# EEL 4930/ 5934

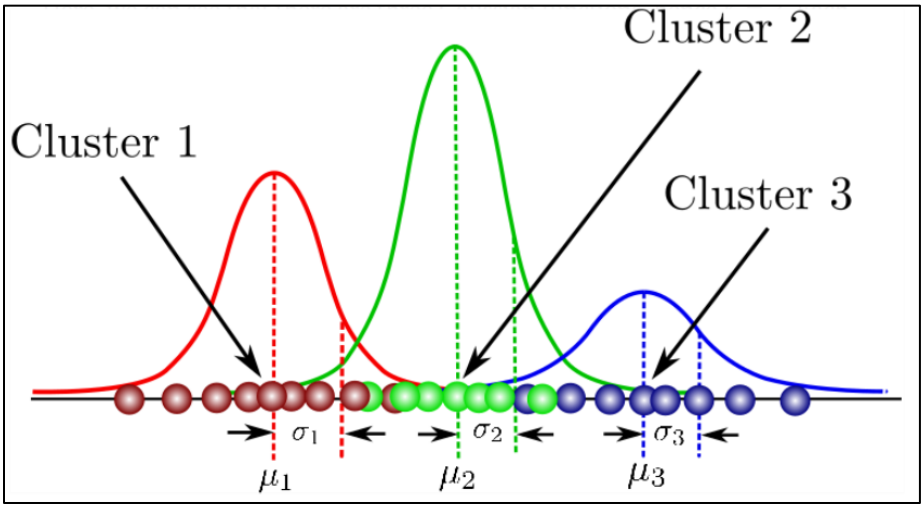
# Introduction to Biomedical Image Analysis

# Assignment – 11

# Due: 04/18/2024, Noon

**Image Segmentation via Clustering Algorithms:**

During the process of image segmentation, we look for characteristic features of the structures we are interested in extracting from the image. These features can be related to color, size, shape, and texture. Separating the objects in the image that we are interested in from objects in the background can also be accomplished using clustering algorithms. In class we went through an example using K-Means on the RGB channels of an image. For this assignment we will be looking at the Mixture of Gaussians algorithm which uses an iterative technique to estimate the parameters (mean and standard deviation) of a certain number of Gaussian distributions that are combined to mimic the distribution of the provided data. An example of this is shown below.




Taken from: https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95

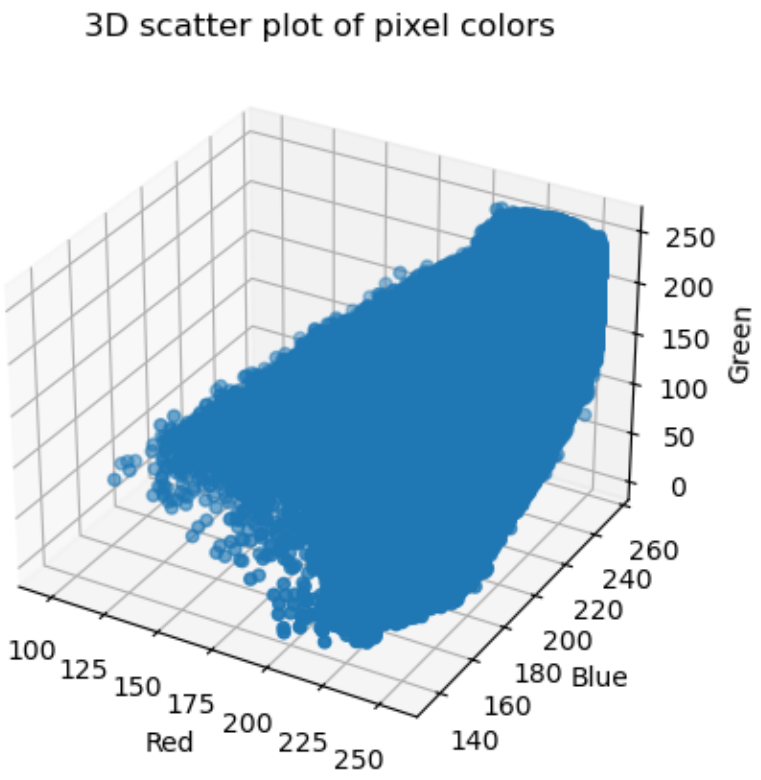
**Part 1: Preparing Image data for clustering.**

(a) Read in the image into your workspace (‘*HE_img.PNG’)*.

