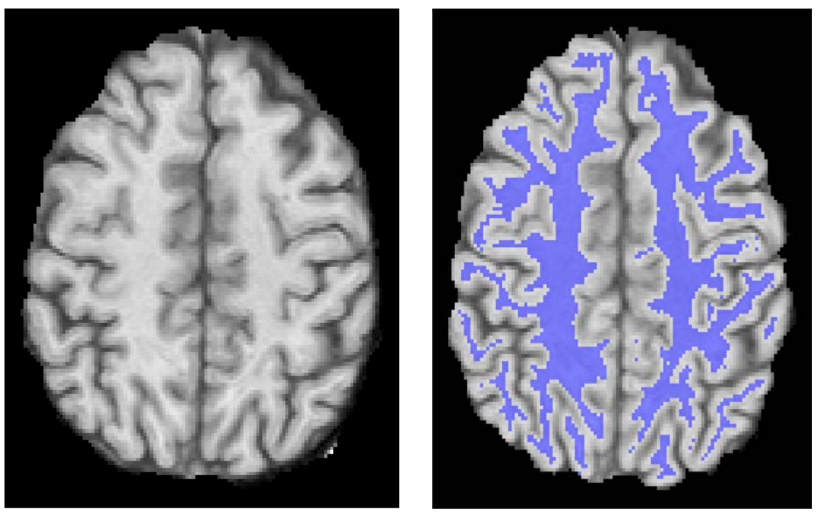
ECSE-626

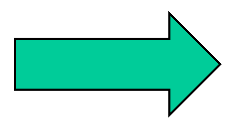
Statistical Computer Vision

Gaussian Mixture Models, EM, Kernel Density Estimation

Introduction

- We often want to estimate a probability density function from a set of observations.

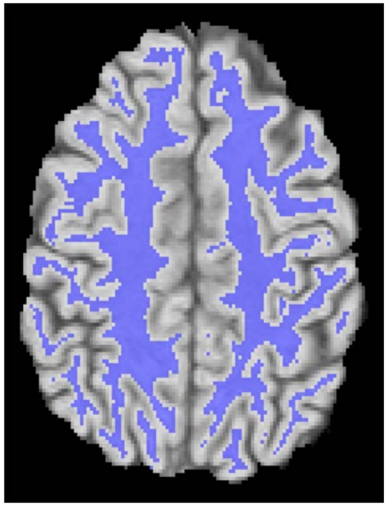


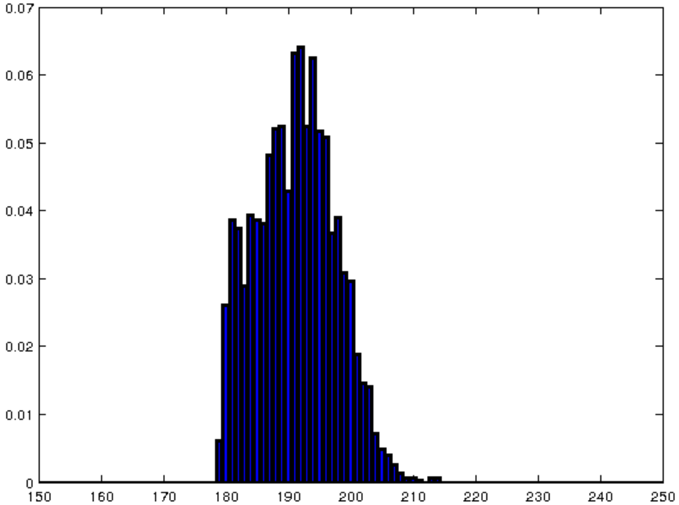


?

Modeling Densities

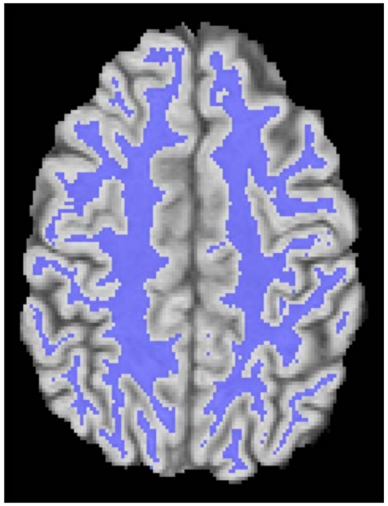
- Given a set of observations, how do we model a probability density?

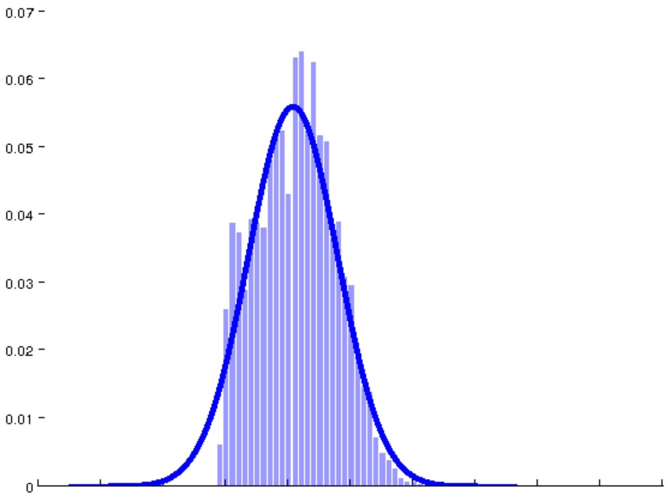




Modeling Densities

- Given a set of observations, how do we model a probability density?





- Often we model a set of samples as having a Gaussian density

Gaussian Densities

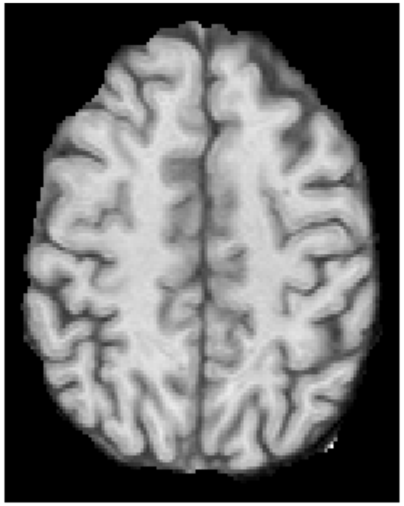
- Gaussian densities can be compactly represented by their mean and variance
- Sample mean and variance can be computed in linear time

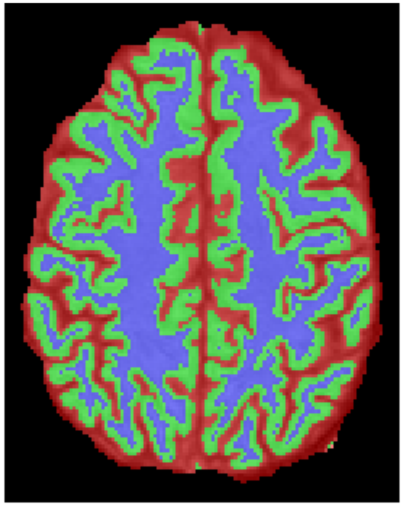
$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

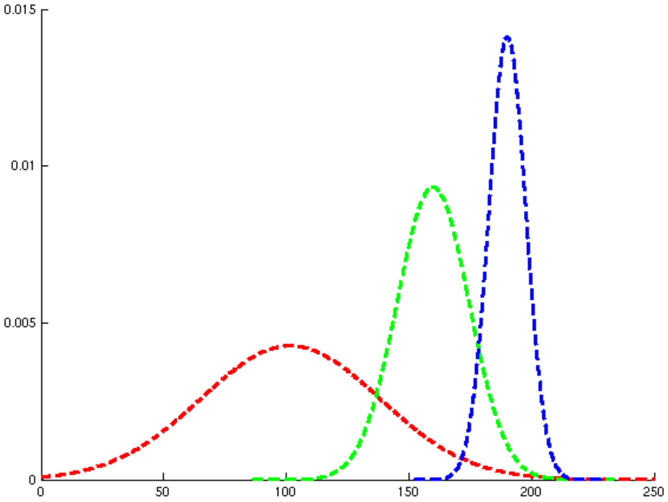
$$s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2$$

Gaussian Densities

- Gaussian densities have many convenient mathematical properties and serve as a reasonable approximation for many true densities.

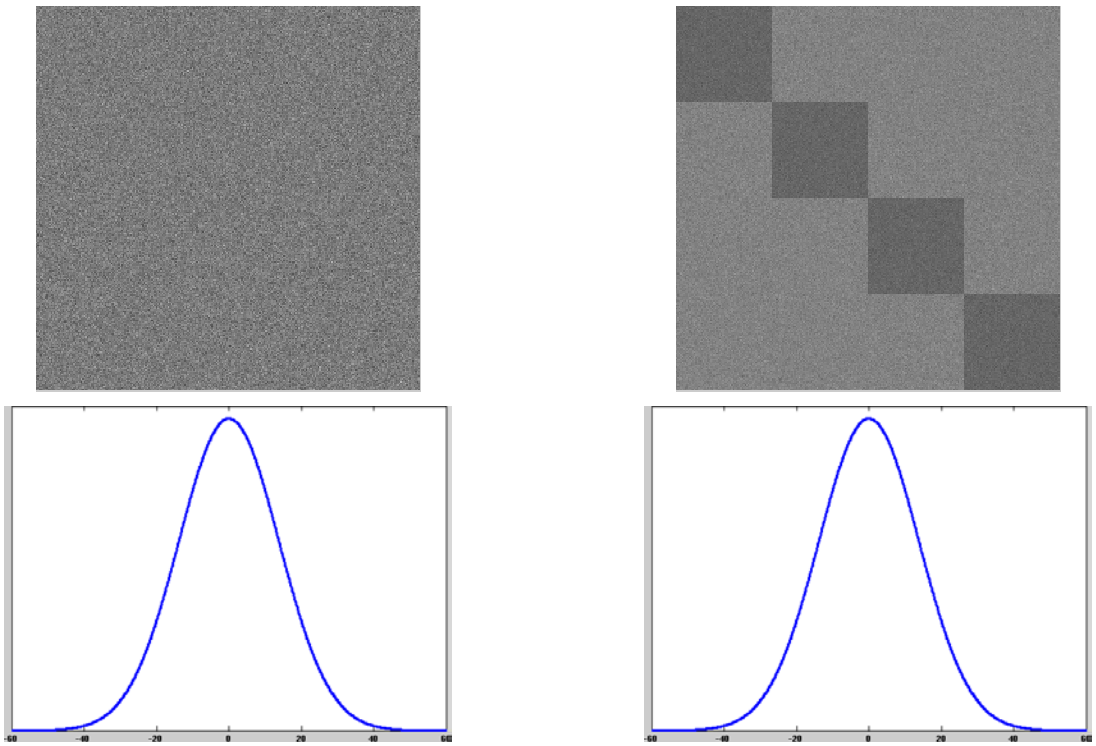






Gaussian Densities

- However, many real densities are not well modeled by Gaussians.

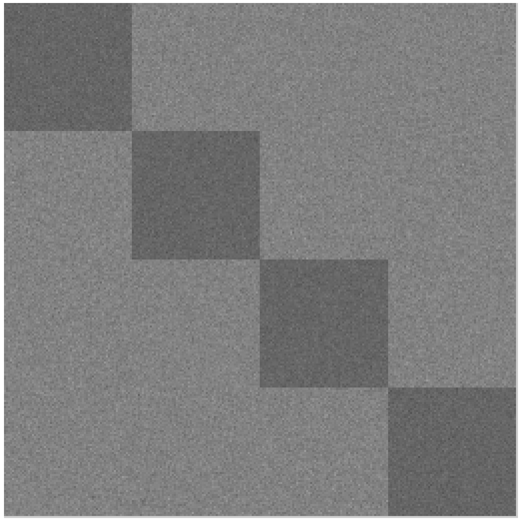


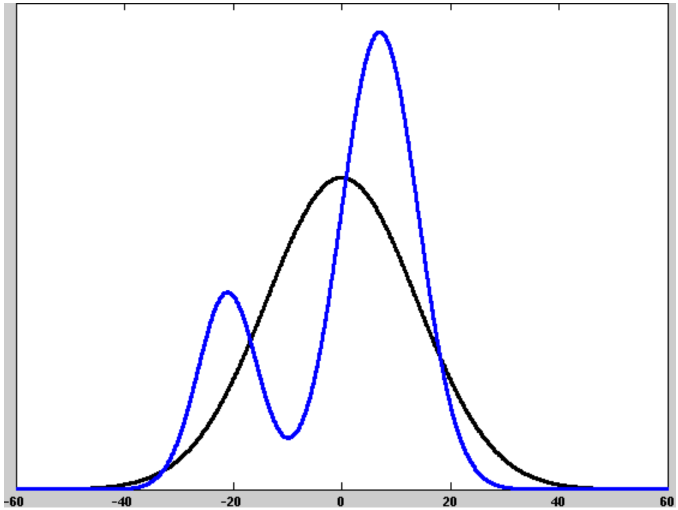
Gaussian Densities

- Parametric densities such as Gaussians, and all exponential densities, are unimodal (have a single local maximum).
- Many practical problems involve multi-modal densities.

Multi-modal Densities

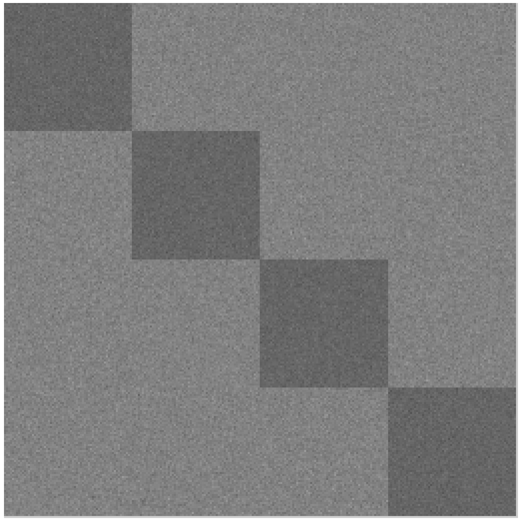
- Gaussian is poor representation of densities that have multiple modes.

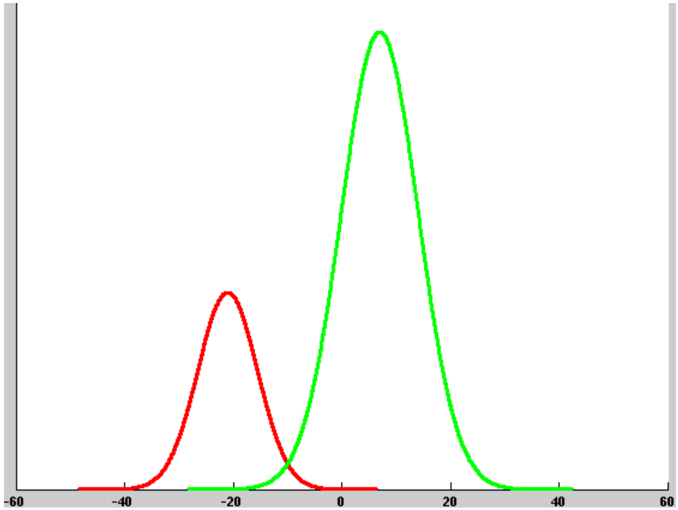




Gaussian Mixture Model (GMM)

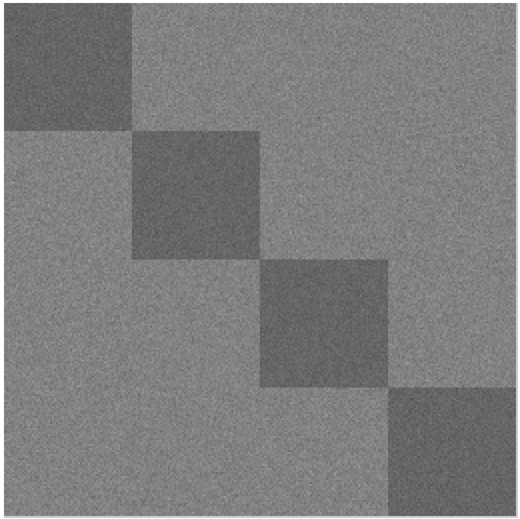
- Allow us to represent densities as weighted sum (or mixture) of multiple Gaussians

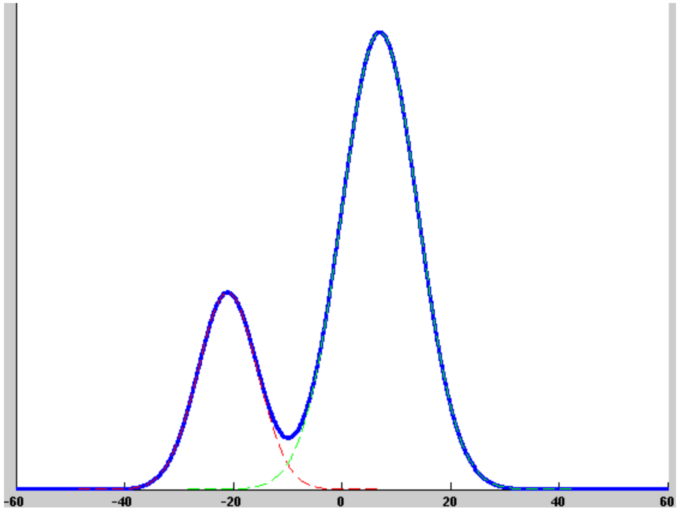




Gaussian Mixture Model (GMM)

- Allow us to represent densities as weighted sum (or mixture) of multiple Gaussians





Gaussian Mixture Models (GMM)

- Mixtures of Gaussian functions are well-suited to modeling clusters of points.
- Each cluster is assigned a Gaussian, with its mean in the middle of the cluster and with a standard deviation measuring its spread.

Gaussian Mixture Models (GMM)

K : number of gaussian components

μ_k : mean of k^{th} gaussian component

σ_k : std of k^{th} gaussian component

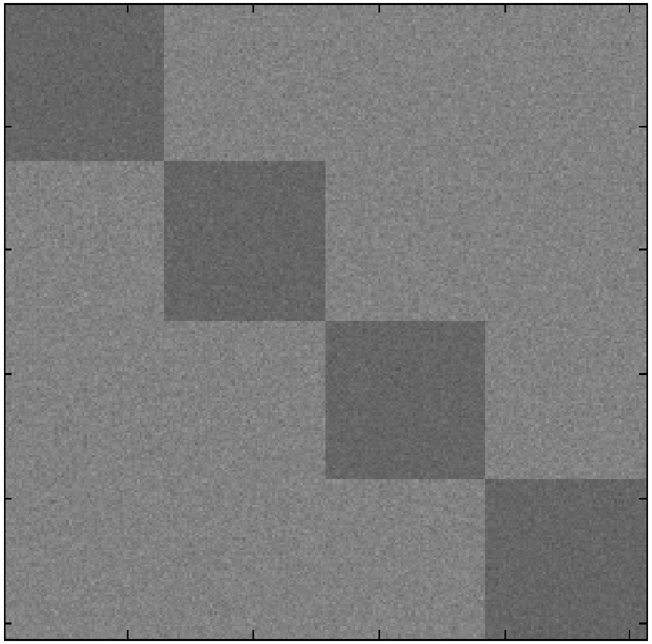
α_k : mixing probability (weight) of k^{th} gaussian component

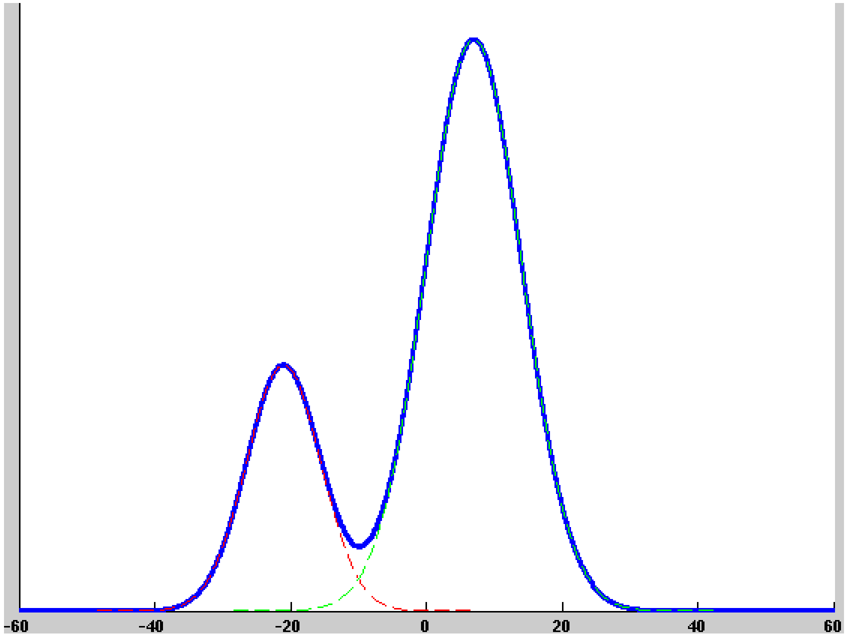
$$\boldsymbol{\Theta} = \{\alpha_1, ..., \alpha_k, \mu_1, ..., \mu_k, \sigma_1, ..., \sigma_k\}$$

$$f(x|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(x; \mu_k, \sigma_k)$$

Gaussian Mixture Models

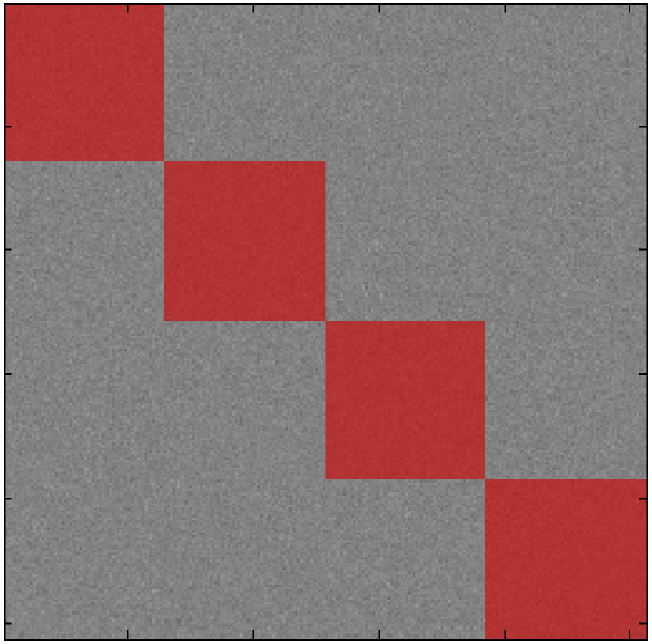
$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$

