Joseph Bell
ECE 271A

Homework 3 and 4 Computer Problem

Dataset 1

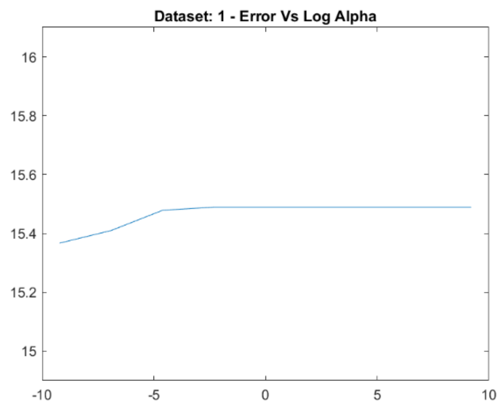

Figure 1: Dataset 1/strategy 1 percent error vs log(alpha) for predictive equation

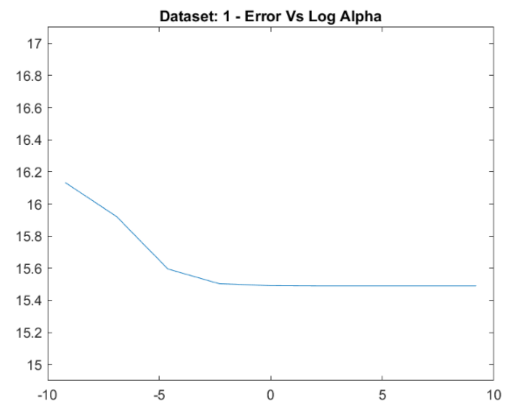

Figure 2: Dataset 1/strategy 2 percent error vs log(alpha) for predictive equation

