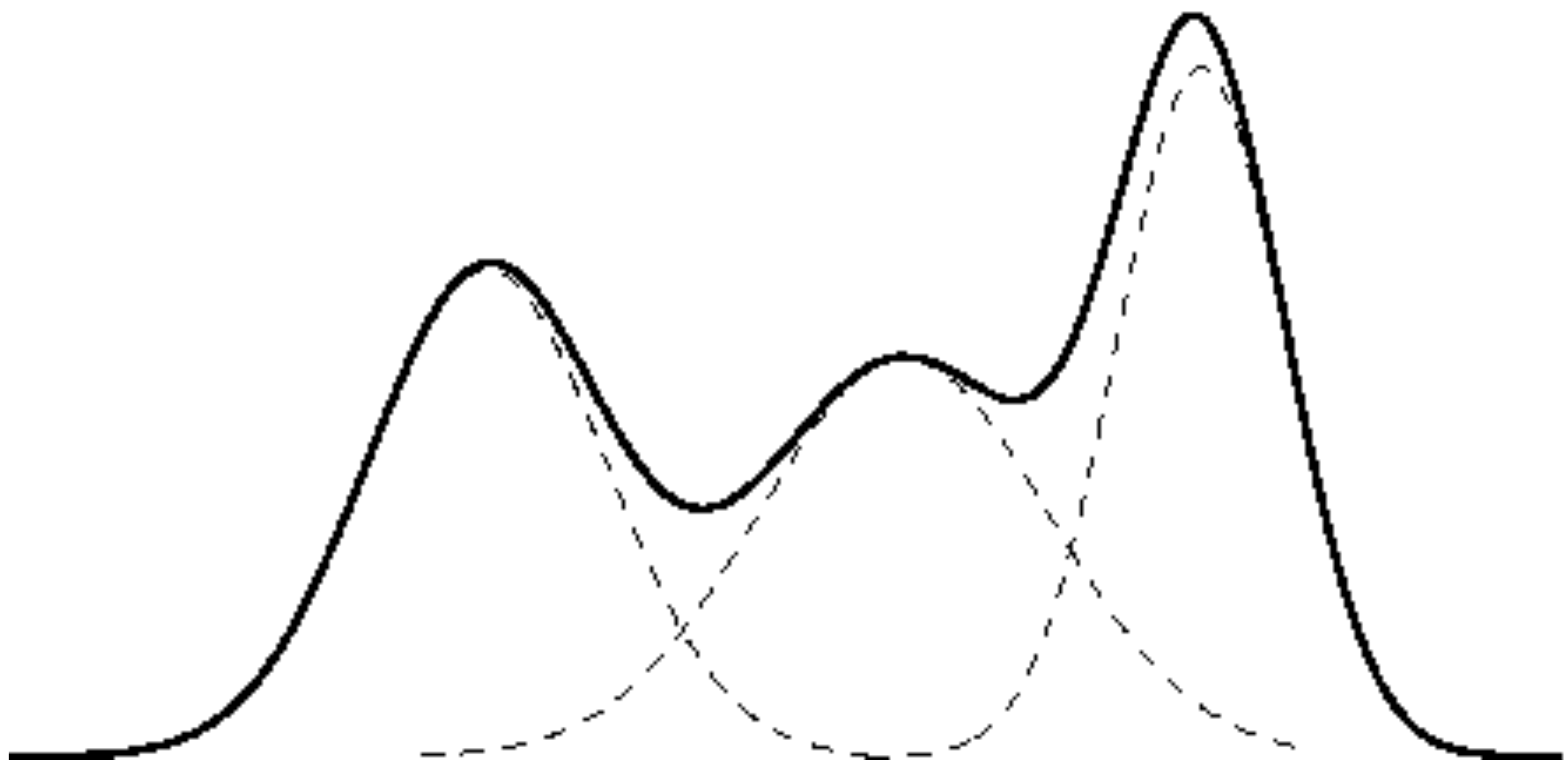
- Probabilistic PCA and factor analysis
  - common in psychology
- Mixture models
- Hidden Markov Models
  - commonly used for NLP and modeling dynamical systems

Probabilistic PCA

- In PCA, we learned p(x l D, h)
  - What were the assumptions on p(x l D, h)?
- For Probabilistic PCA, we learn p(x l D)
- Given some prior p(h), we have

$$p(\mathbf{x}|\mathbf{D}) = \int_{\mathcal{H}} p(\mathbf{x}|\mathbf{D}, \mathbf{h}) p(\mathbf{h}) d\mathbf{h}$$

Gaussian mixture model



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

$$p(x|\theta) = \sum_{j=1}^{m} w_j p(x|\theta_j).$$

Differences to PPCA

- Hidden variable is a discrete number in set {1,…, k}: represents the cluster/label that a sample could belong too
- In PPCA, hidden variable was the right singular vector, of continuous values
- The same lower bound applies, but we use a sum for mixture models (to sum over h) and an integral for PPCA

