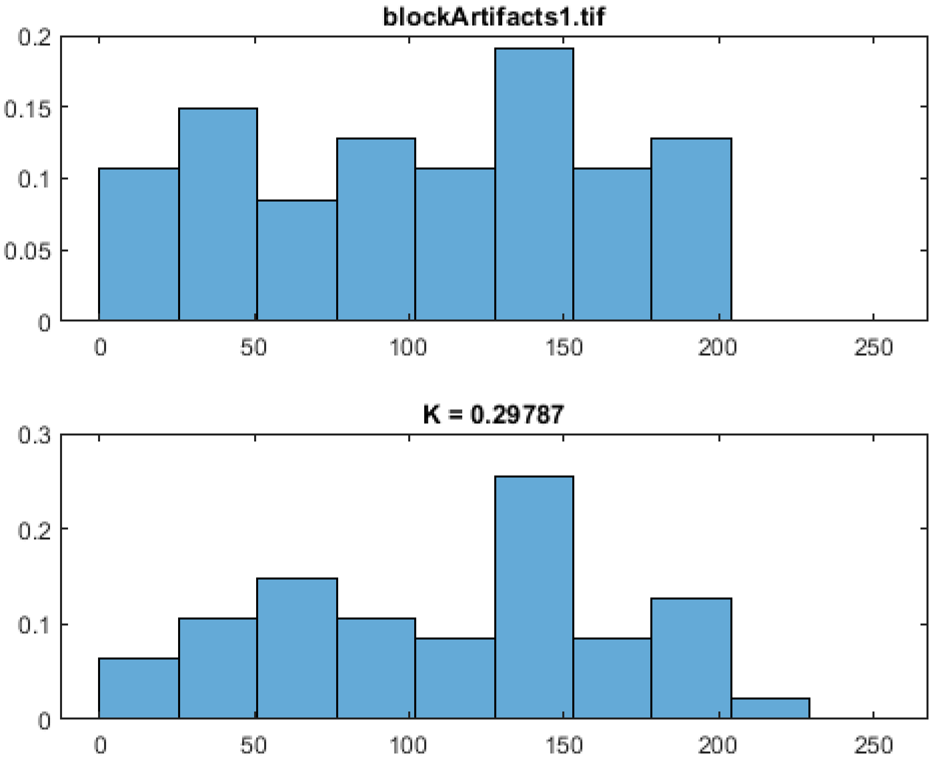
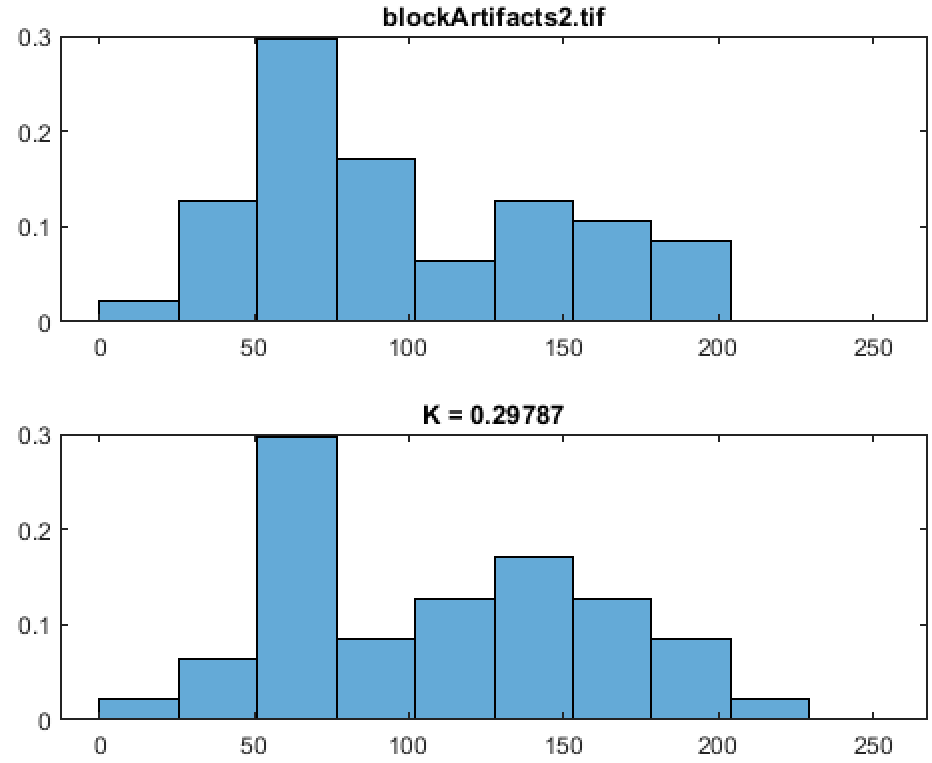
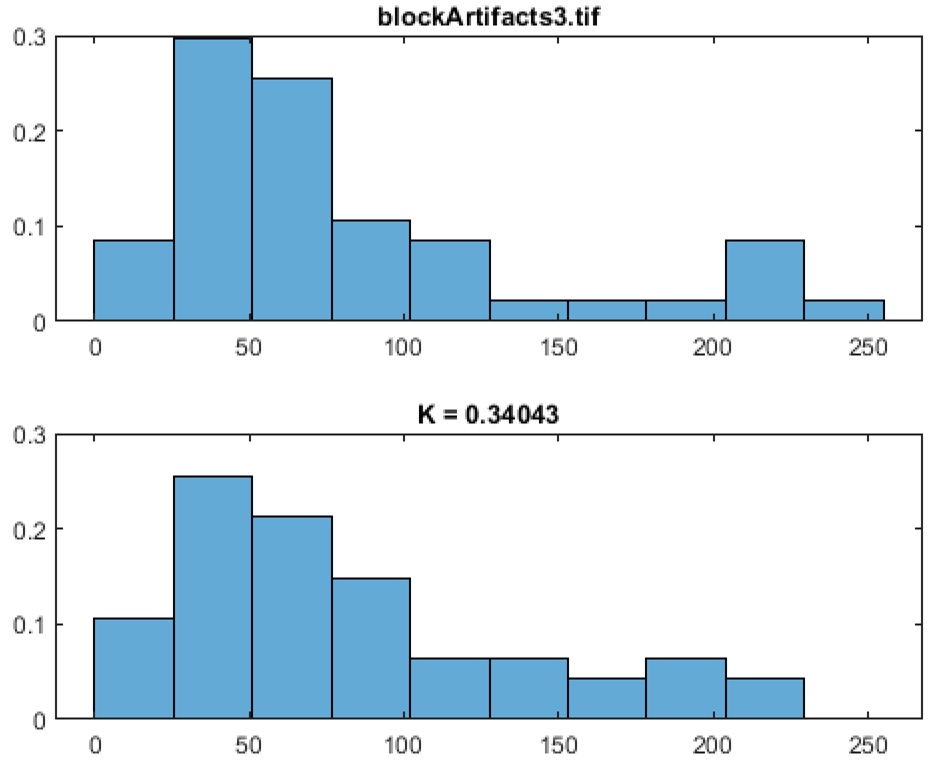
```
%ECES435 Assignment 4 Part 2 - By Wanyu Li and John Seitz
close all; clear all; clc;

% Examing 3 block artifact images using the created function to
 implement
% Fan and de Quieroz's JPEG blocking artifact detection algorithm

Imgs =
 {'blockArtifacts1.tif','blockArtifacts2.tif','blockArtifacts3.tif'};

for i=1:length(Imgs)
    figure(i)
    K = ArtifactDetector(Imgs{i});

    if K > 0.25 % Detect evidence of JPEG compression if K>n and
 classifies image as never compressed if K<n
    sprintf('Evidence of JPEG Compression')
else
    sprintf('Never Compressed')

    end
end
```

```
ans =
    'Evidence of JPEG Compression'
ans =
    'Evidence of JPEG Compression'
ans =
    'Evidence of JPEG Compression'
```


