```
load('TrainingSamplesDCT_subsets_8.mat')
cheetah_img=imread('cheetah.bmp');
mask=imread('cheetah_mask.bmp');
[m,n]=size(cheetah_img);

%Strategy 1
load('Prior_1.mat');
load('Alpha.mat');
alpha=load('Alpha.mat').alpha;

%Dataset 1
[numSamplesBG, ~] = size(D1_BG);
[numSamplesFG, ~] = size(D1_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D1_BG(:, 1:64), 1);
backgroundCovariance = (D1_BG(:, 1:64) - backgroundMean)' * (D1_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D1_FG(:, 1:64), 1);
foregroundCovariance = (D1_FG(:, 1:64) - foregroundMean)' * (D1_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
```

```
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalBackgroundCovariance)) + log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalForegroundCovariance)) + log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                     - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);

            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
```

```
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end
```

```
figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 1 Strategy 1');
legend('Location', 'best', 'FontSize', 10);
set(gca);



%  %Dataset 2
[numSamplesBG, ~] = size(D2_BG);
[numSamplesFG, ~] = size(D2_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D2_BG(:, 1:64), 1);
backgroundCovariance = (D2_BG(:, 1:64) - backgroundMean)' * (D2_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D2_FG(:, 1:64), 1);
foregroundCovariance = (D2_FG(:, 1:64) - foregroundMean)' * (D2_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
```

```
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalBackgroundCovariance)) + log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalForegroundCovariance)) + log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
```

```
            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
```

```
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 2 Strategy 1');
legend('Location', 'best', 'FontSize', 10);
set(gca);


%Dataset 3
[numSamplesBG, ~] = size(D3_BG);
[numSamplesFG, ~] = size(D3_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D3_BG(:, 1:64), 1);
backgroundCovariance = (D3_BG(:, 1:64) - backgroundMean)' * (D3_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D3_FG(:, 1:64), 1);
foregroundCovariance = (D3_FG(:, 1:64) - foregroundMean)' * (D3_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
```

```
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalBackgroundCovariance)) + log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalForegroundCovariance)) + log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
```

```
log(probabilityBG);

            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                      - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
```

```
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 3 Strategy 1');
legend('Location', 'best', 'FontSize', 10);
set(gca);

% Dataset 4
[numSamplesBG, ~] = size(D4_BG);
[numSamplesFG, ~] = size(D4_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D4_BG(:, 1:64), 1);
backgroundCovariance = (D4_BG(:, 1:64) - backgroundMean)' * (D4_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D4_FG(:, 1:64), 1);
foregroundCovariance = (D4_FG(:, 1:64) - foregroundMean)' * (D4_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
```

```
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalBackgroundCovariance)) + log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 *
det(finalForegroundCovariance)) + log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
```

```
log(probabilityBG);

            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                      - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
```

```
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 4 Strategy 1');
legend('Location', 'best', 'FontSize', 10);
set(gca);

% Strategy 2
clear;
load('TrainingSamplesDCT_subsets_8.mat')
cheetah_img=imread('cheetah.bmp');
mask=imread('cheetah_mask.bmp');
[m,n]=size(cheetah_img);
load('Prior_2.mat');
load('Alpha.mat');
alpha=load('Alpha.mat').alpha;

%Dataset 1
[numSamplesBG, ~] = size(D1_BG);
[numSamplesFG, ~] = size(D1_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D1_BG(:, 1:64), 1);
backgroundCovariance = (D1_BG(:, 1:64) - backgroundMean)' * (D1_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D1_FG(:, 1:64), 1);
foregroundCovariance = (D1_FG(:, 1:64) - foregroundMean)' * (D1_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
ML = [];
```

```
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalBackgroundCovariance))
+ log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalForegroundCovariance))
+ log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
            end
```

```
            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);

            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
            end
```

```
            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 1 Strategy 2');
legend('Location', 'best', 'FontSize', 10);
set(gca);


%  %Dataset 2
[numSamplesBG, ~] = size(D2_BG);
[numSamplesFG, ~] = size(D2_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D2_BG(:, 1:64), 1);
backgroundCovariance = (D2_BG(:, 1:64) - backgroundMean)' * (D2_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D2_FG(:, 1:64), 1);
foregroundCovariance = (D2_FG(:, 1:64) - foregroundMean)' * (D2_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
MAP = [];
```

```
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalBackgroundCovariance))
+ log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalForegroundCovariance))
+ log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                PD_BG = PD_BG + 1;
```

```
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                PD_FG = PD_FG + 1;
            end


            % MAP
            i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);

            i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


            if i1_MAP > i2_MAP
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                MAP_BG = MAP_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                MAP_FG = MAP_FG + 1;
            end

            index = index + 1;


            % ML
            i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
            i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                    - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

            if i1_ML > i2_ML
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
                ML_BG = ML_BG + 1;
```

```
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 2 Strategy 2');
legend('Location', 'best', 'FontSize', 10);
set(gca);


%Dataset 3
[numSamplesBG, ~] = size(D3_BG);
[numSamplesFG, ~] = size(D3_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D3_BG(:, 1:64), 1);
backgroundCovariance = (D3_BG(:, 1:64) - backgroundMean)' * (D3_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D3_FG(:, 1:64), 1);
foregroundCovariance = (D3_FG(:, 1:64) - foregroundMean)' * (D3_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
```

```
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalBackgroundCovariance))
+ log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalForegroundCovariance))
+ log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
```

```
            PD_BG = PD_BG + 1;
        end

        if uint8(A(j, h)) == 0 && mask(j, h) == 255
            PD_FG = PD_FG + 1;
        end


        % MAP
        i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                 - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);

        i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                 - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


        if i1_MAP > i2_MAP
            A(j, h) = 0;
        else
            A(j, h) = 255;
        end

        if uint8(A(j, h)) == 255 && mask(j, h) == 0
            MAP_BG = MAP_BG + 1;
        end

        if uint8(A(j, h)) == 0 && mask(j, h) == 255
            MAP_FG = MAP_FG + 1;
        end

        index = index + 1;


        % ML
        i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
        i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

        if i1_ML > i2_ML
            A(j, h) = 0;
        else
            A(j, h) = 255;
        end

        if uint8(A(j, h)) == 255 && mask(j, h) == 0
```