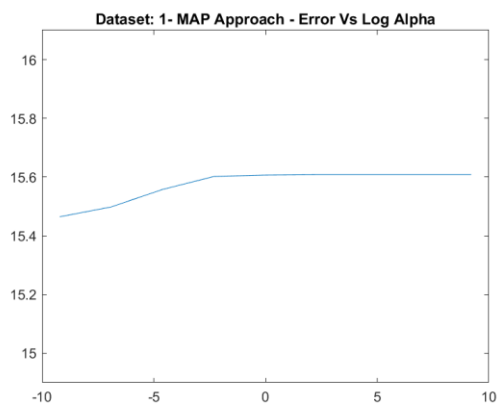

Figure 3: Dataset 1/strategy 1 percent error vs log(alpha) for MAP approach

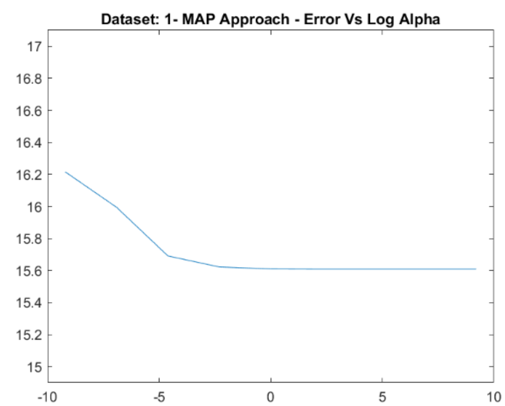

Figure 4: Dataset 1/strategy 2 percent error vs log(alpha) for MAP approach

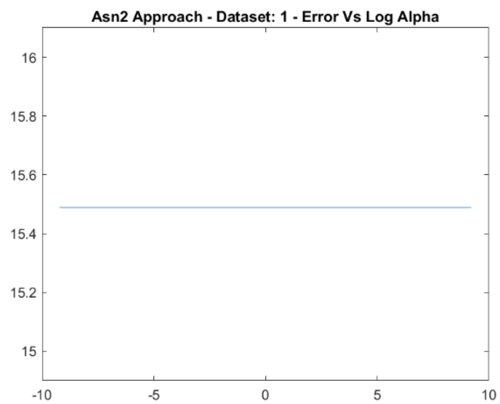

Figure 5: Dataset 1/strategy 1 percent error vs log(alpha) for ML approach

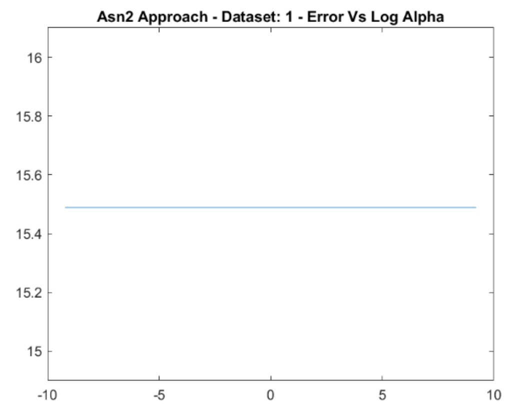

Figure 6: Dataset 1/strategy 2 percent error vs log(alpha) for ML approach

Dataset 2

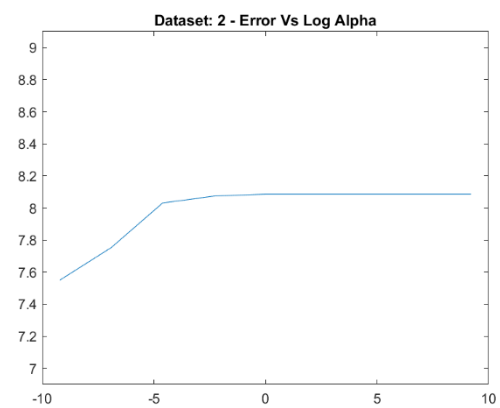

Figure 7: Dataset 2/strategy 1 percent error vs log(alpha) for predictive equation

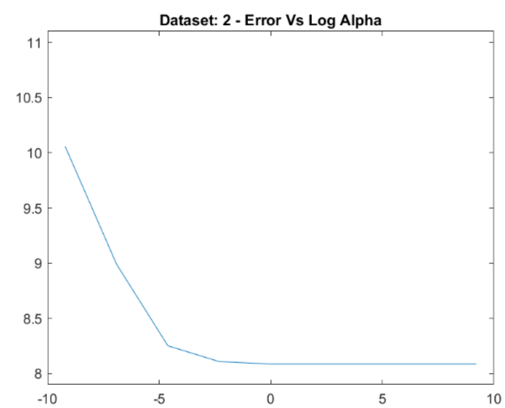

Figure 8: Dataset 2/strategy 2 percent error vs log(alpha) for predictive equation

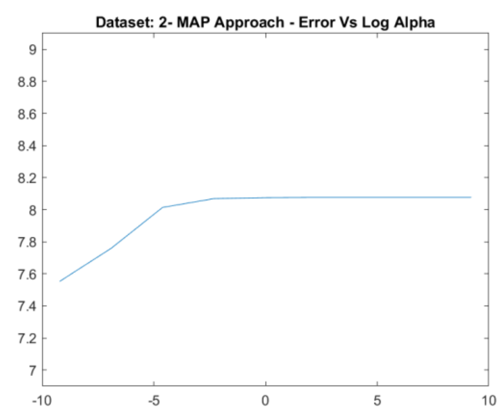

Figure 9: Dataset 2/strategy 1 percent error vs log(alpha) for MAP approach

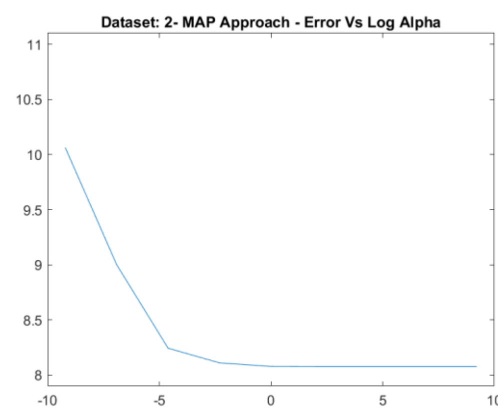

Figure 10: Dataset 2/strategy 2 percent error vs log(alpha) for MAP approach

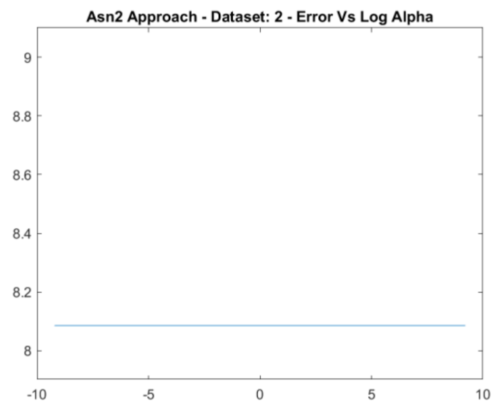

Figure 11: Dataset 2/strategy 1 percent error vs log(alpha) for ML approach

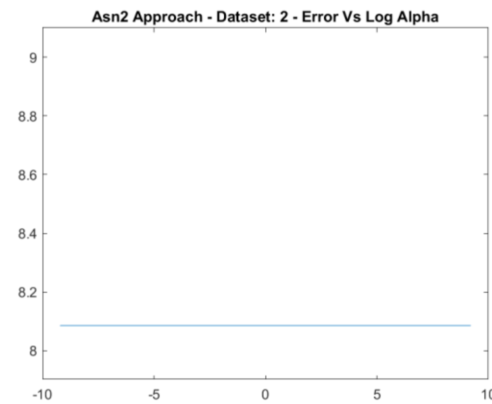

Figure 12: Dataset 2/strategy 2 percent error vs log(alpha) for ML approach

Dataset 3

Figure 13: Dataset 3/strategy 1 percent error vs log(alpha) for predictive equation