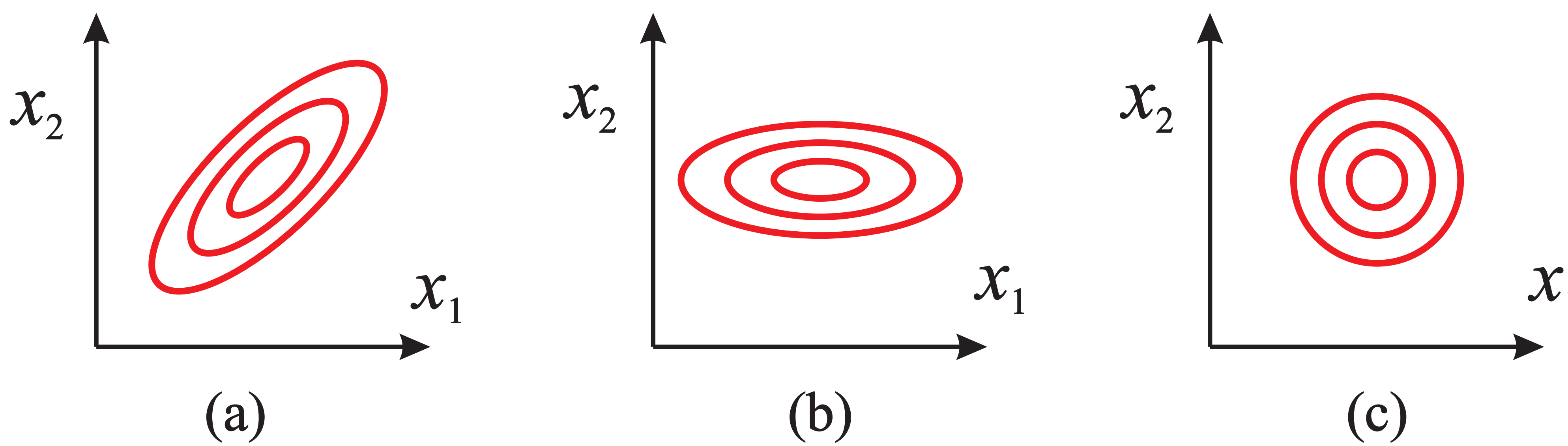
Gaussian distribution

- Multivariate Gaussian

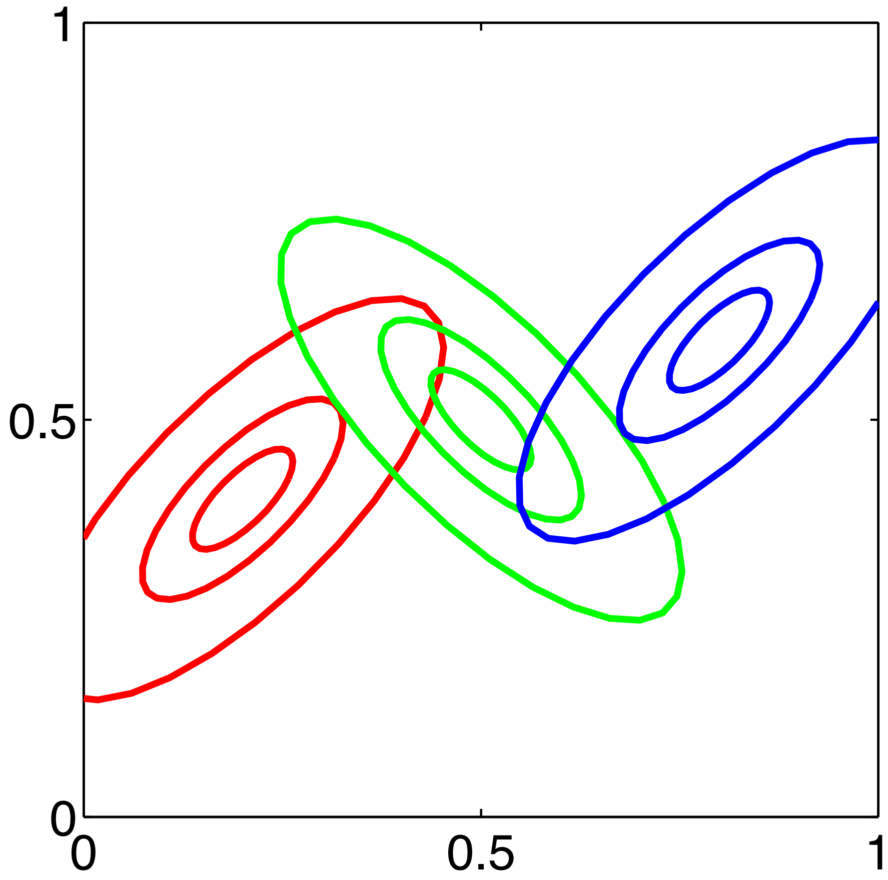
$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

