機械学習特論

~ 理論とアルゴリズム ~

(Clustering: k-means and EM)

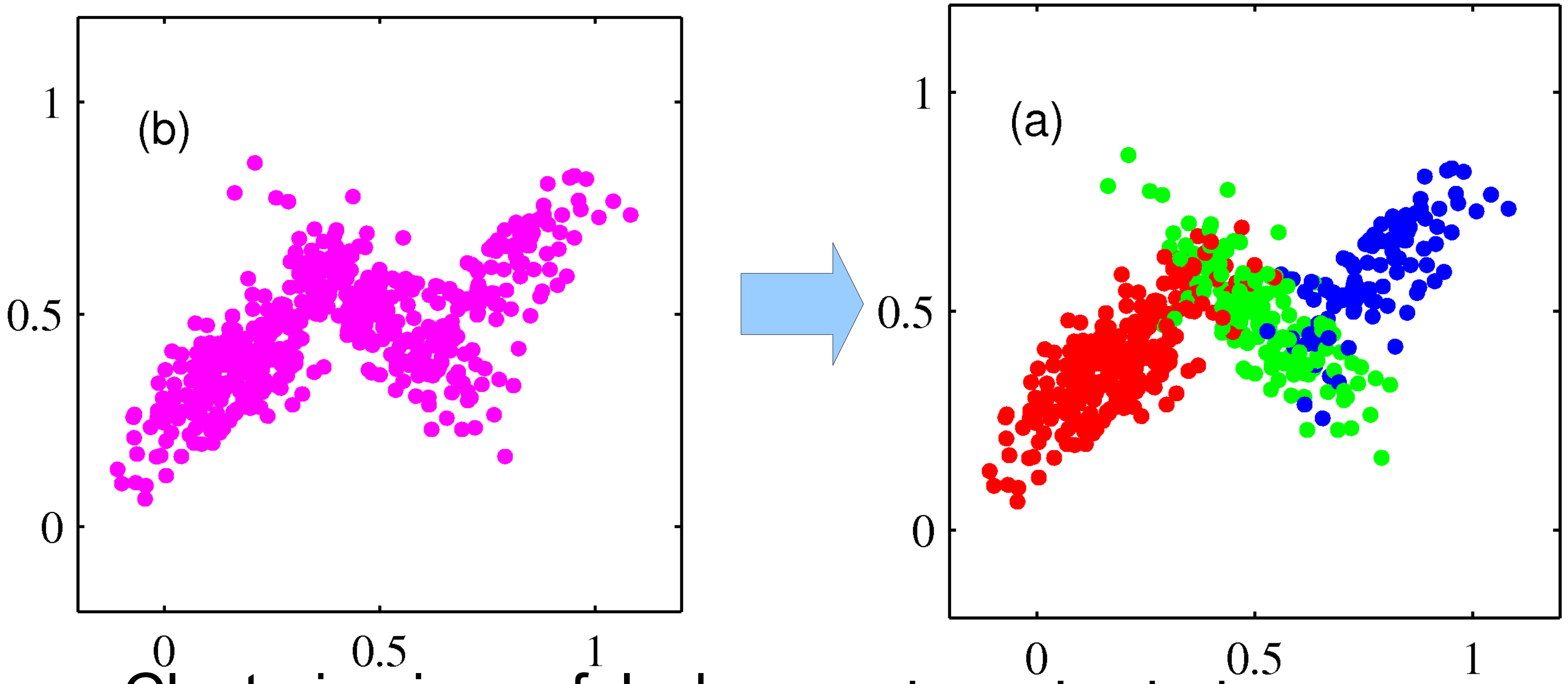
講師：西郷浩人

Supervised and Unsupervised Learning

- Supervised Learning(教師あり学習)
  - Classification(分類)
    - Given observed data(explanatory variables) $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ predict a category $\boldsymbol{y} \in \mathbb{N}^n$
  - Regression(回帰)
    - Given observed data $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ predict real-valued response $\boldsymbol{y} \in \mathbb{R}^n$
- Unsupervised Learning(教師なし学習)
  - Clustering
    - Given observed data $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ generate categories and classify datum into one of them

Clustering(クラスタリング)

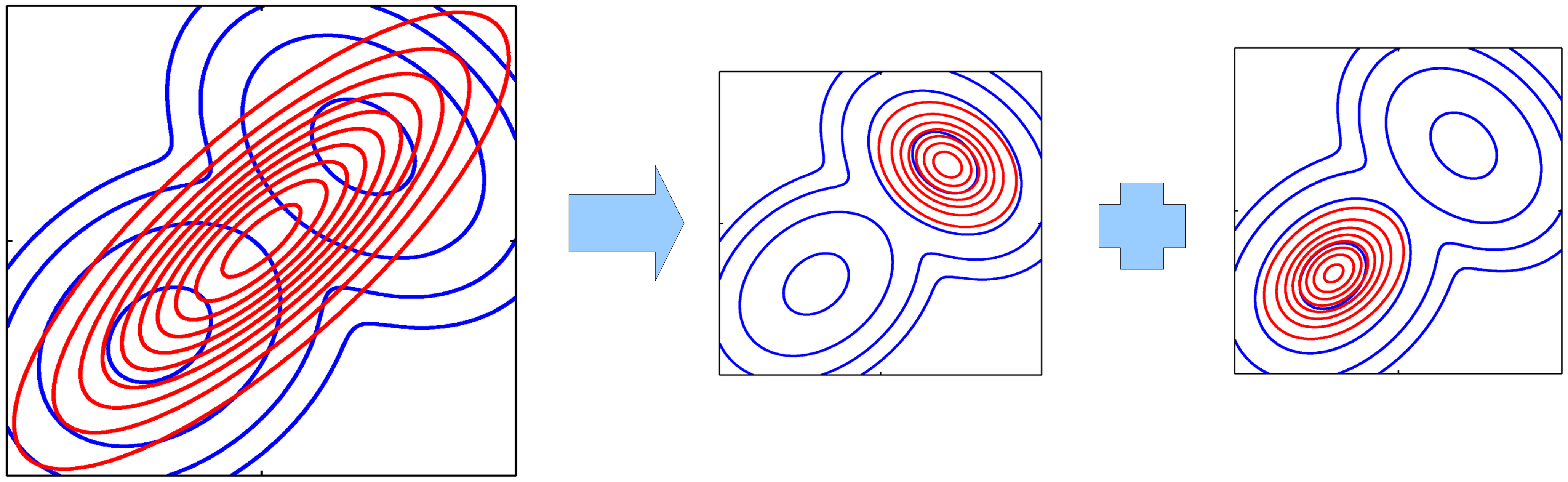
- The goal is to classify data points into categories. In this case, into 3 categories.