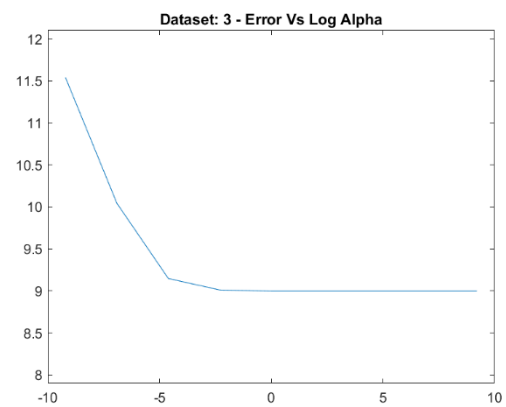

Figure 14: Dataset 3/strategy 2 percent error vs log(alpha) for predictive equation

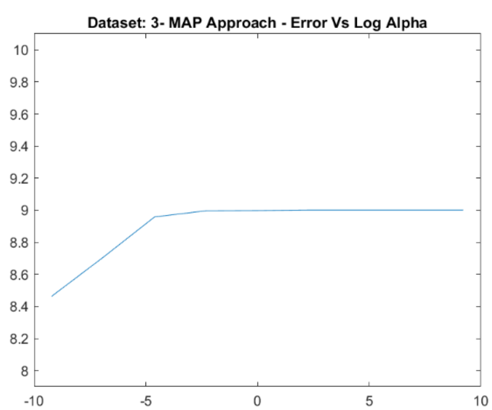

Figure 15: Dataset 3/strategy 1 percent error vs log(alpha) for MAP approach

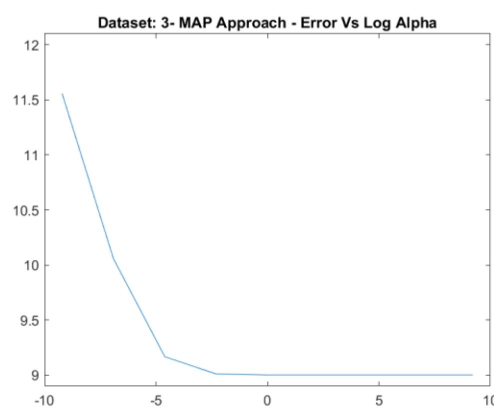

Figure 16: Dataset 3/strategy 2 percent error vs log(alpha) for MAP approach

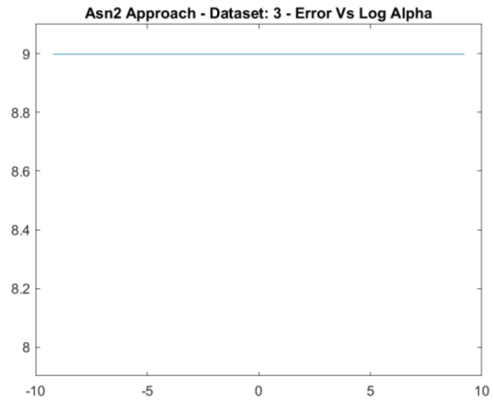

Figure 17: Dataset 3/strategy 1 percent error vs log(alpha) for ML approach

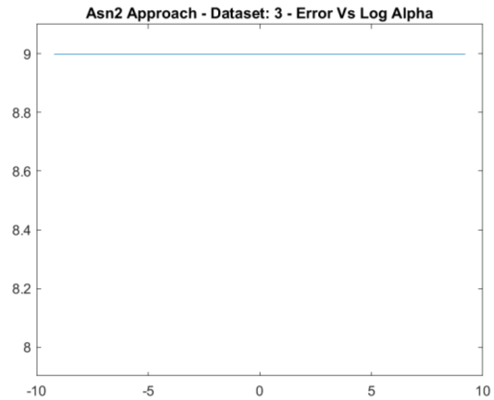

Figure 18: Dataset 3/strategy 2 percent error vs log(alpha) for ML approach

Dataset 4

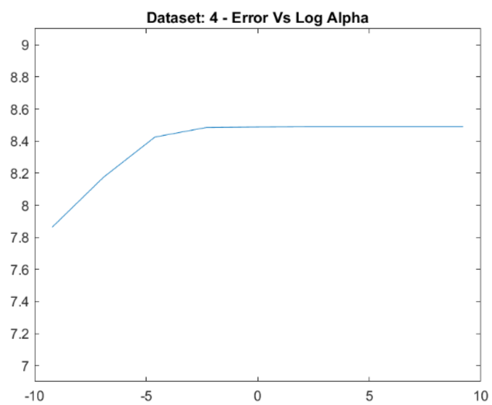

Figure 19: Dataset 4/strategy 1 percent error vs log(alpha) for predictive equation