mean

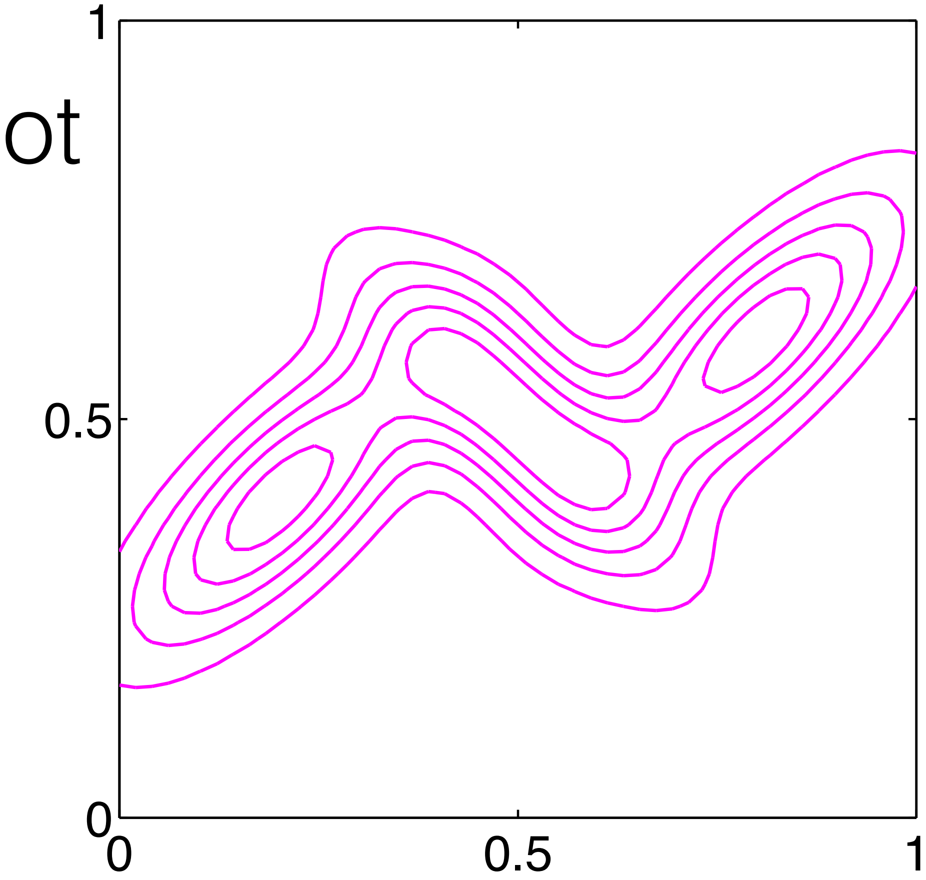
covariance



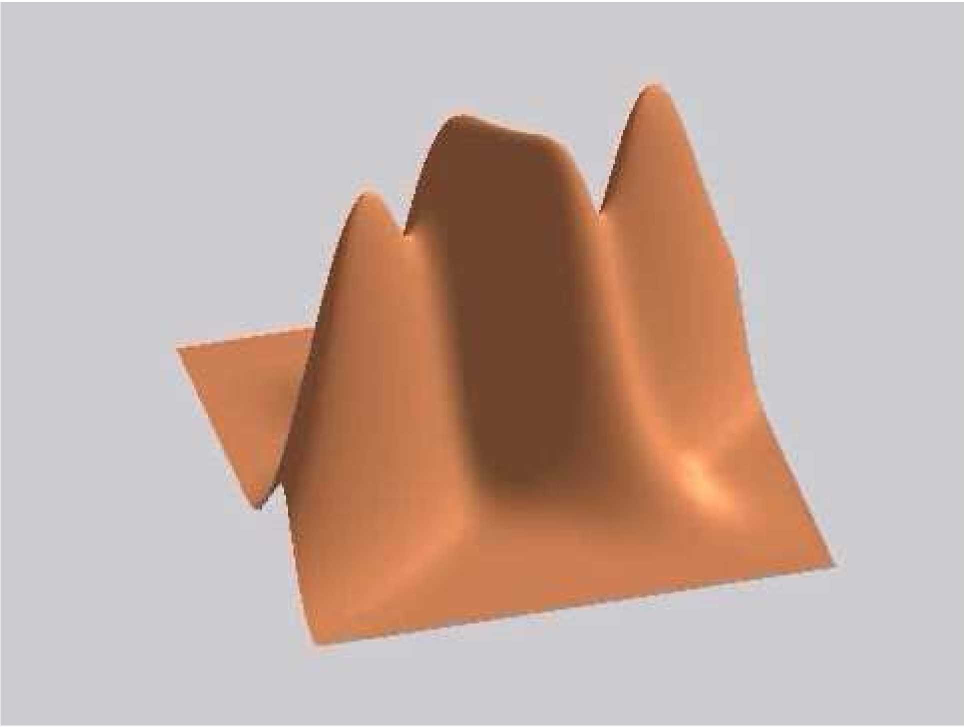
Mixture of 3 Gaussians



Contour plot



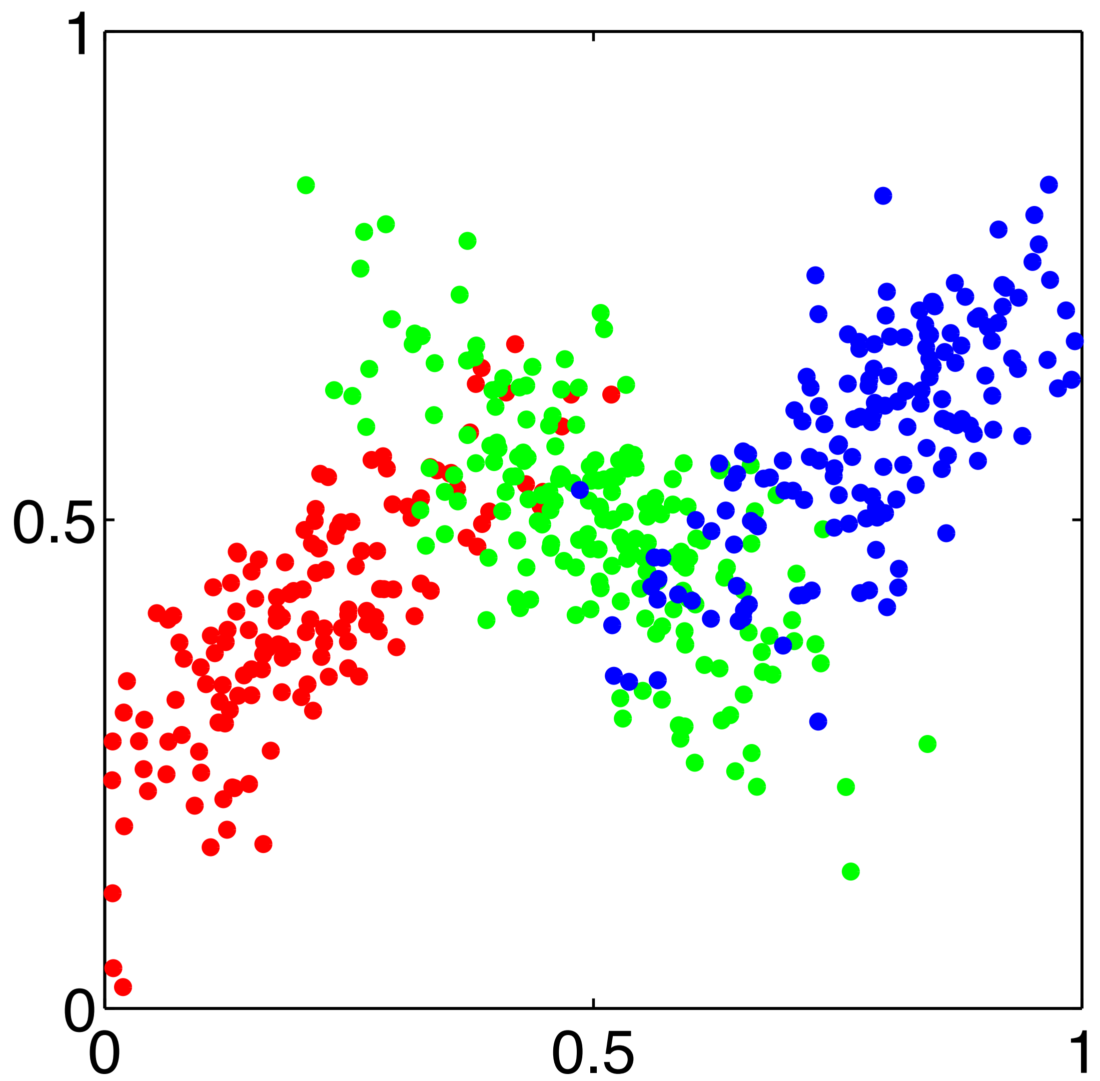
3 contours



Surface plot