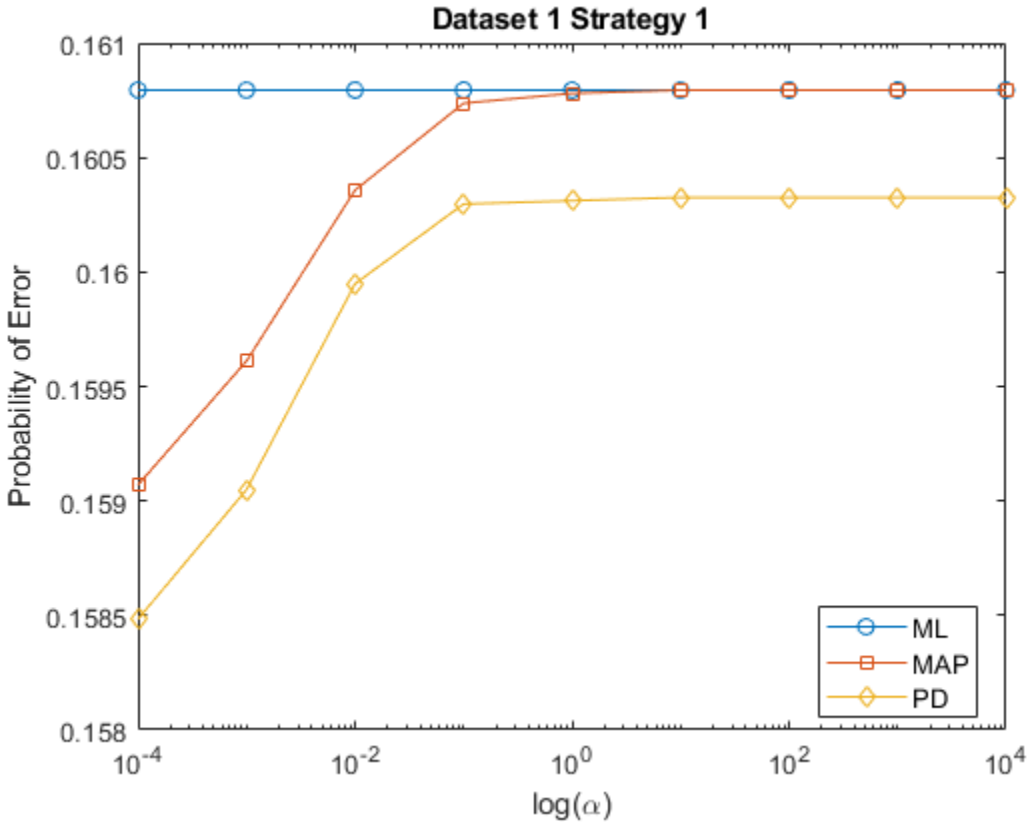
```
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 3 Strategy 2');
legend('Location', 'best', 'FontSize', 10);
set(gca);

% Dataset 4
[numSamplesBG, ~] = size(D4_BG);
[numSamplesFG, ~] = size(D4_FG);
probabilityBG = numSamplesBG / (numSamplesBG + numSamplesFG);
probabilityFG = numSamplesFG / (numSamplesBG + numSamplesFG);
backgroundMean = mean(D4_BG(:, 1:64), 1);
backgroundCovariance = (D4_BG(:, 1:64) - backgroundMean)' * (D4_BG(:, 1:64)
- backgroundMean) / numSamplesBG;
foregroundMean = mean(D4_FG(:, 1:64), 1);
foregroundCovariance = (D4_FG(:, 1:64) - foregroundMean)' * (D4_FG(:, 1:64)
- foregroundMean) / numSamplesFG;
priorCovariance = diag(W0);
index = 1;
zigzagPatternData = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPatternData(:)' + 1;

for row = 1:m-7
    for col = 1:n-7
        featureMatrix(:, index) = reshape(dct2(double(cheetah_img(row:row+7,
col:col+7)) / 255.0)', [64, 1]);
        featureMatrix(zigzagPattern, index) = featureMatrix(:, index);
        index = index + 1;
    end
end

sumFG = sum(mask(:) == 255);
sumBG = numel(mask) - sumFG;
```