- Clustering is useful when we have limited knowledge about the data. (e.g., we don't even know the names of fish.)

Mixture of Gaussians

- Superimposing several gaussian(normal) distribution enables us to represent various types of distributions.
- Below we assume that our data consist of mixture of gaussians (especially in the EM case)



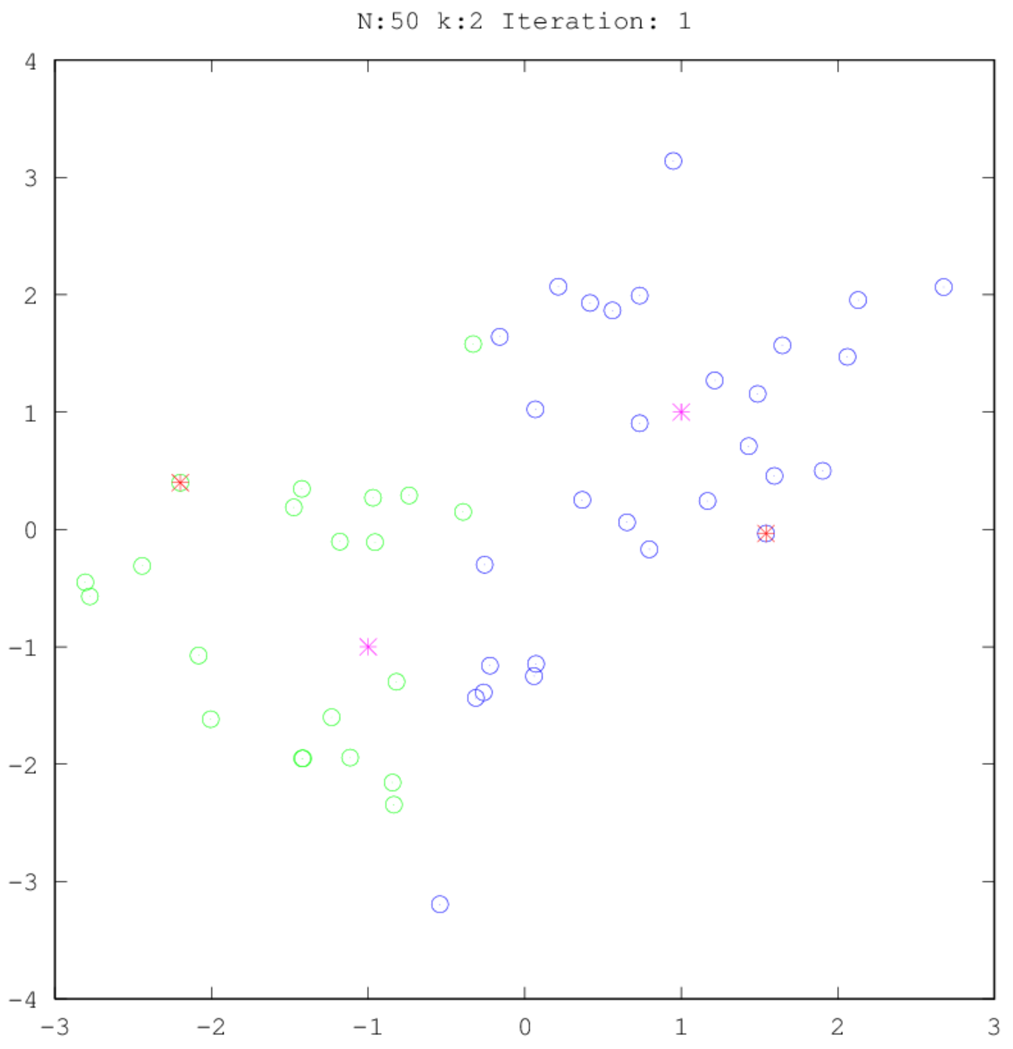
Clustering algorithms

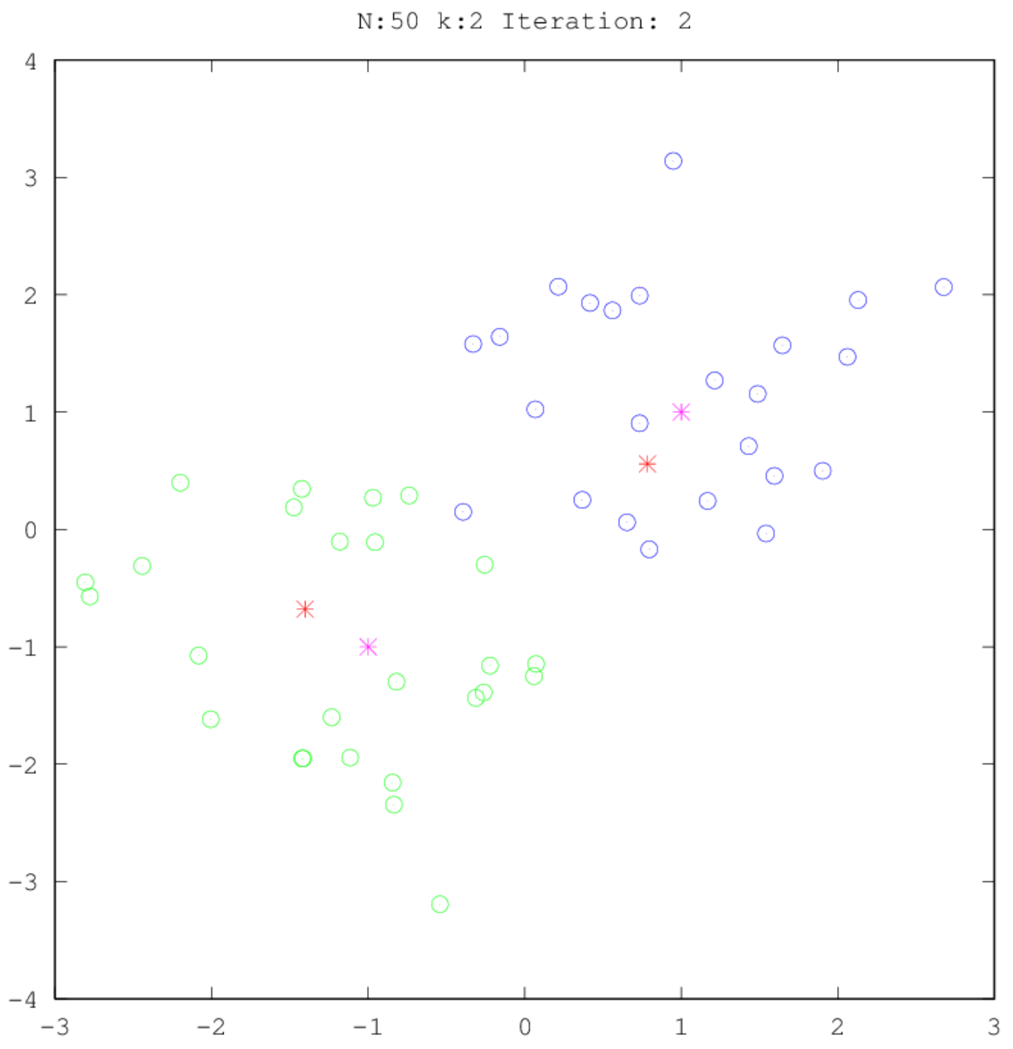
- k-means
  - Simple
  - Often gives us satisfactory performance
  - Only estimates means of clusters

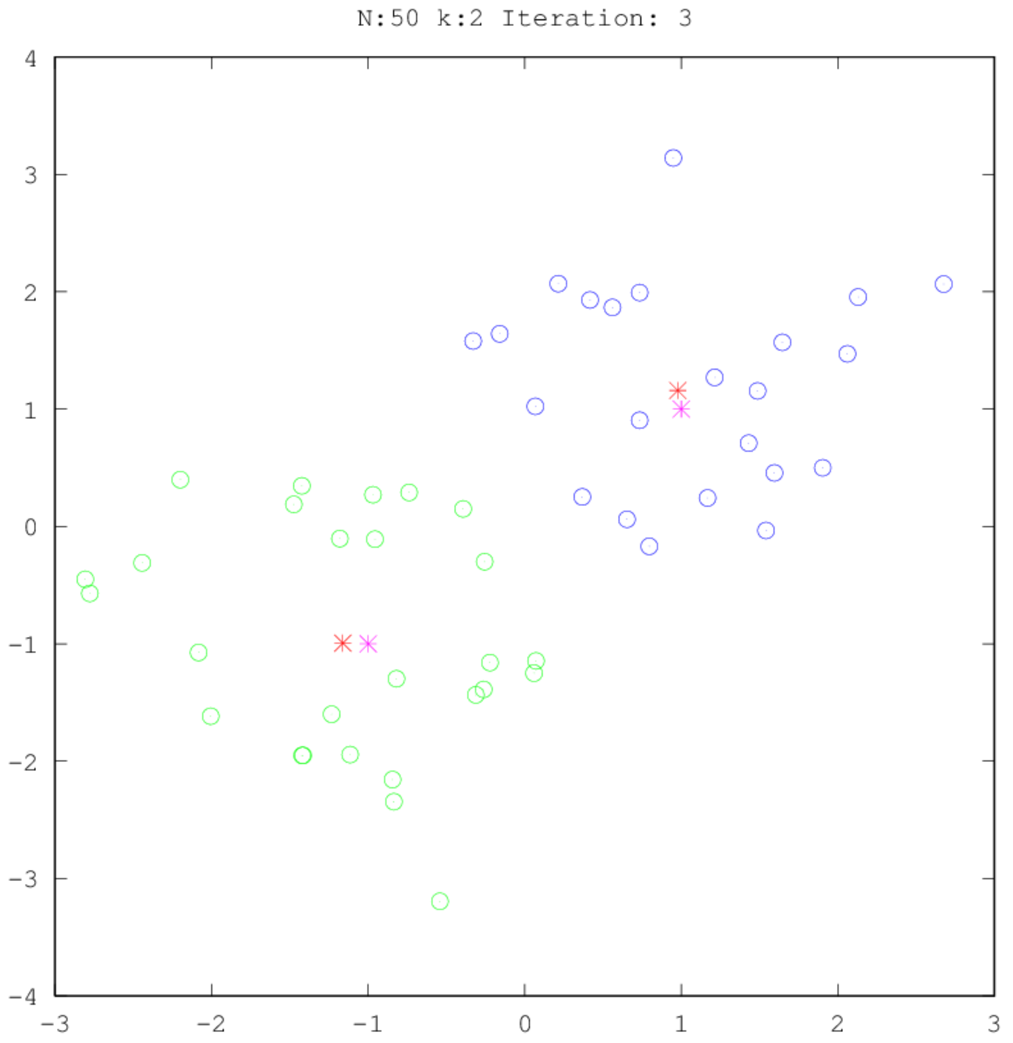
- Expectation Maximization
  - More complicated
  - Estimates means and covariances of clusters