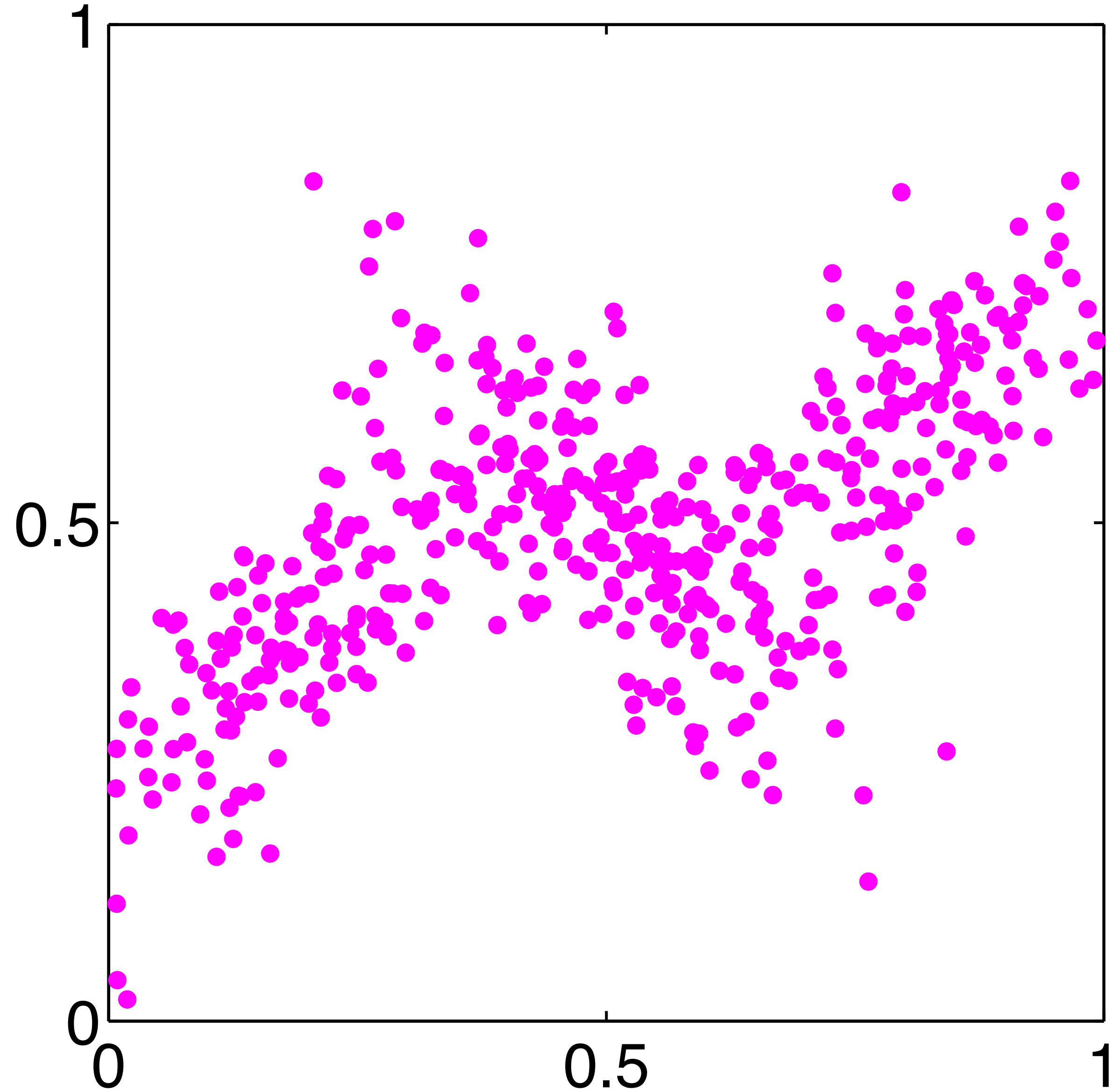
Generating synthetic data

- To generate a point:
  - pick one of the components with probability w_i (e.g., using np.random.choice)
  - draw a sample x from that component (e.g., using np.random.normal)



Other direction: estimation

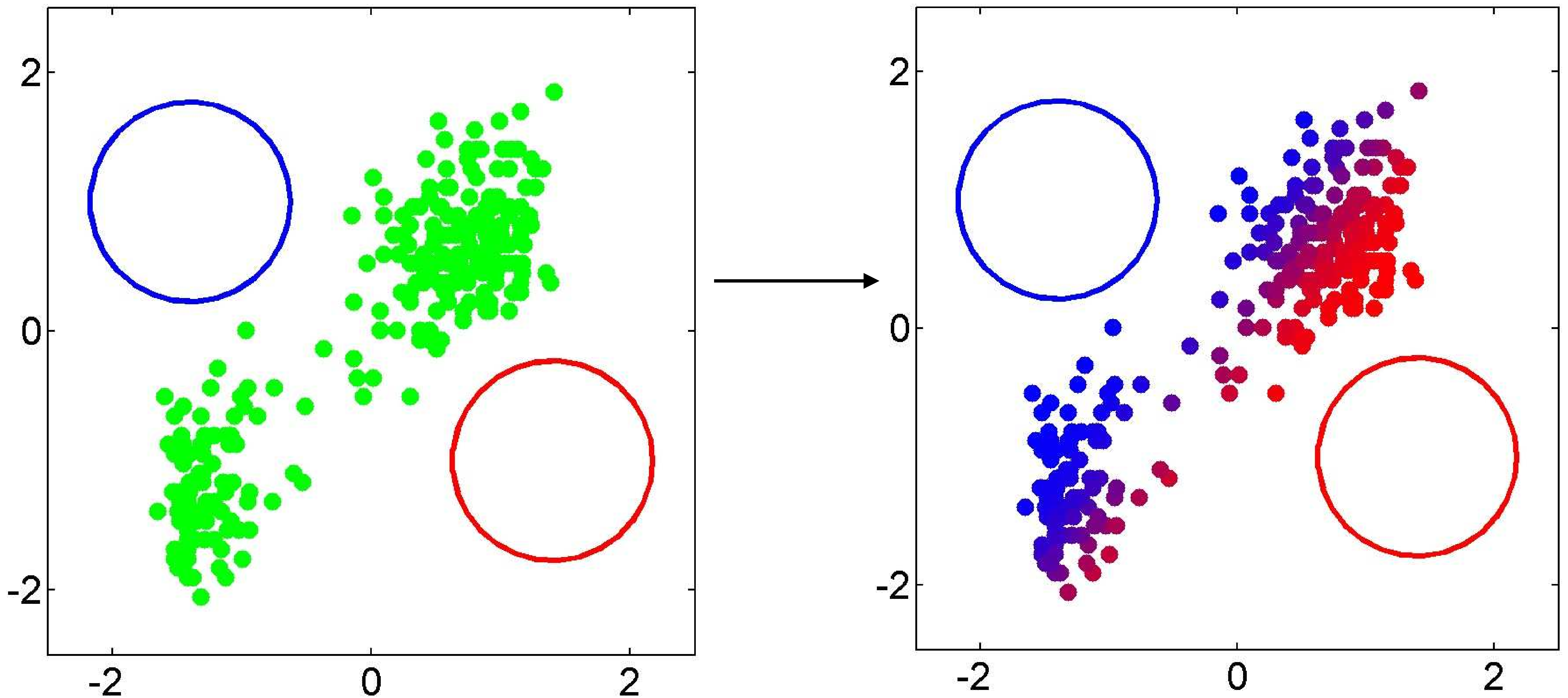
- Given parameters, easy to see how data generated
- Given data, now want to learn/estimate parameters



