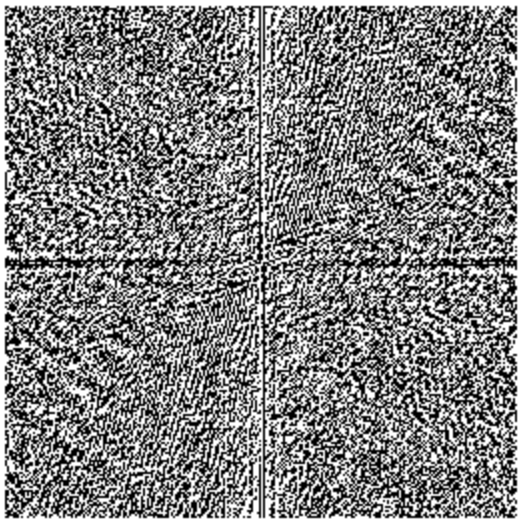
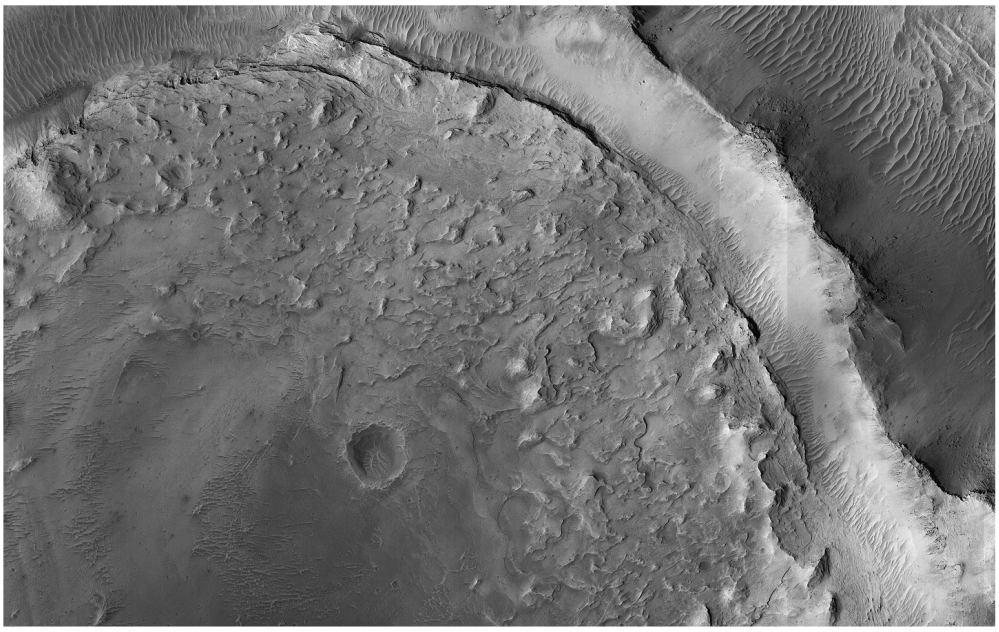
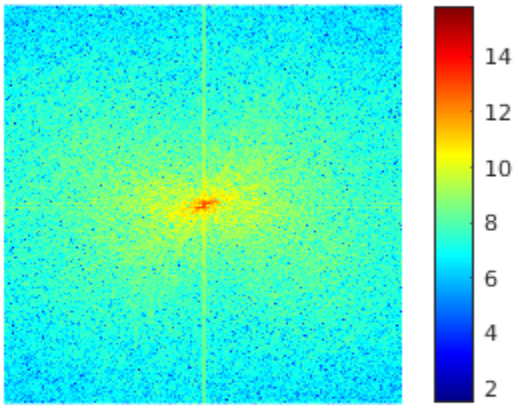
```
clear;close all; clc;
% Prepare image
f = imread('ImgPIA.jpg');
imshow(f);
%Convert to gray
Igray = rgb2gray(f);
%Reszie the image to 256,256
resized = imresize(Igray, [256 256]);
%2D fast fourier transform
f = fft2(resized, 256, 256);
%Shift to centre
f = fftshift(f);
%Show new image
figure;
imshow(f);
%Show the colormap of the transformed image
figure;
transformed_img = log(1+abs(f));
imshow(transformed_img,[]); colormap(jet); colorbar;

%log the image for analysis
transformed_img_log = log(transformed_img);
%S(R) array for values
SR = [];
%S(theta) array for values
ST = [];


%For the radius r (half of image since it will go to edge)
for r = 0:127
    %set S_R to 0
    S_R = 0;
    %for theta between 0 and -pi
    for theta = 0:-1:-180
    %degrees to radians
    theta = deg2rad(theta);
    %transform the polar coordinates from polar to cartesian
    [x, y] = pol2cart(theta, r);
    %round the value add the other half of the image
    x = round(x + 128);
    y = round(y + 128);
    %S_R is S_R + the location of x and y in the log of the fourier
    %transform image
    S_R = S_R + transformed_img_log(x, y);
    end
    %SR to plot is SR and S_R location
    SR = [SR; S_R];
end
%plot figure for S(R)
figure;
plot(log(SR))
xlabel('Radius 0 to -pi')
```