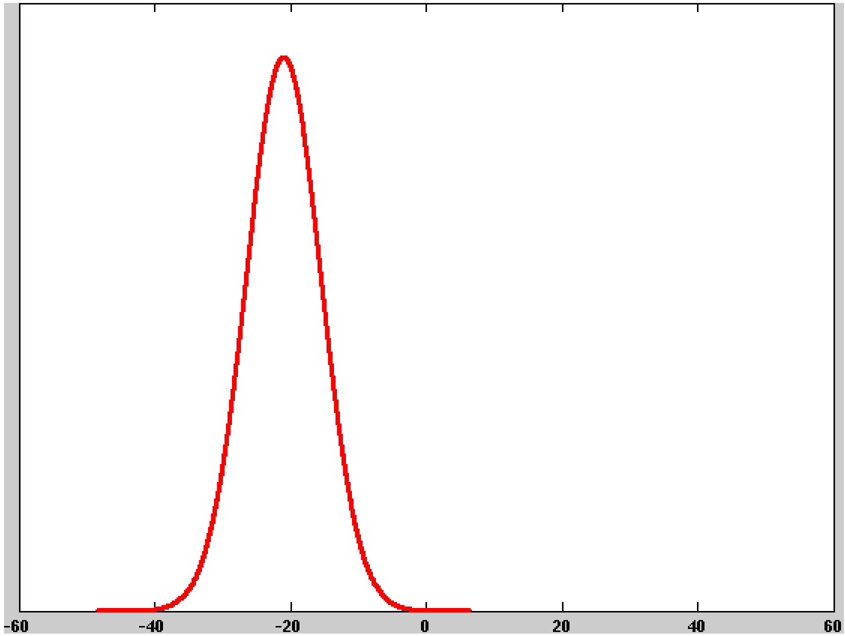




Gaussian Mixture Models

$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$

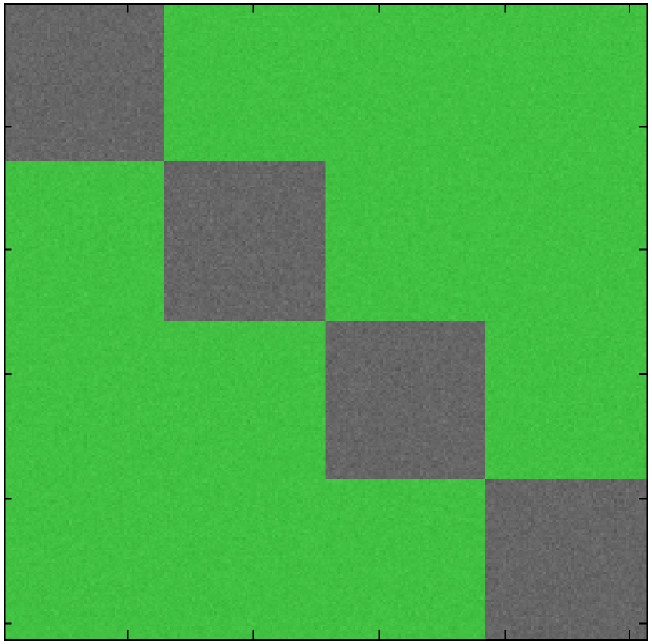


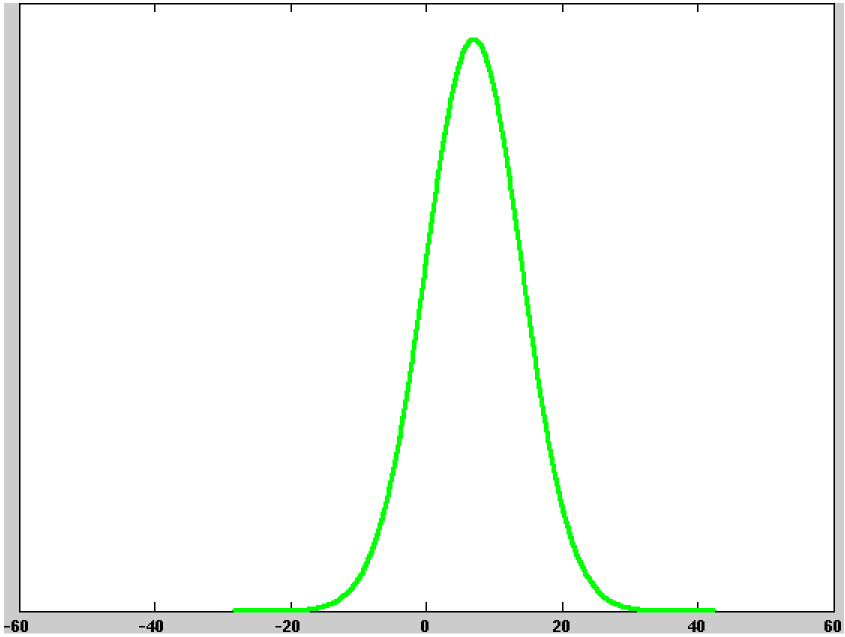


k = 1

Gaussian Mixture Models

$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$

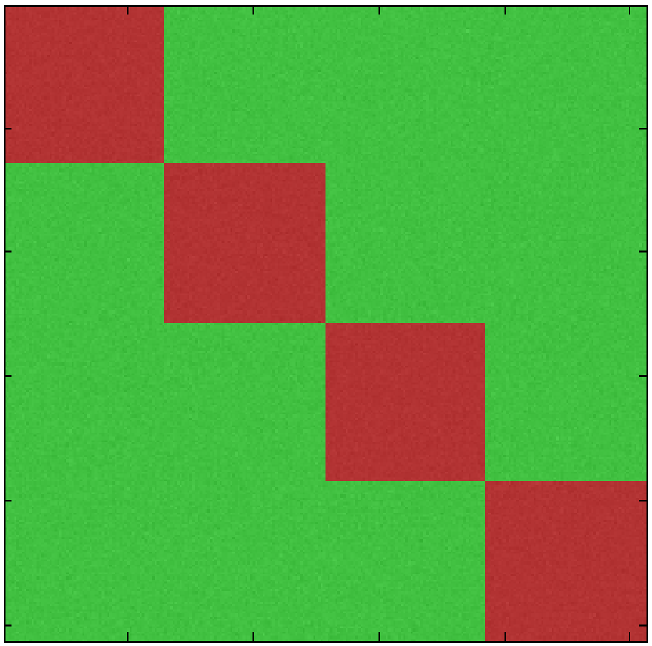


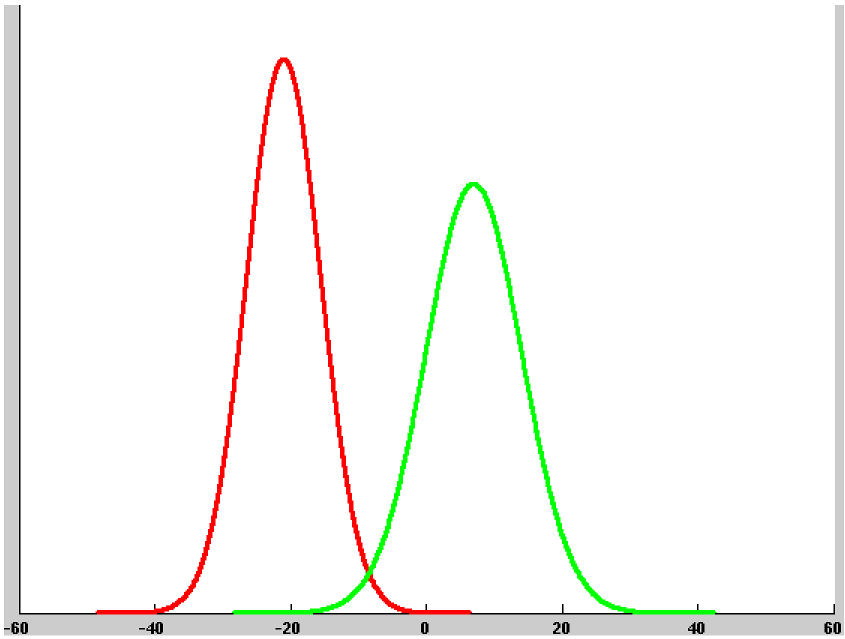


k = 2

Gaussian Mixture Models

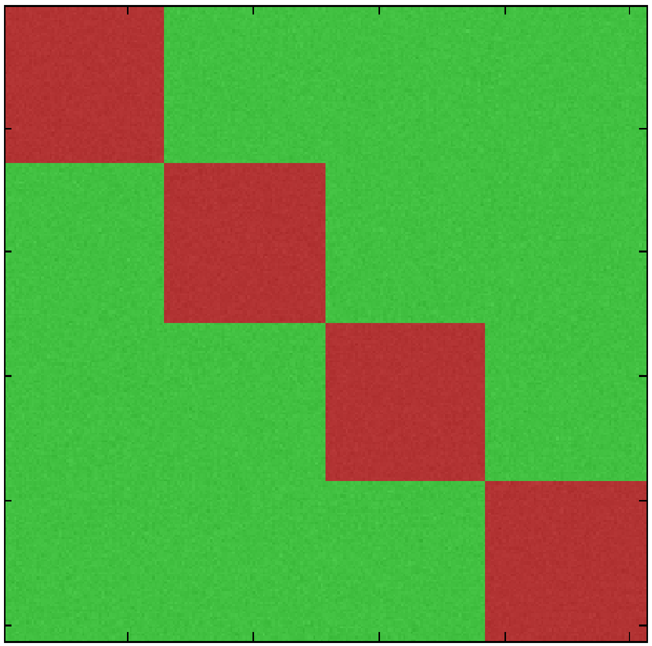
$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$

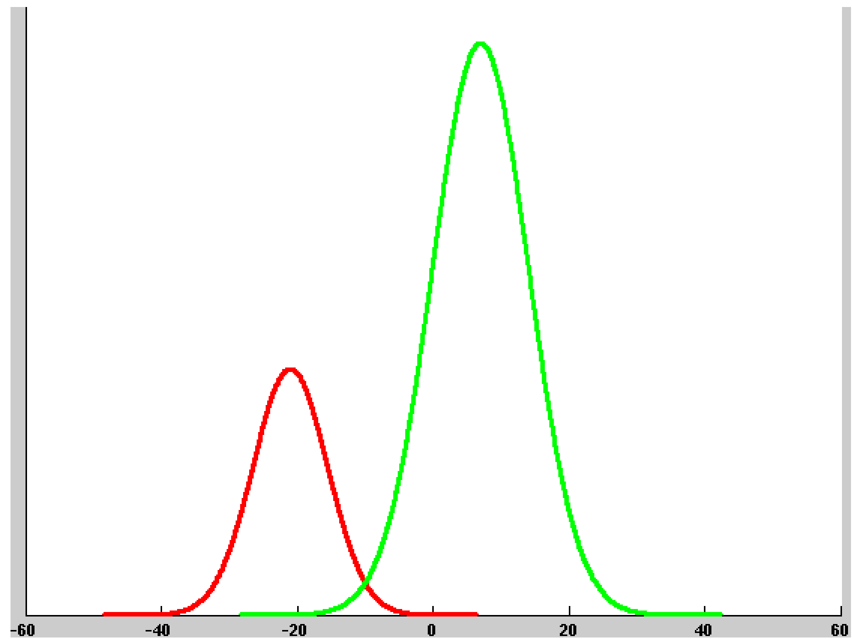




Gaussian Mixture Models

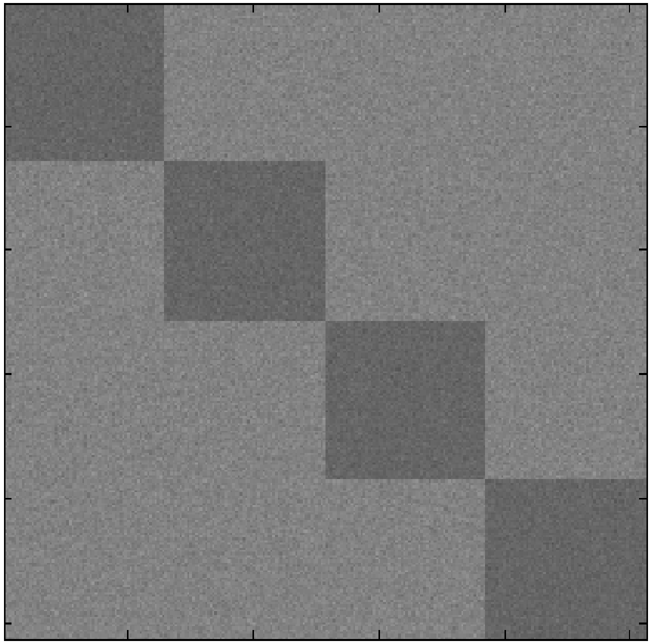
$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$

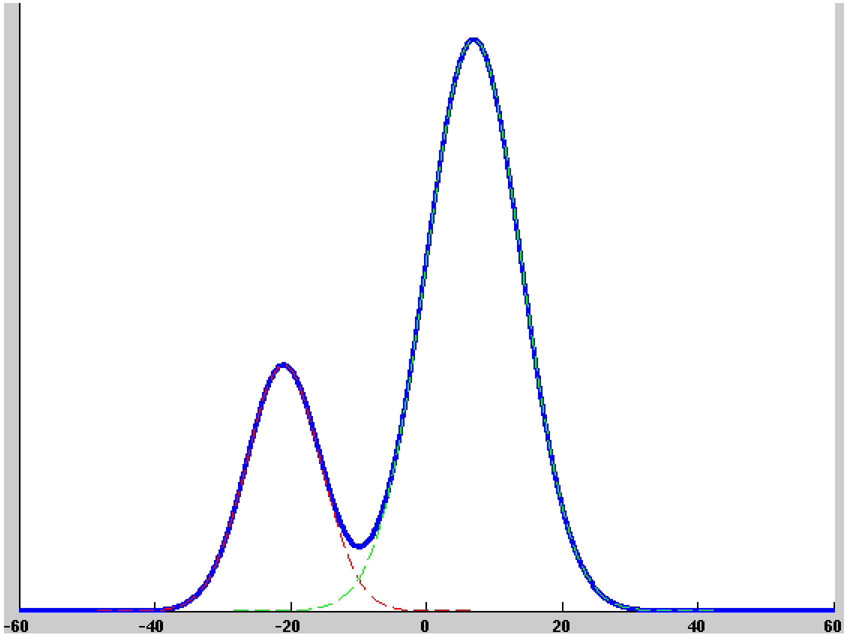




Gaussian Mixture Models

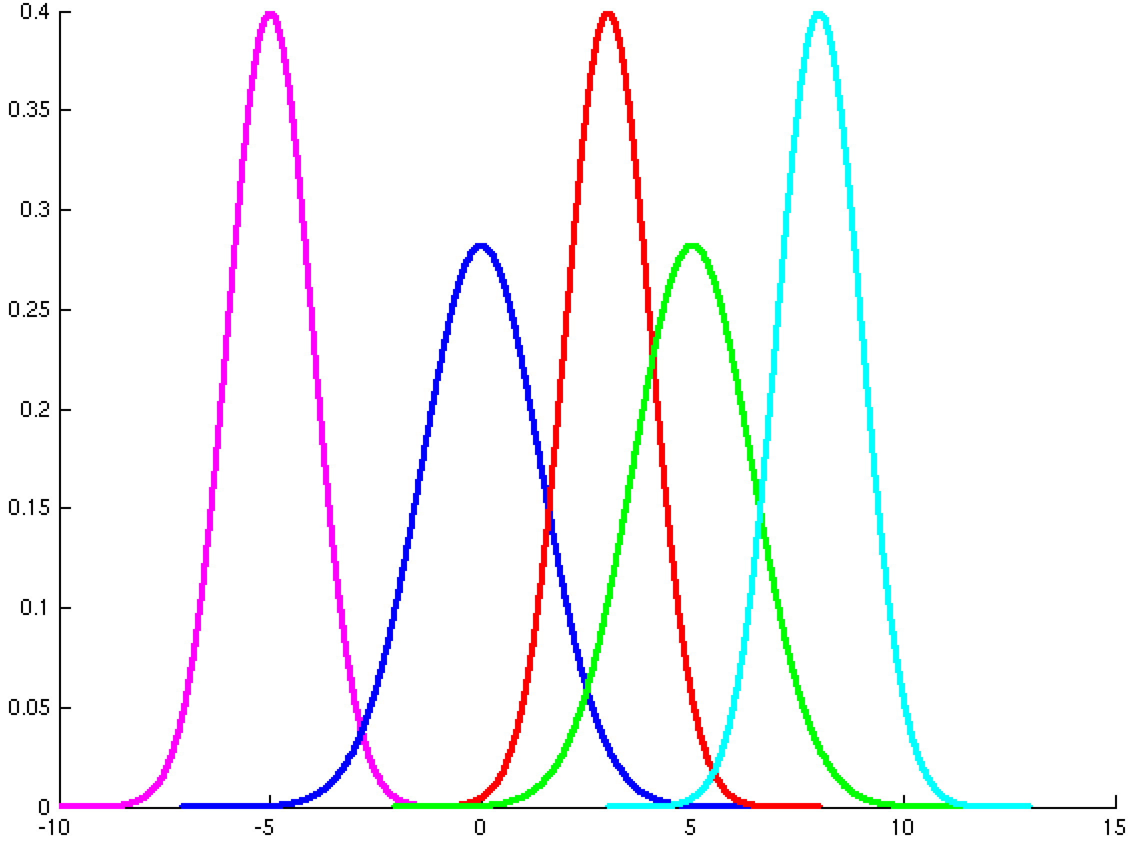
$$K = 2,\ \mu = \{-21, 7.00\},\ \sigma = \{30, 50\},\ \alpha = \{0.25, 0.75\}$$





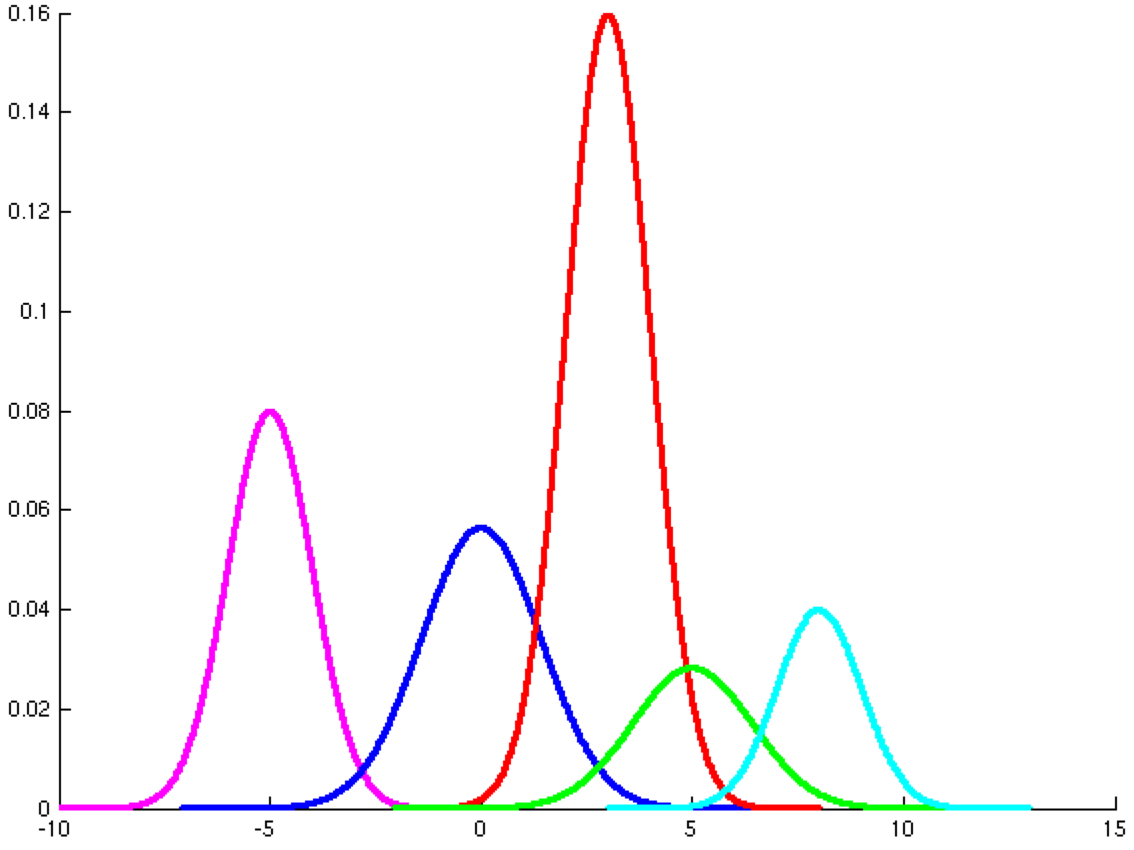
Gaussian Mixture Models

$K = 5,\ \mu = \{-5, 0, 3, 5, 8\},\ \sigma = \{1, 2, 1, 2, 1\},\ \alpha = \{0.2, 0.2, 0.4, 0.1, 0.1\}$



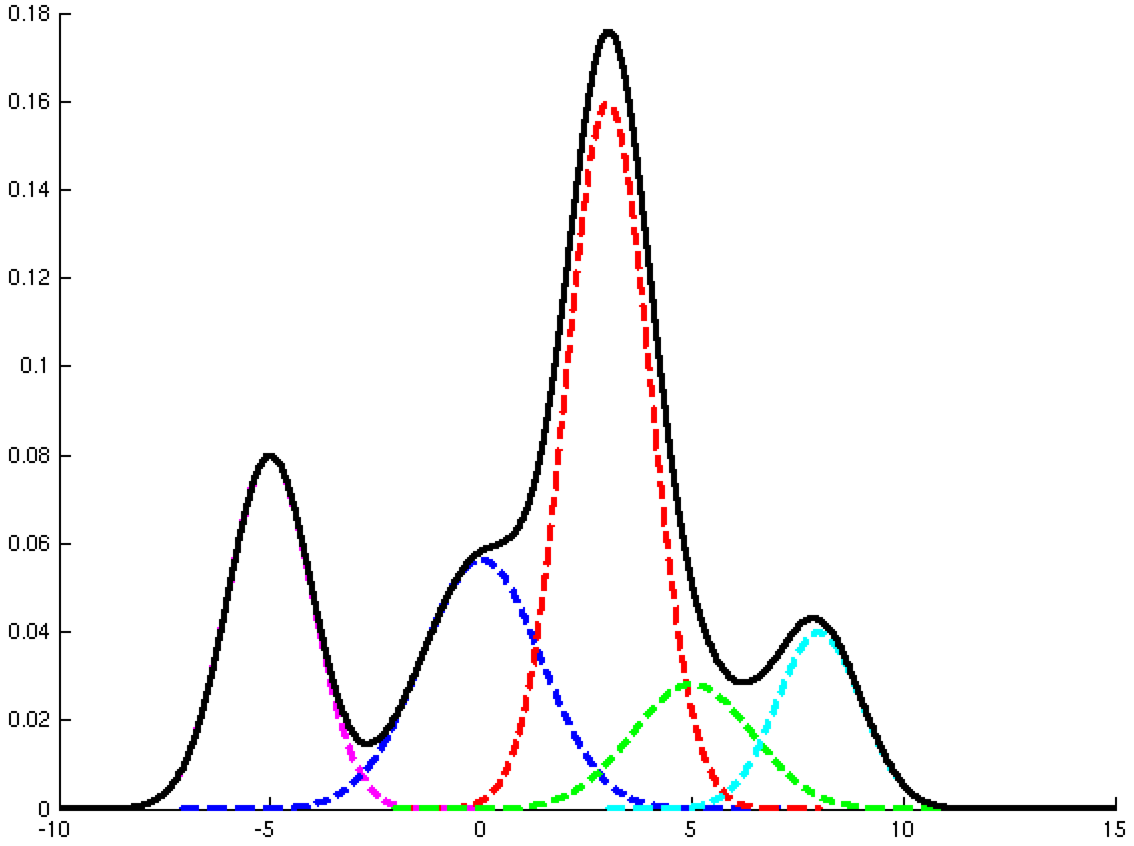
Gaussian Mixture Models

$$K = 5,\ \mu = \{-5, 0, 3, 5, 8\},\ \sigma = \{1, 2, 1, 2, 1\},\ \alpha = \{0.2, 0.2, 0.4, 0.1, 0.1\}$$



Gaussian Mixture Models

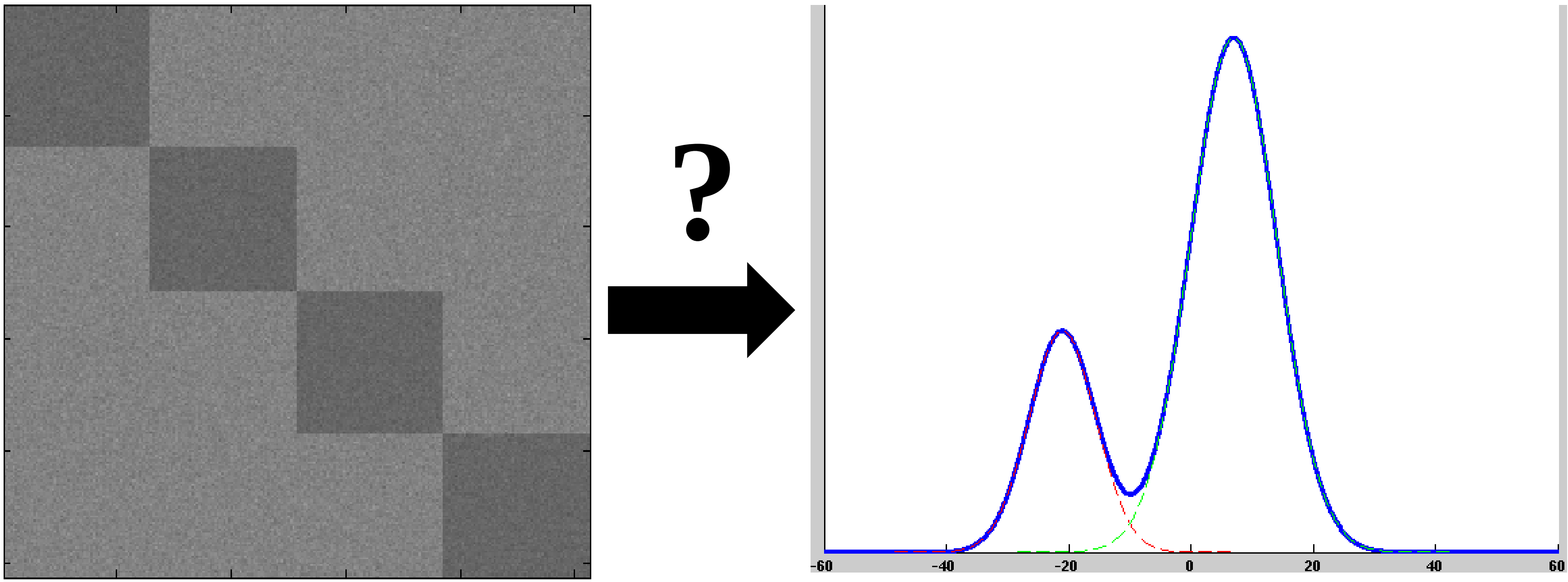
$$K = 5,\ \mu = \{-5, 0, 3, 5, 8\},\ \sigma = \{1, 2, 1, 2, 1\},\ \alpha = \{0.2, 0.2, 0.4, 0.1, 0.1\}$$



Gaussian Mixture Models

- For most real problems, we don't know how many or which Gaussians the data are sampled from.
- How do we automatically determine the parameters of a GMM from observed samples?

Unsupervised Learning - GMMs



- Given an image we want to determine parameters $\boldsymbol{\Theta}$:

$$\boldsymbol{\Theta} = \{\alpha_1, \ldots, \alpha_k, \mu_1, \ldots, \mu_k, \sigma_1, \ldots, \sigma_k\}$$

Maximum Likelihood

- Assume we know form of model (GMM) :

$$p(x|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \alpha_k \, \mathcal{N}(x; \mu_k, \sigma_k)$$

Maximum Likelihood

- Assume we know form of model (GMM) :

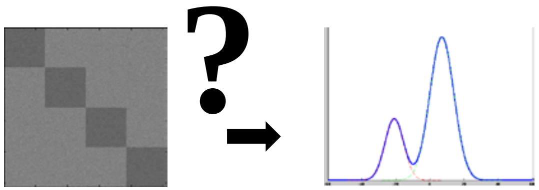
$$p(x|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \alpha_k \, \mathcal{N}(x; \mu_k, \sigma_k)$$

- Assume observed samples, x_i, are i.i.d.:

$$p(\mathcal{X}|\boldsymbol{\Theta}) = \prod_{i=1}^{N} p(x_i|\boldsymbol{\Theta})$$

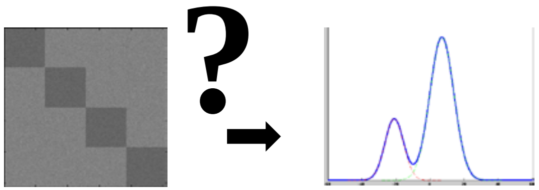
Maximum Likelihood

$$p(\mathcal{X}|\mathbf{\Theta}) = \prod_{i=1}^{N} p(x_i|\mathbf{\Theta})$$



N : # samples, i : sample (pixel) index, $\mathbf{\Theta}$: GMM parameters, K : # components in GMM

Maximum Likelihood

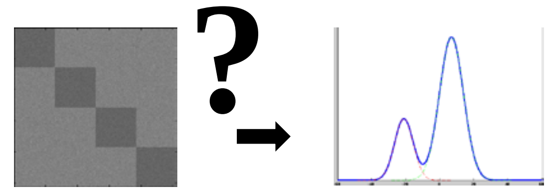


$$p(\mathcal{X}|\mathbf{\Theta}) = \prod_{i=1}^{N} p(x_i|\mathbf{\Theta})$$

$$= \prod_{i=1}^{N} \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_i; \mu_k, \sigma_k)$$

N : # samples, i : sample (pixel) index, $\mathbf{\Theta}$: GMM parameters, K : # components in GMM

Maximum Likelihood



$$p(\mathcal{X}|\boldsymbol{\Theta}) = \prod_{i=1}^{N} p(x_i|\boldsymbol{\Theta})$$

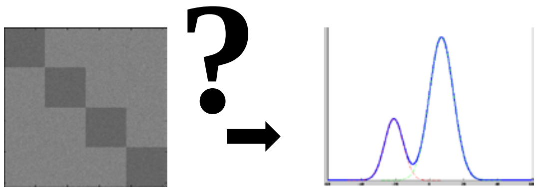
$$= \prod_{i=1}^{N} \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_i; \mu_k, \sigma_k)$$

- Want to find model parameters, $\boldsymbol{\Theta}$, that maximize likelihood.