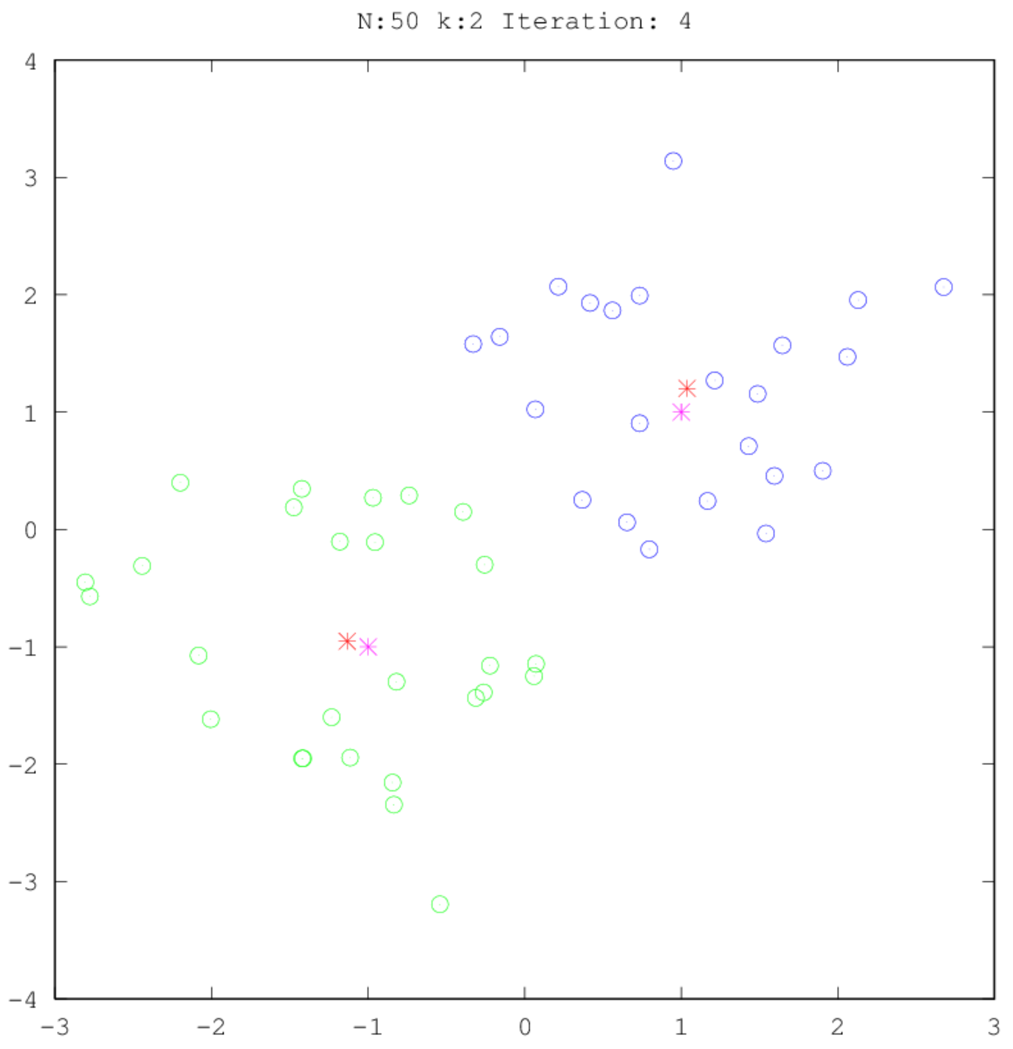
k-means (k- 平均法)

- Algorithm k-means
  - Input: data points (n), number of clusters (k)
  - Outputs: assingnment of data points to each cluster
  - Initialize:
    - set cluster centers randomly
  - Iteration:
    - assign each data point to the nearest cluster centers
    - renew cluster centers by taking mean of data points.