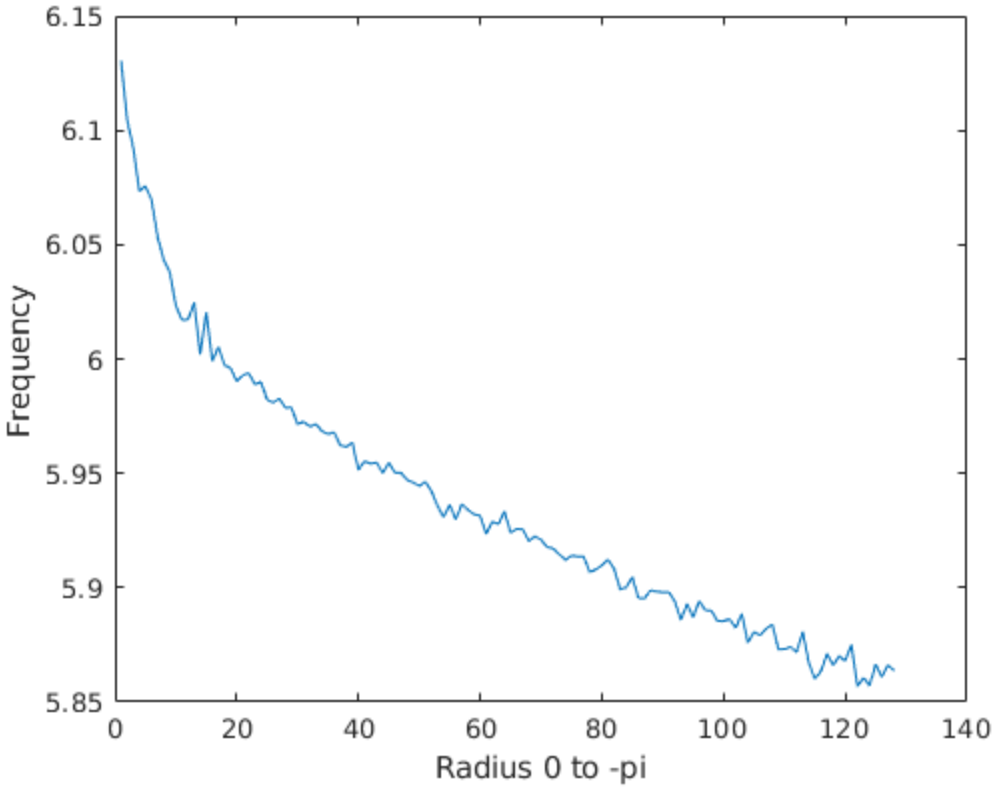
```
ylabel('Frequency')

%Theta for loop 0 to -pi
for theta = 0:-1:-180
    %degrees to radians
    theta = deg2rad(theta);
    %set S_T to 0
    S_T = 0;
    %For the radius r (half of image since it will go to edge)
    for r = 0:127
        %transform the polar coordinates from polar to cartesian
        [x,y] = pol2cart(theta, r);
        %round the value add the other half of the image
        x = round(x + 128);
        y = round(y + 128);
        %S_T is S_T + the location of x and y in the log of the
 fourier
        %transform image
        S_T = S_T + transformed_img_log(x, y);
    end
    %ST to plot is ST and S_T location
    ST = [ST; S_T];
end
%plot figure for S(theta)
figure;
plot(log(ST))
xlabel('Radius 0 to -pi')
ylabel('Frequency')

%For the radius r (half of image since it will go to edge)
for r = 0:127
    %set S_R to 0
    S_R = 0;
    %for theta between 0 and pi
    for theta = 0:-1:180
    %degrees to radians
    theta = deg2rad(theta);
    %transform the polar coordinates from polar to cartesian
    [x, y] = pol2cart(theta, r);
    %round the value add the other half of the image
    x = round(x + 128); %Half of the image length
    y = round(y + 128);
    %S_R is S_R + the location of x and y in the log of the fourier
    %transform image
    S_R = S_R + transformed_img_log(x, y);
    end
    %SR to plot is SR and S_R location
    SR = [SR; S_R];
end
figure;
plot(log(SR))
xlabel('Radius 0 to pi')
ylabel('Frequency')
```