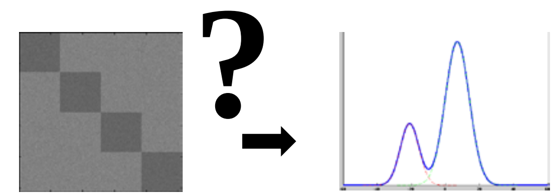
N : # samples, i : sample (pixel) index, $\boldsymbol{\Theta}$: GMM parameters, K : # components in GMM

Maximum Likelihood for GMM

$$\mathbf{\Theta}^* = \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, p(\mathcal{X}|\mathbf{\Theta})$$



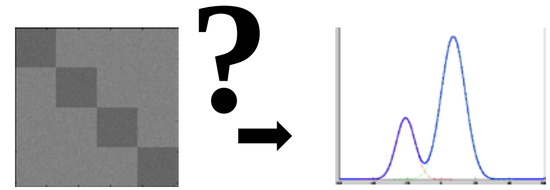
Maximum Likelihood for GMM



$$\mathbf{\Theta}^* = \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, p(\mathcal{X}|\mathbf{\Theta})$$

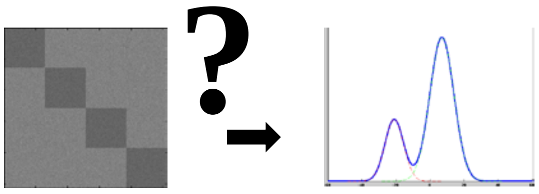
$$= \underset{\mathbf{\Theta}}{\operatorname{argmax}} \log p(\mathcal{X}|\mathbf{\Theta})$$

Maximum Likelihood for GMM



$$
\begin{aligned}
\mathbf{\Theta}^* &= \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, p(\mathcal{X}|\mathbf{\Theta}) \\
&= \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, \log p(\mathcal{X}|\mathbf{\Theta}) \\
&= \underset{\mathbf{\Theta}}{\operatorname{argmax}}\, \log \prod_{i=1}^{N} \sum_{k=1}^{K} \alpha_k\, \mathcal{N}(x_i; \mu_k, \sigma_k)
\end{aligned}
$$

Maximum Likelihood for GMM



$$
\begin{aligned}
\boldsymbol{\Theta}^* &= \operatorname*{argmax}_{\boldsymbol{\Theta}} p(\mathcal{X}|\boldsymbol{\Theta}) \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \log p(\mathcal{X}|\boldsymbol{\Theta}) \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \log \prod_{i=1}^{N} \sum_{k=1}^{K} \alpha_k \, \mathcal{N}(x_i; \mu_k, \sigma_k) \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \sum_{i=1}^{N} \log(\sum_{k=1}^{K} \alpha_k \, \mathcal{N}(x_i; \mu_k, \sigma_k))
\end{aligned}
$$

Maximum Likelihood for GMM

$$\mathbf{\Theta}^* = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \sum_{i=1}^{N} \log\left(\sum_{k=1}^{K} \alpha_k \, \mathcal{N}(x_i; \mu_k, \sigma_k)\right)$$

- Difficult to optimize because contains log of summation

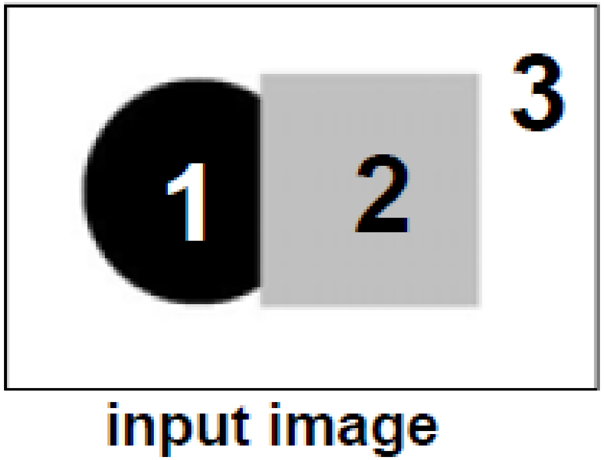
Expectation Maximization (EM)

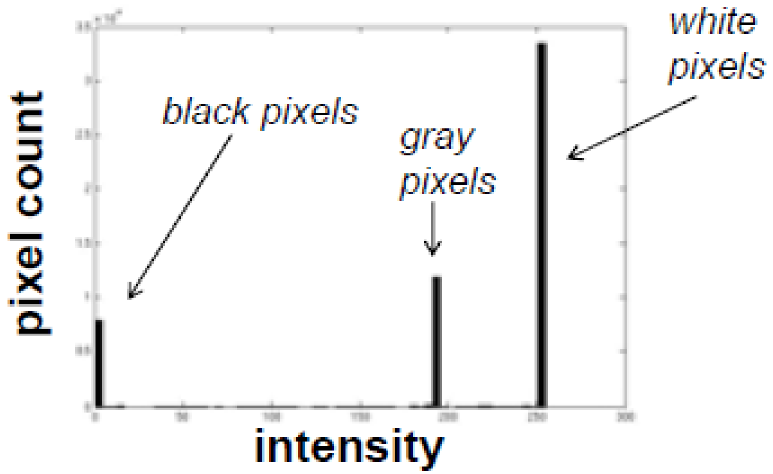
- Iterative algorithm for finding maximum-likelihood estimate of model parameters from a given data set.
- Expectation : determine which Gaussian component(s) each sample comes from based on current estimate of model parameters
- Maximization : maximize model parameters given current estimates of Gaussian sources of each sample

GMMs for Segmentation

- We have started to look at the use of GMMs / EM for estimating multi-modal probability densities.
- GMMs / EM are also often used for segmentation or clustering tasks
- EM can be thought of as conceptually similar to K-means
- Recall K-means for segmentation….

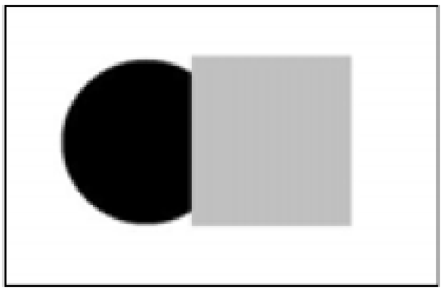
Image Segmentation: toy example



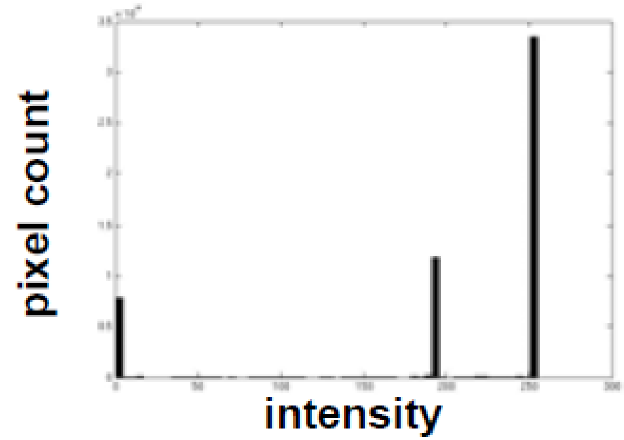


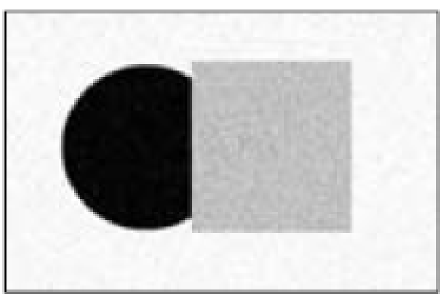
- These intensities define the three groups
- We could label every pixel in the image according to which of these primary intensities it is
  - i.e., *segment* the image based on the intensity feature.
- What if the image isn't quite so simple?

Image Segmentation: toy example

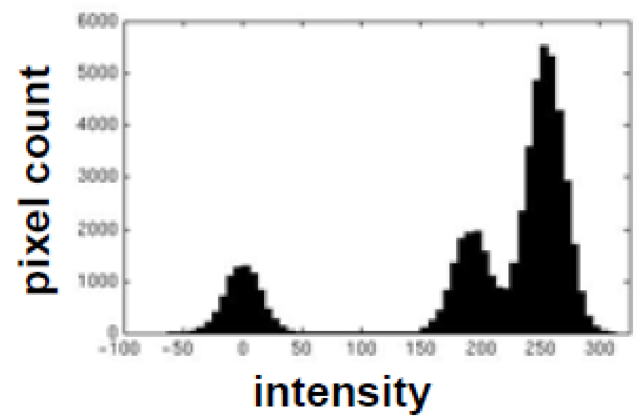


input image



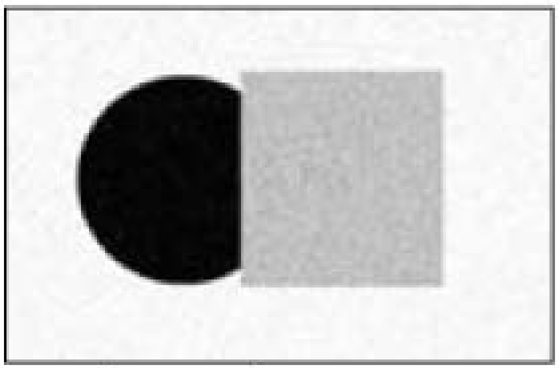


input image

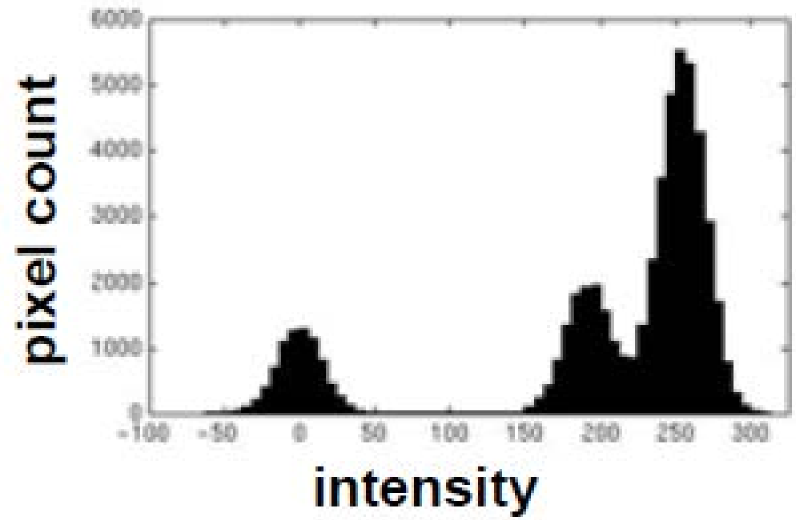


Kristen Grauman, Cs 378, UT Austin

Image Segmentation: toy example

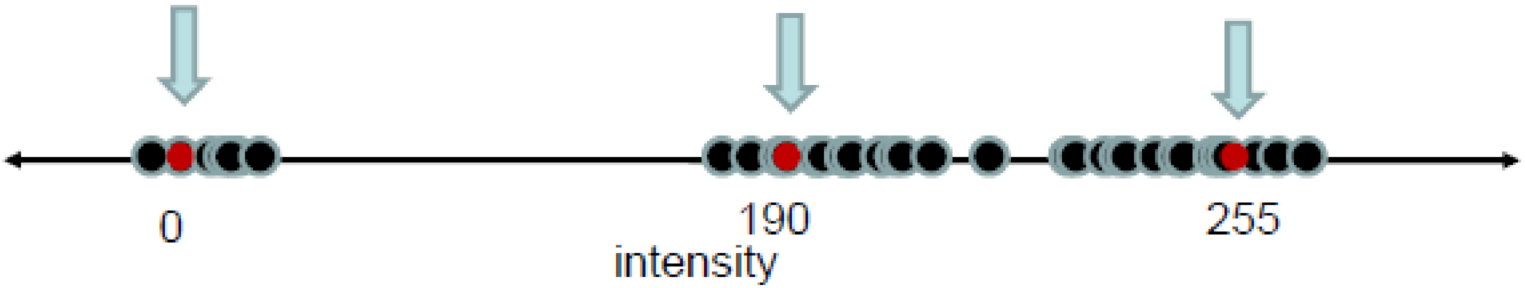


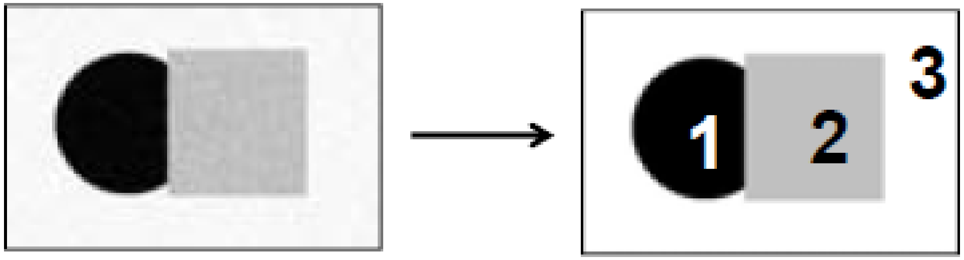
input image



- Now how to determine the three main intensities that define our groups?
- We need to *cluster*

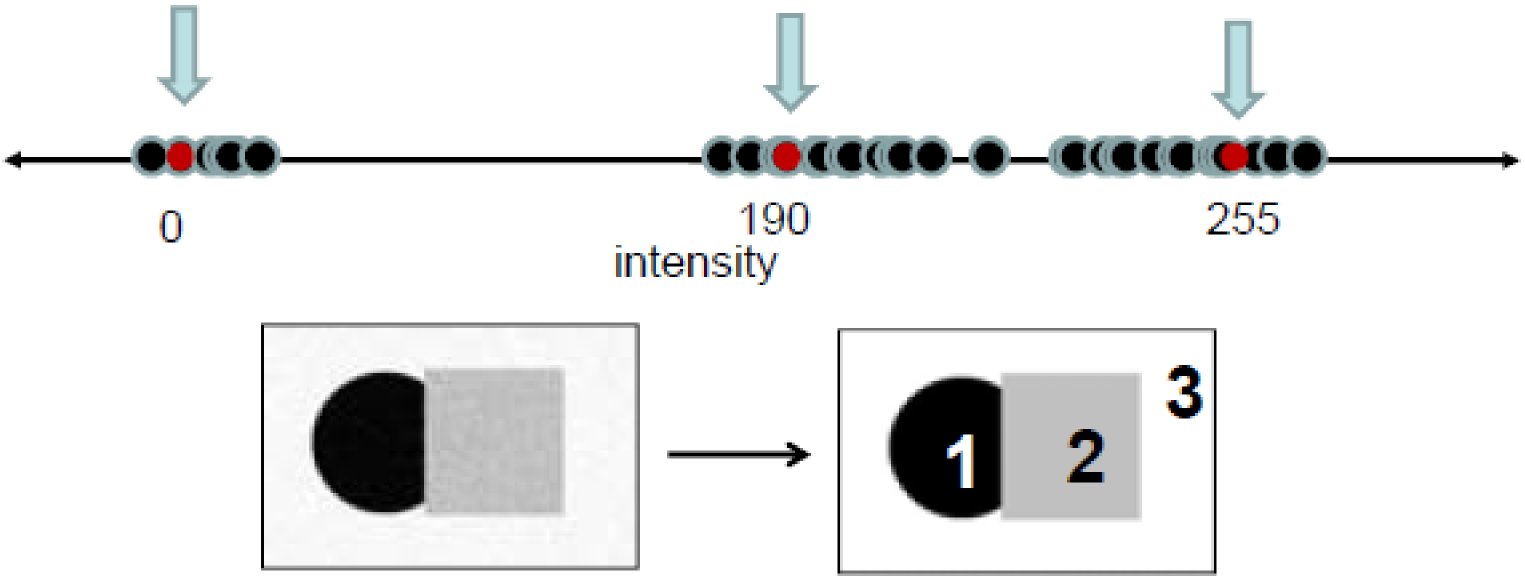
Image Segmentation: Clustering





- Goal: choose three *centers* as the representative intensities, and label every pixel according to which of these centers it is nearest to

Image Segmentation: Clustering

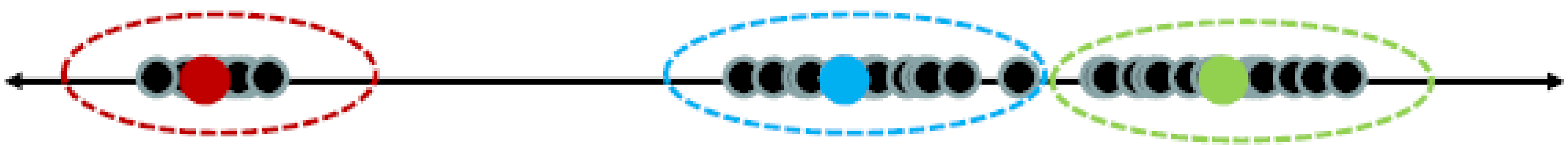


- Best cluster centers are those that minimize SSD between all points and their nearest cluster center c_i

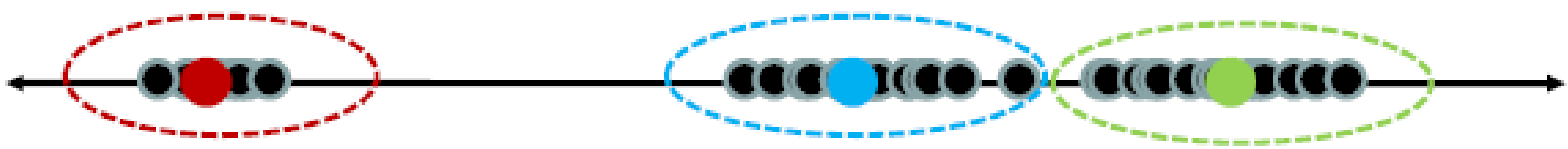
$$\sum_{\text{cluster } i} \sum_{\text{points } p \text{ in cluster } i} \|p - c_i\|^2$$

Image Segmentation: Clustering

- With this objective, it is a “chicken and egg” problem
  - If we knew the cluster centers, we could allocate points to groups by assigning each to its closest center



  - If we knew the group memberships, we could get the centers by computing the mean per group



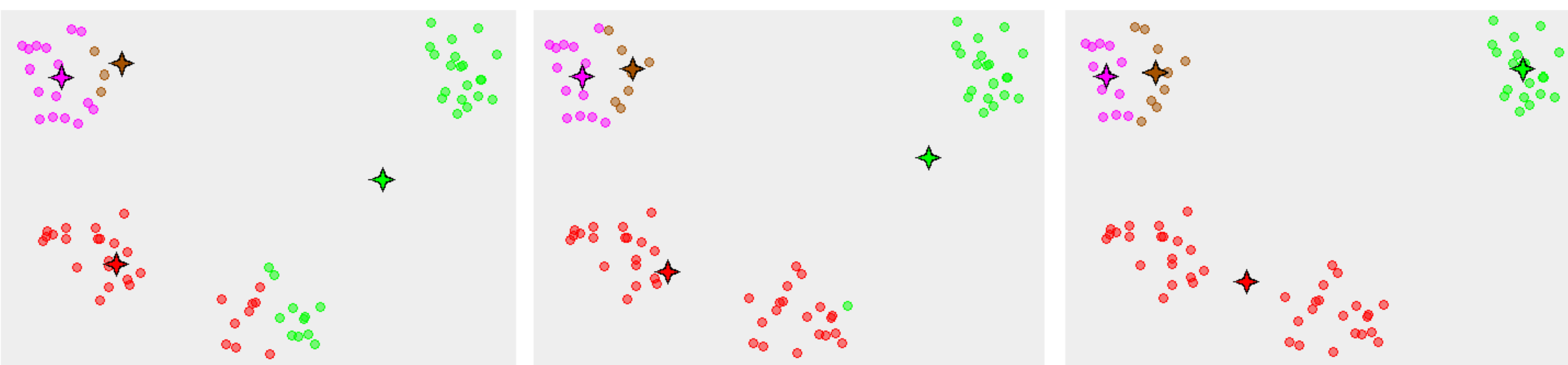
K-means Clustering Algorithm

- Randomly initialize the K cluster centers, and iterate between the two steps we just saw

1. Randomly initialize the cluster centers $c_1, \ldots, c_K$
2. Given the cluster centers, determine points in each cluster
   - For each point p, find the closest c_i. Put p into cluster i
3. Given points in each cluster, solve for c_i
   - Set c_i to be the mean of points in cluster i
4. If c_i have changed, repeat Step 2

K-means Clustering

- Converging to a local minimum



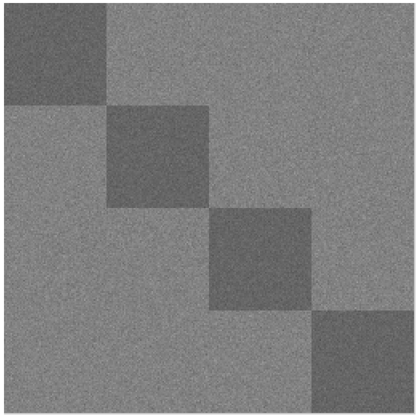
GMMs for Segmentation

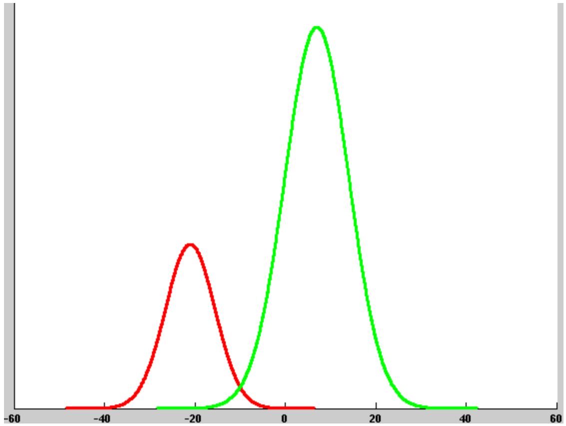
- GMMs / EM are also often used for segmentation or clustering tasks
- We can think of each of the K components in our GMM as representing a distinct *class* and each sample as having a membership to one or more of the K classes.

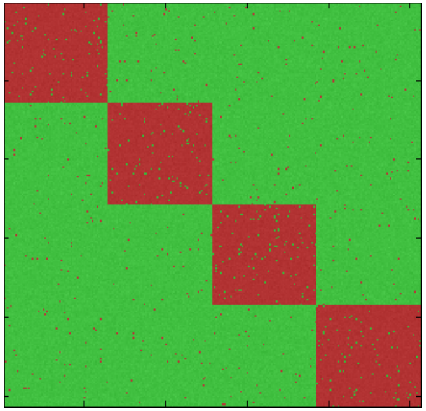
GMMs for Segmentation

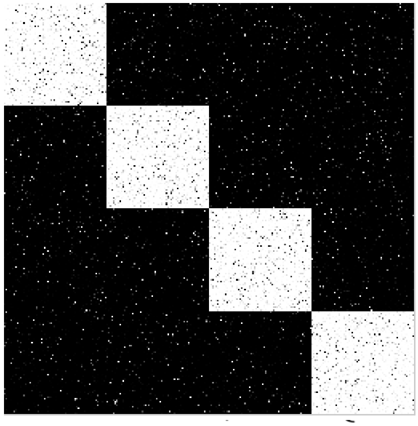
- We can model our image as a GMM consisting of $\boldsymbol{K}$ components, each representing a distinct class.
- We can then assign a likelihood of each of the $\boldsymbol{K}$ classes for every sample in our image

GMMs for Segmentation

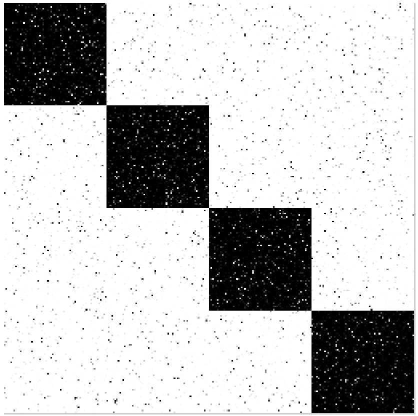






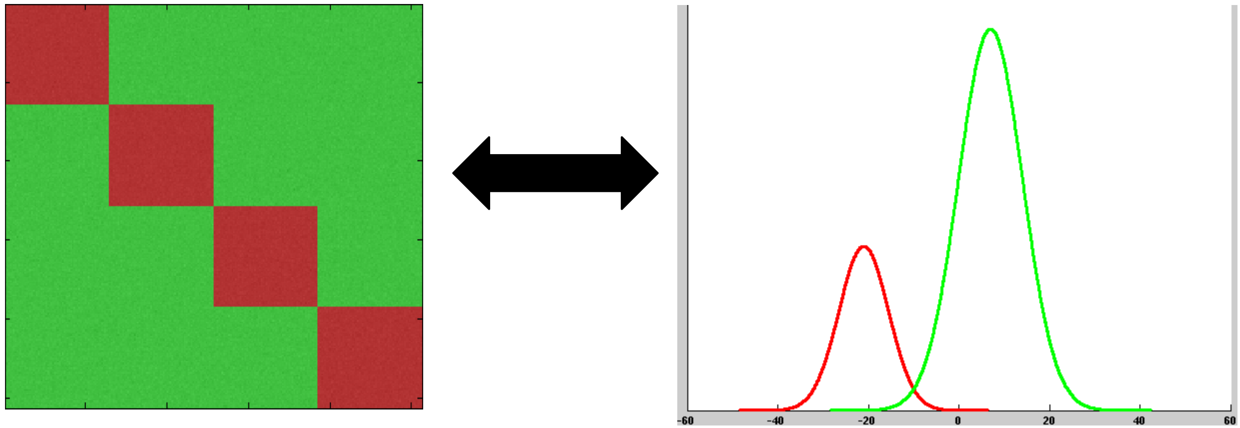


$p(y_i = 1 | x_i, \mathbf{\Theta})$



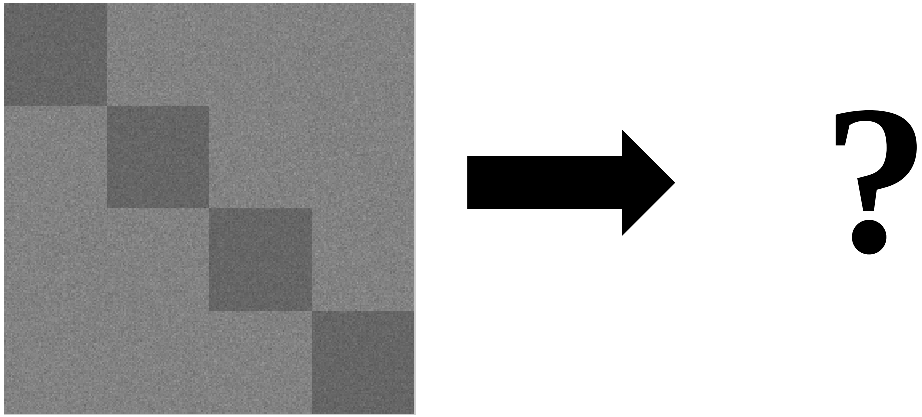
$p(y_i = 2 | x_i, \mathbf{\Theta})$

Expectation Maximization



- If we know which class (gaussian component) each sample came from, the we can estimate the parameters of our GMM.
- If we know the parameters of the GMM, we can determine the likelihood of each sample belonging to each class.