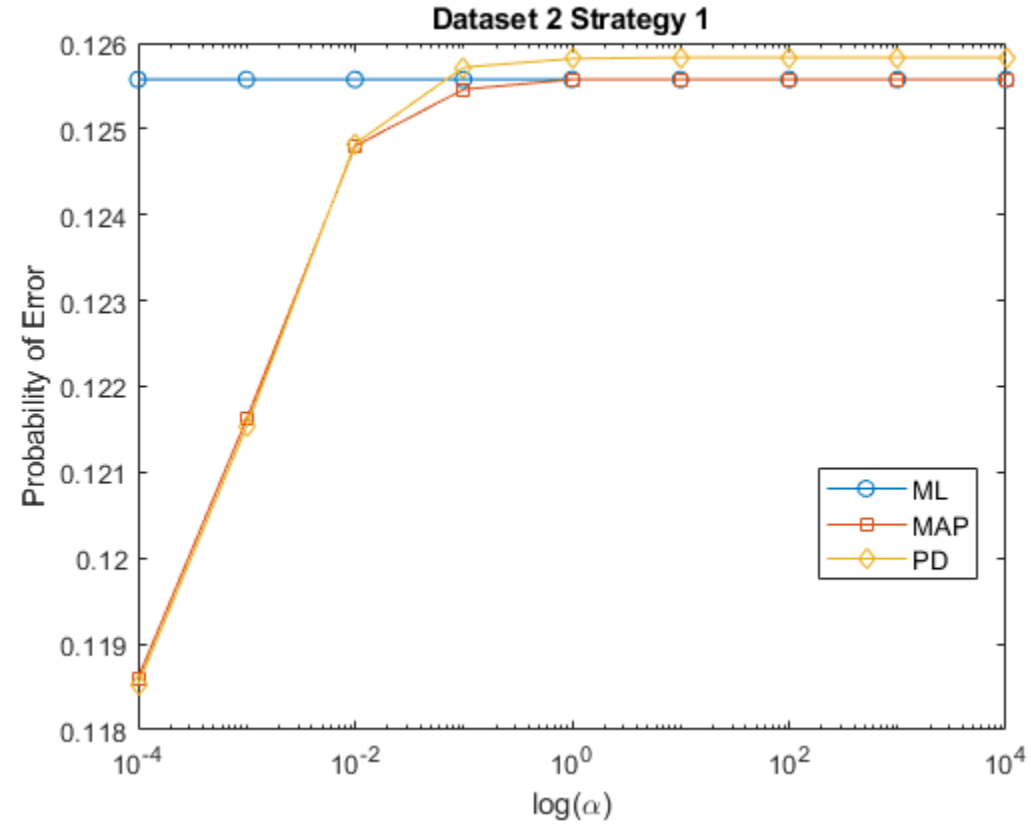
```
MAP = [];
ML = [];
PD = [];

for i = 1:9
    currentAlpha = alpha(i);
    adjustedPriorCovariance = currentAlpha * priorCovariance;
    updatedForegroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG)
* foregroundMean' + foregroundCovariance / numSamplesFG *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) * mu0_FG';
    updatedBackgroundMean = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG)
* backgroundMean' + backgroundCovariance / numSamplesBG *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) * mu0_BG';
    updatedForegroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + foregroundCovariance / numSamplesFG) *
foregroundCovariance / numSamplesFG;
    updatedBackgroundCovariance = adjustedPriorCovariance *
inv(adjustedPriorCovariance + backgroundCovariance / numSamplesBG) *
backgroundCovariance / numSamplesBG;
    finalForegroundCovariance = updatedForegroundCovariance +
foregroundCovariance;
    finalBackgroundCovariance = updatedBackgroundCovariance +
backgroundCovariance;
    index = 1;
    PD_FG = 0;
    PD_BG = 0;
    ML_FG = 0;
    ML_BG = 0;
    MAP_FG = 0;
    MAP_BG = 0;

    for j = 1:m-7
        for h = 1:n-7
            X = featureMatrix(1:64, index);

            % Predictive
            i1_PD = -0.5 * (X - updatedBackgroundMean)' *
inv(finalBackgroundCovariance) * (X - updatedBackgroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalBackgroundCovariance))
+ log(probabilityBG);
            i2_PD = -0.5 * (X - updatedForegroundMean)' *
inv(finalForegroundCovariance) * (X - updatedForegroundMean) ...
                    - 0.5 * log((2 * pi)^64 * det(finalForegroundCovariance))
+ log(probabilityFG);

            if i1_PD > i2_PD
                A(j, h) = 0;
            else
                A(j, h) = 255;
            end

            if uint8(A(j, h)) == 255 && mask(j, h) == 0
```