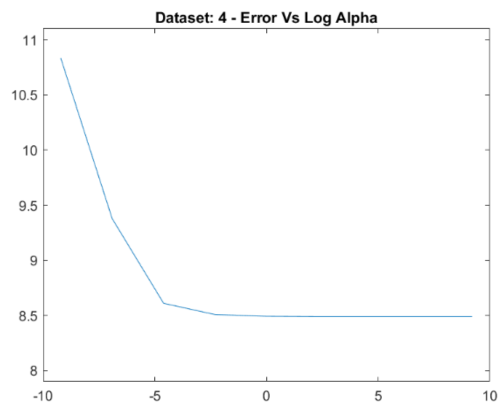

Figure 20: Dataset 4/strategy 2 percent error vs log(alpha) for predictive equation

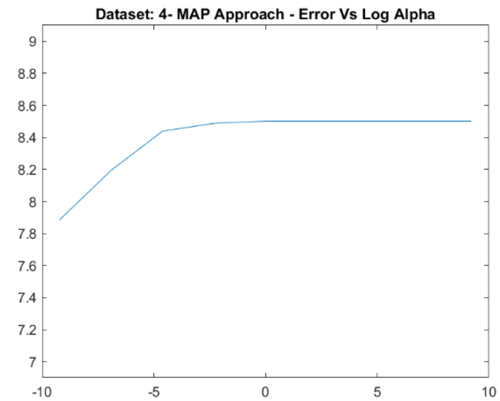

Figure 21: Dataset 4/strategy 1 percent error vs log(alpha) for MAP approach

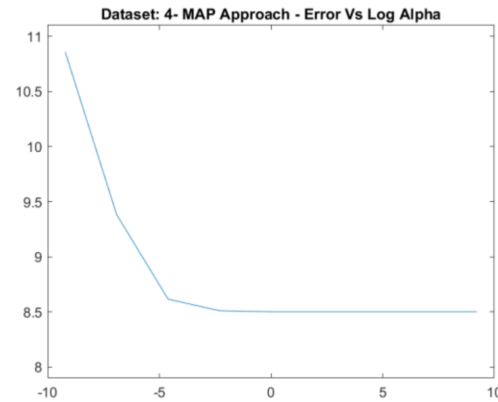

Figure 22: Dataset 4/strategy 2 percent error vs log(alpha) for MAP approach

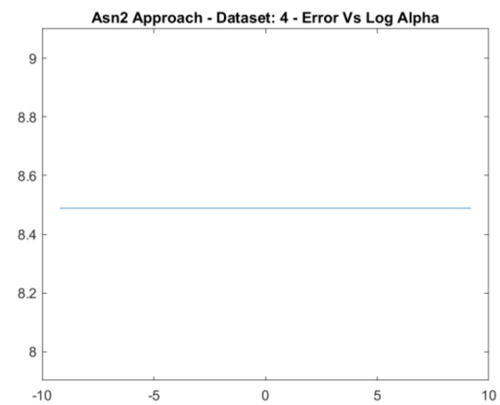

Figure 23: Dataset 4/strategy 1 percent error vs log(alpha) for ML approach

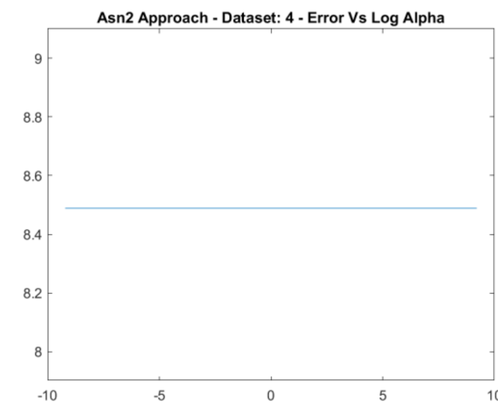

Figure 24: Dataset 4/strategy 2 percent error vs log(alpha) for ML approach

Joseph Bell
ECE 271A

Analysis

All the curves for the predictive equation and MAP approaches are asymptotic, and the "curves" for the ML approach are linear (as the function for ML is not dependent on alpha). For strategy 1, as alpha increases the error percentage increases toward an upper limit. For strategy 2, as alpha increases the error percentage decreases toward a lower limit. These upper and lower limits, however, are equivalent (i.e. strategy 1 increases toward the limit, whereas strategy 2 decreases toward that same limit value). This limit value, spanning all datasets (with one exception) and strategy types is the error percentage calculated by the ML approach. The one exception to this trend is observed in dataset 1. In dataset 1 (for both strategies) the MAP approach tends toward an asymptote of approximately 15.6, whereas the predictive equation approach tends toward an asymptote of approximately 15.5. The ML approach has an error percentage of approximately 15.5. From this observation, it seems as if the MAP approach will result in a larger percentage of error relative to the predictive equation approach when the dataset is small*. As the dataset size increases (dataset 2, 3, and 4 are all larger than dataset 1), however, both the MAP and predictive equation approaches approach the same asymptote value – this asymptote value is the value of the ML percentage of error.