Expectation Maximization



- In practice we don't know the GMM parameters or the classes of each sample but only have the image.

Expectation Maximization

$$\mathbf{\Theta}^* = \underset{\mathbf{\Theta}}{\operatorname{argmax}} \sum_{i=1}^{N} \log(\sum_{k=1}^{K} \alpha_k \mathcal{N}(x_i; \mu_k, \sigma_k))$$

- To simplify our optimization, we introduce a hidden parameter, ***Y*,** that acts as a class label (corresponding to one of ***K*** gaussian clusters) for each sample, x_i:

$$\mathcal{Y} = \{y_i\}_{i=1}^{N}, y_i \in 1, \ldots, K$$

Expectation Maximization

$$\mathcal{Y} = \{y_i\}_{i=1}^{N}, y_i \in 1, ..., K$$

- For segmentation, we can think of each $\boldsymbol{y_i}$ as a class label for sample $\boldsymbol{x_i}$

Expectation Maximization

$$\mathcal{Y} = \{y_i\}_{i=1}^{N}, y_i \in 1, ..., K$$

- For segmentation, we can think of each $\boldsymbol{y_i}$ as a class label for sample $\boldsymbol{x_i}$
- For density estimation, $\boldsymbol{y_i}$ acts as a hidden parameter that will allow us to solve for our GMM parameters, $\boldsymbol{\Theta}$.

Expectation Maximization

$$\mathcal{Y} = \{y_i\}_{i=1}^{N}, y_i \in 1, ..., K$$

- For segmentation, we can think of each $\boldsymbol{y_i}$ as a class label for sample $\boldsymbol{x_i}$
- For density estimation, $\boldsymbol{y_i}$ acts as a hidden parameter that will allow us to solve for our GMM parameters, $\boldsymbol{\Theta}$.
- In general, each $\boldsymbol{y_i}$ will be represented as a distribution over all $\boldsymbol{K}$ possible classes.

Expectation Maximization

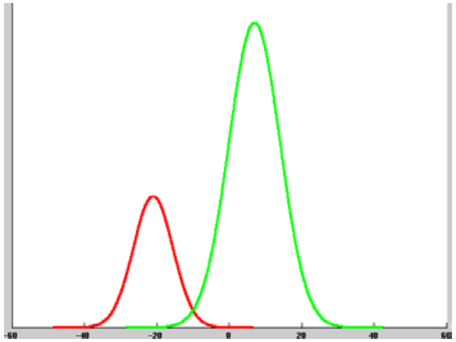
$$p(x_i, y_i|\mathbf{\Theta}) = p(x_i|y_i, \mathbf{\Theta})p(y_i|\mathbf{\Theta})$$

Expectation Maximization

$$p(x_i, y_i | \mathbf{\Theta}) = p(x_i | y_i, \mathbf{\Theta}) p(y_i | \mathbf{\Theta})$$

- We can simplify by observing the following:

$$p(y_i | \mathbf{\Theta}) = \alpha_{y_i}$$



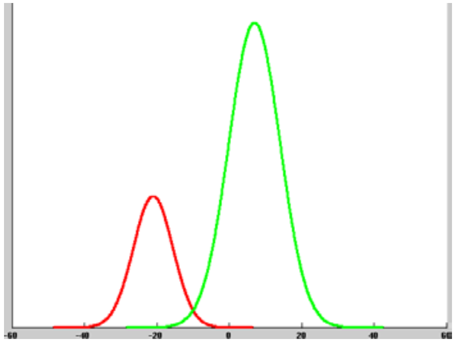
Expectation Maximization

$$p(x_i, y_i|\mathbf{\Theta}) = p(x_i|y_i, \mathbf{\Theta})p(y_i|\mathbf{\Theta})$$

- We can simplify by observing the following:

$$p(y_i|\mathbf{\Theta}) = \alpha_{y_i}$$

$$p(x_i|y_i, \mathbf{\Theta}) = \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})$$



Expectation Maximization

$$p(x_i, y_i|\mathbf{\Theta}) = p(x_i|y_i, \mathbf{\Theta})p(y_i|\mathbf{\Theta})$$

- We can simplify by observing the following:

$$p(y_i|\mathbf{\Theta}) = \alpha_{y_i}$$

$$p(x_i|y_i, \mathbf{\Theta}) = \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})$$

$$p(x_i, y_i|\mathbf{\Theta}) = \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})\alpha_{y_i}$$

Expectation Maximization

$$p(\mathcal{X}, \mathcal{Y} | \mathbf{\Theta}) = \prod_{i=1}^{N} p(x_i, y_i | \mathbf{\Theta})$$

Expectation Maximization

$$p(\mathcal{X}, \mathcal{Y}|\mathbf{\Theta}) = \prod_{i=1}^{N} p(x_i, y_i|\mathbf{\Theta})$$

$$= \prod_{i=1}^{N} \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})\alpha_{y_i}$$

Expectation Maximization

$$\mathbf{\Theta}^* = \operatorname*{argmax}_{\mathbf{\Theta}} p(\mathcal{X}, \mathcal{Y} | \mathbf{\Theta})$$

Expectation Maximization

$$\begin{aligned}\mathbf{\Theta}^* &= \operatorname*{argmax}_{\mathbf{\Theta}} p(\mathcal{X}, \mathcal{Y} | \mathbf{\Theta}) \\ &= \operatorname*{argmax}_{\mathbf{\Theta}} \log p(\mathcal{X}, \mathcal{Y} | \mathbf{\Theta})\end{aligned}$$

Expectation Maximization

$$\begin{aligned}\mathbf{\Theta}^* &= \operatorname*{argmax}_{\mathbf{\Theta}} p(\mathcal{X}, \mathcal{Y}|\mathbf{\Theta}) \\ &= \operatorname*{argmax}_{\mathbf{\Theta}} \log p(\mathcal{X}, \mathcal{Y}|\mathbf{\Theta}) \\ &= \operatorname*{argmax}_{\mathbf{\Theta}} \log \prod_{i=1}^{N} \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i}) \alpha_{y_i}\end{aligned}$$

Expectation Maximization

$$
\begin{aligned}
\boldsymbol{\Theta}^* &= \operatorname*{argmax}_{\boldsymbol{\Theta}} p(\mathcal{X}, \mathcal{Y} | \boldsymbol{\Theta}) \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \log p(\mathcal{X}, \mathcal{Y} | \boldsymbol{\Theta}) \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \log \prod_{i=1}^{N} \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i}) \alpha_{y_i} \\
&= \operatorname*{argmax}_{\boldsymbol{\Theta}} \sum_{i=1}^{N} \log \alpha_{y_i} \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})
\end{aligned}
$$

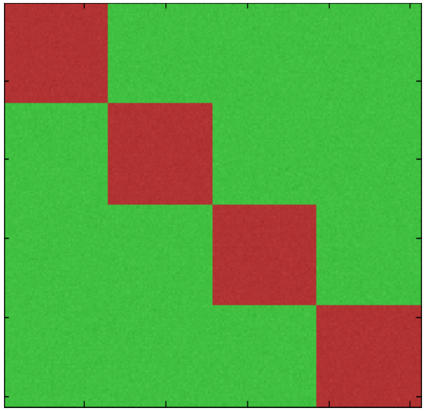
Expectation Maximization

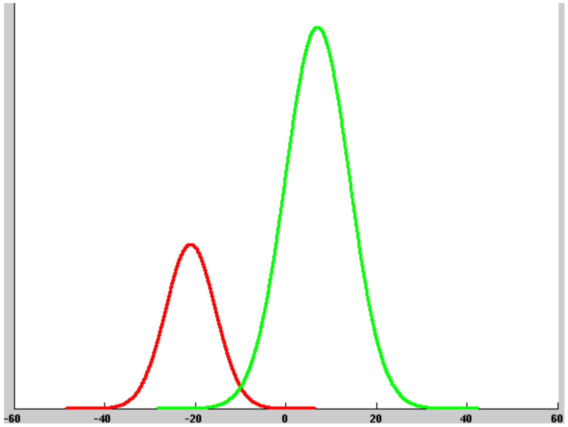
$$\boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}} \sum_{i=1}^{N} \log \alpha_{y_i} \mathcal{N}(x_i; \mu_{y_i}, \sigma_{y_i})$$

- One problem : we don't actually have $\boldsymbol{y_i}$…
- Iterative optimization
  - E-step : estimate $\boldsymbol{y_i}$ based on current estimate of parameters
  - M-step : find parameters, $\boldsymbol{\Theta}$, that maximize expectation of log-likelihood
- Requires initial estimate of parameters, $\boldsymbol{\Theta}$, or of $\boldsymbol{y_i}$

E-step

- Determine density of the hidden parameters, $\boldsymbol{y_i}$, based on current estimate of model parameters

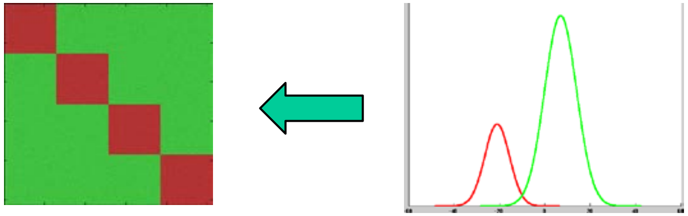




$$p(y_i | x_i, \boldsymbol{\Theta}^t)$$

E-step

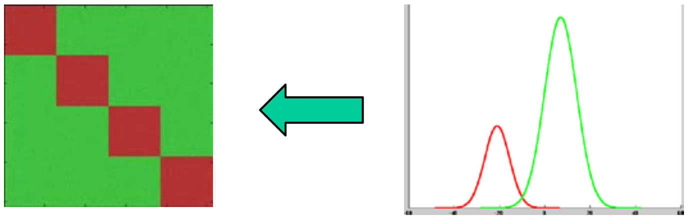
$$p(y_i|x_i,\mathbf{\Theta}^t) = \frac{p(x_i|y_i,\mathbf{\Theta}^t)p(y_i|\mathbf{\Theta}^t)}{p(x_i|\mathbf{\Theta}^t)}$$



E-step

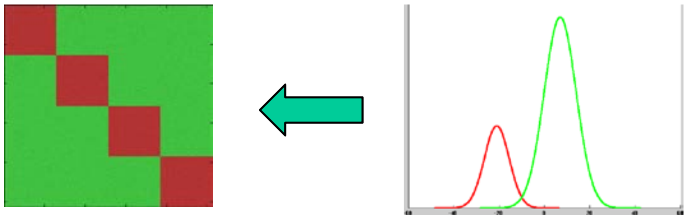
$$p(y_i|x_i, \mathbf{\Theta}^t) = \frac{p(x_i|y_i, \mathbf{\Theta}^t)p(y_i|\mathbf{\Theta}^t)}{p(x_i|\mathbf{\Theta}^t)}$$

$$= \frac{\mathcal{N}(x_i; \mu_{y_i}^t, \sigma_{y_i}^t)\alpha_{y_i}^t}{p(x_i|\mathbf{\Theta}^t)}$$



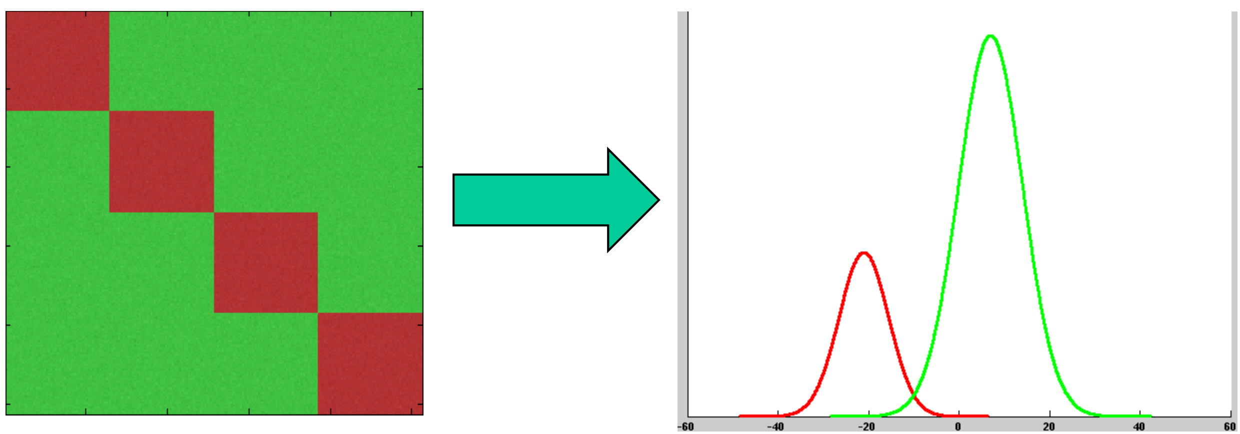
E-step

$$
\begin{aligned}
p(y_i|x_i, \mathbf{\Theta}^t) &= \frac{p(x_i|y_i, \mathbf{\Theta}^t)p(y_i|\mathbf{\Theta}^t)}{p(x_i|\mathbf{\Theta}^t)} \\
&= \frac{\mathcal{N}(x_i; \mu_{y_i}^t, \sigma_{y_i}^t)\alpha_{y_i}^t}{p(x_i|\mathbf{\Theta}^t)} \\
&= \frac{\mathcal{N}(x_i; \mu_{y_i}^t, \sigma_{y_i}^t)\alpha_{y_i}^t}{\sum_{k=1}^{K} \alpha_k^t \, \mathcal{N}(x_i; \mu_k^t, \sigma_k^t)}
\end{aligned}
$$



M-step

- Find model parameters that maximize expectation of log likelihood based on current estimate of density of hidden parameters

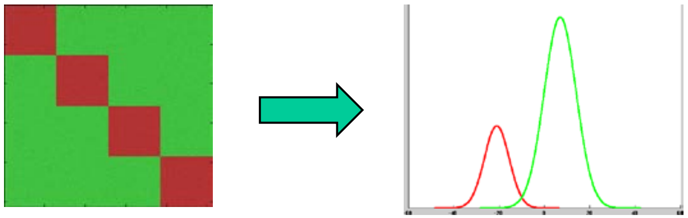


M-step

$$\mathrm{E}[\log p(\mathcal{X},\mathcal{Y}|\mathbf{\Theta})|\mathcal{X},\mathbf{\Theta}^t] = \sum_{\mathbf{y}\in\mathcal{Y}} \log p(\mathcal{X},\mathcal{Y}|\mathbf{\Theta})p(\mathbf{y}|\mathcal{X},\mathbf{\Theta}^t)$$

$$Q(\mathbf{\Theta},\mathbf{\Theta}^t) = \mathrm{E}[\log p(\mathcal{X},\mathcal{Y}|\mathbf{\Theta})|\mathcal{X},\mathbf{\Theta}^t]$$

$$\mathbf{\Theta}^{t+1} = \operatorname*{argmax}_{\mathbf{\Theta}} Q(\mathbf{\Theta},\mathbf{\Theta}^t)$$



M-step

- After a long derivation, we can get analytical equations for our M-step if using GMMs:

$$\alpha_k^{t+1} = \frac{1}{N}\sum_{i=1}^{N} p(y_i = k|x_i, \mathbf{\Theta}^t)$$

$$\mu_k^{t+1} = \frac{\sum_{i=1}^{N} x_i p(y_i = k|x_i, \mathbf{\Theta}^t)}{\sum_{i=1}^{N} p(y_i = k|x_i, \mathbf{\Theta}^t)}$$

$$\sigma_k^{t+1} = \frac{\sum_{i=1}^{N} p(y_i = k|x_i, \mathbf{\Theta}^t)(x_i - \mu_k^t)^2}{\sum_{i=1}^{N} p(y_i = k|x_i, \mathbf{\Theta}^t)}$$

M-step

- The update equations are actually quite intuitive:

$$\alpha_k^{t+1} = \frac{1}{N} \sum_{i=1}^{N} p(y_i = k | x_i, \mathbf{\Theta}^t)$$

- This represents the total relative weight of samples that are of class ***k*** (or gaussian component ***k***), based on estimates of class labels (or hidden parameters), $\boldsymbol{y_i}$

M-step

$$\mu_k^{t+1} = \frac{\sum_{i=1}^{N} x_i p(y_i = k | x_i, \mathbf{\Theta}^t)}{\sum_{i=1}^{N} p(y_i = k | x_i, \mathbf{\Theta}^t)}$$

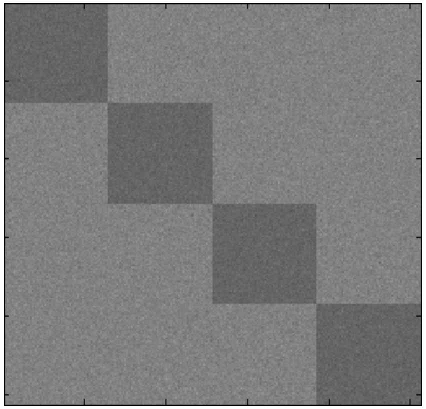
- This represents the weighted sample mean of samples that are of class ***k*** (or gaussian component ***k***), based on estimates of class labels (or hidden parameters), $\boldsymbol{y_i}$

M-step

$$\sigma_k^{t+1} = \frac{\sum_{i=1}^{N} p(y_i = k | x_i, \mathbf{\Theta}^t)(x_i - \mu_k^t)^2}{\sum_{i=1}^{N} p(y_i = k | x_i, \mathbf{\Theta}^t)}$$

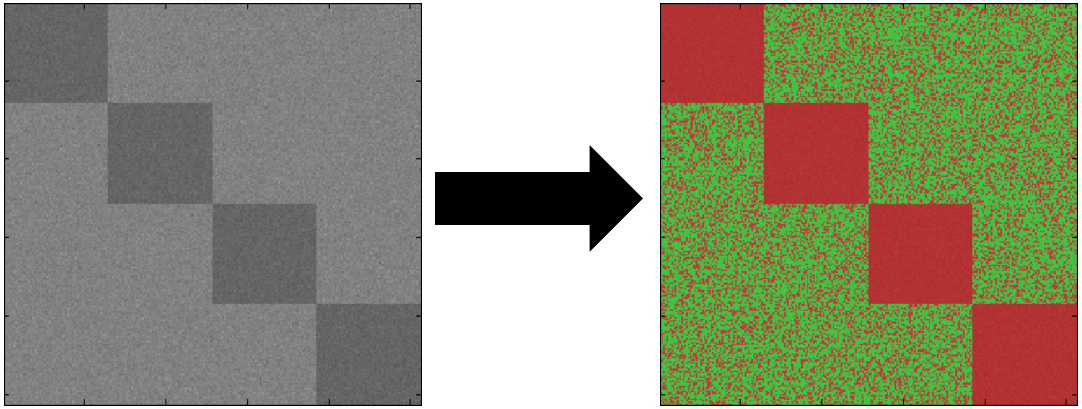
- This represents the weighted sample variance of samples that are of class ***k*** (or gaussian component ***k***), based on estimates of class labels (or hidden parameters), $\boldsymbol{y_i}$

EM in practice



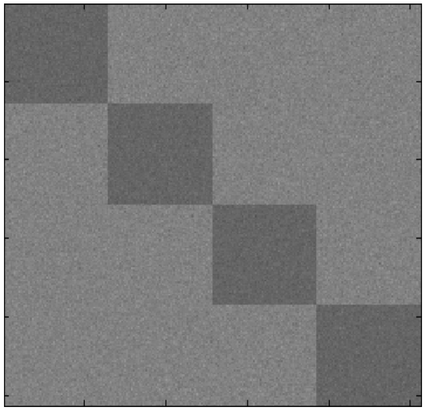
- Assume $\boldsymbol{K} = 2$
- Initialize $\boldsymbol{y_i}$ such that samples with lowest 50% of intensities are assigned a label of 1, while highest 50% are assigned to label 2

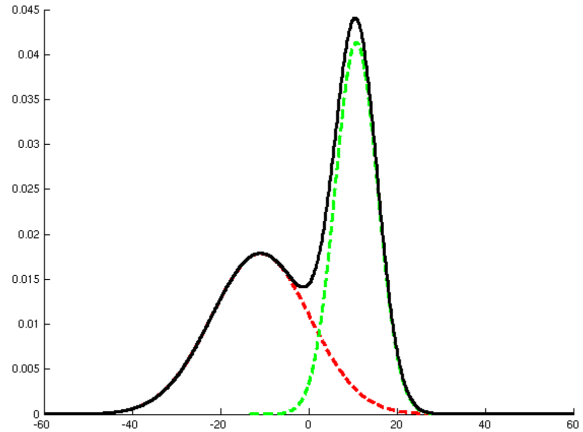
EM in practice



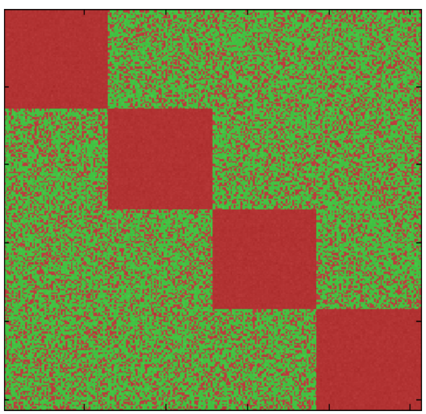
- Assume $\boldsymbol{K} = 2$
- Initialize $\boldsymbol{y_i}$ such that samples with lowest 50% of intensities are assigned a label of 1, while highest 50% are assigned to label 2