N:50 k:2 Iteration: 1
4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3


N:50 k:2 Iteration: 2
4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3


N:50 k:2 Iteration: 3
4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3


N:50 k:2 Iteration: 4
4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3

Ex. 1

- Download today's data, and observe the behavior of k-means algorithm.
- Inputs
  - X: data (explanatory variables)
  - k: number of clusters
- Observe the behavior of the algorithm by changing k.
  - Usage: [estimated_mu] = kmeans(X,k)

About the data

- X (50 x 2 matrix)
  - The former 25 data points are generated from gaussian distribution with mean (1,1)
  - The latter 25 data points are generated from gaussian distribution with mean (-1,-1)
- Generated by gen2Ddata.m

gen2Ddata.m

```
function [X] = gen2Ddata(N,meanX,meanY)
% generate 2D data from gaussian disturibution
% Inputs
% N: number of data points
% meanX: 2-dimensional vector specifying two x coordinates
% meanY: 2-dimensional vector specifying two y coordinates
%
% example: meanX=[-1 1], meanY=[-1 1], X=gen2Ddata(50,meanX,meanY);

var = 1;
mid = floor(N/2);
for i=1:mid
  x(i)=normrnd(meanX(1),var);
  y(i)=normrnd(meanY(1),var);
end
for i=mid+1:N
  x(i)=normrnd(meanX(2),var);
  y(i)=normrnd(meanY(2),var);
end
X=[x' y'];
```

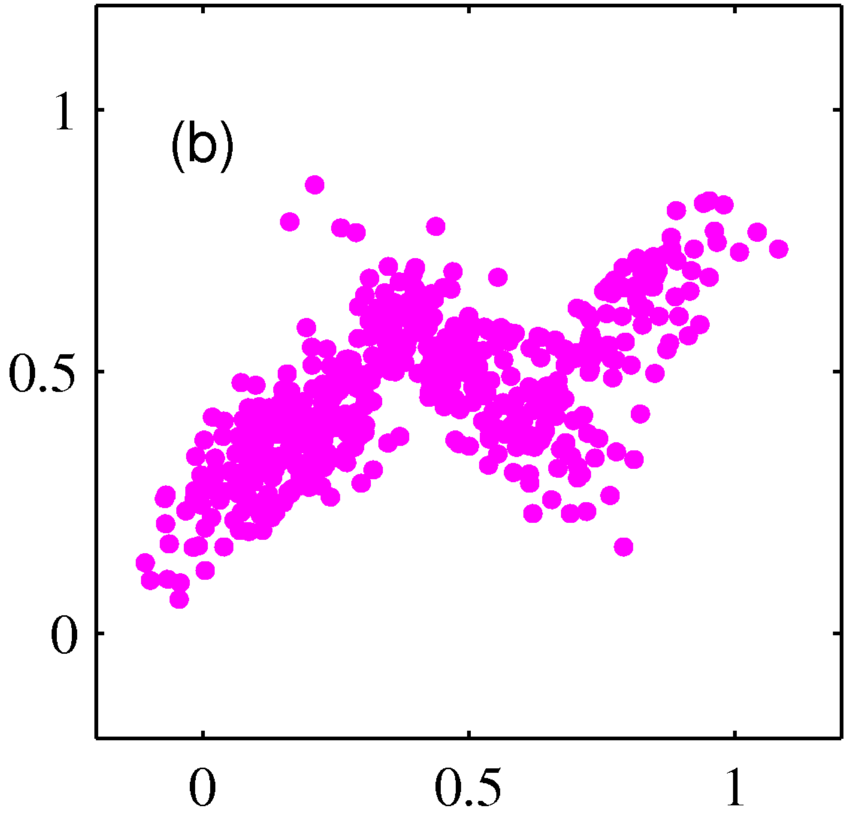
Note: Your enviranment may not have function 'nornrnd'. If so, you can search for 'normrnd.m' on the web, and download it to your current directory

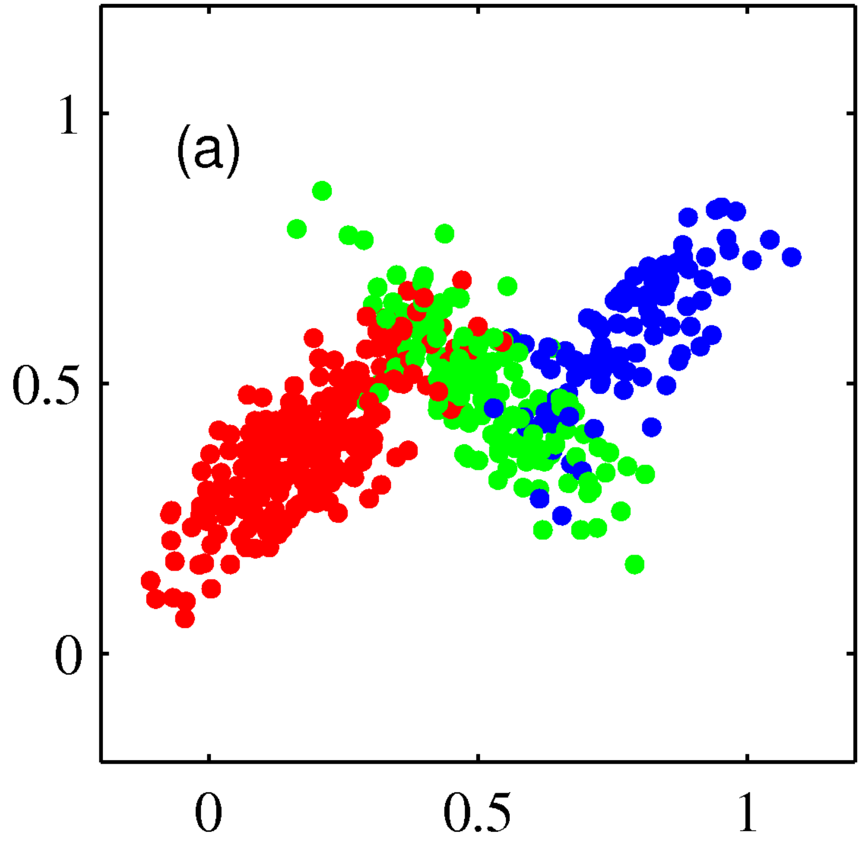
Ex1 continued.