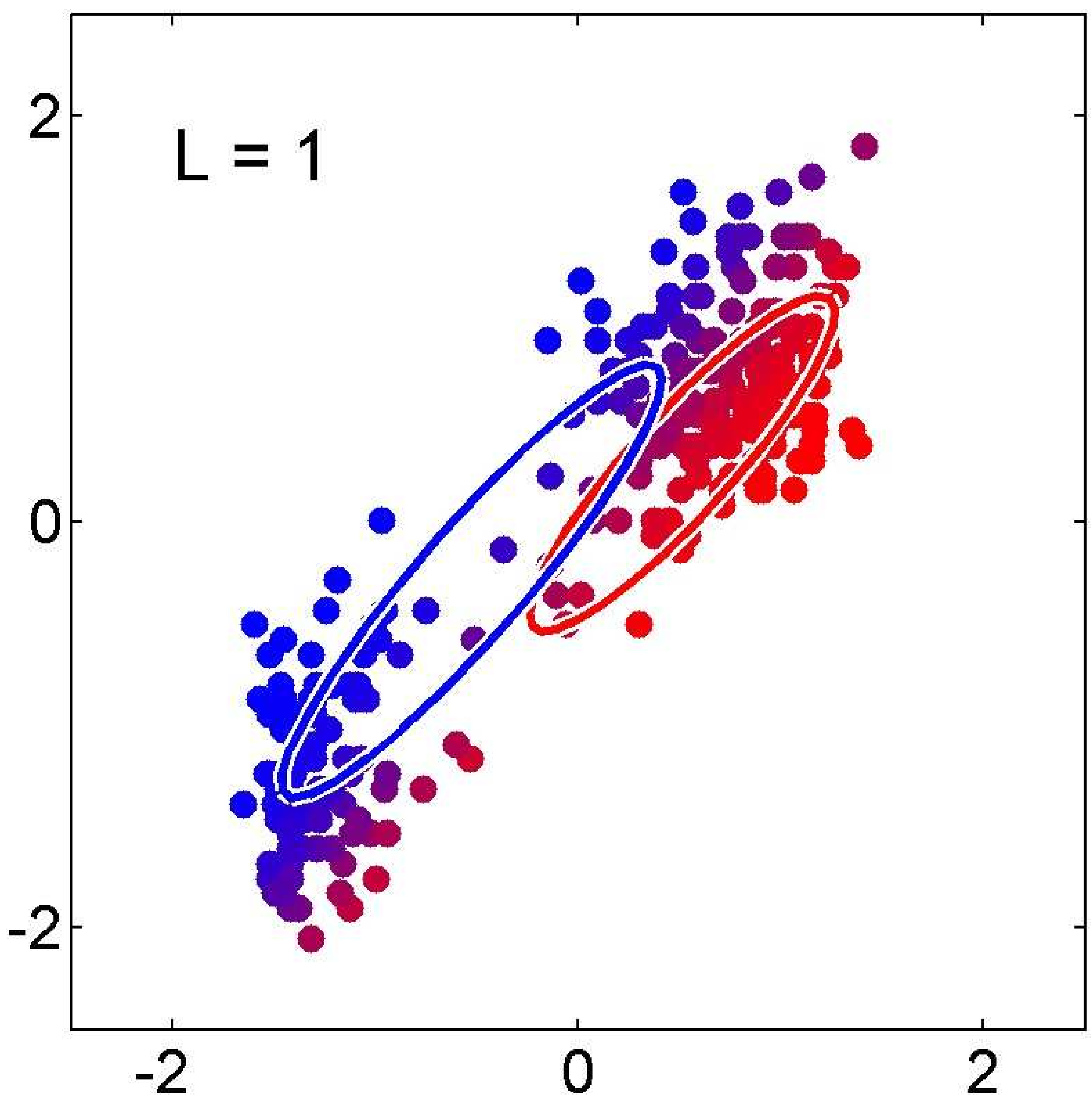
EM algorithm for mixtures

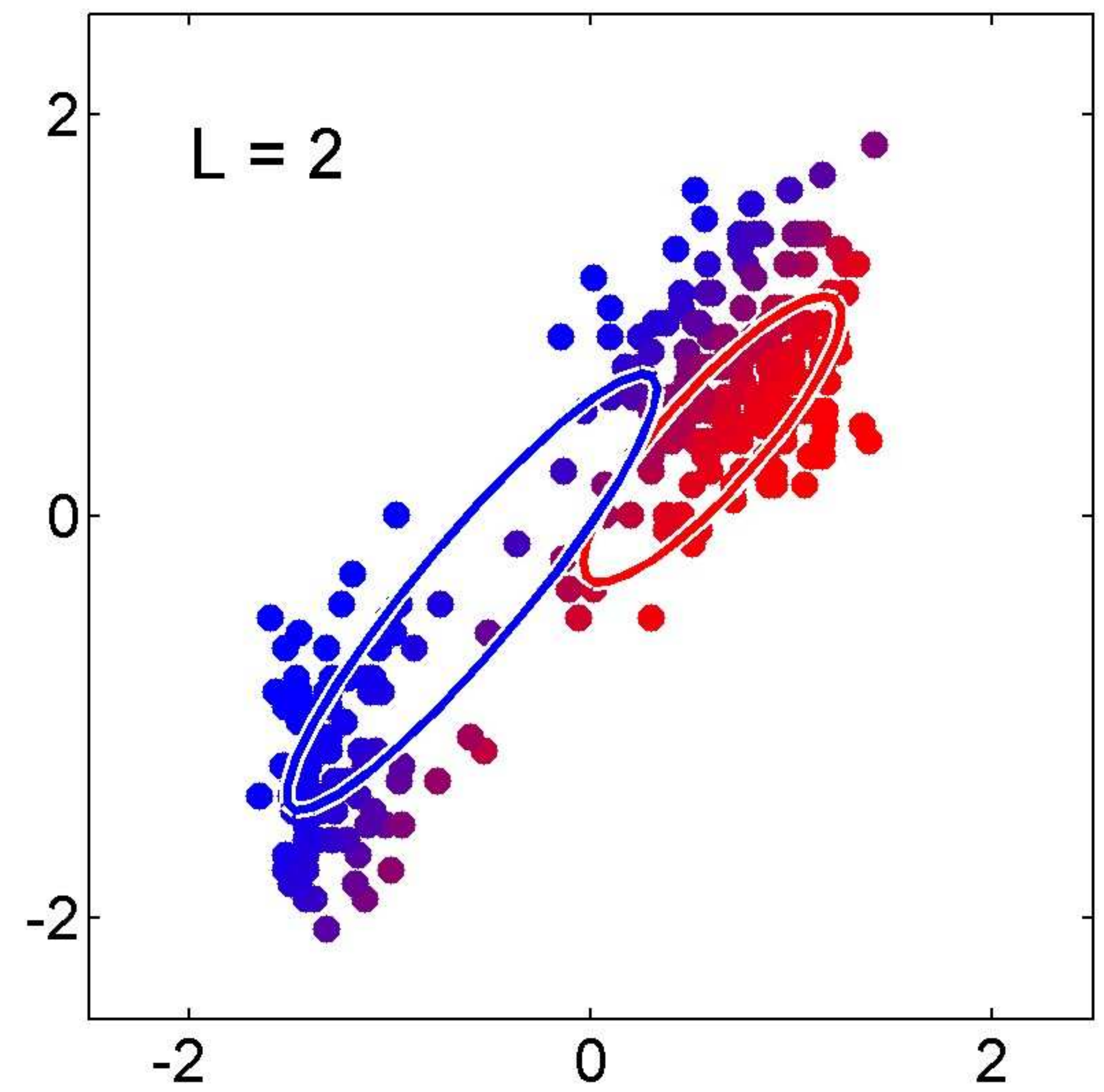
- We will use EM to obtain an algorithm for estimating the parameters
- Procedure: initialize parameters to some initial guess/random
- Alternate between:
  - E-step: fix parameter, approximate p(h l x, theta)
  - M-step: fix p(h l x, theta) obtaining maximum likelihood parameters for means, covariances and weights on each distribution
- Each cycle guaranteed not to decrease likelihood, converge to a local minimum