Accomplished in submitted code.

(b) Reshape the image matrix from (m x n x 3) to (L x 3). This puts the different color channels into columns.
Accomplished in submitted code.

(c) Generate a 3-D scatterplot of RGB values. Label each axis according to the color channel those values are derived from.




**(10 pts.)**

**Part 2: Clustering the data.**

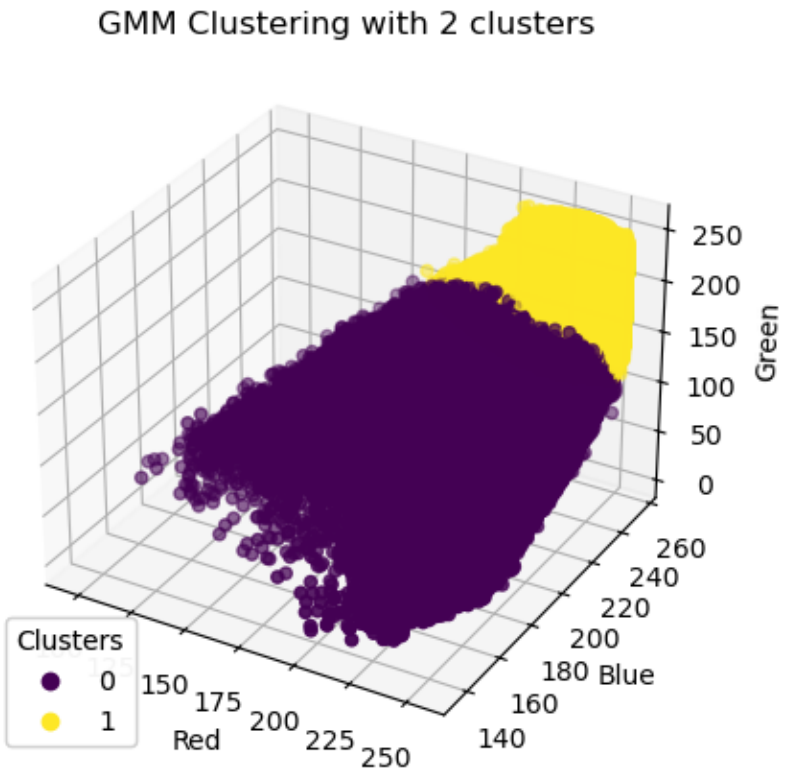
(a) The command for fitting a mixture of Gaussians in Matlab is *fitgmdist()*. Look at the documentation page by typing *fitgmdist()* in the search bar in the upper-right corner. What is the name of the algorithm used to optimize Gaussian parameter values? What is the default method used to initialize distributions?

The likelihood optimization algorithm for MATLAB's GMM function is the Expectctation-Maximization (EM) algorithm. The default method to initialize distributions is called the k-means++ algorithm.

(b) Run the following lines of code. (Replace 'reshaped_img' with the name of your 3 column matrix from Part 1).