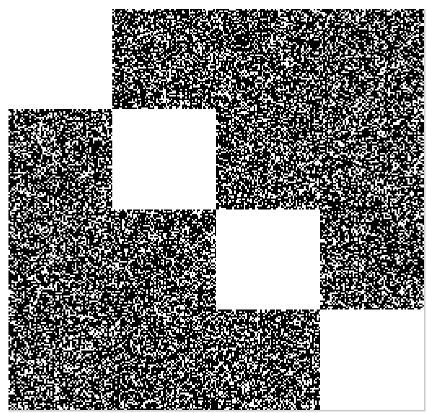
EM in practice : M-step



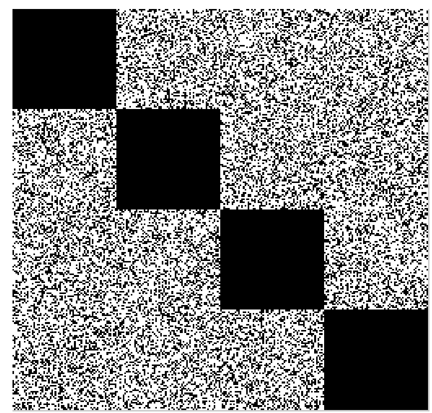


$p(\mathcal{X}|\mathbf{\Theta})$



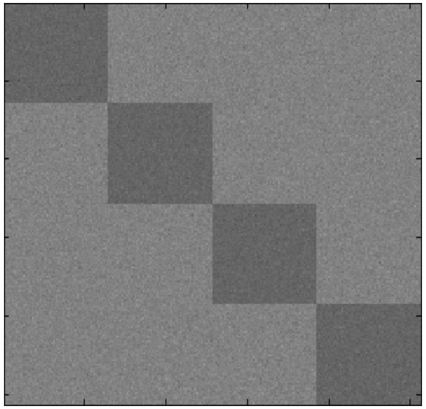


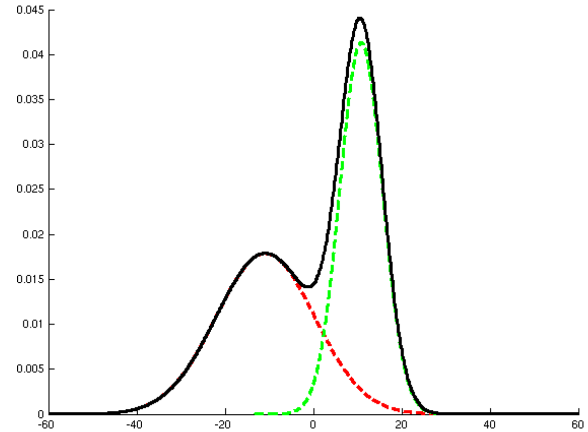
$p(y_i = 1|x_i, \mathbf{\Theta})$



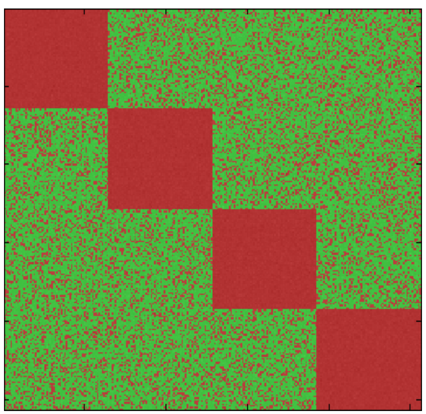
$p(y_i = 2|x_i, \mathbf{\Theta})$

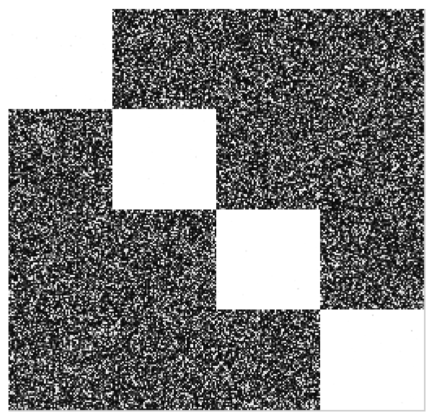
EM in practice : E-step



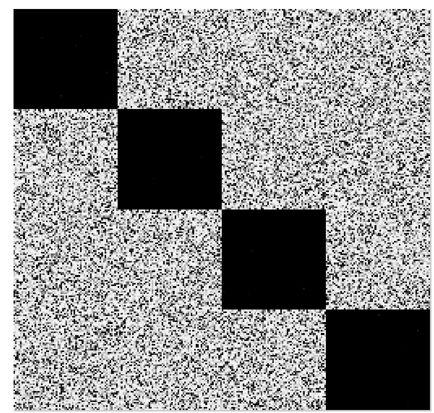


$p(\mathcal{X}|\boldsymbol{\Theta})$



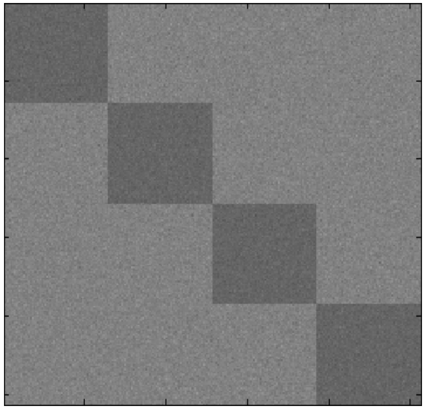


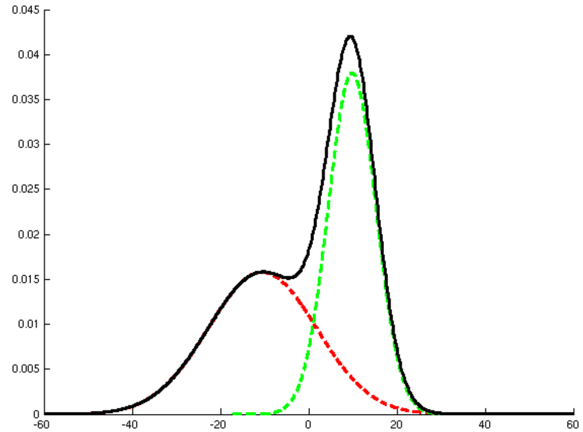
$p(y_i = 1|x_i, \boldsymbol{\Theta})$



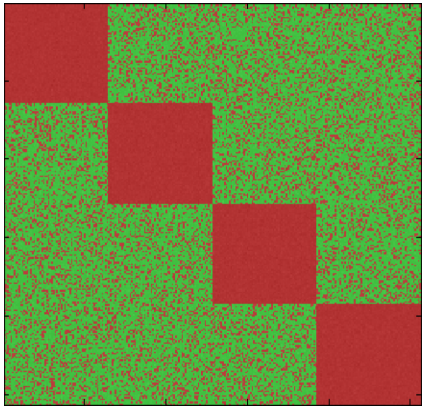
$p(y_i = 2|x_i, \boldsymbol{\Theta})$

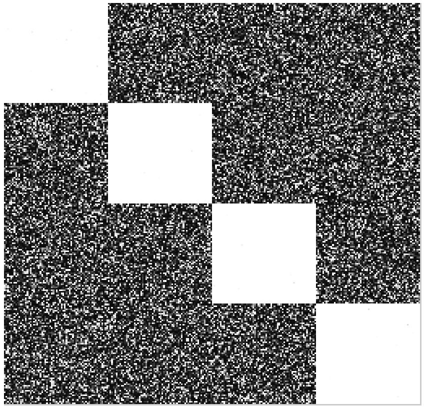
EM in practice : M-step



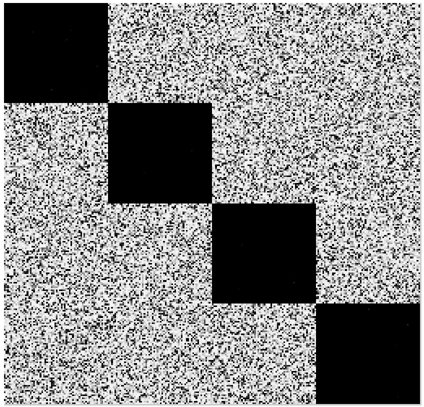


$p(\mathcal{X}|\mathbf{\Theta})$



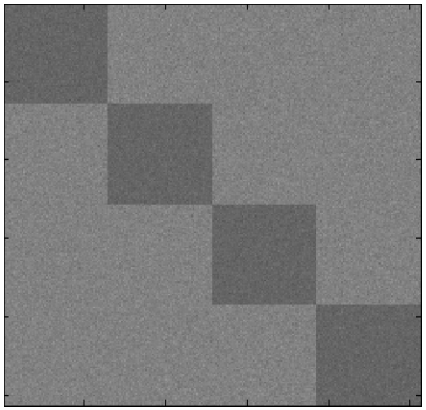


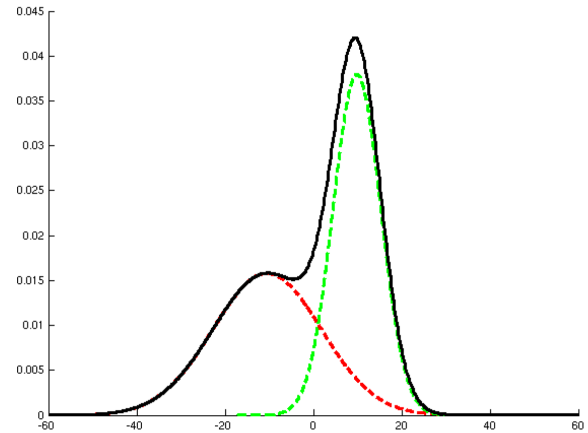
$p(y_i = 1|x_i, \mathbf{\Theta})$



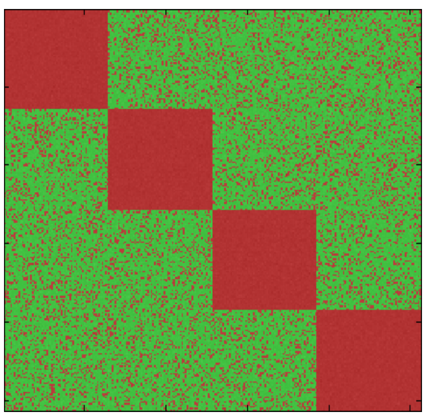
$p(y_i = 2|x_i, \mathbf{\Theta})$

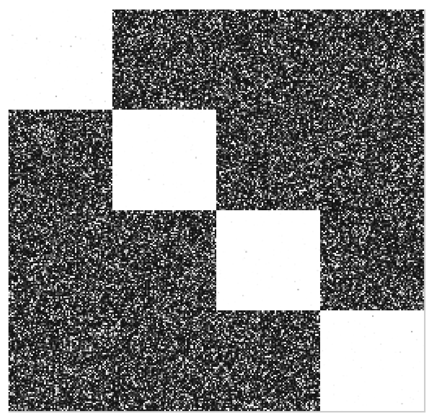
EM in practice : E-step



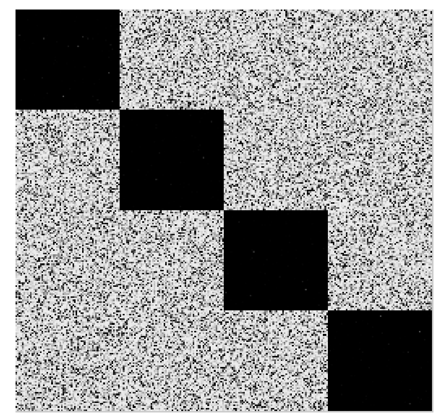


$p(\mathcal{X}|\mathbf{\Theta})$



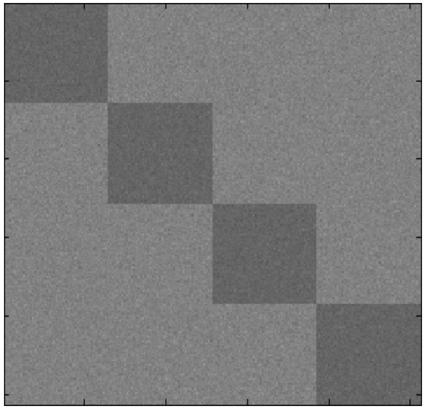


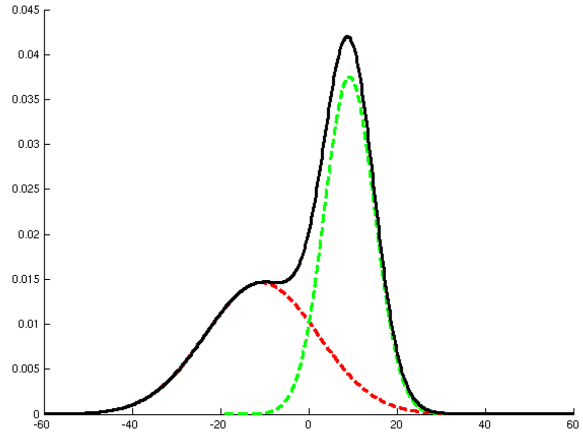
$p(y_i = 1|x_i, \mathbf{\Theta})$



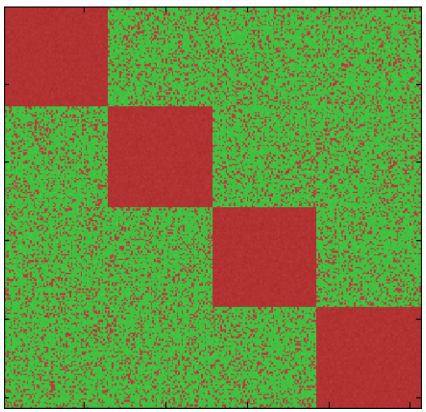
$p(y_i = 2|x_i, \mathbf{\Theta})$

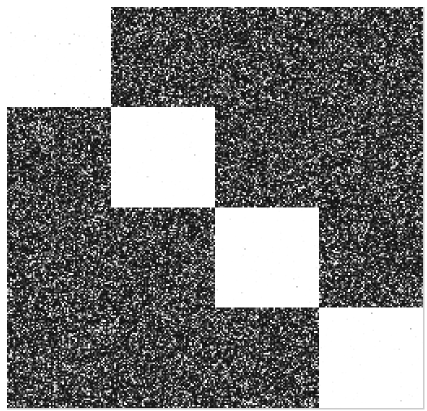
EM in practice : M-step



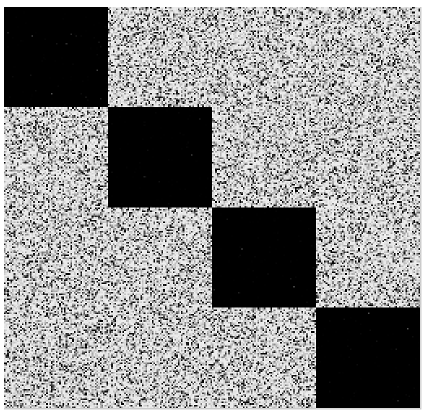


$p(\mathcal{X}|\mathbf{\Theta})$



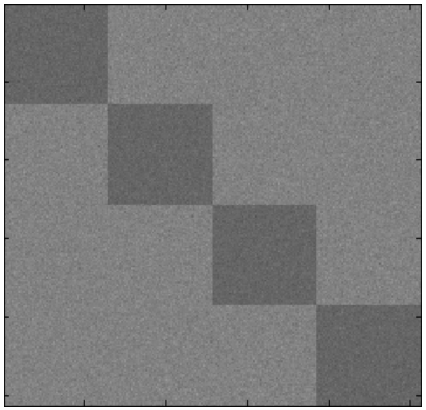


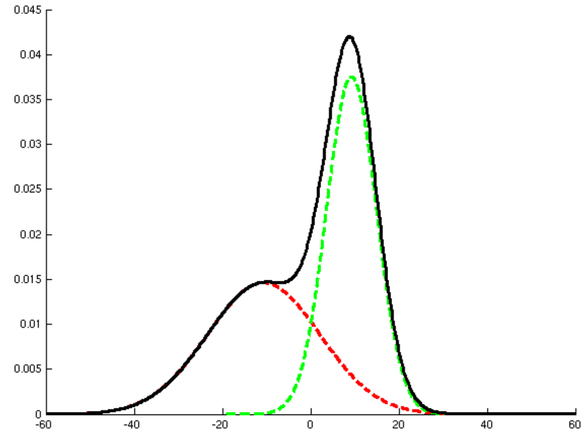
$p(y_i = 1|x_i, \mathbf{\Theta})$



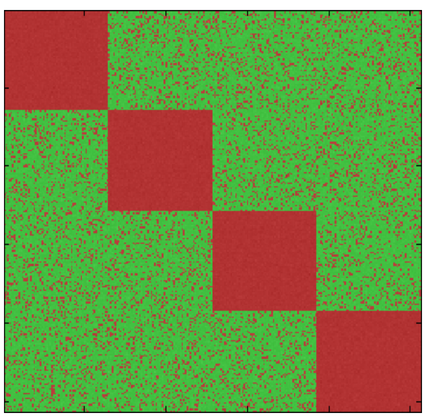
$p(y_i = 2|x_i, \mathbf{\Theta})$

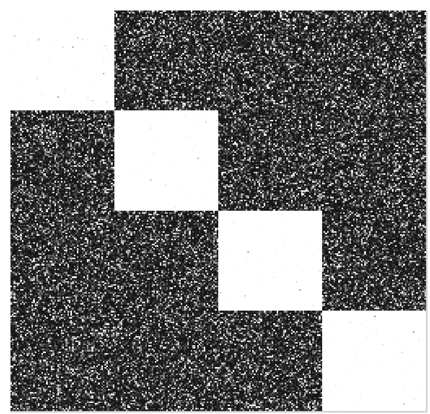
EM in practice : E-step



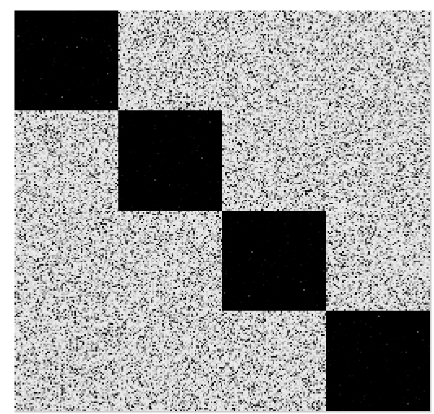


$p(\mathcal{X}|\boldsymbol{\Theta})$



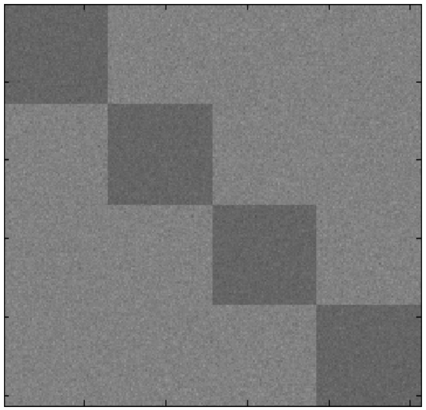


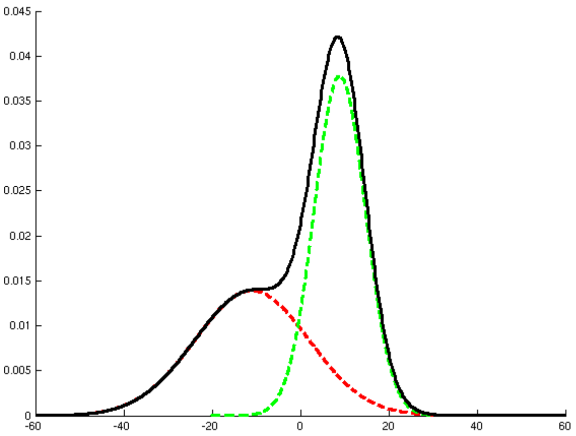
$p(y_i = 1|x_i, \boldsymbol{\Theta})$



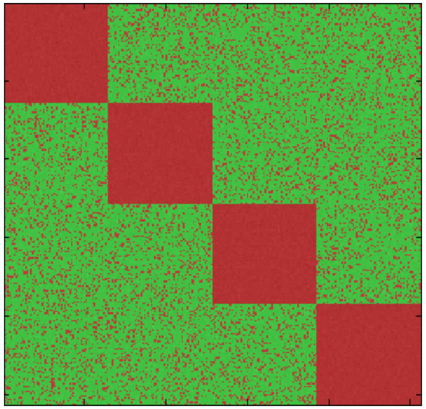
$p(y_i = 2|x_i, \boldsymbol{\Theta})$

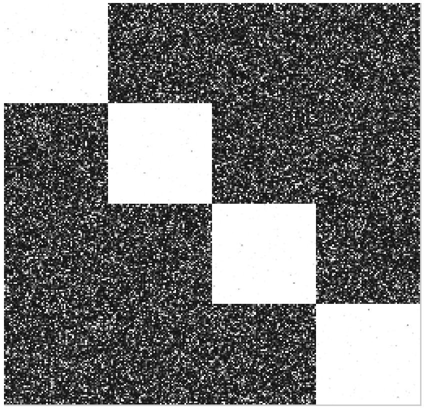
EM in practice : M-step



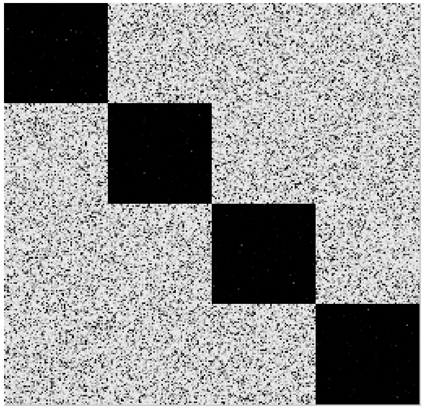


$p(\mathcal{X}|\mathbf{\Theta})$



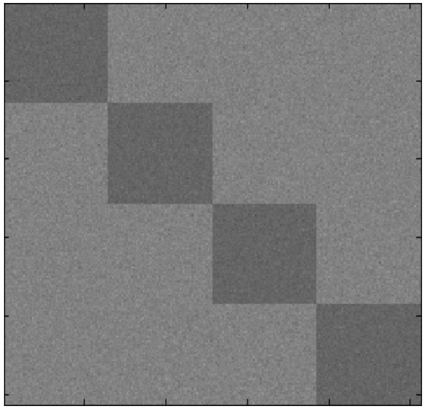


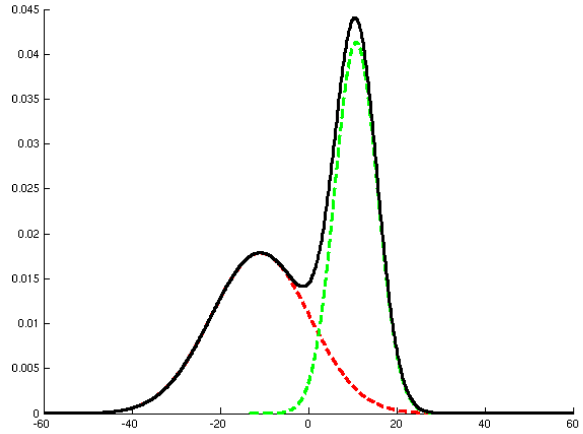
$p(y_i = 1|x_i, \mathbf{\Theta})$



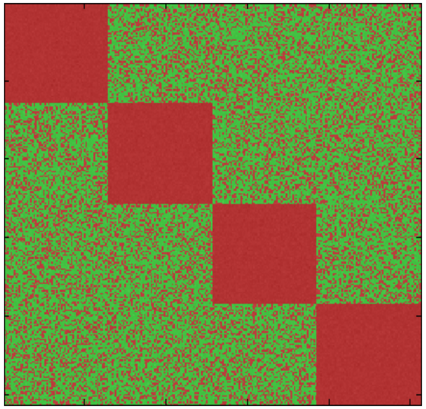
$p(y_i = 2|x_i, \mathbf{\Theta})$

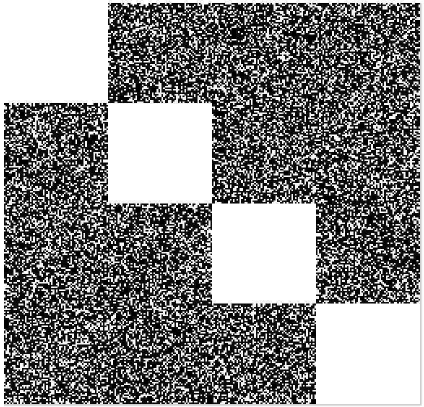
EM in practice



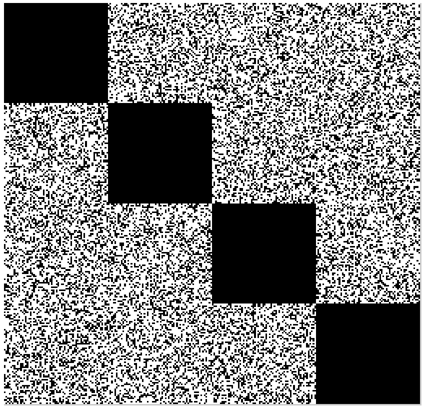


$p(\mathcal{X}|\boldsymbol{\Theta})$



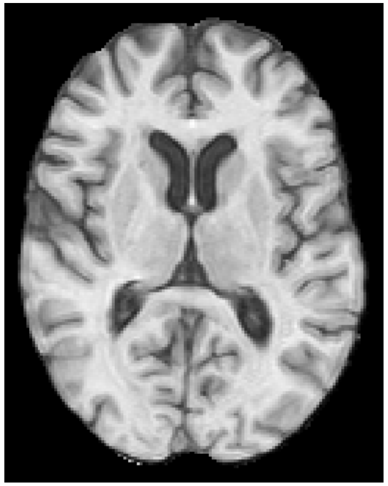


$p(y_i = 1|x_i, \boldsymbol{\Theta})$



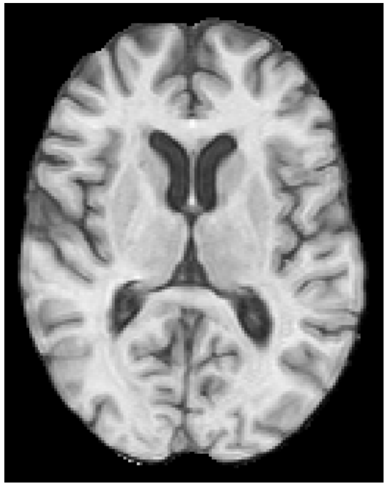
$p(y_i = 2|x_i, \boldsymbol{\Theta})$

EM in practice



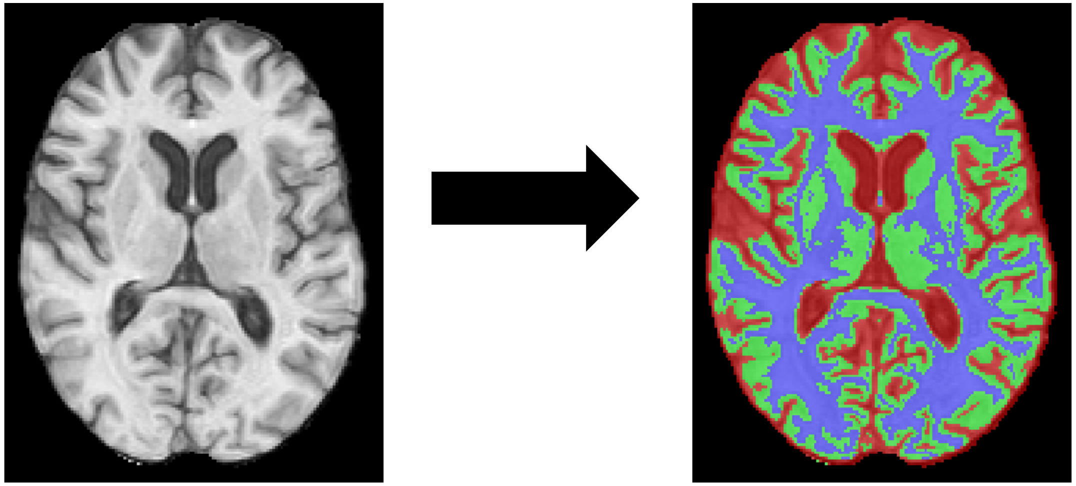
- Want to segment brain MRI into 3 tissue classes:
  - cerebro-spinal fluid (csf)
  - gray matter
  - white matter

EM in practice



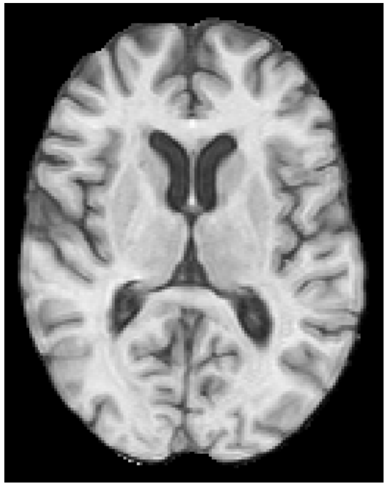
- Initialize $\boldsymbol{y_i}$ by assigning lowest 1/3 of intensities to csf, middle 1/3 to gray matter, and highest 1/3 to white matter