- Algorithm chooses cluster centers randomly, therefore the results you obtained can be different each time even though the input does not change.
- What do you observe by increasing the number of clusters to 2, 3 ?

K-means as *hard* clustering

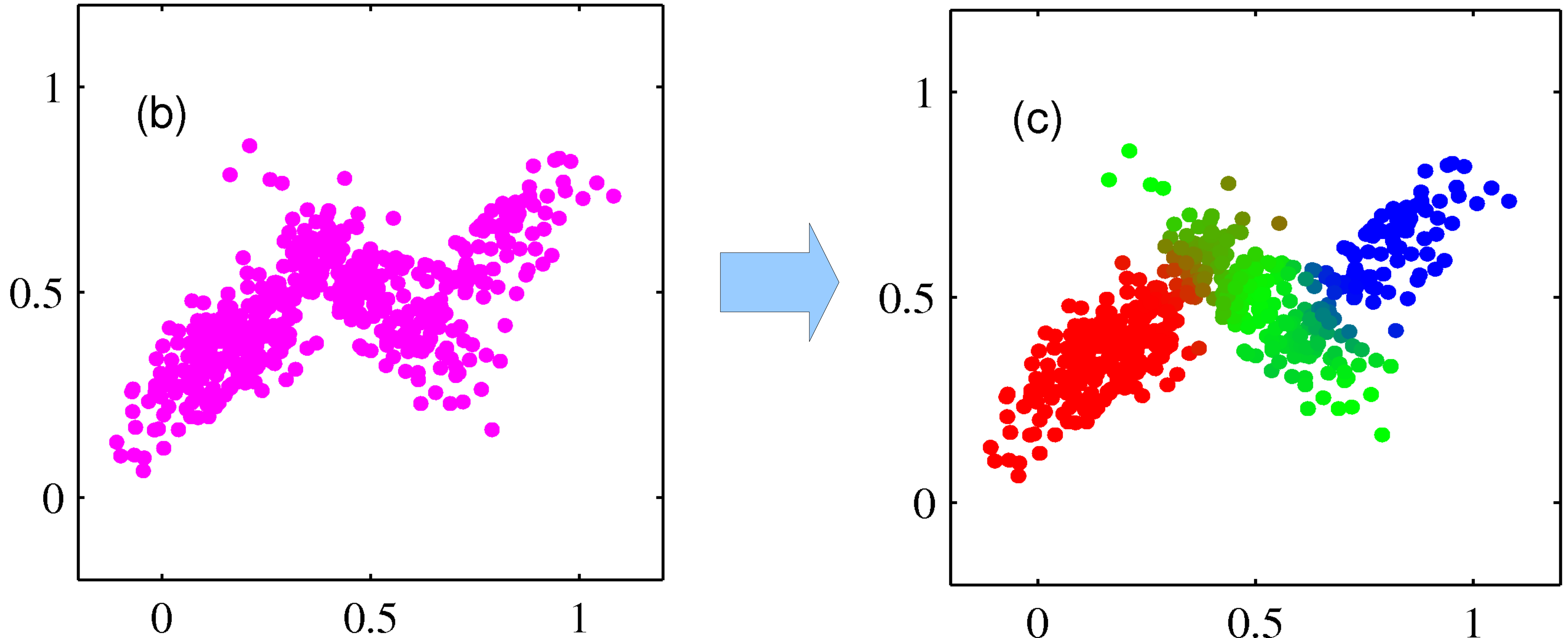
- Each data point is either assigned to cluster(s) or not.
- In this sense, k-means is a hard clustering algorithm.





Soft clustering

- Now we allow each data point to be shared by multiple clusters with arbitrary mixing ratio, then membership probability can take either [1, 0, 0], [0.5, 0.5, 0], [0.3, 0.3, 0.3] etc.
- Notice the data points in between red and green (or green and blue) in the figure below.



Preparation

- Let us define mixture weights (prior probability) as propotion of each clusters that sum to one:

$$\sum_{k=1}^{K} \alpha_k = 1$$

- Then probability of each data point is weighted sum of probabilities

$$p(\boldsymbol{x_i}) = \sum_{k=1}^{K} \alpha_k \, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$$

where

$$p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma_y}|^{1/2}} \exp\left(\frac{-1}{2} (\boldsymbol{x_i} - \boldsymbol{\mu_k})' \boldsymbol{\Sigma^{-1}} (\boldsymbol{x_i} - \boldsymbol{\mu_k}) \right)$$

Posterior probability

- Then posterior probability of a point belonging to class k is determined by Bayes' rule.
  - Mixture weights alpha can be thought as priors.

$$w_{i,k} = p(C=k \mid \boldsymbol{x_i}, \boldsymbol{\mu_k} \boldsymbol{\Sigma_k}) = \frac{\alpha_k \, p_k(\boldsymbol{x_i} \mid \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{k=1}^{K} \alpha_k \, p_k(\boldsymbol{x_i} \mid \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}$$

Bayes' rule

$$\boxed{p(y \mid x) = \frac{p(y)\, p(x \mid y)}{p(x)} = \frac{p(y)\, p(x \mid y)}{\sum_y p(y)\, p(x \mid y)}}$$