Simulation of EM for mixtures

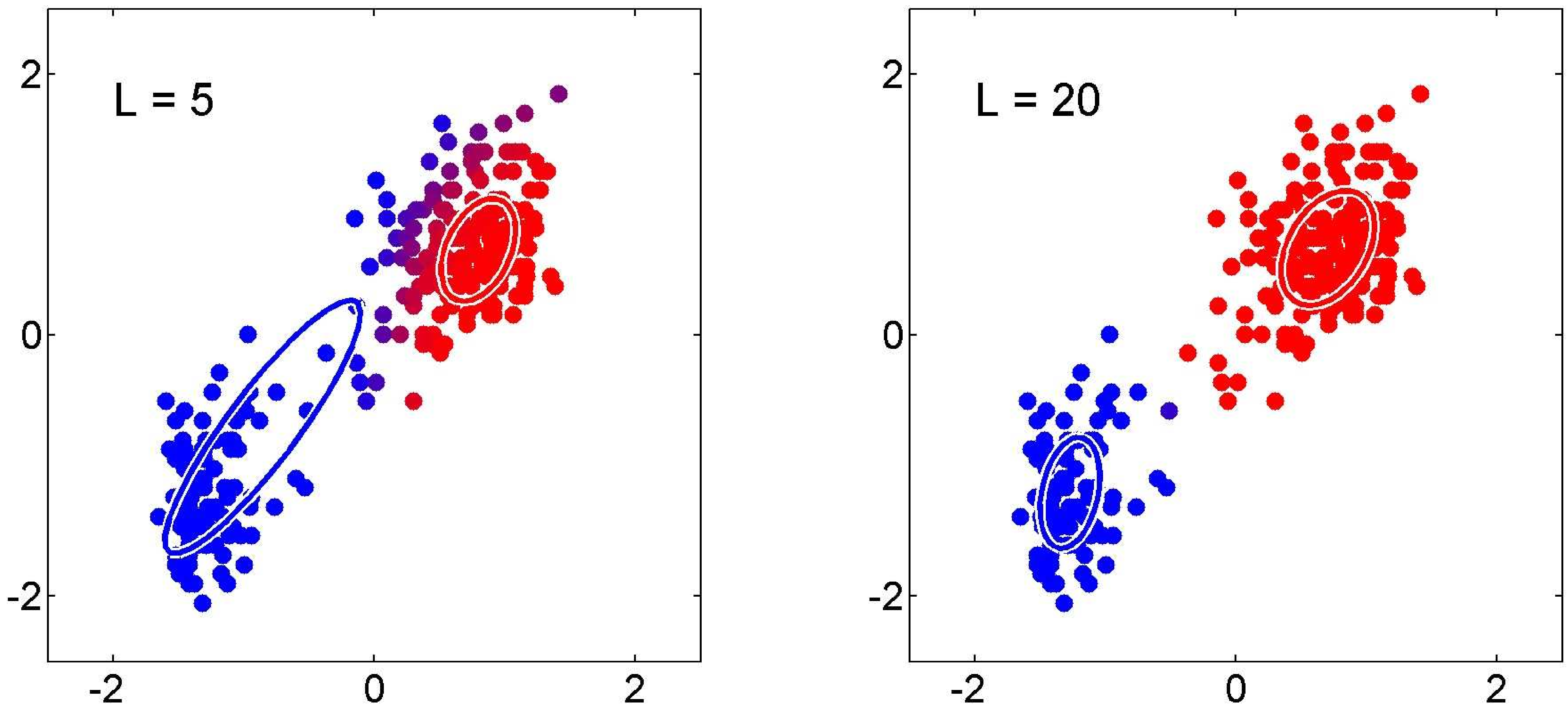


Simulation of EM for mixtures





Simulation of EM for mixtures



Mixture models

- In general, we can take mixtures of other types of distributions
- Example: mixture of exponential distributions
- The algorithm itself will be different, as start with different distributions and then obtain different E and M steps

Demo

- Estimate parameters for the Gaussian mixture models
- Can formulate as k-means problem, learning only means and assuming fixed, unit covariances
  - using Lloyds algorithm
- Can formulate more generally as to learn means, covariances and weights
  - can learn these parameters using an EM-approach
  - EM is a general solution approach (like gradient-descent), rather than an algorithm specifically for mixture models

Motivation for summing rather than maximizing

- Could simple pick the maximum/best hidden variable, as is done for the factorizations we looked at and k-means

- Summing over values can give better performance, and is solving for the parameters to a distribution

- Depends on your assumptions/needs
  - for representations, we want the “best” representation
  - for generative models, we want to appropriately approximate the model we have specified that integrates out the variables

- In some cases, it is worth the speed of the approximate solution for estimating distributions (hard EM or viterbi EM)

Whiteboard

- Expectation-maximization for mixture models
- Expectation-maximization for probabilistic PCA