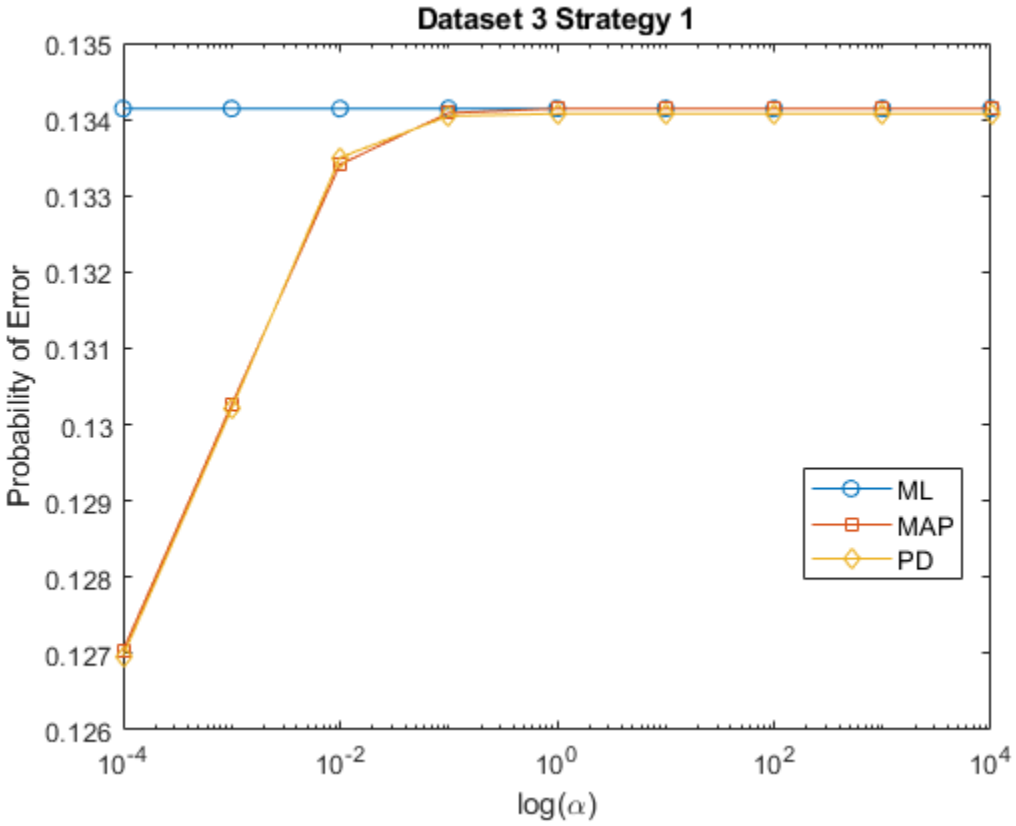
```
            PD_BG = PD_BG + 1;
        end

        if uint8(A(j, h)) == 0 && mask(j, h) == 255
            PD_FG = PD_FG + 1;
        end


        % MAP
        i1_MAP = -0.5 * (X - updatedBackgroundMean)' *
inv(backgroundCovariance) * (X - updatedBackgroundMean) ...
                 - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);

        i2_MAP = -0.5 * (X - updatedForegroundMean)' *
inv(foregroundCovariance) * (X - updatedForegroundMean) ...
                 - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);


        if i1_MAP > i2_MAP
            A(j, h) = 0;
        else
            A(j, h) = 255;
        end

        if uint8(A(j, h)) == 255 && mask(j, h) == 0
            MAP_BG = MAP_BG + 1;
        end

        if uint8(A(j, h)) == 0 && mask(j, h) == 255
            MAP_FG = MAP_FG + 1;
        end

        index = index + 1;


        % ML
        i1_ML = -0.5 * (X - backgroundMean')' *
inv(backgroundCovariance) * (X - backgroundMean') ...
                - 0.5 * log((2 * pi)^64 * det(backgroundCovariance)) +
log(probabilityBG);
        i2_ML = -0.5 * (X - foregroundMean')' *
inv(foregroundCovariance) * (X - foregroundMean') ...
                - 0.5 * log((2 * pi)^64 * det(foregroundCovariance)) +
log(probabilityFG);

        if i1_ML > i2_ML
            A(j, h) = 0;
        else
            A(j, h) = 255;
        end

        if uint8(A(j, h)) == 255 && mask(j, h) == 0
```