Maximumizing Loglikelihood

$$
\begin{aligned}
l &= \log \prod_{i=1}^{N} p(\boldsymbol{x_i}) \\
&= \log \prod_{i=1}^{N} \sum_{k=1}^{K} \alpha_k \, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \\
&= \sum_{i=1}^{N} \log \sum_{k=1}^{K} \alpha_k \, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})
\end{aligned}
$$

- Parameters that maximize the ablve log-likelihood is obtained by solving the following set of equations.

$$\frac{\partial l}{\partial \boldsymbol{\alpha_k}} = 0 \quad \Longleftrightarrow \quad \alpha_k^{new} = \frac{1}{N} \sum_{i=1}^{N} w_{i,k}$$

$$\frac{\partial l}{\partial \boldsymbol{\mu_k}} = 0 \quad \Longleftrightarrow \quad \mu_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} \, \boldsymbol{x_i}$$

$$\frac{\partial l}{\partial \boldsymbol{\Sigma_k}} = 0 \quad \Longleftrightarrow \quad \Sigma_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} (\boldsymbol{x_i} - \mu_k^{new})(\boldsymbol{x_i} - \mu_k^{new})'$$

Wait a moment..

$$\alpha_k^{new} = \frac{1}{N} \sum_{i=1}^{N} w_{i,k}$$

$$\mu_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} \boldsymbol{x_i}$$

$$\Sigma_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} (\boldsymbol{x_i} - \mu_k^{new})(\boldsymbol{x_i} - \mu_k^{new})'$$

- Solving the above equations is not trivial, since they all depend on unknown posterior probability $w_{i,k}$, which recursively depend on $\alpha_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$

$$w_{i,k} = \frac{\alpha_k p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{k=1}^{K} \alpha_k p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}$$

EM (Expectation Maximization) algorithm for gaussian mixtures

$$w_{i,k} = \frac{\alpha_k \, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{k=1}^{K} \alpha_k \, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}$$

$$\alpha_k^{new} = \frac{1}{N} \sum_{i=1}^{N} w_{i,k}$$

$$\mu_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} \, \boldsymbol{x_i} \longleftarrow$$

Weighted mean vector (p x 1)

$$\Sigma_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} (\boldsymbol{x_i} - \mu_k^{new})(\boldsymbol{x_i} - \mu_k^{new})' \longleftarrow$$

Weighted covariance matrix (p x p)

- So we initialize $\alpha_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$ by some arbitrary values, and compute $w_{i,k}$
- Then recompute $\alpha_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$, and iterates...

EM (Expectation Maximization) algorithm for gaussian mixtures

E-step

$$w_{i,k} = \frac{\alpha_k\, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}{\sum_{k=1}^{K} \alpha_k\, p_k(\boldsymbol{x_i} | \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})}$$

M-step

$$\alpha_k^{new} = \frac{1}{N} \sum_{i=1}^{N} w_{i,k}$$

$$\mu_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k}\, \boldsymbol{x_i}$$

$$\Sigma_k^{new} = \left(\frac{1}{\sum_{i=1}^{N} w_{i,k}} \right) \sum_{i=1}^{N} w_{i,k} (\boldsymbol{x_i} - \mu_k^{new})(\boldsymbol{x_i} - \mu_k^{new})'$$

- In the framework of EM algorithm, computation of $\alpha_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$ is called M-step, and that of $w_{i,k}$ is called E-step.

Convergence of EM algorithm

- Convergence of the algorithm is checked by observing the change of log-likelihood.
- **EM algorihtm for gaussian mixtures only converges to its local maximum, but not to global maximum.**
- For searching global maximum..
  - Algorithm can be run again and again using different initial parameters.

Ex. 2

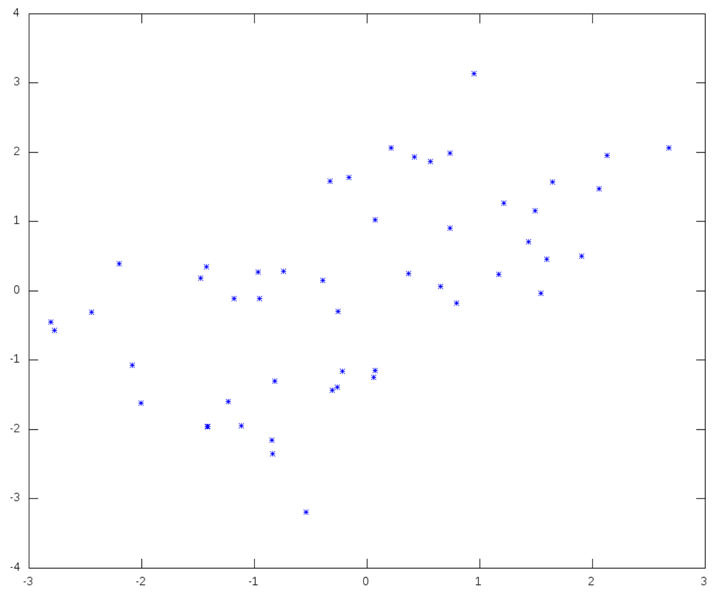
- Download "EM.m" and observe the behavior of the EM algorithm. Read the code by yourself.
- Inputs:
  - X: data
  - k: number of clusters
  - usage: [alpha,mu,sigma,w]=EM(X,k)