*****Small in this case is 300 DCT coefficient arrays for the background and 75 DCT coefficient arrays for the foreground.

Joseph Bell
ECE 271A

Code

```
%Joseph Bell
%ECE271A HW3 and 4

clc;
clear;
load('TrainingSamplesDCT_subsets_8.mat');
load('Alpha.mat');



%%%%% Starting off with Data Set 1 and strategy 1%%%%%
% Strategy 1 = ?0 is smaller for the (darker) cheetah class (?0 = 1)
% and larger for the (lighter) grass class (?0 = 3)

%%%%% Part a %%%%%
%load('Prior_1.mat'); %priors for strategy 1 - consists of weights, mu0_FG,
                     %and mu0_BG
load('Prior_2.mat'); %priors for strategy 2 - consists of weights, mu0_FG,
                     %and mu0_BG

% Reading in cheetah image and mask
cheetah_img = imread('cheetah.bmp');
cheetah_img = im2double(cheetah_img); %converting to double values since training data
is of type double

cheetah_mask = imread('cheetah_mask.bmp');
cheetah_mask = im2double(cheetah_mask);

[cheetah_rows, cheetah_cols] = size(cheetah_img);
cheetah_img = cheetah_img(1:8*floor(cheetah_rows/8),1:8*floor(cheetah_cols/8));
%modifying image so it can be split into 8x8 blocks
[cheetah_rows, cheetah_cols] = size(cheetah_img); %overwriting for modified dimensions

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% STARTING LOOP %%%%%%%%%% Loops 4 times due to 4 data sets
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for d=1:4
    alpha_values = []; %array of alpha values used for plotting
    error_percentages = []; %array of error percentages per alpha used for plotting
    dataset_FG = []; %stores selected foreground data set
    dataset_BG = []; %stores selected background data set

    %%%% IF/ELSEIF for selecting data set to operate on
    if d == 1
```

```
        dataset_FG = D1_FG;
        dataset_BG = D1_BG;
    elseif d == 2
        dataset_FG = D2_FG;
        dataset_BG = D2_BG;
    elseif d == 3
        dataset_FG = D3_FG;
        dataset_BG = D3_BG;
    elseif d == 4
        dataset_FG = D4_FG;
        dataset_BG = D4_BG;
    end

    %Taking rows and cols used for calculations
    [data_rows_FG, data_cols_FG] = size(dataset_FG);
    [data_rows_BG, data_cols_BG] = size(dataset_BG);


    %zig zag scanning of data to order coefficents from 1-64
    %1 row per DC coefficient (First row should have largest values)
    cheetah_zigzag = zeros(64, data_rows_FG);
    grass_zigzag = zeros(64, data_rows_BG);

    %Credit to Alexey Sokolov from
https://www.mathworks.com/matlabcentral/fileexchange/15317-zigzag-scan
    %for the zig zag code

    %Reading into zig zag and transposing to create n*64 matrix
    for row=1:data_rows_FG
        cheetah_zigzag(:,row) = zigzag(dataset_FG(row,:));
    end

    for row=1:data_rows_BG
        grass_zigzag(:,row) = zigzag(dataset_BG(row,:));
    end

    means_variance_cheetah = zeros(64, 1, 2); %(:,:,1) = mean
    means_variance_grass = zeros(64, 1, 2); %(:,:,2) = variance

    % Calculating mean and variance of each coefficient
    % CODE COPIED FROM MY ASSIGNMENT 2

    %cheetah loop
    for row=1:data_cols_FG
        N = data_rows_FG;
        sample_mean = 0;
        sample_variance = 0;

        sample = cheetah_zigzag(row,:); %grab each coefficient row

        for i=1:N
```

```
            sample_mean = sample_mean + sample(1,i);
        end
        sample_mean = sample_mean/N;

        for i=1:N
            sample_variance = sample_variance + (sample(1,i) - sample_mean)^2;
        end
        sample_variance = sample_variance/N;

        means_variance_cheetah(row,1,1) = sample_mean;
        means_variance_cheetah(row,1,2) = sample_variance;
    end

    %grass loop
    for row=1:data_cols_BG
        N = data_rows_BG;
        sample_mean = 0;
        sample_variance = 0;

        sample = grass_zigzag(row,:); %grab each coefficient row

        for i=1:N
            sample_mean = sample_mean + sample(1,i);
        end
        sample_mean = sample_mean/N;

        for i=1:N
            sample_variance = sample_variance + (sample(1,i) - sample_mean)^2;
        end
        sample_variance = sample_variance/N;

        means_variance_grass(row,1,1) = sample_mean;
        means_variance_grass(row,1,2) = sample_variance;
    end