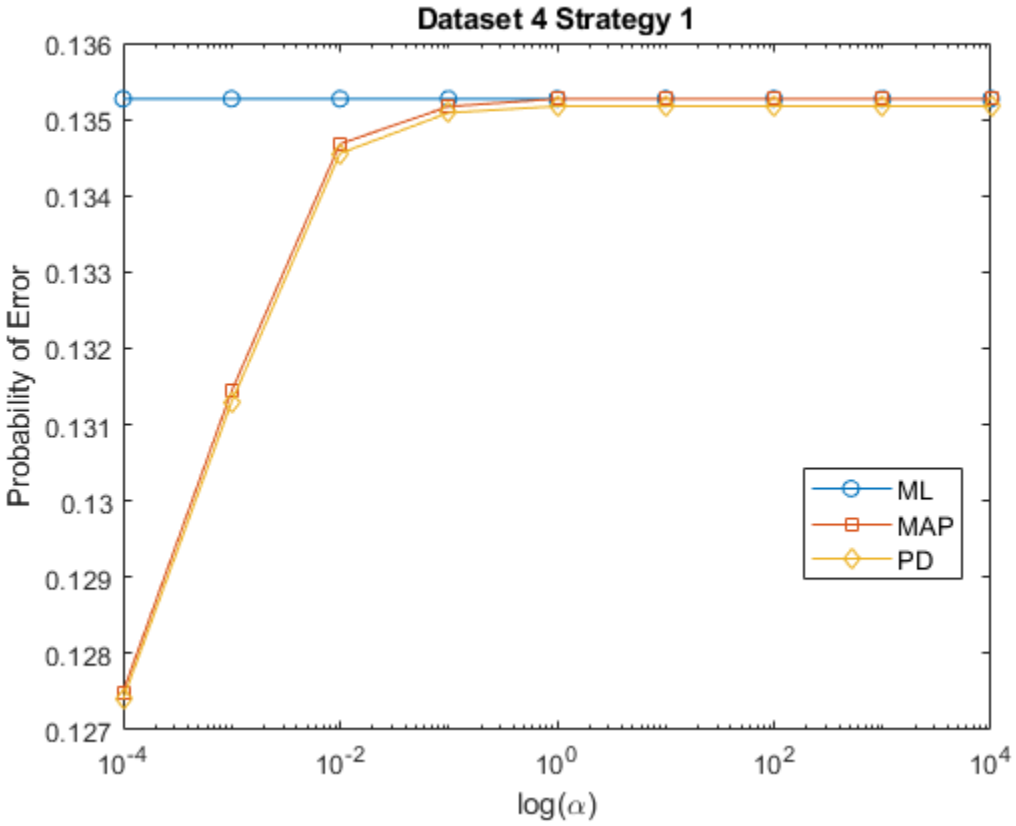
```
                ML_BG = ML_BG + 1;
            end

            if uint8(A(j, h)) == 0 && mask(j, h) == 255
                ML_FG = ML_FG + 1;
            end

        end
    end

    PD(i) = PD_BG / sumBG * probabilityBG + PD_FG / sumFG * probabilityFG;
    ML(i) = ML_BG / sumBG * probabilityBG + ML_FG / sumFG * probabilityFG;
    MAP(i) = MAP_BG / sumBG * probabilityBG + MAP_FG / sumFG * probabilityFG;
end

figure;
plot(alpha, ML, '-o', 'DisplayName', 'ML');
hold on;
plot(alpha, MAP, '-s', 'DisplayName', 'MAP');
plot(alpha, PD, '-d', 'DisplayName', 'PD');
set(gca, 'XScale', 'log');
xlabel('log(\alpha)');
ylabel('Probability of Error');
title('Dataset 4 Strategy 2');
legend('Location', 'best', 'FontSize', 10);
set(gca);
```

