Student Name: Lin-Hsi (Ethen) Tsao
Student PID: A59015911

Statistical Learning 1 (ECE 271A)

HW5

(a) **For each class, learn 5 mixtures of $C = 8$ components, using a random initialization (recall that the mixture weights must add up to one). Plot the probability of error vs. dimension for each of the 25 classifiers obtained with all possible mixture pairs. Comment the dependence of the probability of error on the initialization.**

We can see that different combination of the foreground and the background mixtures would lead to different error rate since we use the random initialization during the EM. From the provided plot, we can also see that the more dim, the better performance (most of the time). This is because we provide more features. However, if we use 64 dim for BDR, the performance is not the best. One possible reason is that some feature is not as good as others. For example, from the previous HW, we used the best 8 features to do the BDR, leading to a better performance. According to the observation, the best dim is around 40 to 60.

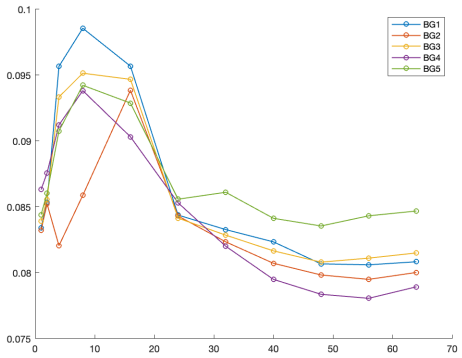


Figure 1: Error rate curve with 1st foreground mixture with different background mixture.

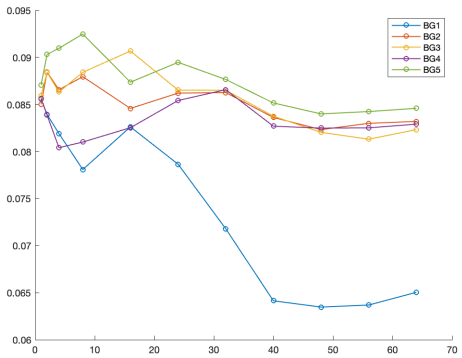


Figure 2: Error rate curve with 2nd foreground mixture with different background mixture.

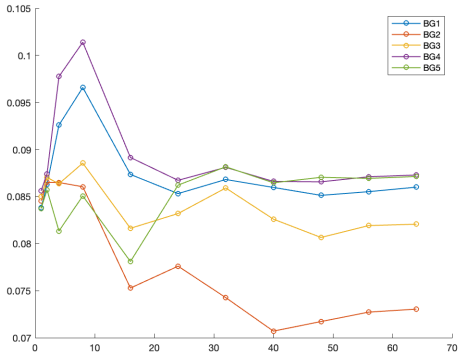


Figure 3: Error rate curve with 3rd foreground mixture with different background mixture.

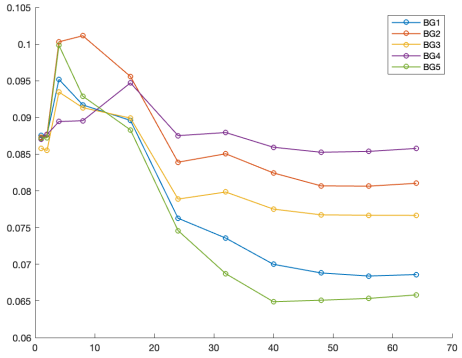


Figure 4: Error rate curve with 4th foreground mixture with different background mixture.

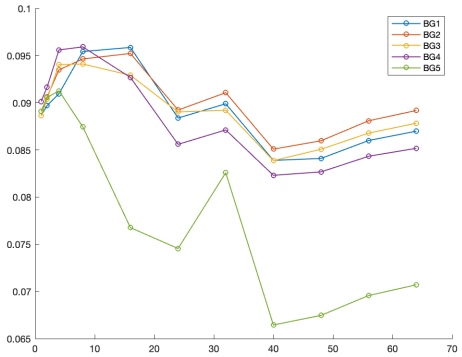


Figure 5: Error rate curve with 5th foreground mixture with different background mixture.

(b) **For each class, learn mixtures with $C \in \{1, 2, 4, 8, 16, 32\}$. Plot the probability of error vs. dimension for each number of mixture components. What is the effect of the number of mixture components on the probability of error?**

We can see that the more component, the better performance (most of the time). This is because if we only use 1 component, it is hard to fit a real data distribution. That is to say, the real data distribution is not just a simple multivariate Gaussian distribution.

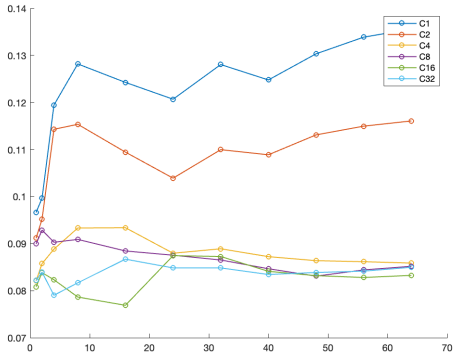


Figure 6: Error rate curve with different component.