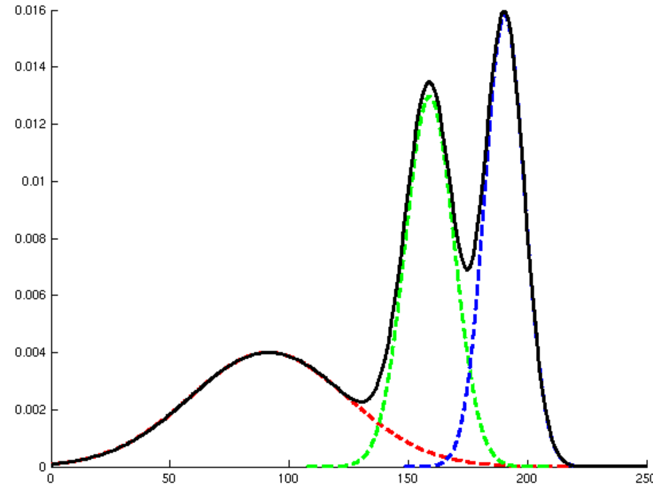
EM in practice



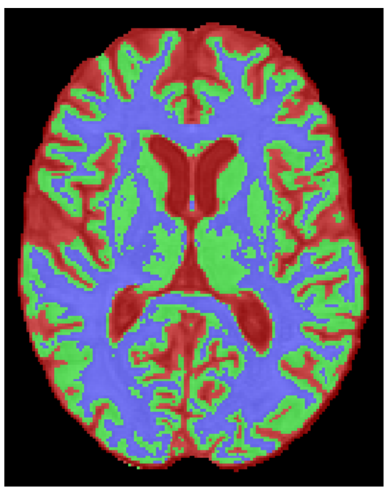
- Initialize y_i by assigning lowest 1/3 of intensities to csf, middle 1/3 to gray matter, and highest 1/3 to white matter

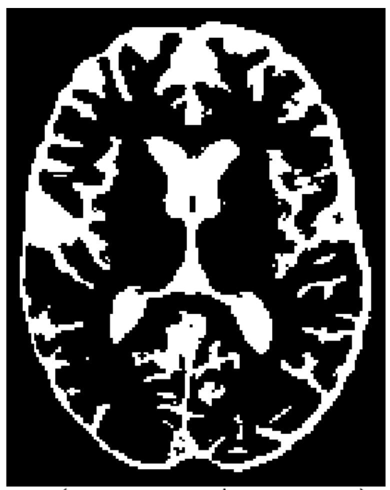
EM in practice : M-step





$p(\mathcal{X}|\mathbf{\Theta})$



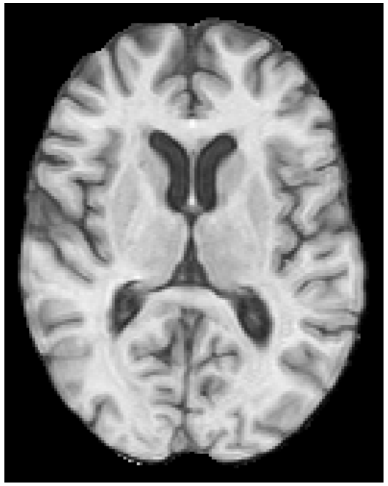


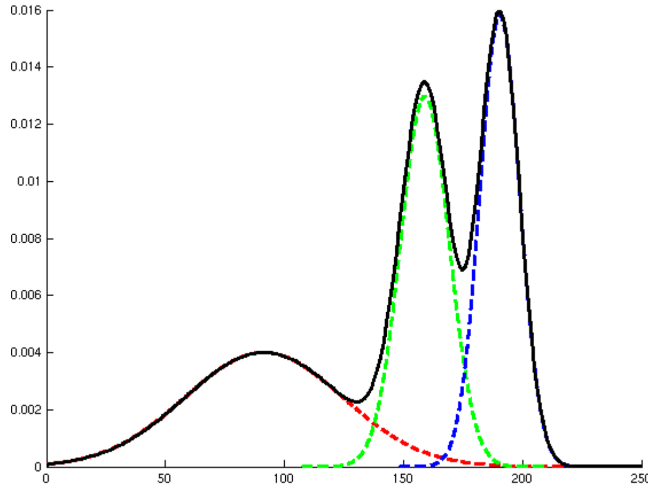
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

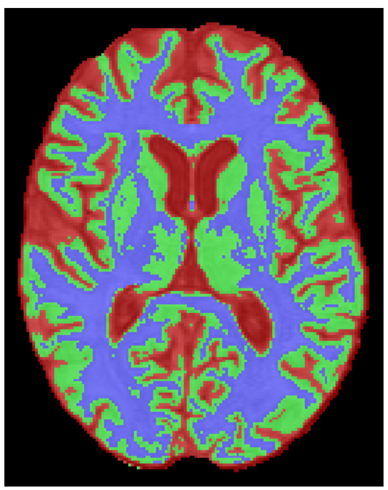
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice : E-step





$p(\mathcal{X}|\mathbf{\Theta})$

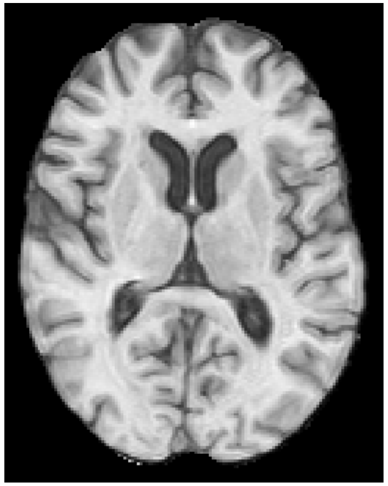


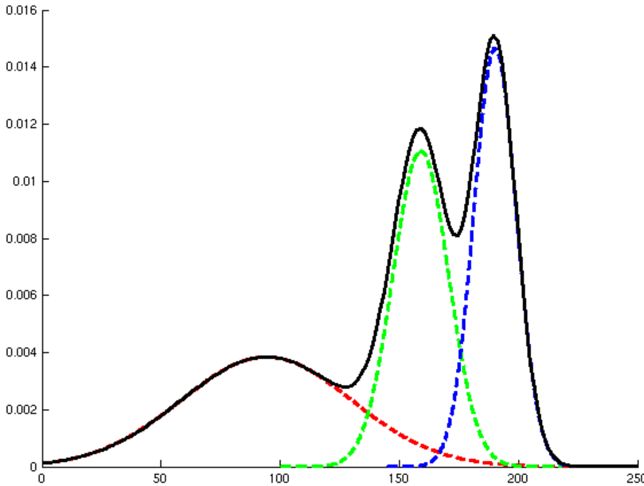
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

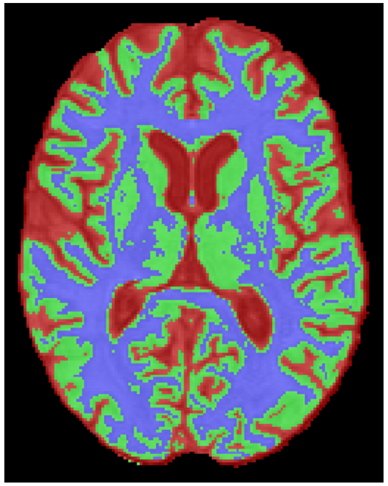
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice : M-step





$p(\mathcal{X}|\mathbf{\Theta})$

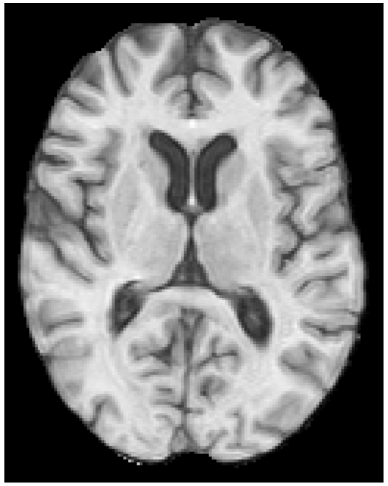


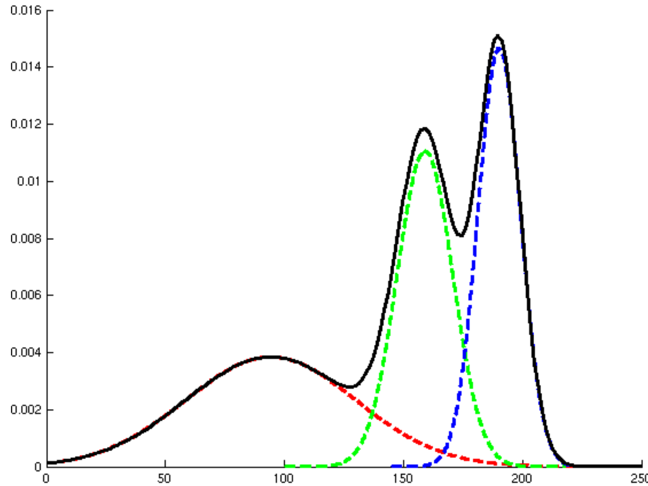
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

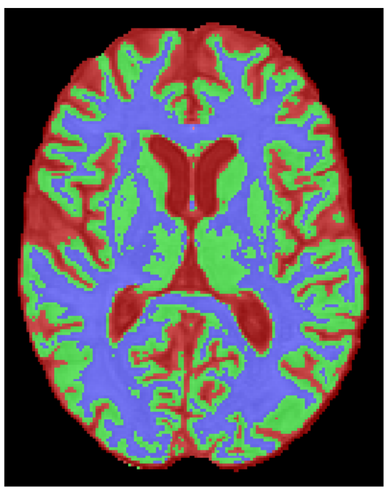
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice : E-step





$p(\mathcal{X}|\mathbf{\Theta})$

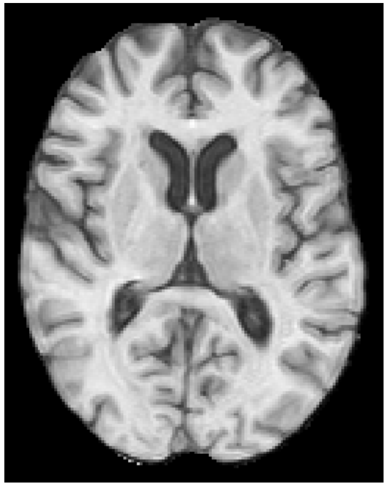


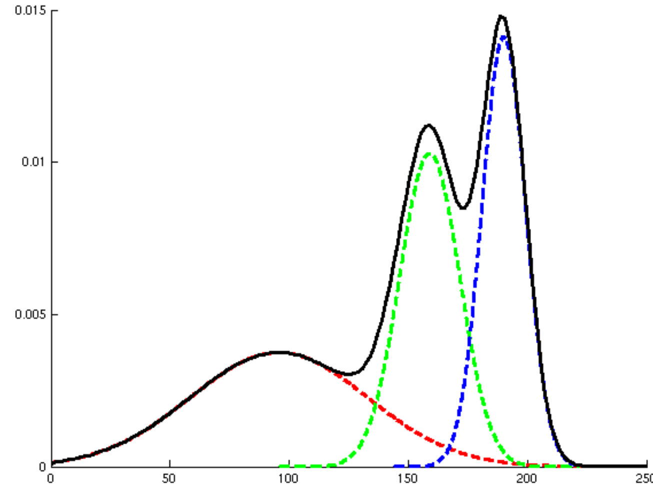
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

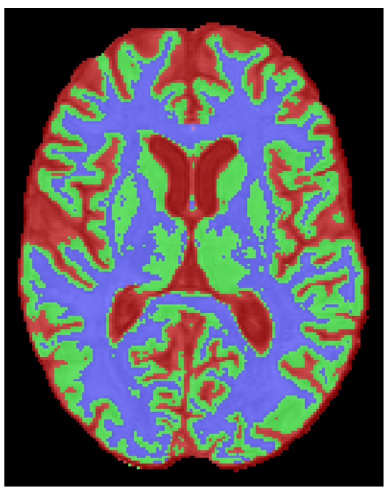
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice : M-step





$p(\mathcal{X}|\mathbf{\Theta})$

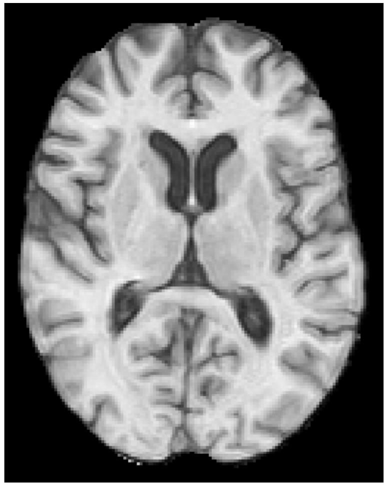


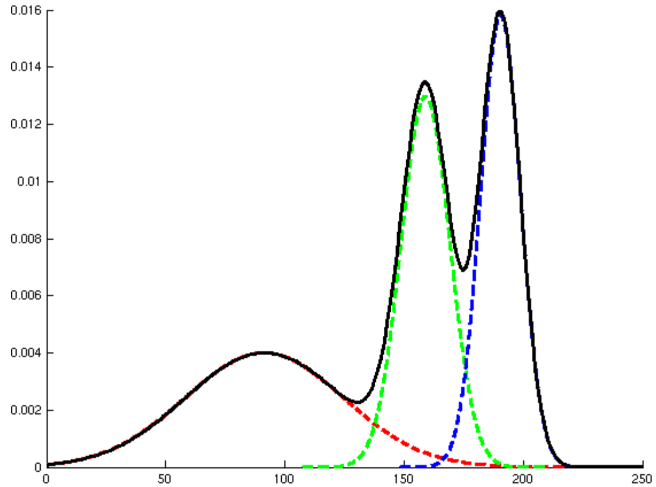
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

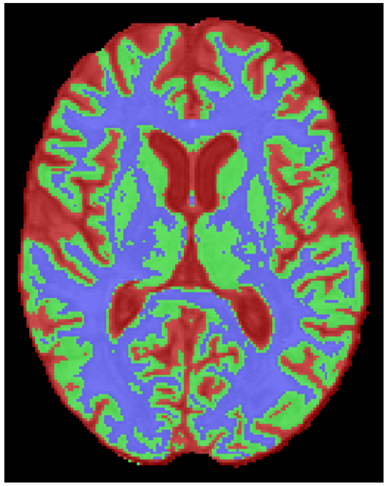
$p(y_i = 3|x_i, \mathbf{\Theta})$

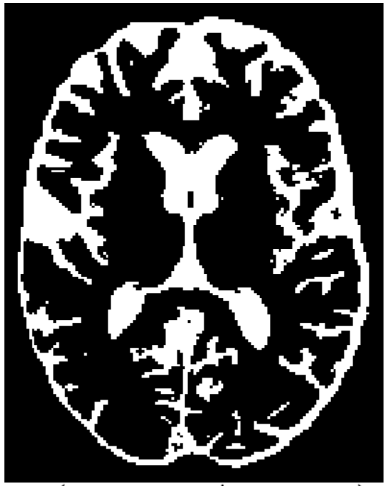
EM in practice





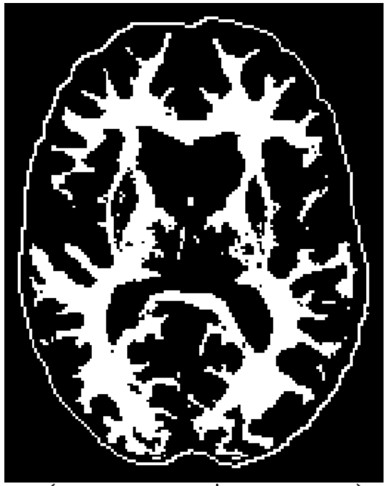
$p(\mathcal{X}|\mathbf{\Theta})$





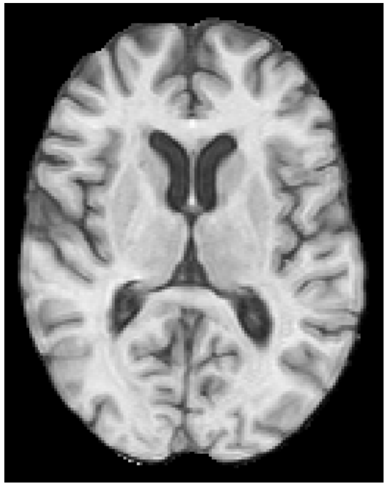
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$



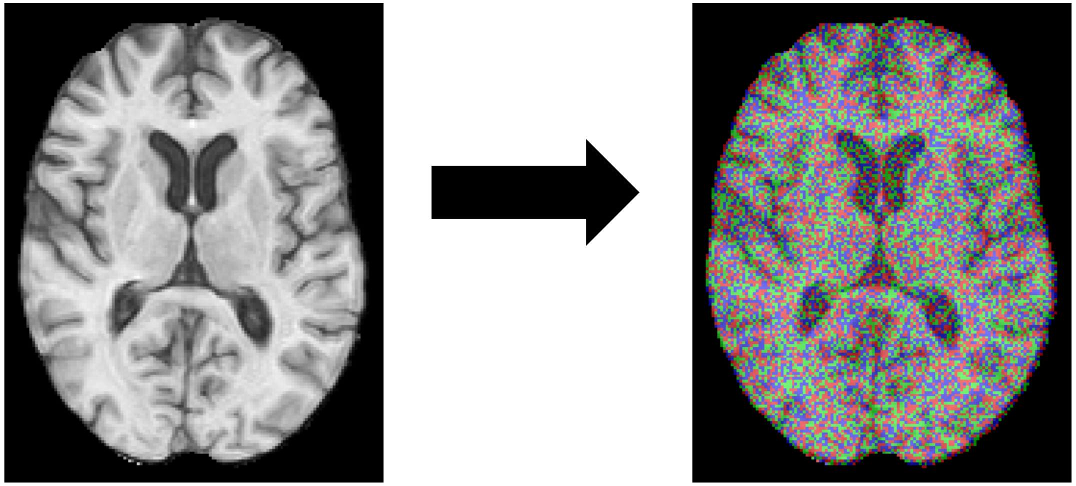
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice



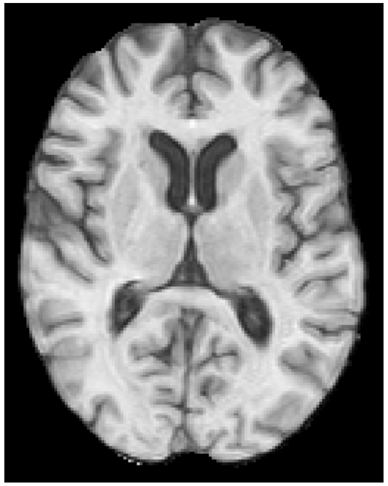
- What if we start with a random initialization?

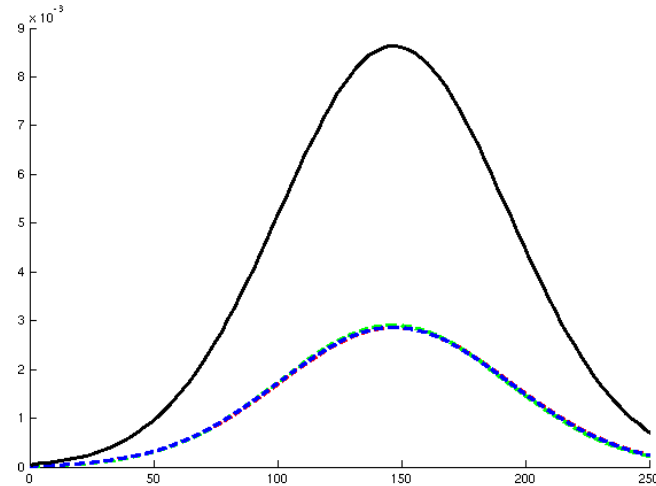
EM in practice



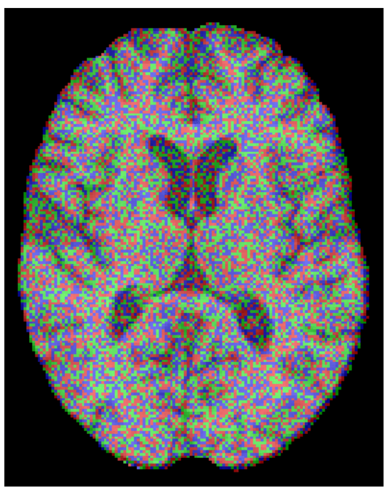
- What if we start with a random initialization?

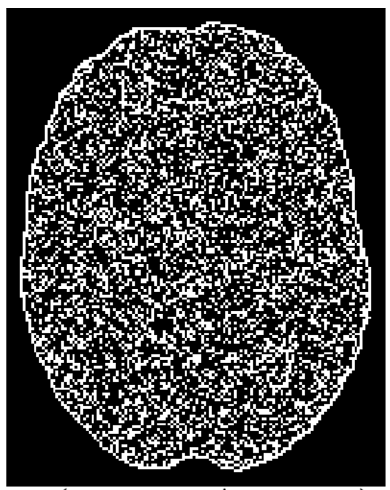
EM in practice



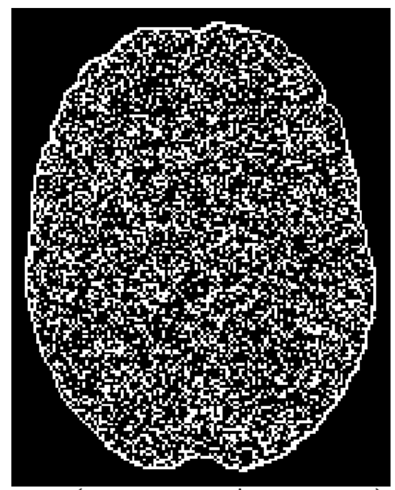


$p(\mathcal{X}|\mathbf{\Theta})$

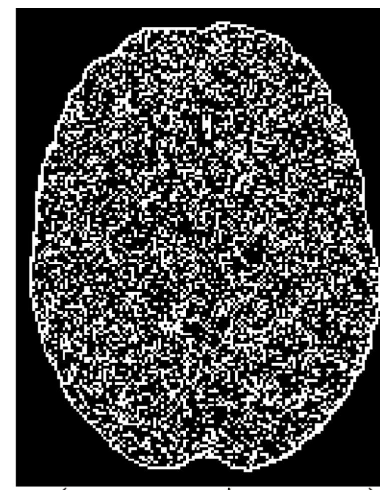




$p(y_i = 1|x_i, \mathbf{\Theta})$

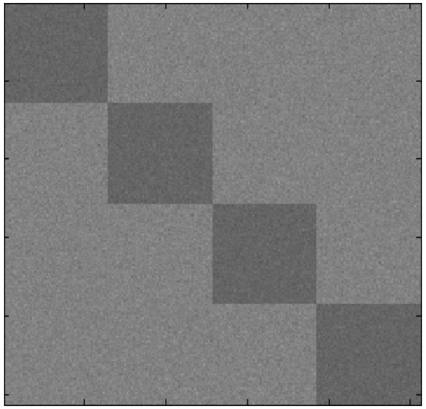


$p(y_i = 2|x_i, \mathbf{\Theta})$



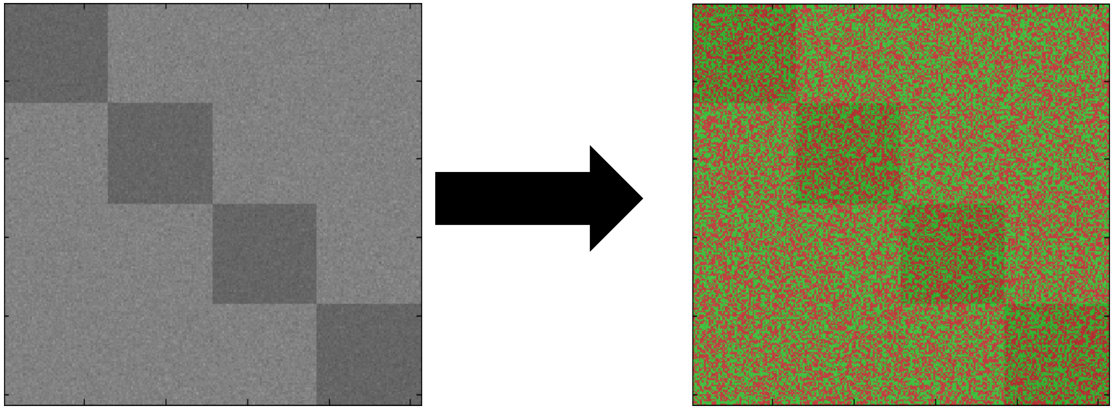
$p(y_i = 3|x_i, \mathbf{\Theta})$

EM in practice



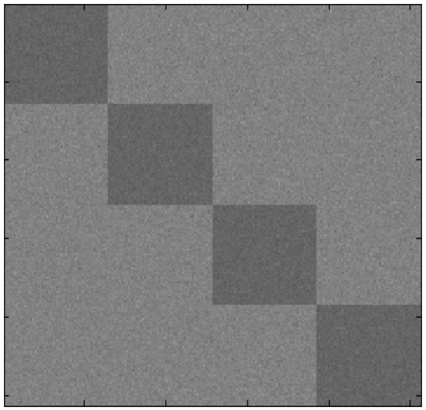
- Assume $\boldsymbol{K} = 2$
- Random initialization