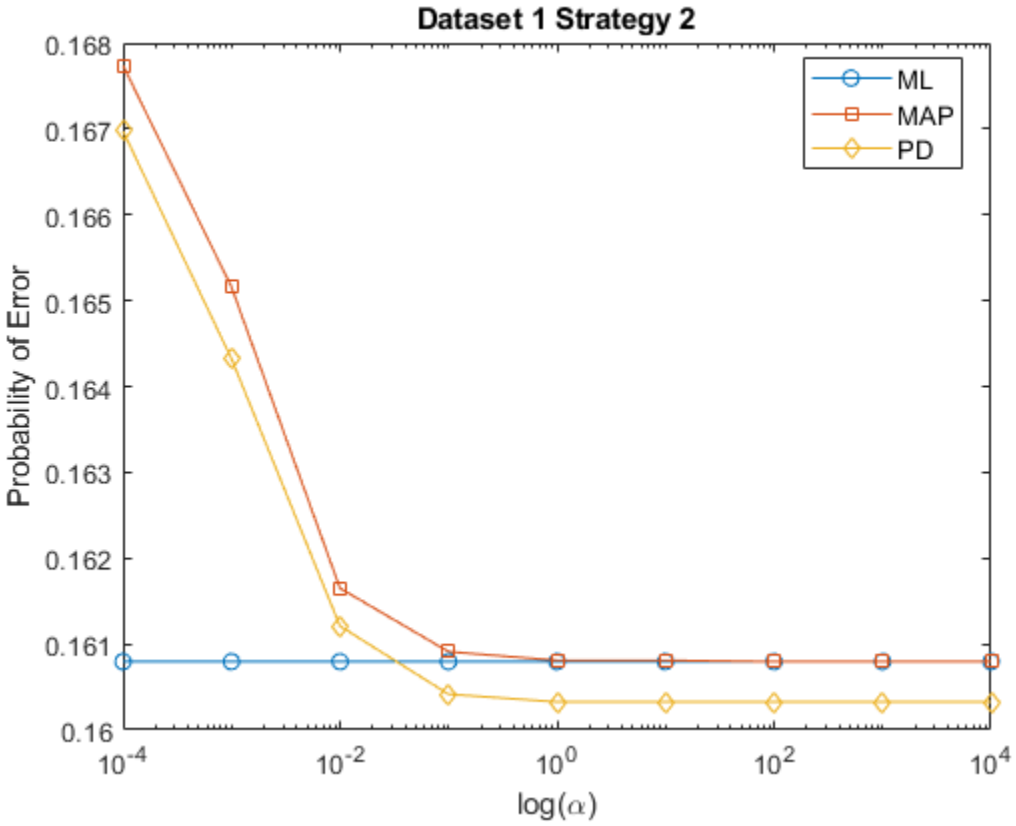
```
                      ALim: {}
                   ALimMode: {'auto'  'manual'}
                 AlphaScale: {'linear'  'log'}
                   Alphamap: {}
           AmbientLightColor: {1×0 cell}
                        Box: {[on]  [off]}
                  BoxStyle: {'full'  'back'}
                BusyAction: {'queue'  'cancel'}
             ButtonDownFcn: {}
                      CLim: {}
                   CLimMode: {'auto'  'manual'}
              CameraPosition: {}
            CameraPositionMode: {'auto'  'manual'}
                 CameraTarget: {}
               CameraTargetMode: {'auto'  'manual'}
                 CameraUpVector: {}
             CameraUpVectorMode: {'auto'  'manual'}
                CameraViewAngle: {}
            CameraViewAngleMode: {'auto'  'manual'}
                    Children: {}
                    Clipping: {[on]  [off]}
                ClippingStyle: {'rectangle'  '3dbox'}
                        Color: {1×0 cell}
                   ColorOrder: {}
               ColorOrderIndex: {}
                    ColorScale: {'linear'  'log'}
                      Colormap: {}
```