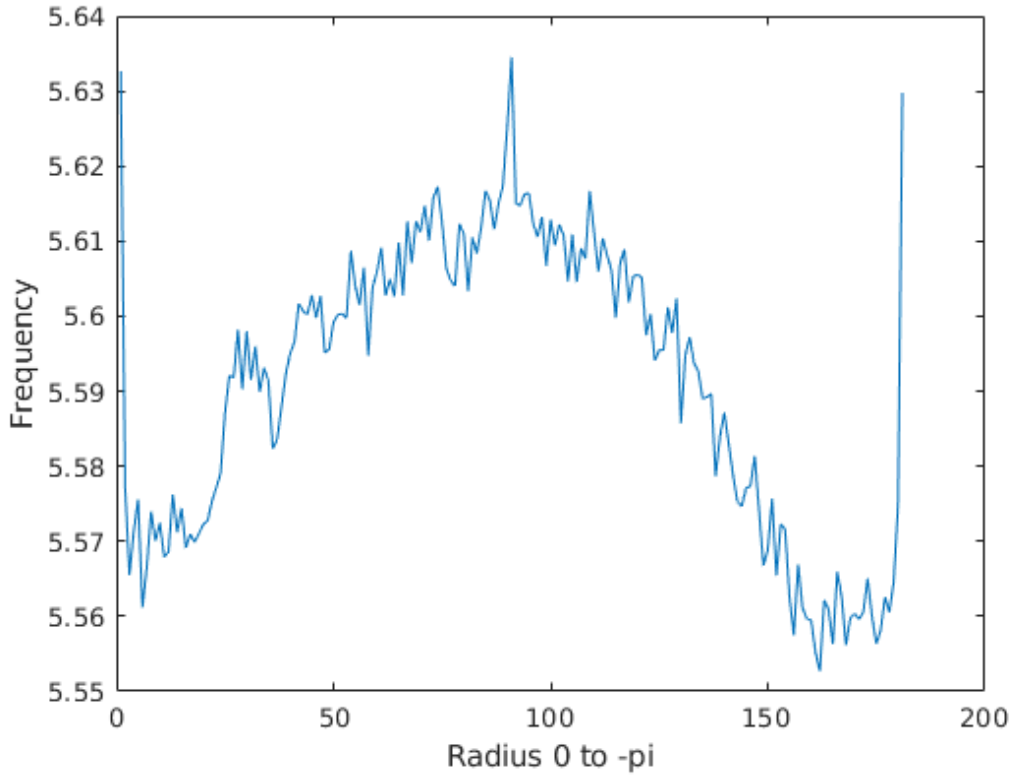
```
%Theta for loop 0 to pi
for theta = 0:1:180
    %degrees to radians
    theta = deg2rad(theta);
    %set S_T to 0
    S_T = 0;
    %For the radius r (half of image since it will go to edge)
    for r = 0:127
        %transform the polar coordinates from polar to cartesian
        [x,y] = pol2cart(theta, r);
        %round the value add the other half of the image
        x = round(x + 128);
        y = round(y + 128);
        %S_T is S_T + the location of x and y in the log of the
 fourier
        %transform image
        S_T = S_T + transformed_img_log(x, y);
    end
    %ST to plot is ST and S_T location
    ST = [ST; S_T];
end
%plot figure for S(theta)
figure;
plot(log(ST))
xlabel('Radius 0 to pi')
ylabel('Frequency')
```