blockArtifacts1.tif
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
K = 0.29787
0.3
0.2
0.1
0
0
50
100
150
200
250


blockArtifacts2.tif
0.3
0.2
0.1
0
0
50
100
150
200
250
K = 0.29787
0.3
0.2
0.1
0
0
50
100
150
200
250



Function of Artifact Detector is below

```
type ArtifactDetector.m


function [K_prime] = ArtifactDetector(Imgs)

%This function is to implement the Fan and de Quieroz's JPEG blocking
%artifact detection algorithm following the steps outlined in the
%assignment PDF.

Img = imread(Imgs); % Read in the image

[x,y] = size(Img); % Set variables x and y to the size of the image
x = floor(x/8)-1; % Round the dimenions of the image, leaving out the
 right and bottom sides
y = floor(y/8)-1; % Round the dimenions of the image
Z_prime = zeros(x,y); % Create matrix of zeros of matching size
Z_2prime = zeros(x,y); % Create matrix of zeros of matching size

for i = 1:x % Repeat for each
    for j = 1:y
        new_1 = 8*(i-1)+4; % Where Z_prime is in the center of the
 block
        Z_prime(i,j) = abs(Img(new_1,new_1)-Img(new_1,new_1+1)-
Img(new_1+1,new_1)+Img(new_1+1,new_1+1));
```

```
        new_2 = 8*i; % Where Z_2prime is at the corner of blocks
        Z_2prime(i,j) = abs(Img(new_2,new_2)-Img(new_2,new_2+1)-
Img(new_2+1,new_2)+Img(new_2+1,new_2+1));
    end
end

bound = linspace(0,255,11); % Create boundaries for both histograms to
 ensure the same x axis

subplot(2,1,1) % plot the first histogram
H_1 =
 histogram(Z_prime,'BinEdges',bound,'Normalization','probability');
title(string(Imgs)) % Title is the name of the image file
K_prime = H_1.Values;

subplot(2,1,2) % plot the second histogram
H_11 =
 histogram(Z_2prime,'BinEdges',bound,'Normalization','probability');
K_2prime = H_11.Values;
K_prime = sum(abs(K_prime-K_2prime)); % Get the K value from the sum
 of the absolute value of H_11 minus H_11
title(['K = ',num2str(K_prime)]); % Title is the calculated K value

end
```

Published with MATLAB® R2019b