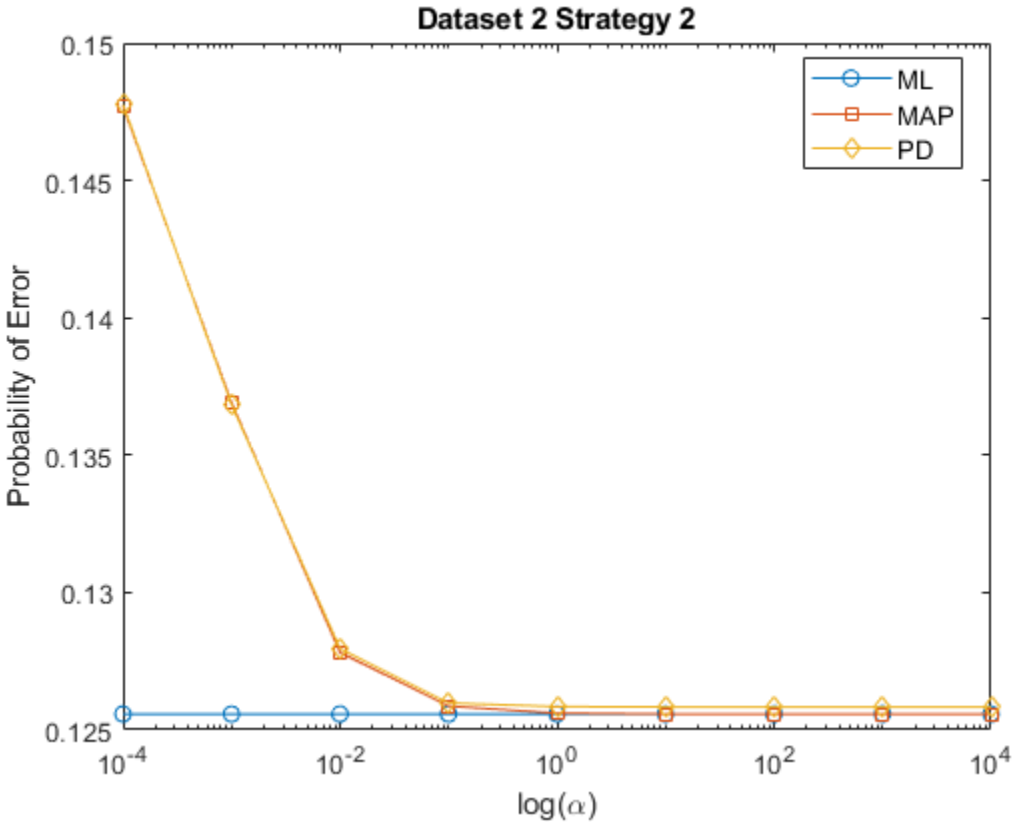

Dataset 1 Strategy 1
Probability of Error
0.161
0.1605
0.16
0.1595
0.159
0.1585
0.158
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)
ML
MAP
PD