    % Calculating covariance matrices for foreground (cheetah)
    % and background (grass)
    % CODE COPIED FROM MY ASSIGNMENT 2
    cheetah_covariances = zeros(64,64);
    grass_covariances = zeros(64,64);

    %calculating covariance matrix for cheetah
    for i=1:64
        p1 = cheetah_zigzag(i,:); %grab coefficient row
        mu_1 = means_variance_cheetah(i,1,1);
        for j=1:64
            p2 = cheetah_zigzag(j,:); %grab coefficient row
            mu_2 = means_variance_cheetah(j,1,1);
            temp = 0;
            for k=1:data_rows_FG
```

```
                temp = temp + (p1(1,k) - mu_1)*(p2(1,k) - mu_2);
            end
            cheetah_covariances(i,j) = temp/data_rows_FG;
        end
    end

    %calculating covariance matrix for grass
    for i=1:64
        p1 = grass_zigzag(i,:); %grab coefficient row
        mu_1 = means_variance_grass(i,1,1);
        for j=1:64
            p2 = grass_zigzag(j,:); %grab coefficient row
            mu_2 = means_variance_grass(j,1,1);
            temp = 0;
            for k=1:data_rows_BG
                temp = temp + (p1(1,k) - mu_1)*(p2(1,k) - mu_2);
            end
            grass_covariances(i,j) = temp/data_rows_BG;
        end
    end

    %Used for probability calculations
    mu_n_hat_cheetah1 = zeros(1,64);
    mu_n_hat_grass1 = zeros(1,64);
    mu_n_cheetah1 = zeros(1,64);
    mu_n_grass1 = zeros(1,64);

    %Used for MAP part
    prior_cheetah = data_rows_FG/(data_rows_FG + data_rows_BG);
    prior_grass = data_rows_BG/(data_rows_BG+data_rows_FG);

    %%%%% Calculating mu_n_hat values %%%%
    for i=1:64
        for j=1:data_rows_FG
            mu_n_hat_cheetah1(1,i) = mu_n_hat_cheetah1(1,i) + dataset_FG(j,i);
        end
        mu_n_hat_cheetah1(1,i) = mu_n_hat_cheetah1(1,i)/data_rows_FG;
    end

    for i=1:64
        for j=1:data_rows_BG
            mu_n_hat_grass1(1,i) = mu_n_hat_grass1(1,i) + dataset_BG(j,i);
        end
        mu_n_hat_grass1(1,i) = mu_n_hat_grass1(1,i)/data_rows_BG;
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%% STARTING LOOP FOR ALPHA %%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
    for a=1:9


        new_image = zeros(cheetah_rows, cheetah_cols); %Returned image

        %Values used for calculations%
        sigma_1 = alpha(1,a)*diag(W0);
        mu_n_cheetah1 = sigma_1 * inv(sigma_1 +
1/data_rows_FG*cheetah_covariances)*transpose(mu_n_hat_cheetah1) +
1/data_rows_FG*cheetah_covariances*inv(sigma_1+1/data_rows_FG*cheetah_covariances)*tra
nspose(mu0_FG);
        mu_n_grass1 = sigma_1 * inv(sigma_1 +
1/data_rows_BG*grass_covariances)*transpose(mu_n_hat_grass1) +
1/data_rows_BG*grass_covariances*inv(sigma_1+1/data_rows_BG*grass_covariances)*transpo
se(mu0_BG);

        sigma_n_cheetah = sigma_1*inv(sigma_1 +
1/data_rows_FG*cheetah_covariances)*1/data_rows_FG*cheetah_covariances;
        sigma_n_grass = sigma_1*inv(sigma_1 +
1/data_rows_BG*grass_covariances)*1/data_rows_BG*grass_covariances;

        sigma_cheetah_total = cheetah_covariances + sigma_n_cheetah;
        sigma_grass_total = grass_covariances + sigma_n_grass;

        %Multivariate Gaussian Distribution PDF functions for part a
        fun_cheetah = @(x) 1/sqrt((det(sigma_cheetah_total)*(2*pi)^64))*exp(-
1/2*transpose(x-mu_n_cheetah1)*inv(sigma_cheetah_total)*(x-mu_n_cheetah1));
        fun_grass= @(x) 1/sqrt((det(sigma_grass_total)*(2*pi)^64))*exp(-
1/2*transpose(x-mu_n_grass1)*inv(sigma_grass_total)*(x-mu_n_grass1));

        %%%%%%%%%% Classifying %%%%%%%%%%%%
        for i=1:cheetah_cols-7 %shift scan pointer over a column
            for j=1:cheetah_rows-7

                block = cheetah_img(j:7+j,i:7+i); %grab 8x8 block
                block_dct = dct2(block);
                zzblock_dct = transpose(zigzag(block_dct));

                P_x_D_cheetah = fun_cheetah(zzblock_dct);
                P_x_D_grass = fun_grass(zzblock_dct);
                if P_x_D_cheetah * prior_cheetah > P_x_D_grass * prior_grass
                    new_image(j:j,i:i) = 1;
                end
            end
        end
        f = figure() % Returning image per alpha
        imagesc(new_image);
        colormap(gray(255));