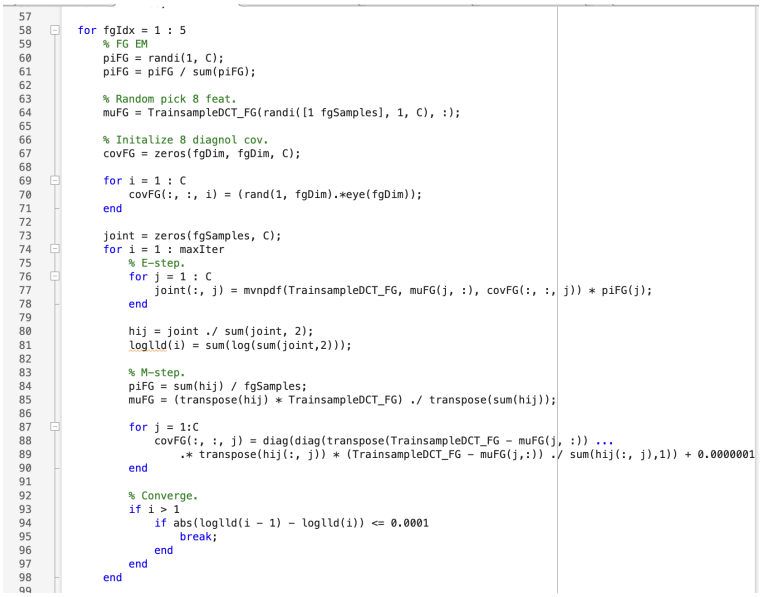
Code

```
% Load training data.
load('hw5Data/TrainingSamplesDCT_8_new.mat');

% Load image, mask.
img = imread('hw5Data/cheetah.bmp');
img = double(img) / 255;
mask = imread('hw5Data/cheetah_mask.bmp');

% Load zigzag pattern.
pattern = readmatrix('hw5Data/Zig-Zag Pattern.txt');

% Padding img with zero.
I = zeros(263, 278);
for row = 5 : 259
    for col = 5 : 274
        I(row, col) = img(row - 4, col - 4);
    end
end

% Base info.
fgSamples = size(TrainsampleDCT_FG, 1);
fgDim = size(TrainsampleDCT_FG, 2);
bgSamples = size(TrainsampleDCT_BG, 1);
bgDim = size(TrainsampleDCT_BG, 2);
rows = size(img, 1);
cols = size(img, 2);
```

Figure 7: Load data and set the basic info of the dataset.

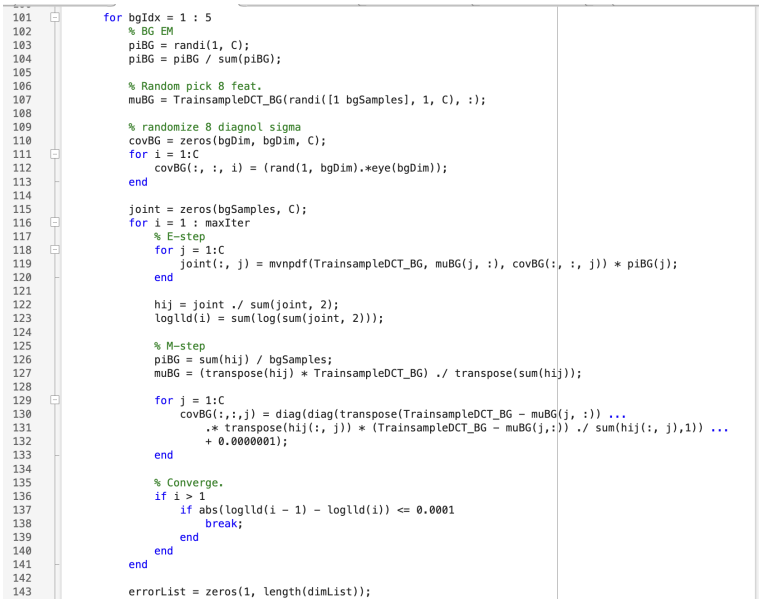
```
% Get dct feature.
A = zeros(rows * cols, 64);
for i = 1 : rows
    for j = 1 : cols
        block = zeros(8,8);
        for r = i : i + 7
            for c = j : j + 7
                block(r - i + 1, c - j + 1) = I(r, c);
            end
        end

        % Compute DCT.
        dct2Block = dct2(block);
        flatBlock = zeros(1, 64);

        for r = 1 : 8
            for c = 1 : 8
                flatBlock(1, pattern(r, c) + 1) = dct2Block(r, c);
            end
        end

        A((i - 1) * cols + j, :) = flatBlock;
    end
end

% Hyper parameters.
C = 8;
dimList = [1, 2, 4, 8, 16, 24, 32, 40, 48, 56, 64];
maxIter = 200;
```