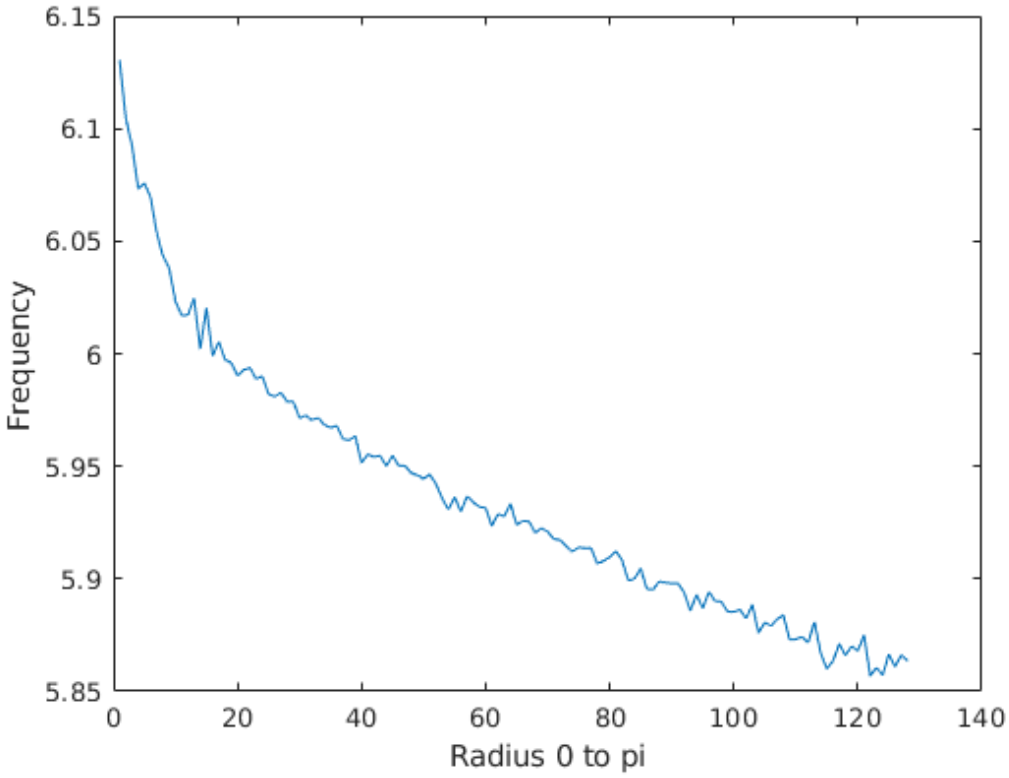


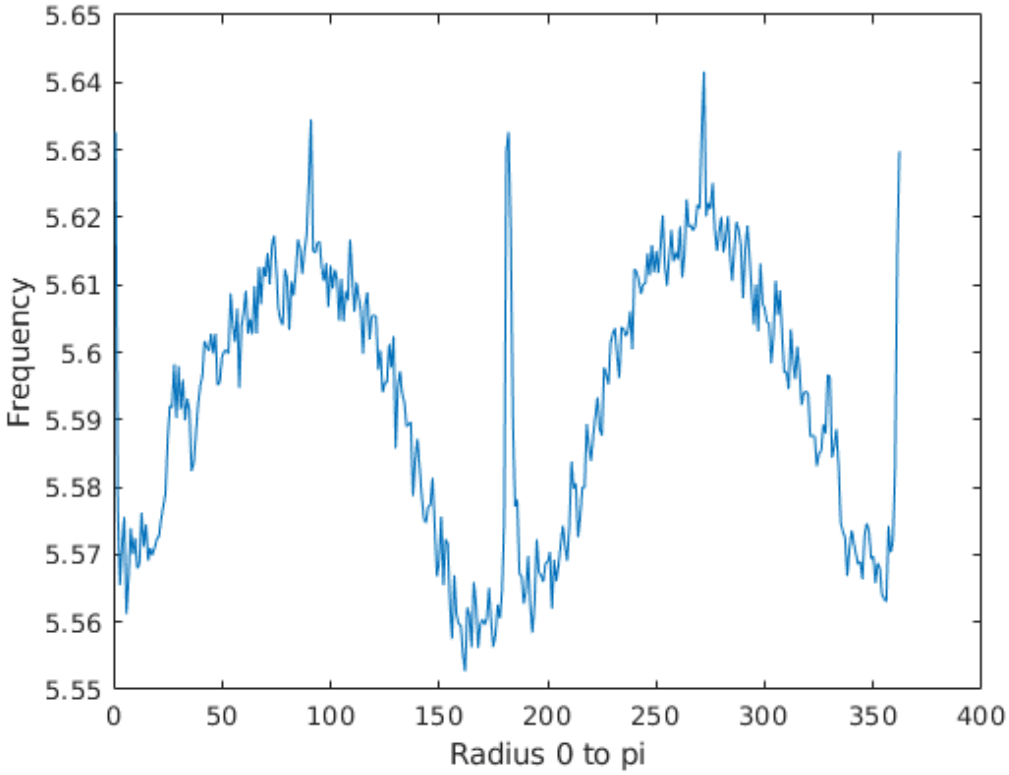



14
12
10
8
6
4
2


6.15
6.1
6.05
6
5.95
5.9
5.85
Frequency
0
20
40
60
80
100
120
140
Radius 0 to -pi


5.64
5.63
5.62
5.61
5.6
5.59
5.58
5.57
5.56
5.55
Frequency
0
50
100
150
200
Radius 0 to -pi


6.15
6.1
6.05
6
5.95
5.9
5.85
Frequency
0
20
40
60
80
100
120
140
Radius 0 to pi


5.65
5.64
5.63
5.62
5.61
5.6
5.59
5.58
5.57
5.56
5.55
Frequency
0
50
100
150
200
250
300
350
400
Radius 0 to pi

Published with MATLAB® R2020b