        title(['Dataset: ', num2str(d),' - Alpha: ', num2str(alpha(1,a))])
        saveas(f,[pwd,'/results/D_',num2str(d),'_A_',num2str(a),'.png']);
        %%%%% Calculating percent error for part a Alpha values %%%%%
```

```
        counter_correct = 0;
        total_pixels = cheetah_rows*cheetah_cols;
        for i=1:cheetah_rows
            for j=1:cheetah_cols
                if cheetah_mask(i,j) ==  new_image(i,j)
                    counter_correct = counter_correct + 1;
                end

            end
        end
        percent_correct = counter_correct/total_pixels*100;
        error_percentage = 100 - percent_correct;

        alpha_values = [alpha_values log(alpha(1,a))];
        error_percentages = [error_percentages error_percentage];

    end% END OF a=1:9 Still in d data set loop

    f = figure() %plotting percent error vs alpha
    plot(alpha_values, error_percentages);
    ylim([floor(min(error_percentages))-0.1 ceil(max(error_percentages))+0.1])

    title(['Dataset: ', num2str(d), ' - Error Vs Log Alpha']);
    saveas(f,[pwd,'/results/error_plot_D_',num2str(d),'.png']);
    %%%%%%%%%%%%%%%%%%%%%%% Assignment 2 Method %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % Starting loop of alphas for assignment 2 approach
    %calculating MLE covariance matrix for cheetah
    cheetah_covariances_MLE = zeros(64,64);
    grass_covariances_MLE = zeros(64,64);
    for i=1:64
        p1 = cheetah_zigzag(i,:); %grab coefficient row
        mu_1 = means_variance_cheetah(i,1,1);
        for j=1:64
            p2 = cheetah_zigzag(j,:); %grab coefficient row
            mu_2 = means_variance_cheetah(j,1,1);
            temp = 0;
            for k=1:data_rows_FG
                temp = temp + (p1(1,k) - mu_1)*(p2(1,k) - mu_2);
            end
            cheetah_covariances_MLE(i,j) = temp/(data_rows_FG-1);
        end
    end

    %calculating covariance matrix for grass
    for i=1:64
        p1 = grass_zigzag(i,:); %grab coefficient row
        mu_1 = means_variance_grass(i,1,1);
        for j=1:64
            p2 = grass_zigzag(j,:); %grab coefficient row
            mu_2 = means_variance_grass(j,1,1);
```

```
            temp = 0;
            for k=1:data_rows_BG
                temp = temp + (p1(1,k) - mu_1)*(p2(1,k) - mu_2);
            end
            grass_covariances_MLE(i,j) = temp/(data_rows_BG-1);
        end
    end
    alpha_values_Asn2 = []; %array of alpha values used for plotting
    error_percentages_Asn2 = []; %array of error percentages per alpha used for
plotting
    for a=1:9
        new_image2 = zeros(cheetah_rows, cheetah_cols);
        fun_cheetah_Asn2 = @(x) 1/sqrt((det(cheetah_covariances_MLE)*(2*pi)^64))*exp(-
1/2*transpose(x-means_variance_cheetah(:,:,1))*inv(cheetah_covariances_MLE)*(x-
means_variance_cheetah(:,:,1)));
        fun_grass_Asn2= @(x) 1/sqrt((det(grass_covariances_MLE)*(2*pi)^64))*exp(-
1/2*transpose(x-means_variance_grass(:,:,1))*inv(grass_covariances_MLE)*(x-
means_variance_grass(:,:,1)));
        for i=1:cheetah_cols-7 %shift scan pointer over a column
            for j=1:cheetah_rows-7
                block = cheetah_img(j:7+j,i:7+i); %grab 8x8 block
                block_dct = dct2(block);
                zzblock_dct = transpose(zigzag(block_dct));

                P_x_y_cheetah = fun_cheetah_Asn2(zzblock_dct);
                P_x_y_grass = fun_grass_Asn2(zzblock_dct);
                if P_x_y_cheetah * prior_cheetah > P_x_y_grass * prior_grass
                        new_image2(j:j,i:i) = 1;
                end
            end
        end


        %Calculating percent error of assignment 2 method
        counter_correct_Asn2 = 0;
        total_pixels = cheetah_rows*cheetah_cols;
        for i=1:cheetah_rows
            for j=1:cheetah_cols
                if cheetah_mask(i,j) ==  new_image2(i,j)
                    counter_correct_Asn2 = counter_correct_Asn2 + 1;
                end

            end
        end
        percent_correct_Asn2 = counter_correct_Asn2/total_pixels*100;
        error_percentage_Asn2 = 100 - percent_correct_Asn2;

        alpha_values_Asn2 = [alpha_values_Asn2 log(alpha(1,a))];
        error_percentages_Asn2 = [error_percentages_Asn2 error_percentage_Asn2];