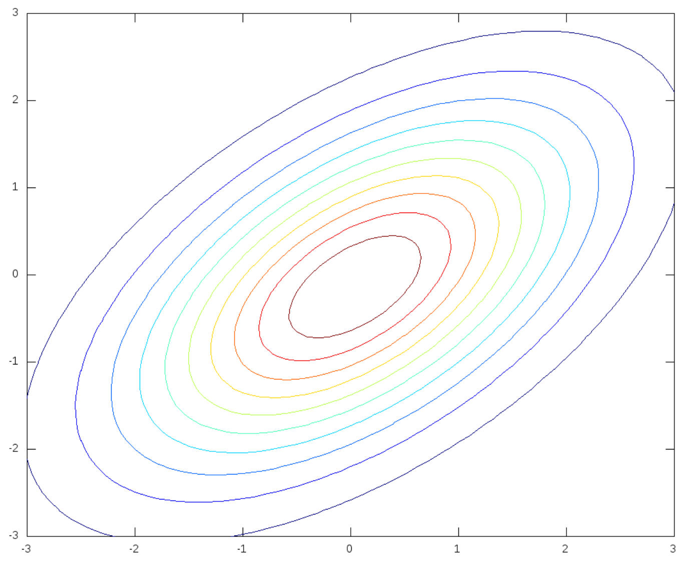
Ex. 2 continued.

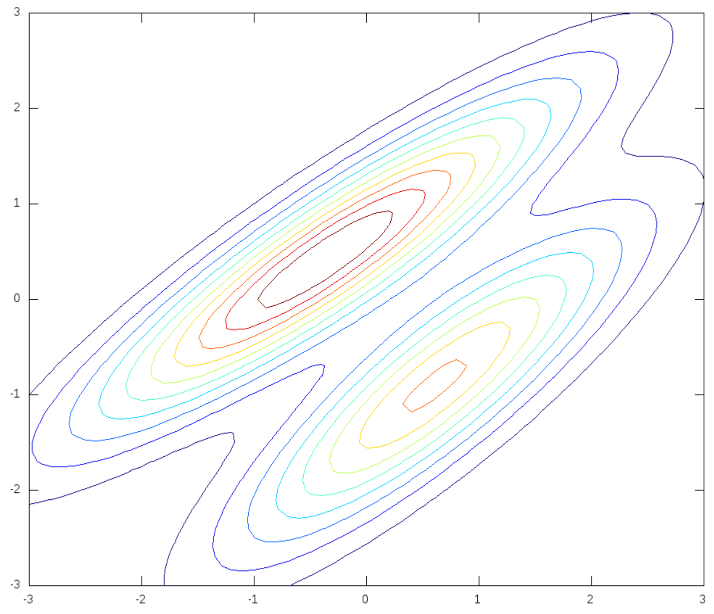
- How do you interpret the ouptuts of EM.m: alpha, mu, sigma, w. Do they look releavant ?
  - Hint: imagesc(w);
    - w contains posterior probabilites for class labels.

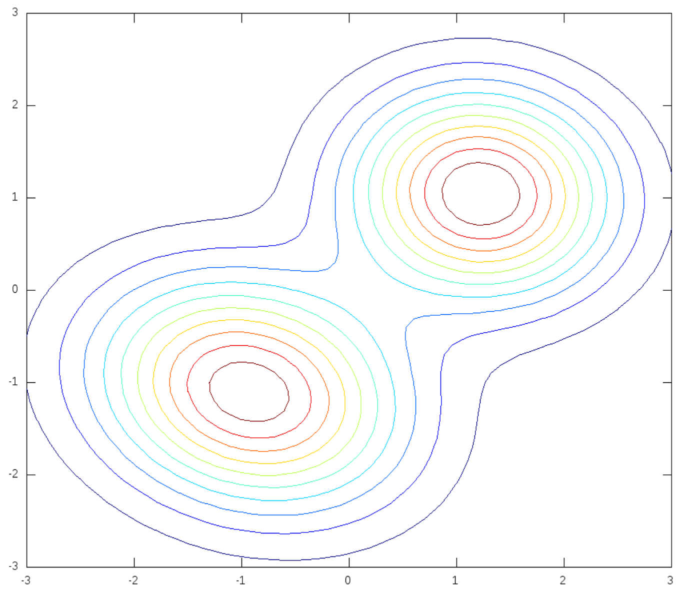
- Generate your own data and try EM algorithm on it !
  - Use of gen2Ddata.m
    - n=50, mu1=[1 1], mu2=[-1 -1]
    - [X]=gen2Ddata(n,mu1,mu2)

Results are different each time









Ex. 3

- Compare K-means and EM in terms of the following points.
  - Speed, accuracy, mean, variance
- For measuring the speed, you should turn visualizaition off by setting the third input to zero, and use 'tic' and 'toc'
  - tic, [mu]=kmeans(X,k,0), toc
  - tic, [alpha mu sigma w]=EM(X,k,0), toc

How to set number of clusters ?

- In clustering, *how to set the number of clusters* is an open problem.
- However, EM algorithm returns maximized likelihood, thus one can make use of information critera such as
  - AIC = - 2 loglikelihood + 2 p
    - where p is the number of parameters
  - BIC = - 2 loglikelihood + p log n
    - where n is the number of data points.

Appendix

Tips: Number of clusters

- On today's data, EM algorithm was run 100 times for each k, and computed average AIC and BIC. Values inside parentheses show standard deviation.

k	1	2	3	4	5
AIC	435.7(0.00)	434.4 (3.29)	435.5 (5.02)	437.4 (9.20)	443.4 (11.7)
BIC	443.4 (0.00)	451.7 (3.29)	464.2 (5.02)	479.5 (9.20)	500.8 (11.7)

- In this case AIC did a better job, but BIC underestimated the number of clusters.
- A script used for this experiment is included, and named as "checkNumCluster.m"