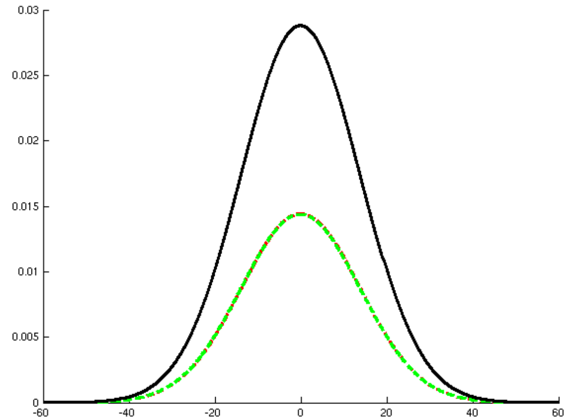
EM in practice



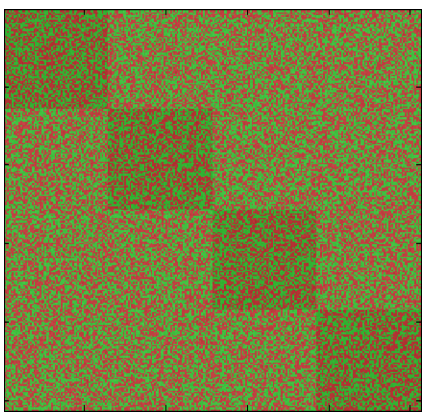
- Assume $\boldsymbol{K} = 2$
- Random initialization

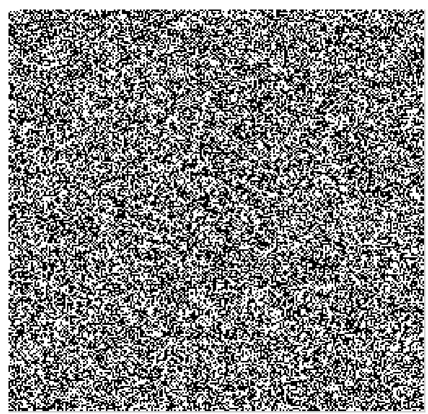
EM in practice



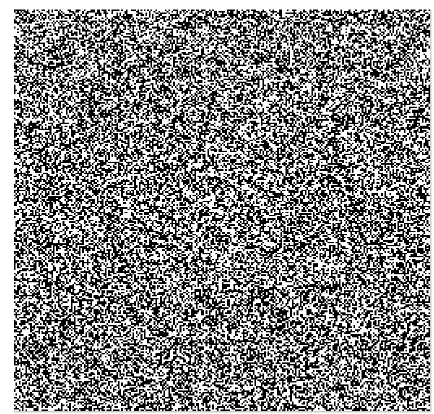


$p(\mathcal{X}|\boldsymbol{\Theta})$



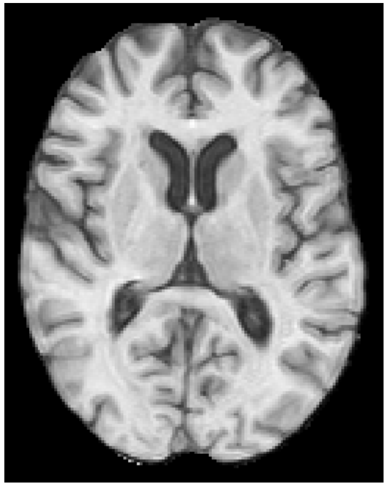


$p(y_i = 1|x_i, \boldsymbol{\Theta})$



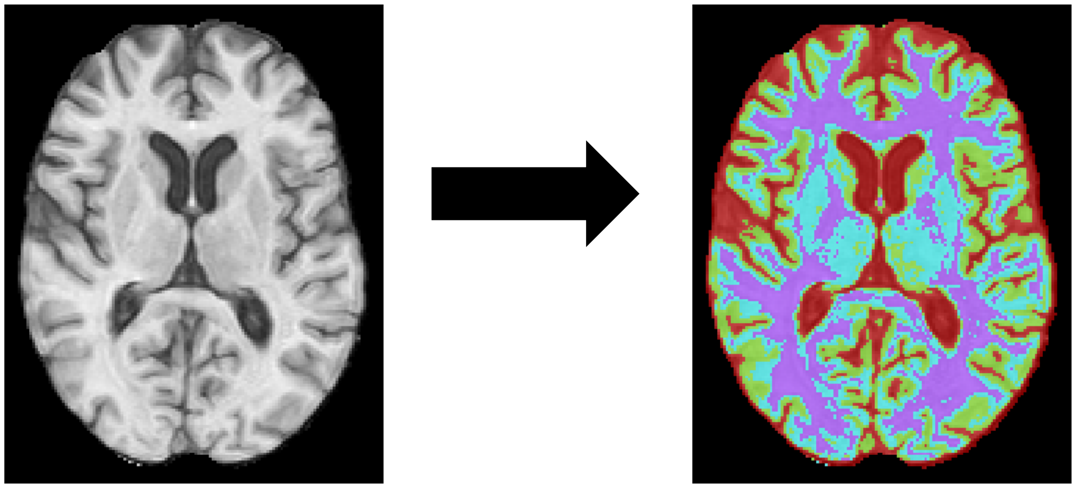
$p(y_i = 2|x_i, \boldsymbol{\Theta})$

EM in practice



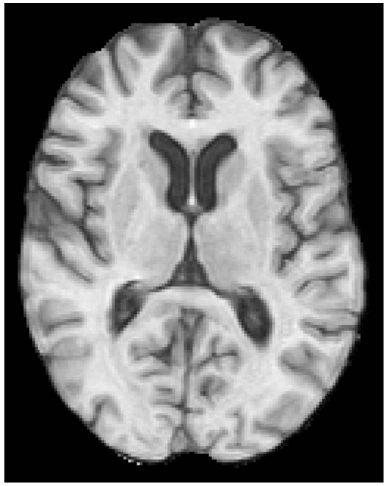
- What if we set $K = 4$?
- Initialize by setting lowest 25% of intensities to class 1, etc…

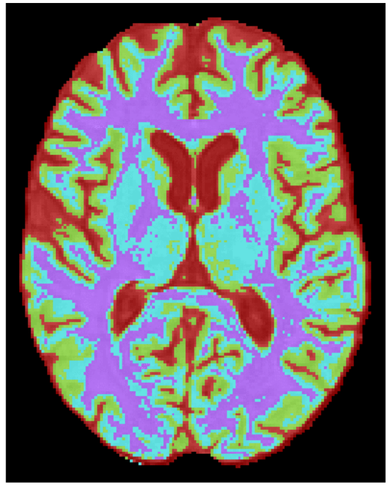
EM in practice

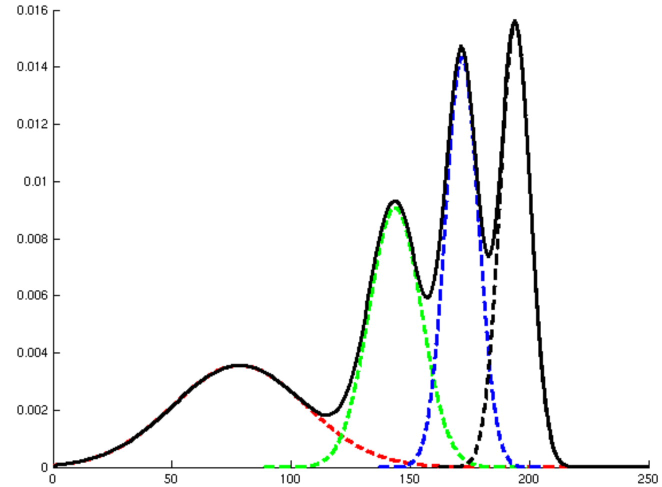


- What if we set $\boldsymbol{K} = 4$?
- Initialize by setting lowest 25% of intensities to class 1, etc…

EM in practice







$p(\mathcal{X}|\mathbf{\Theta})$

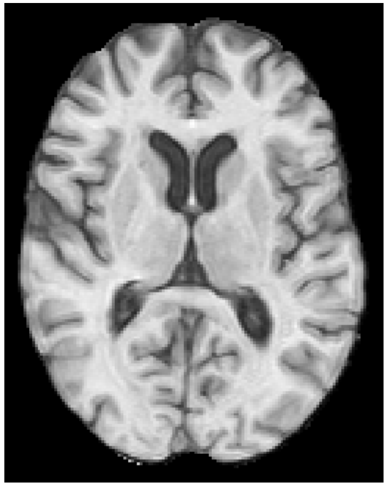
$p(y_i = 1|x_i, \mathbf{\Theta})$

$p(y_i = 2|x_i, \mathbf{\Theta})$

$p(y_i = 3|x_i, \mathbf{\Theta})$

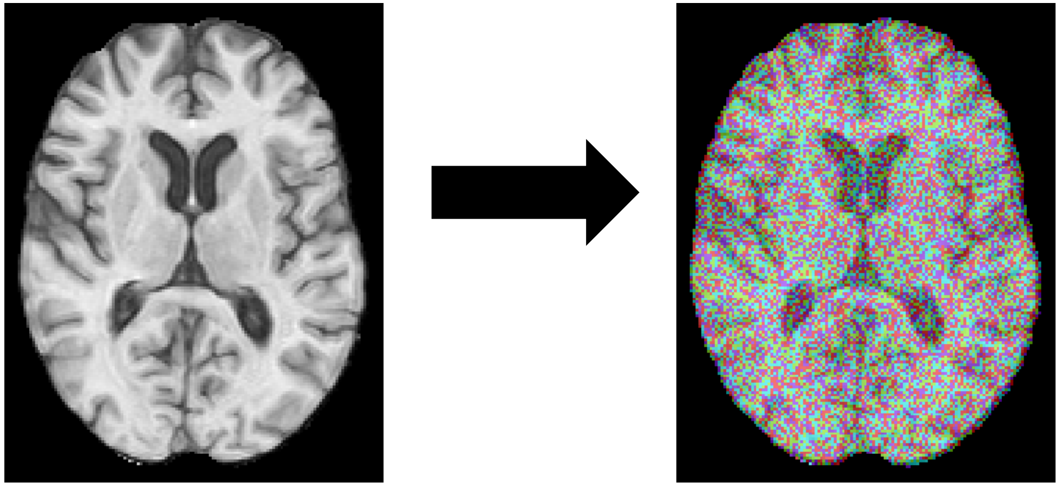
$p(y_i = 4|x_i, \mathbf{\Theta})$

EM in practice



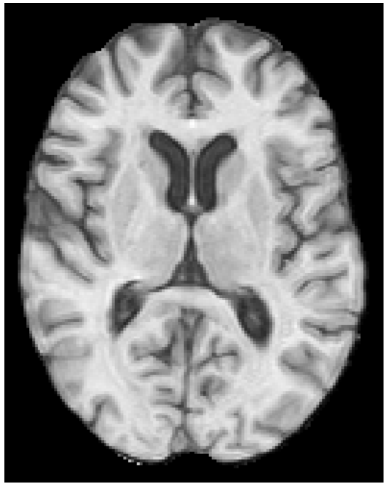
- What if we set $\boldsymbol{K} = 4$ and initialize randomly?

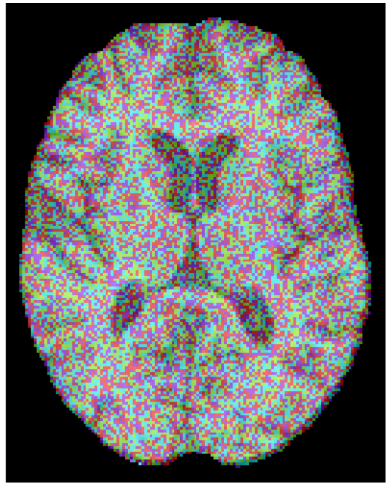
EM in practice

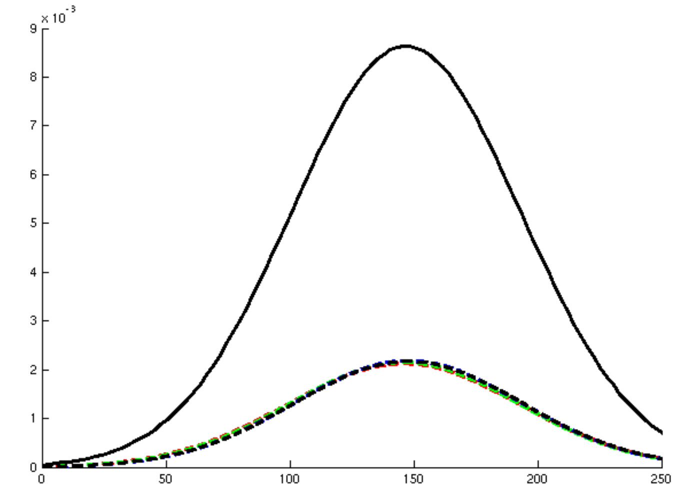


- What if we set $K = 4$ and initialize randomly?

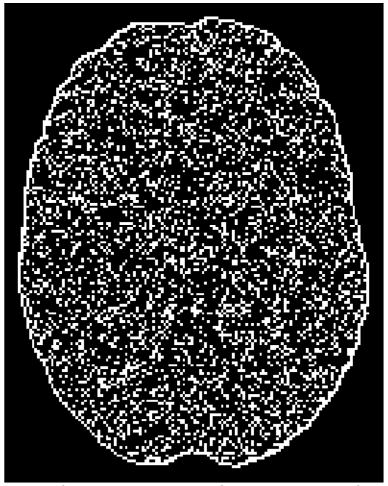
EM in practice



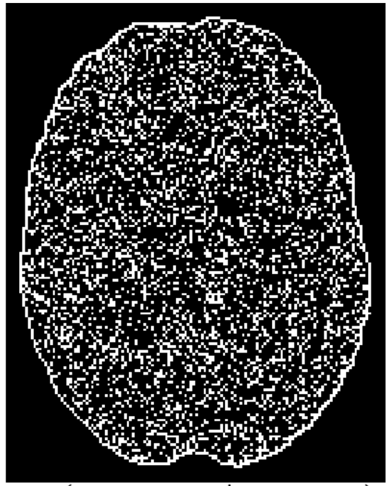




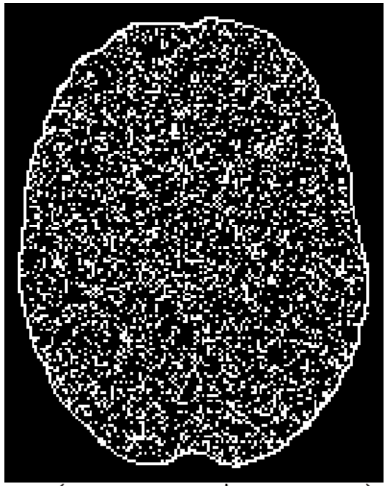
$p(\mathcal{X}|\mathbf{\Theta})$



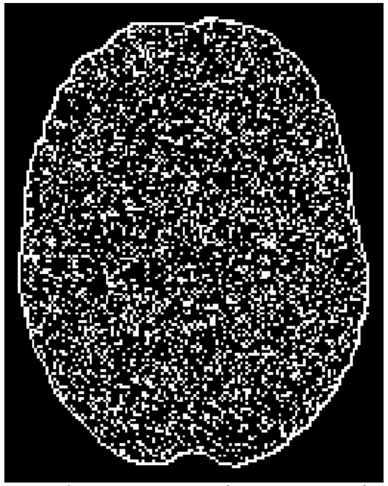
$p(y_i = 1|x_i, \mathbf{\Theta})$



$p(y_i = 2|x_i, \mathbf{\Theta})$

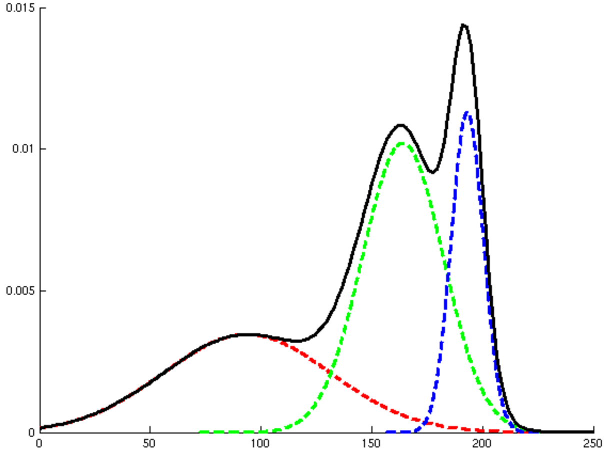


$p(y_i = 3|x_i, \mathbf{\Theta})$

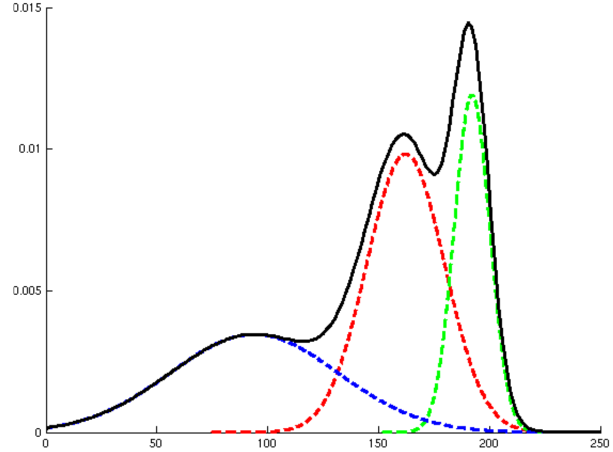


$p(y_i = 4|x_i, \mathbf{\Theta})$

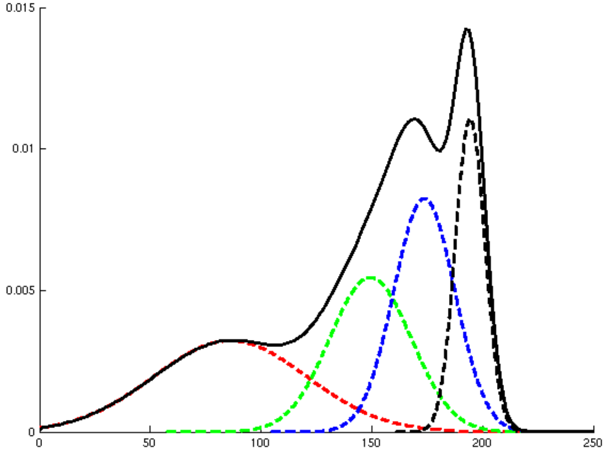
Effect of K and initialization



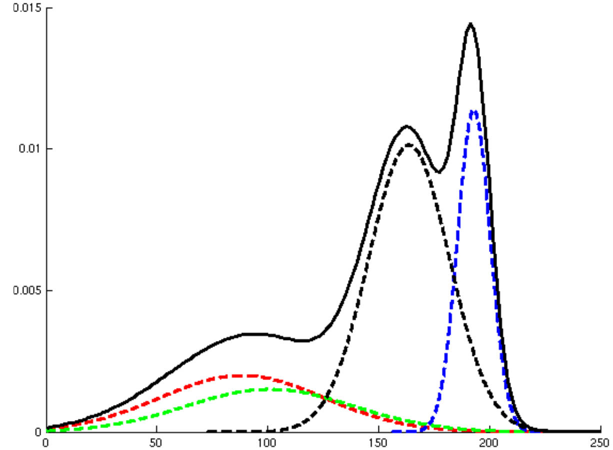
K = 3



K = 3, random



K = 4



K = 4, random

Expectation Maximization

- Iterate until convergence – guaranteed to converge to at least a local maximum
- More details on the derivation of the update equations (and EM in general) can be found here: http://www.icsi.berkeley.edu/ftp/global/pub/techreports/1997/tr-97-021.pdf

GMMs / Expectation Maximization

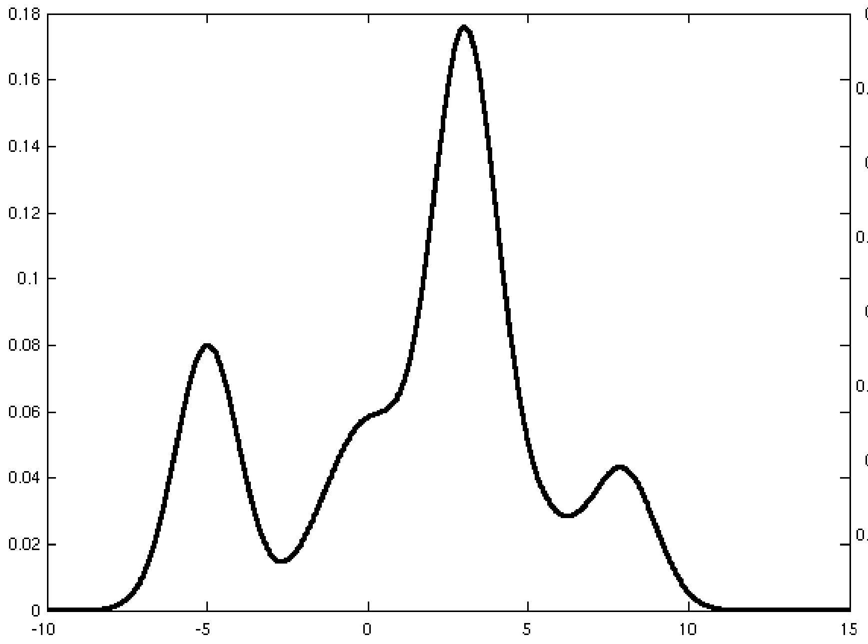
- GMM parameters provide multimodal estimate of a probability density from a set of samples
- Hidden parameters, $\boldsymbol{y_i}$, can be treated as class labels if used in a segmentation or clustering framework and provide likelihood of class $\boldsymbol{k}$ at each sample, where class $\boldsymbol{k}$ is parametrized by the $\boldsymbol{k^{th}}$ Gaussian component of our GMM

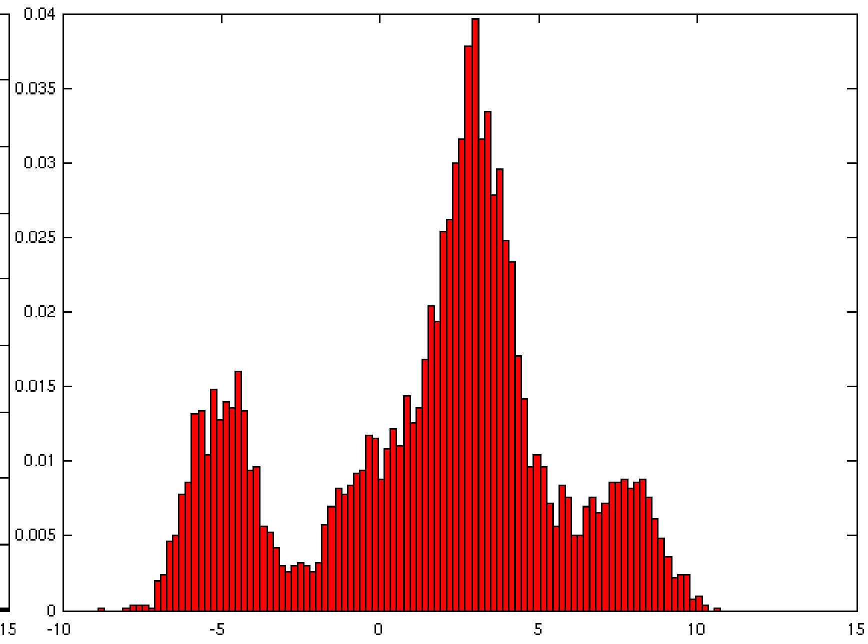
GMMs

- Drawbacks
  - Need to know K or have some way to determine K
  - Final distribution depends on initialization of model parameters
  - Cannot approximate all arbitrary distributions with limited number of Gaussians
  - May take long time to converge
- Advantages
  - Parametric : still have relatively compact representation
  - Can approximate many multimodal distributions reasonably well with relatively small number of Gaussians
  - Relatively easy to implement

Non-parametric densities

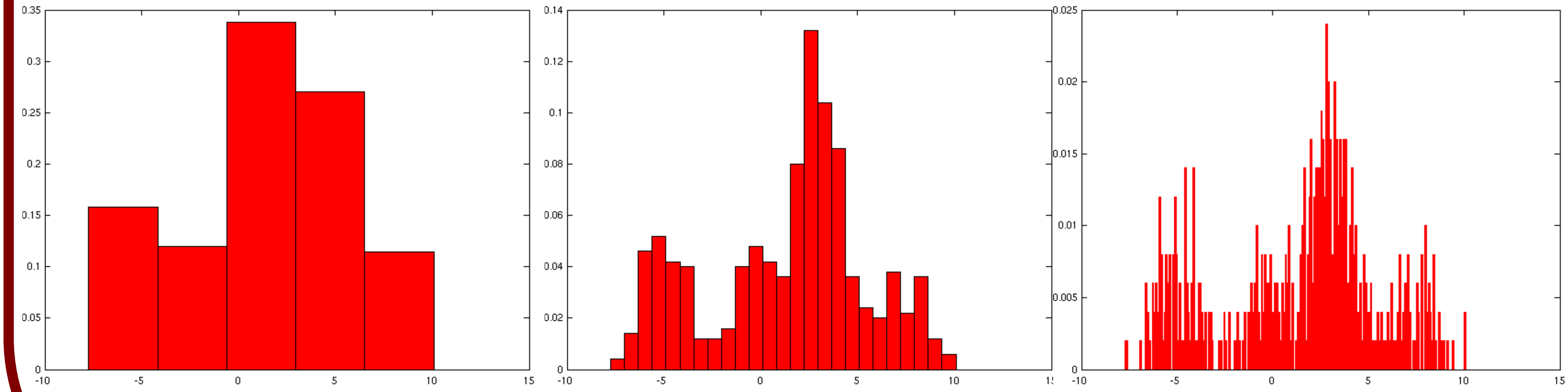
- Non-parametric representations make no assumptions about the form of underlying density





Histogram

- Simple non-parametric representation
- Bins too large : loss of resolution
- Bins too small : zero probability “holes” in density



Kernel Density Estimation

- Method of building non-parametric representations of densities from data samples
- Densities represented as expected value of “smoothed” samples
- Also known as “Parzen Window” method

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K(x - x_i)$$

Kernel Density Estimation

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K(x - x_i)$$

- K(x) is kernel (or window) function

$$\int K(x) = 1$$

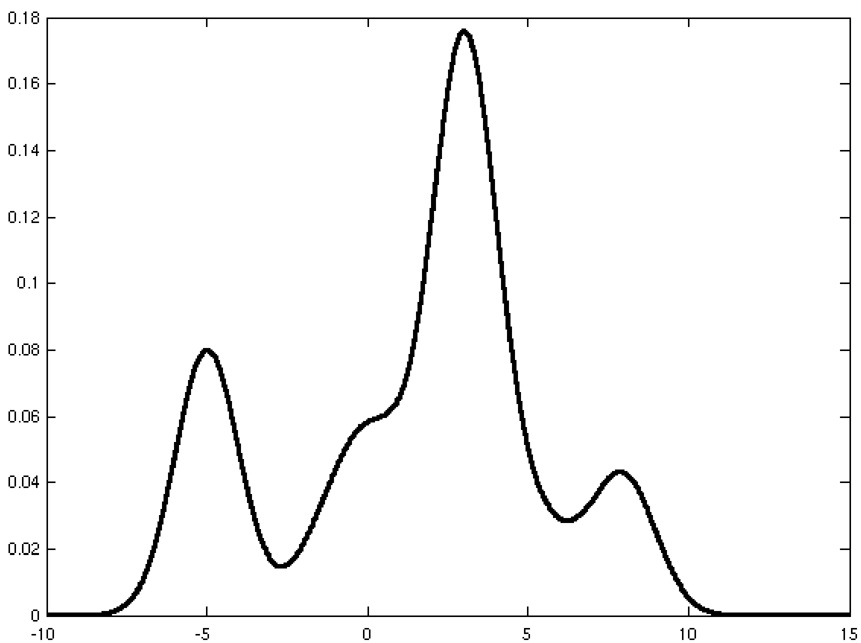
Kernel Density Estimation

- Gaussian is often used as smoothing kernel

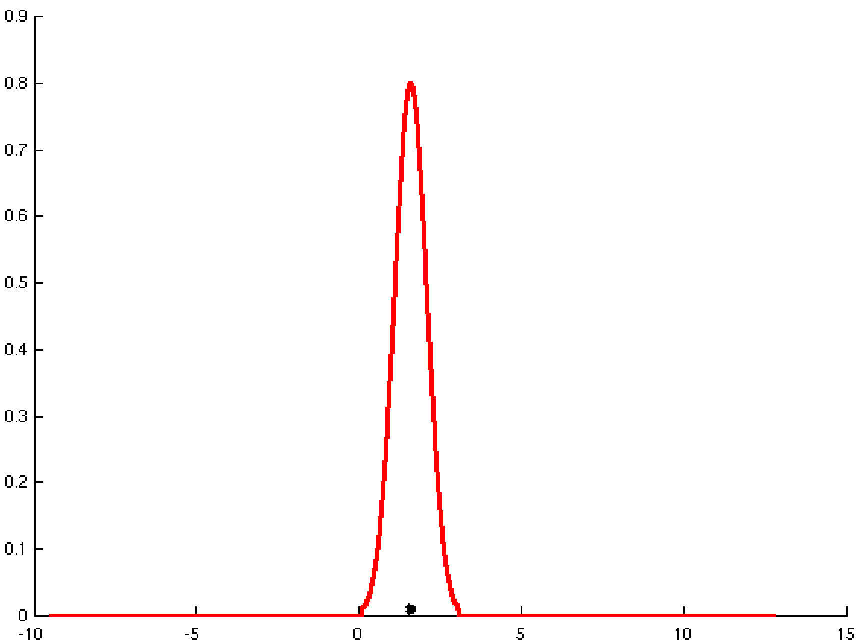
$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{N}(\mu = x_i, \sigma)$$

- Density represented as sum of Gaussians, where each Gaussian represents a data sample and has variance σ^2 and mean value equal to the value of the data sample.