    end%END OF a=1:9 for Assignment 2 MLE method
```

```
    f = figure() %plotting assignment 2 method (1 per data set just plotting last one
they're all the same)
    imagesc(new_image2);      %as I'm not multiplying covariace by alpha for this part
only using MLE
    colormap(gray(255));      %percent error chart should be a horizontal line
    title(['Dataset: ', num2str(d), ' - Assignment 2 Method'])
    saveas(f,[pwd,'/results/Asn2_D_',num2str(d),'.png']);

    f = figure() %plotting percent error vs alpha (should only be 4 of these)
    plot(alpha_values_Asn2, error_percentages_Asn2);
    ylim([floor(min(error_percentages_Asn2))-0.1
ceil(max(error_percentages_Asn2))+0.1])
    title(['Asn2 Approach - Dataset: ', num2str(d), ' - Error Vs Log Alpha']);
    saveas(f,[pwd,'/results/error_plot_Asn2_D_',num2str(d),'.png']);


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%% MAP APPROACH %%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    alpha_values = []; %Stores alpha values for plotting
    error_percentages = []; %Stores error percentage per alpha

    %Same stuff except using MAP for mu
    for a=1:9
        new_image = zeros(cheetah_rows, cheetah_cols);

        sigma_1 = alpha(1,a)*diag(W0);

        mu_n_cheetah1 = sigma_1 * inv(sigma_1 +
1/data_rows_FG*cheetah_covariances)*transpose(mu_n_hat_cheetah1) +
1/data_rows_FG*cheetah_covariances*inv(sigma_1+1/data_rows_FG*cheetah_covariances)*tra
nspose(mu0_FG);
        mu_n_grass1 = sigma_1 * inv(sigma_1 +
1/data_rows_BG*grass_covariances)*transpose(mu_n_hat_grass1) +
1/data_rows_BG*grass_covariances*inv(sigma_1+1/data_rows_BG*grass_covariances)*transpo
se(mu0_BG);;

        %sigma_n_cheetah = sigma_1*inv(sigma_1 +
1/data_rows_FG*cheetah_covariances)*1/data_rows_FG*cheetah_covariances;
        %sigma_n_grass = sigma_1*inv(sigma_1 +
1/data_rows_BG*grass_covariances)*1/data_rows_BG*grass_covariances;

        sigma_n_cheetah = 0;
        sigma_n_grass = 0;

        sigma_cheetah_total = cheetah_covariances + sigma_n_cheetah;
        sigma_grass_total = grass_covariances + sigma_n_grass;
```

```
        fun_cheetah_D1c = @(x) 1/sqrt((det(sigma_cheetah_total)*(2*pi)^64))*exp(-
1/2*transpose(x-mu_n_cheetah1)*inv(sigma_cheetah_total)*(x-mu_n_cheetah1));
        fun_grass_D1c= @(x) 1/sqrt((det(sigma_grass_total)*(2*pi)^64))*exp(-
1/2*transpose(x-mu_n_grass1)*inv(sigma_grass_total)*(x-mu_n_grass1));
        for i=1:cheetah_cols-7 %shift scan pointer over a column
            for j=1:cheetah_rows-7

                block = cheetah_img(j:7+j,i:7+i); %grab 8x8 block
                block_dct = dct2(block);
                zzblock_dct = transpose(zigzag(block_dct));

                P_x_D_cheetah = fun_cheetah_D1c(zzblock_dct);
                P_x_D_grass = fun_grass_D1c(zzblock_dct);

                if P_x_D_cheetah * prior_cheetah > P_x_D_grass * prior_grass
                    new_image(j:j,i:i) = 1;
                end
            end
        end
        f = figure()%Plotting per alpha for MAP approach
        imagesc(new_image);
        colormap(gray(255));
        title(['Dataset: ', num2str(d), ' - MAP Approach - Alpha: ',
num2str(alpha(1,a))])
        saveas(f,[pwd,'/results/MAP_D_',num2str(d),'_A_',num2str(a),'.png']);
        %Calculating percent error for alpha and MAP approach
        counter_correct = 0;
        total_pixels = cheetah_rows*cheetah_cols;
        for i=1:cheetah_rows
            for j=1:cheetah_cols
                if cheetah_mask(i,j) ==  new_image(i,j)
                    counter_correct = counter_correct + 1;
                end

            end
        end
        percent_correct = counter_correct/total_pixels*100;
        error_percentage = 100 - percent_correct;

        alpha_values = [alpha_values log(alpha(1,a))];
        error_percentages = [error_percentages error_percentage];
    end

    f = figure()%Potting percent error vs alpha values for MAP
    plot(alpha_values, error_percentages);
    ylim([floor(min(error_percentages))-0.1 ceil(max(error_percentages))+0.1])
    title(['Dataset: ', num2str(d), '- MAP Approach', ' - Error Vs Log Alpha']);
    saveas(f,[pwd,'/results/error_MAP_D',num2str(d),'.png']);
end
```