Figure 8: Get the dct feature and set the hyper parameter.



```
for fgIdx = 1 : 5
    % FG EM
    piFG = randi(1, C);
    piFG = piFG / sum(piFG);

    % Random pick 8 feat.
    muFG = TrainsampleDCT_FG(randi([1 fgSamples], 1, C), :);

    % Initalize 8 diagnol cov.
    covFG = zeros(fgDim, fgDim, C);

    for i = 1 : C
        covFG(:, :, i) = (rand(1, fgDim).*eye(fgDim));
    end

    joint = zeros(fgSamples, C);
    for i = 1 : maxIter
        % E-step.
        for j = 1 : C
            joint(:, j) = mvnpdf(TrainsampleDCT_FG, muFG(j, :), covFG(:, :, j)) * piFG(j);
        end

        hij = joint ./ sum(joint, 2);
        loglld(i) = sum(log(sum(joint,2)));

        % M-step.
        piFG = sum(hij) / fgSamples;
        muFG = (transpose(hij) * TrainsampleDCT_FG) ./ transpose(sum(hij));

        for j = 1:C
            covFG(:, :, j) = diag(diag(transpose(TrainsampleDCT_FG - muFG(j, :)) ...
                .* transpose(hij(:, j)) * (TrainsampleDCT_FG - muFG(j,:)) ./ sum(hij(:, j),1)) + 0.0000001
        end

        % Converge.
        if i > 1
            if abs(loglld(i - 1) - loglld(i)) <= 0.0001
                break;
            end
        end
    end
```

Figure 9: Estimate the foreground EM.



```
    for bgIdx = 1 : 5
        % BG EM
        piBG = randi(1, C);
        piBG = piBG / sum(piBG);

        % Random pick 8 feat.
        muBG = TrainsampleDCT_BG(randi([1 bgSamples], 1, C), :);

        % randomize 8 diagnol sigma
        covBG = zeros(bgDim, bgDim, C);
        for i = 1:C
            covBG(:, :, i) = (rand(1, bgDim).*eye(bgDim));
        end

        joint = zeros(bgSamples, C);
        for i = 1 : maxIter
            % E-step
            for j = 1:C
                joint(:, j) = mvnpdf(TrainsampleDCT_BG, muBG(j, :), covBG(:, :, j)) * piBG(j);
            end

            hij = joint ./ sum(joint, 2);
            loglld(i) = sum(log(sum(joint, 2)));

            % M-step
            piBG = sum(hij) / bgSamples;
            muBG = (transpose(hij) * TrainsampleDCT_BG) ./ transpose(sum(hij));

            for j = 1:C
                covBG(:,:,j) = diag(diag(transpose(TrainsampleDCT_BG - muBG(j, :)) ...
                    .* transpose(hij(:, j)) * (TrainsampleDCT_BG - muBG(j,:)) ./ sum(hij(:, j),1)) ...
                    + 0.0000001);
            end

            % Converge.
            if i > 1
                if abs(loglld(i - 1) - loglld(i)) <= 0.0001
                    break;
                end
            end
        end

        errorList = zeros(1, length(dimList));
```

Figure 10: Estimate the background EM.

```
        for i = 1 : length(dimList)
            dim = dimList(i);
            result = zeros(rows * cols, 1);

            for x = 1 : length(A)
                probFG = 0;
                probBG = 0;

                for y = 1 : C
                    probFG = probFG + mvnpdf(A(x, 1 : dim), muFG(y, 1 : dim), covFG(1 : dim, 1 : dim, y))
                        * piFG(y);

                    probBG = probBG + mvnpdf(A(x, 1 : dim), muBG(y, 1 : dim), covBG(1 : dim, 1 : dim, y))
                        * piBG(y);
                end

                if probBG <= probFG
                    result(x) = 1;
                end
            end

%             resultImage = zeros(rows, cols);
%             for k = 1 : rows
%                 resultImage(k, :) = transpose(result((k - 1) * cols + 1 : k * cols));
%             end

            for x = 1 : rows
                for y = 1 : cols
                    if mask(x, y) ~= result((x - 1) * cols + y, 1)
                        errorList(1, i) = errorList(1, i) + 1;
                    end
                end
            end

            errorList(1, i) = errorList(1, i) / (rows * cols);
            disp("idxFG " + fgIdx + " idxBG " + bgIdx + " dim " + dim);
        end
        hold on;
        plot(dimList, errorList, 'o-', 'linewidth', 1, 'markersize', 5);
    end
```

Figure 11: Estimate the error rate.

```
% Hyper parameters.
CList = [1, 2, 4, 8, 16, 32];
dimList = [1, 2, 4, 8, 16, 24, 32, 40, 48, 56, 64];
maxIter = 200;

for c = 1 : length(CList)
    C = Clist(1, c);
```

Figure 12: Code for problem 2. Use only 1 foreground EM and 1 background EM with different c components. The other code remains as the problem 1.