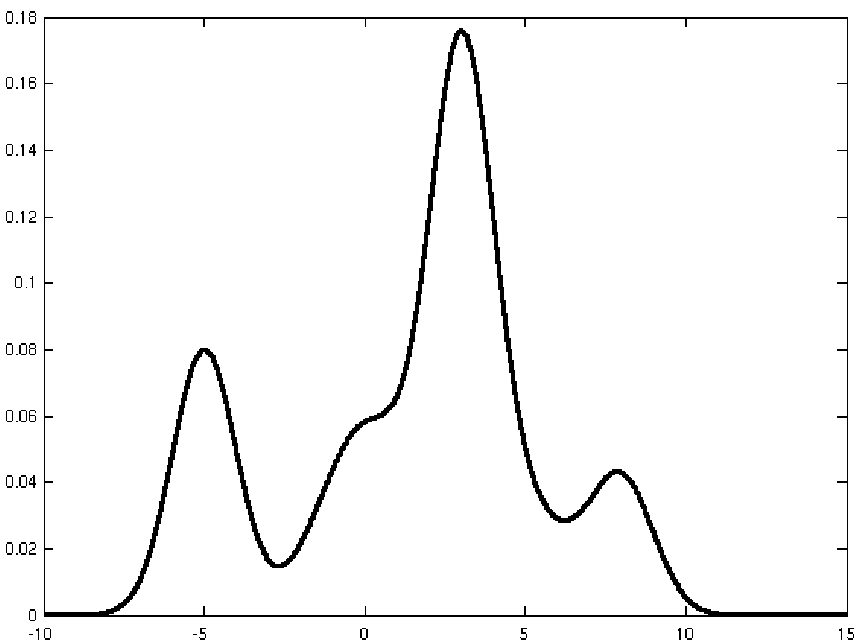
Parzen Windows



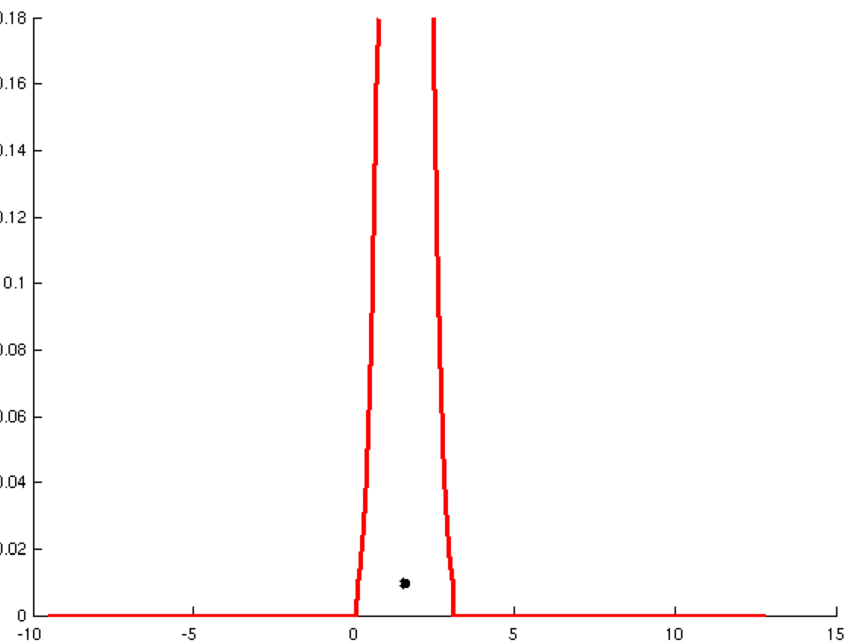
Actual Distribution



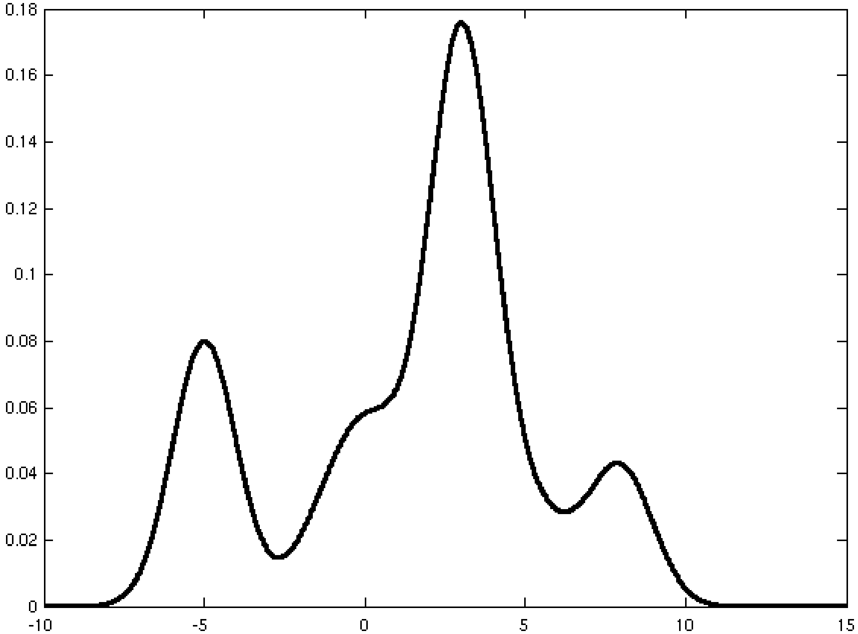
Parzen Windows



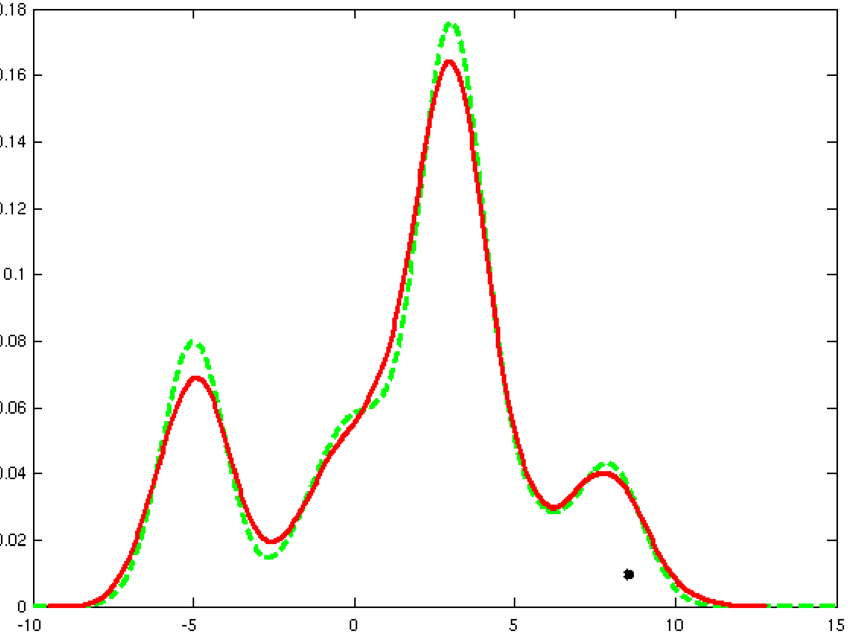
Actual Distribution



Parzen Windows



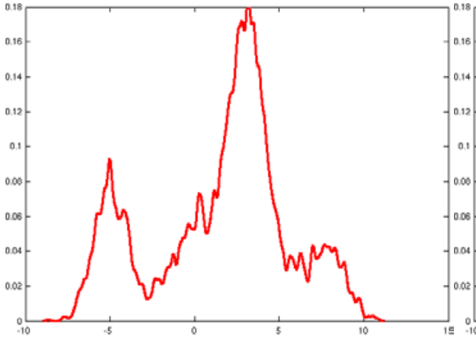
Actual Distribution



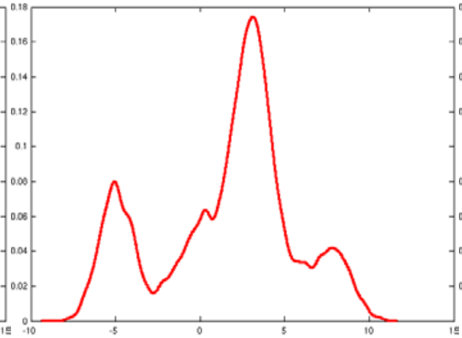
Parzen Windows

- Require bandwidth parameter (covariance of Gaussian kernel)
  - Determines “smoothness” of representation
  - Similar to bin width selection
  - Choice of kernel bandwidth generally based on # data samples and variance of data
  - Several methods exist to select bandwidth based on available sample data

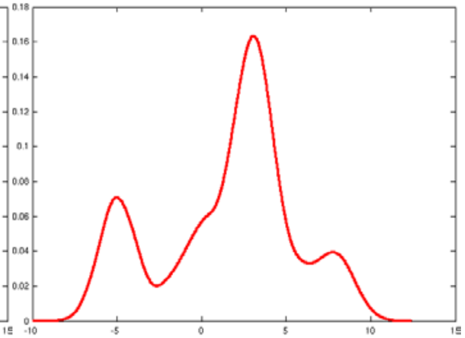
Kernel Bandwidth



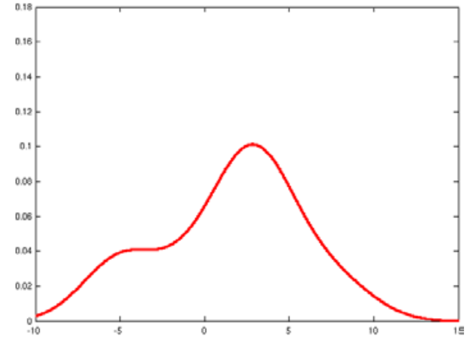
N=5000,σ=0.1



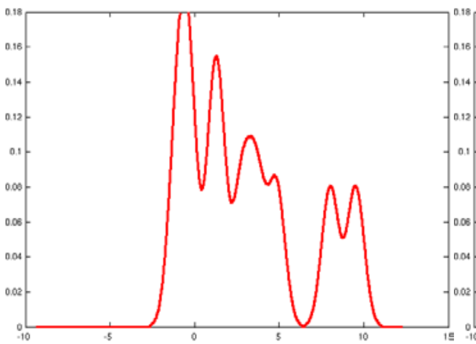
N=5000,σ=0.25



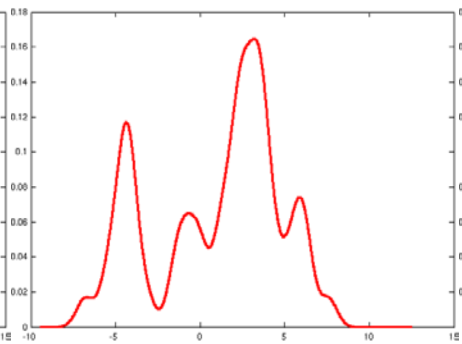
N=5000,σ=0.5



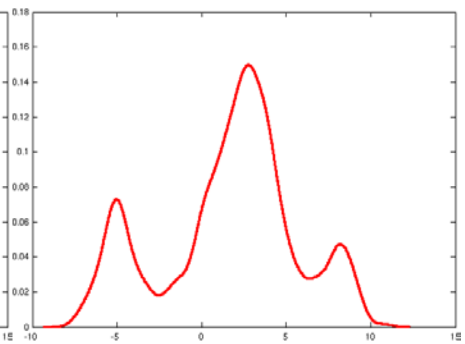
N=5000,σ=2



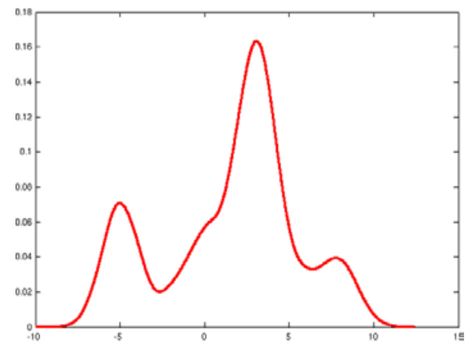
N=10,σ=0.5



N=100,σ=0.5

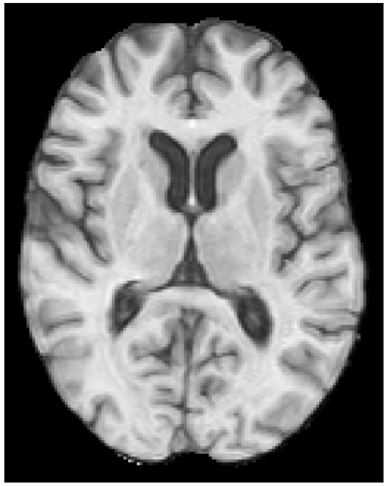


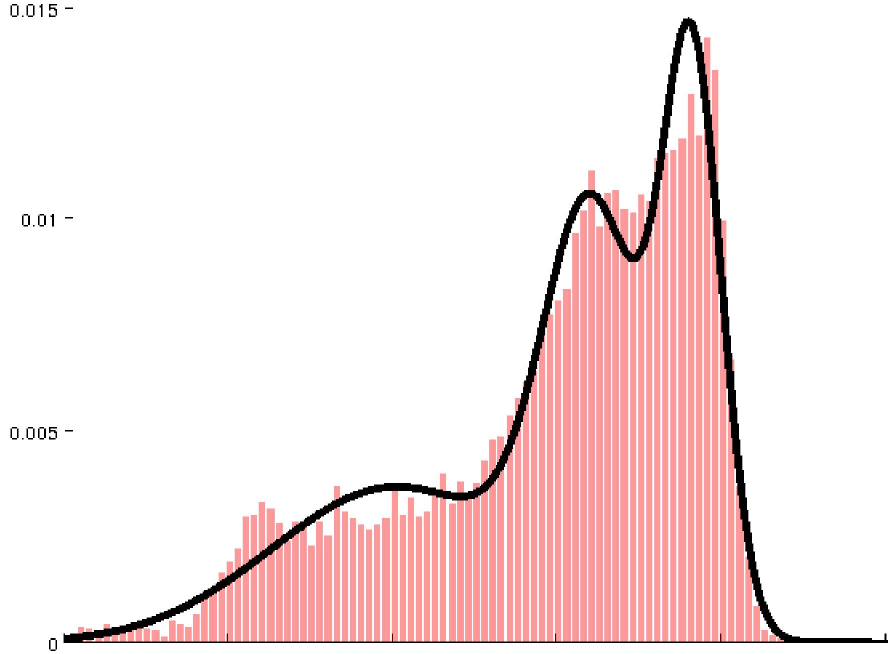
N=500,σ=0.5



N=5000,σ=0.5

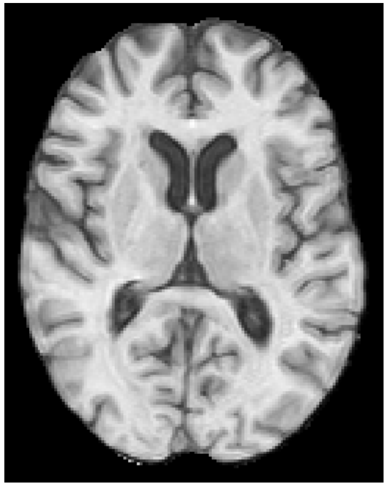
Parzen Windows

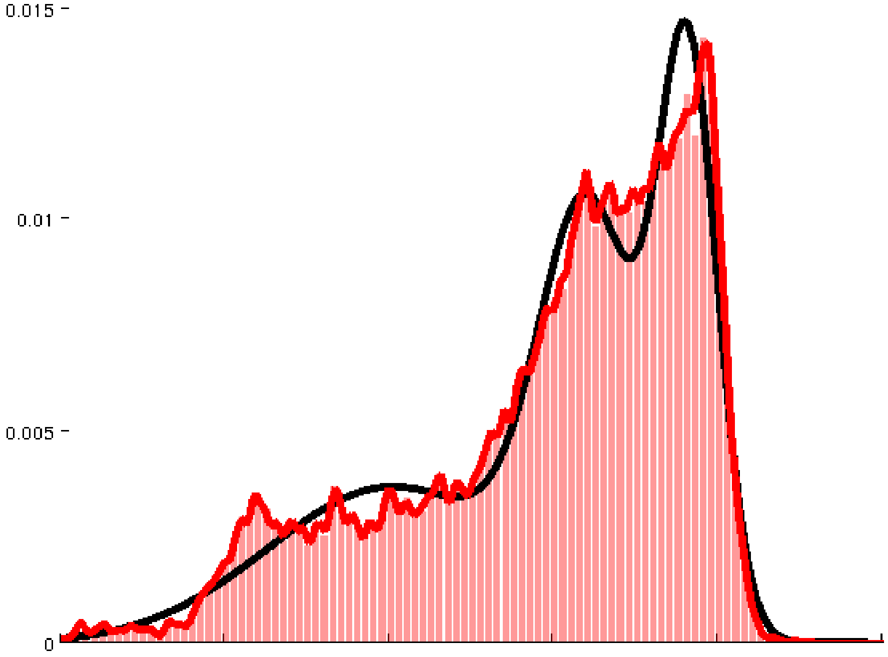




Black = 3 component GMM

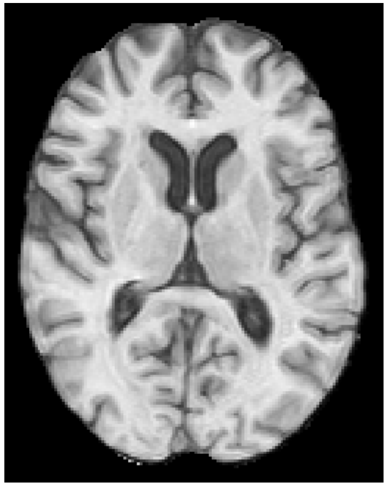
Parzen Windows

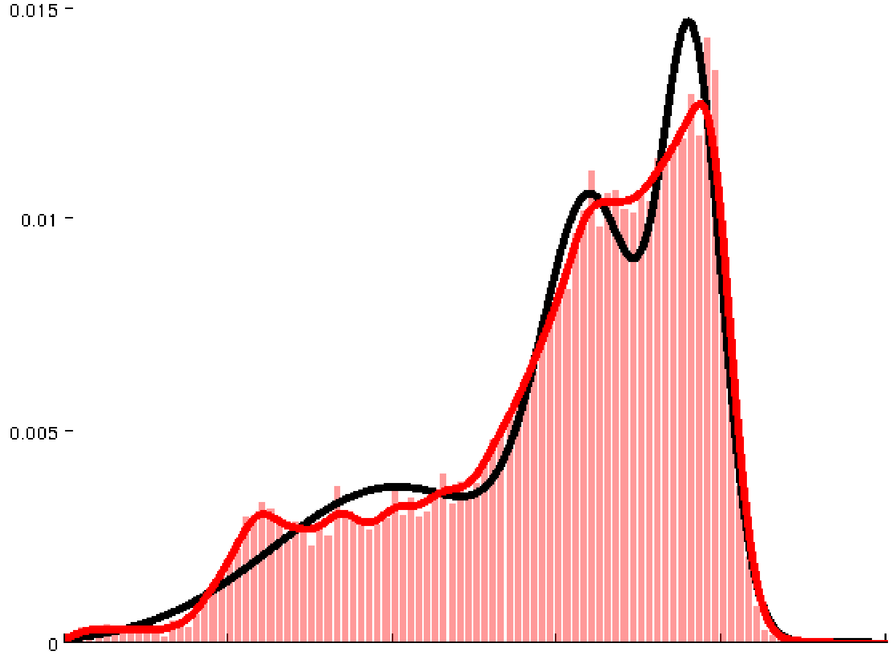




Black = 3 component GMM
Red = Parzen Window (BW = 1)

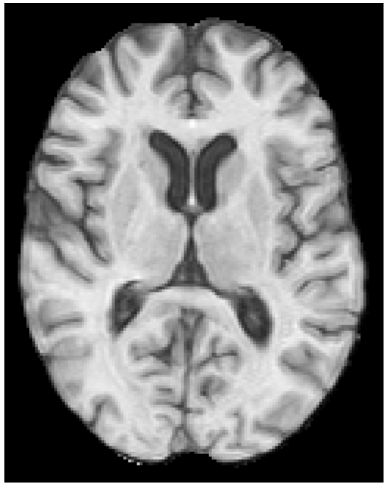
Parzen Windows

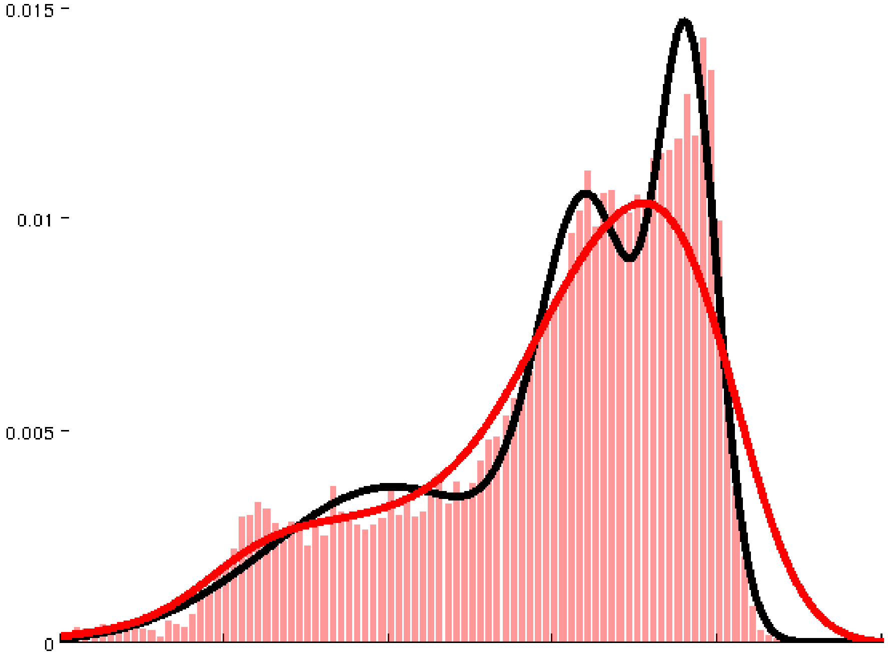




Black = 3 component GMM
Red = Parzen Window (BW = 4)

Parzen Windows





Black = 3 component GMM
Red = Parzen Window (BW = 16)

Densities using Parzen Windows

- Advantages
  - Can represent arbitrary densities
  - Does not require initialization
  - Methods exist to select appropriate kernel bandwidth based on available data samples
- Disadvantages
  - Can not represent density in a compact form
  - Need to determine kernel bandwidth
  - Non-trivial implementation details, especially for higher-dimensional densities

Summary

- GMMs
  - Can approximate multi-modal densities in a relatively compact form.
  - Given ***K***, EM allows us to generate an estimate of the underlying probability density of our samples.
  - For segmentation or clustering tasks, we can model our samples as a GMM and use EM to determine the underlying cluster from which each sample came.
- Parzen Windows
  - Can represent arbitrary probability densities without any initialization
  - Non-compact representation
  - More difficult to implement