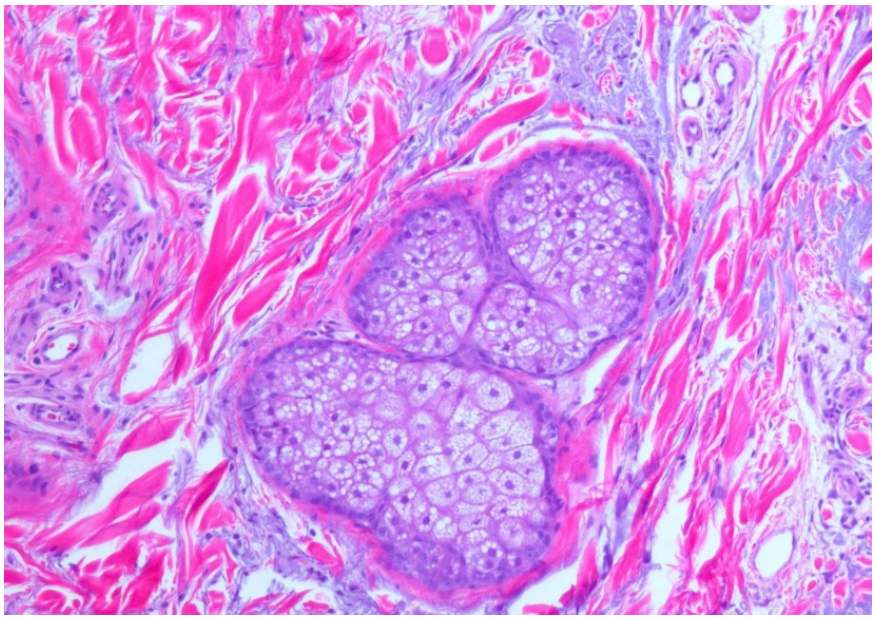
```
num_iter = 25;
num_clust = 2;

options = statset('Display','final','MaxIter',num_iter);
fitted = fitgmdist(double(reshaped_img), num_clust,'CovType','diagonal', 'Options',options);

id = cluster(fitted, double(reshaped_img));
```

I believe that code will display a figure (display being set to final) – I assume the figure would look something like this:

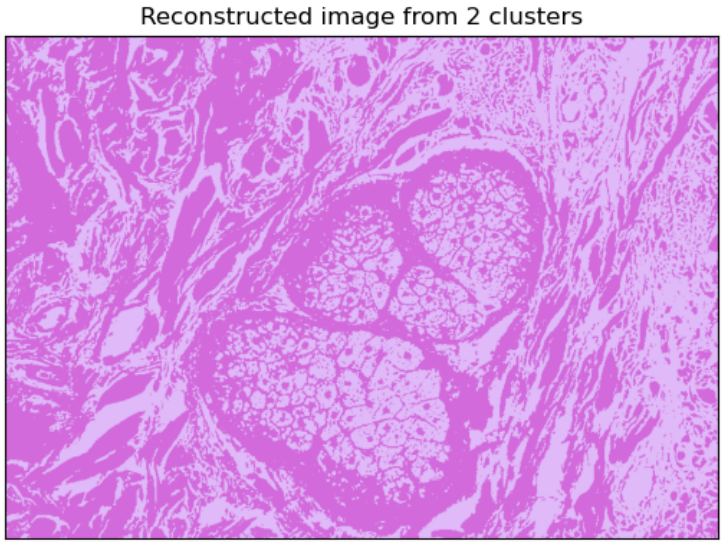



(c) Reshape the 'id' matrix to the (m x n) height and width of the original image and display using *imagesc()*. Include the resulting image and original image below.

Original Image:



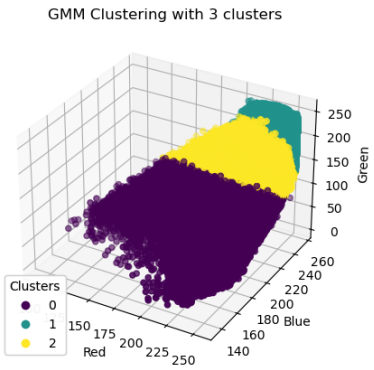
Reconstructed image:

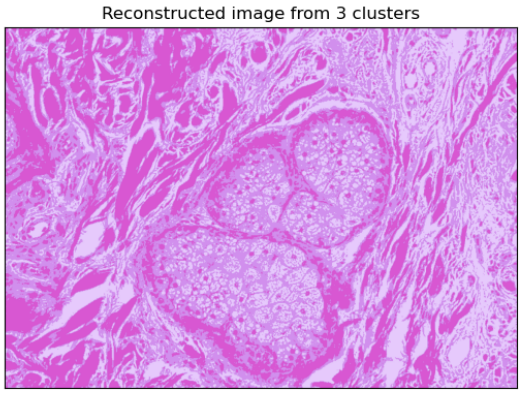



(d) Try out a handful of different values (5) for the 'num_clust' variable. How do the cluster-labeled images change as the number of clusters increases? What are the log-likelihood values for each number of clusters?

As the number of clusters increases, the images get closer and closer to the original image. With 7 clusters, the original image is quite close to the original.

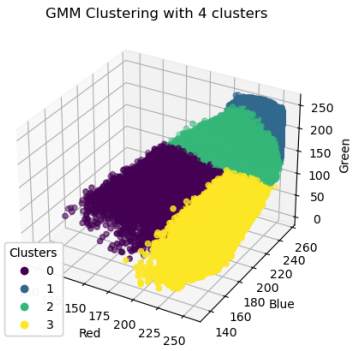
<u>3 clusters:</u>

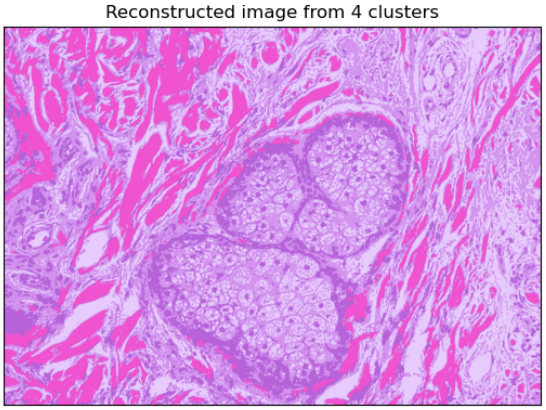



Log-likelihood: 3,003.56

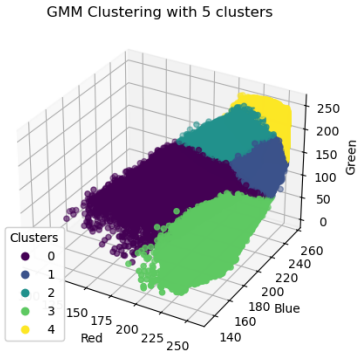
4 clusters:

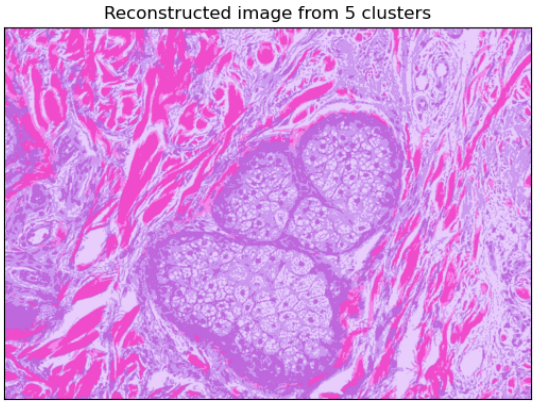



Log-likelihood: 3,534.23

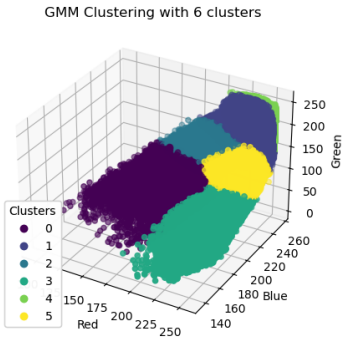
5 clusters:

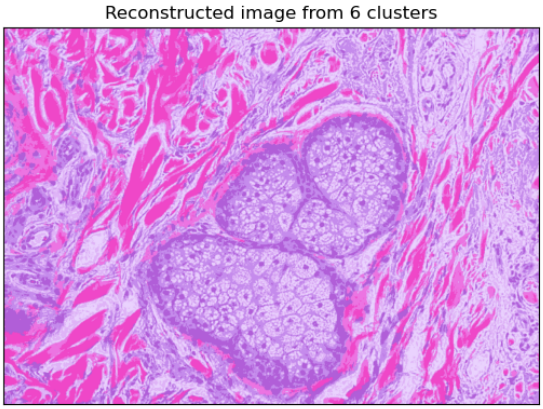



Log-likelihood: 3,920.73

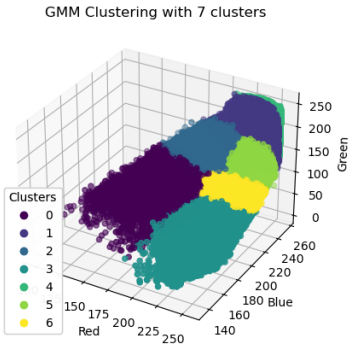
6 clusters:

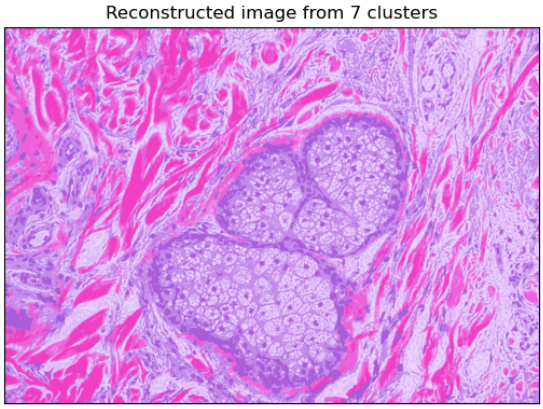



Log-likelihood: 34,578.12

7 clusters:




Log-likelihood: 64,311.87

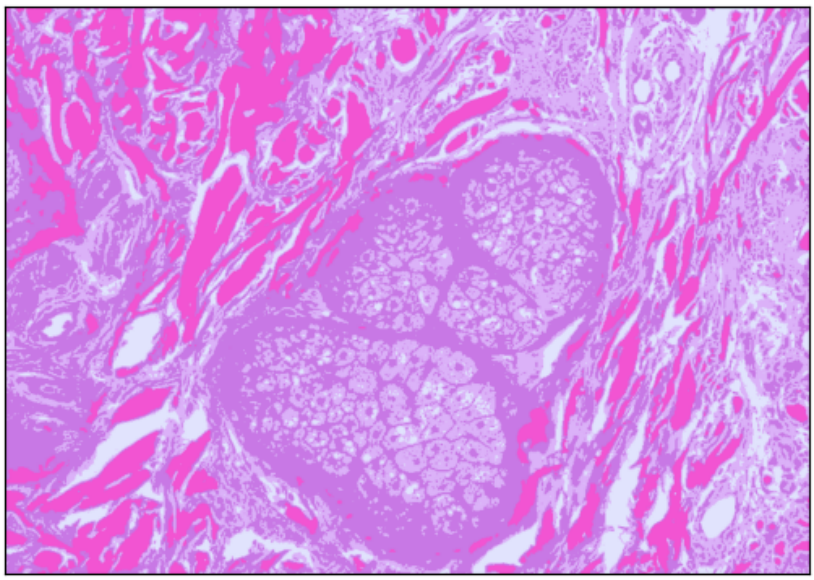
**(20 pts.)**

**Part 3: Clustering in different color spaces.**

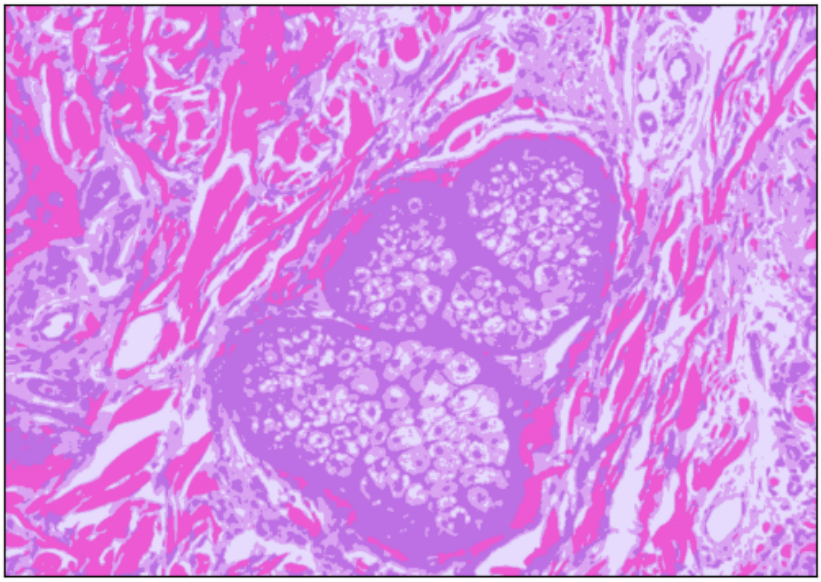
(a) Use the commands *rgb2hsv()*, *rgb2lab()*, and *rgb2ycbr()* to transform your original RGB image into the HSV, LAB, and YCbCr color spaces. Run the mixture of Gaussians clustering again using 'num_clust' equal to 4. Include the results for each color space below. How do the different color spaces compare? Which one seems to best isolate individual cell types within the image?

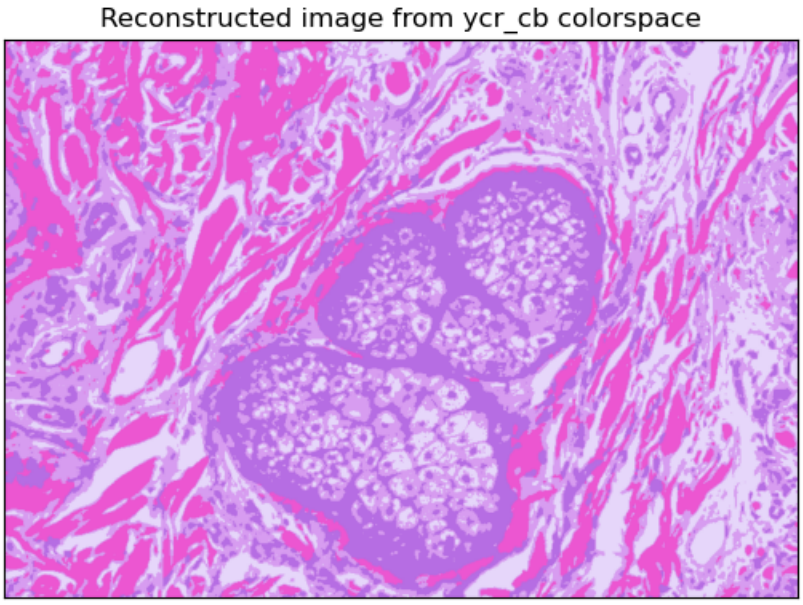
The different color space output (found below, converted back to RGB for plotting) for the GMMs are all similar – HSV has a lot more noise than the other 2, with pixels dotting the interior of most cells. YCbCr also has some noise, though a lot less. LAB colorspace has the lowest amount of noise and seems to best isolate cell types based solely on that merit, though it really isn't great at it. Almost all cells have a "nucleus" cluster, a "cytoplasm" cluster (which, combined would make for a good cell isolation), and some of the "background" cluster. If I were wanting to isolate individual cell types as the question asks, I think the cell types would need to be much more distinct in each color-space.

Reconstructed image from hsv colorspace



Reconstructed image from lab colorspace






**(20 pts.)**