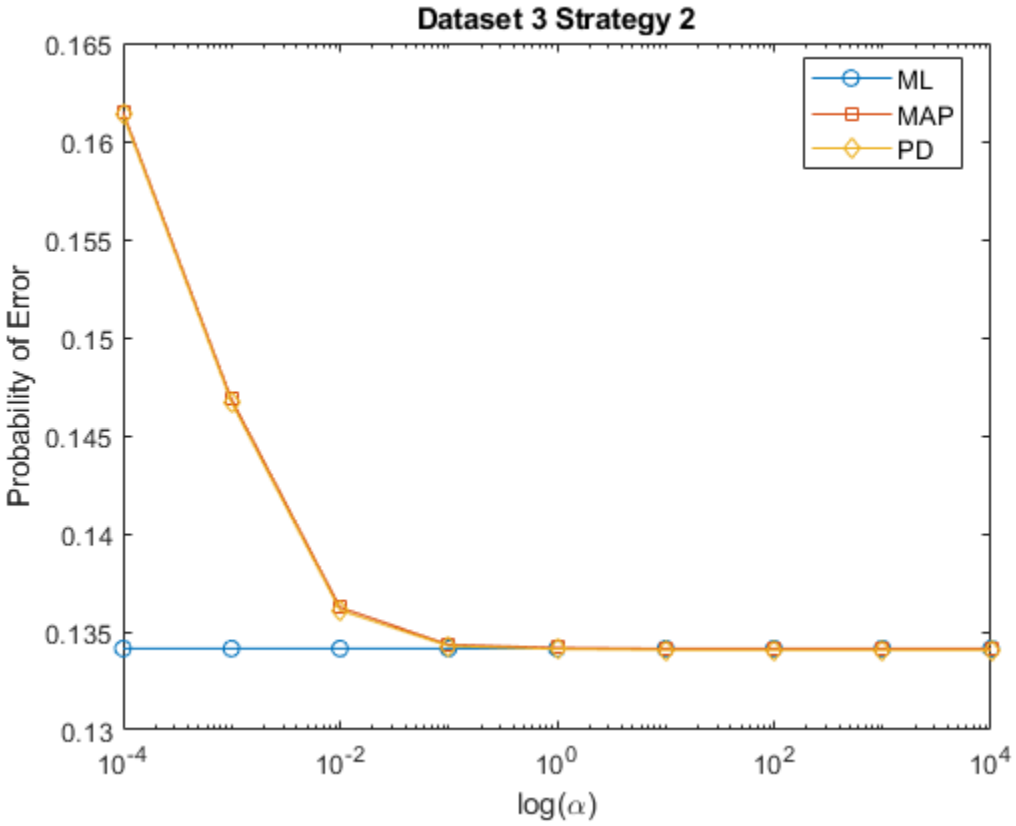

Dataset 2 Strategy 1
Probability of Error
0.126
0.125
0.124
0.123
0.122
0.121
0.12
0.119
0.118
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)
ML
MAP
PD

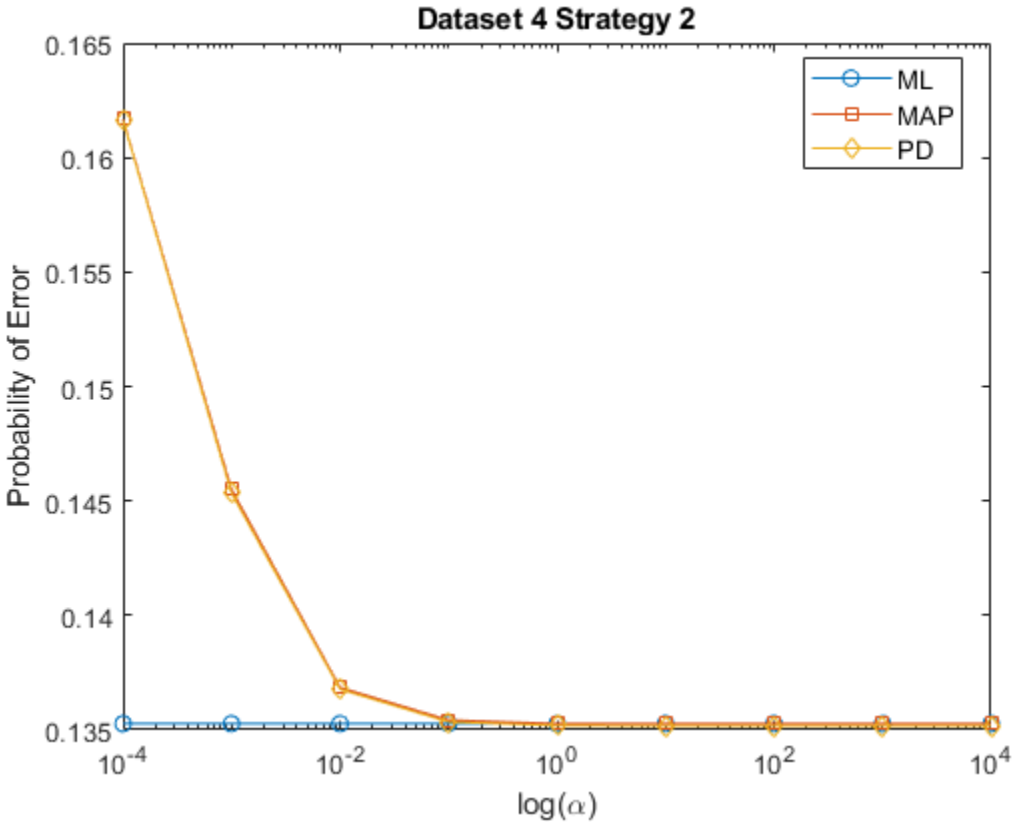

Dataset 3 Strategy 1
Probability of Error
0.135
0.134
0.133
0.132
0.131
0.13
0.129
0.128
0.127
0.126
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)
ML
MAP
PD


Dataset 4 Strategy 1
Probability of Error
0.136
0.135
0.134
0.133
0.132
0.131
0.13
0.129
0.128
0.127
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)
ML
MAP
PD


Dataset 1 Strategy 2
ML
MAP
PD
Probability of Error
0.168
0.167
0.166
0.165
0.164
0.163
0.162
0.161
0.16
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)


Dataset 2 Strategy 2
ML
MAP
PD
Probability of Error
0.15
0.145
0.14
0.135
0.13
0.125
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)


Dataset 3 Strategy 2
ML
MAP
PD
Probability of Error
0.165
0.16
0.155
0.15
0.145
0.14
0.135
0.13
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)


Dataset 4 Strategy 2
ML
MAP
PD
Probability of Error
0.165
0.16
0.155
0.15
0.145
0.14
0.135
10^{-4}
10^{-2}
10^{0}
10^{2}
10^{4}
log(α)