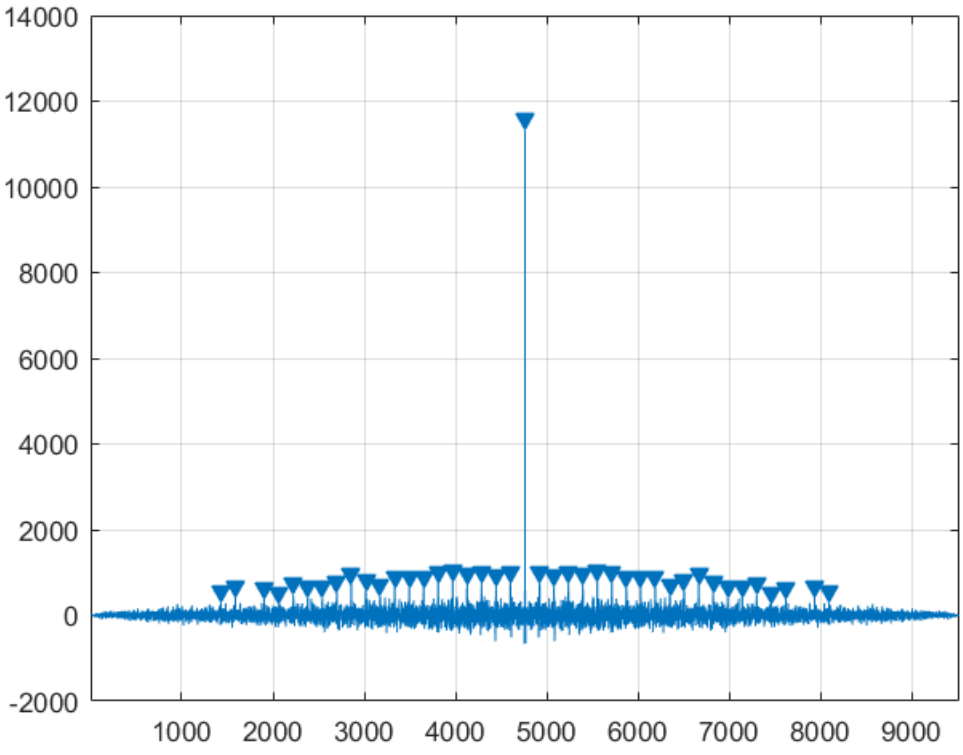
```
clear

A = load('C:\Users\Minh Quan Do\Desktop\GMU\year 3 contents\beng
 320\HW2\ecg_noisy.mat');
y = A.ecg_noisy;
r = xcorr(y,y);
xr = linspace(0, A.Ts, length(r));
xy = linspace(0, A.Ts, length(y));

[pk loc] = findpeaks(r, 'MinPeakDistance', 100, 'MinPeakProminence',
 500);

total_loc = 0;
for i = 1:length(loc)
    if (loc(1,i) < 6000) && (loc(1,i) > 4000)
        range = abs(loc(1,i) - loc(1,i-1));
        total_loc = total_loc + range;
    end
end
loc1 = total_loc;
mean_loc = loc1/(length(loc) - 1)

figure(1)
plot(loc, pk)
findpeaks(r, 'MinPeakDistance', 100, 'MinPeakProminence', 500);

% RR-interval = 47.55
```

```
mean_loc =

   47.5500
```

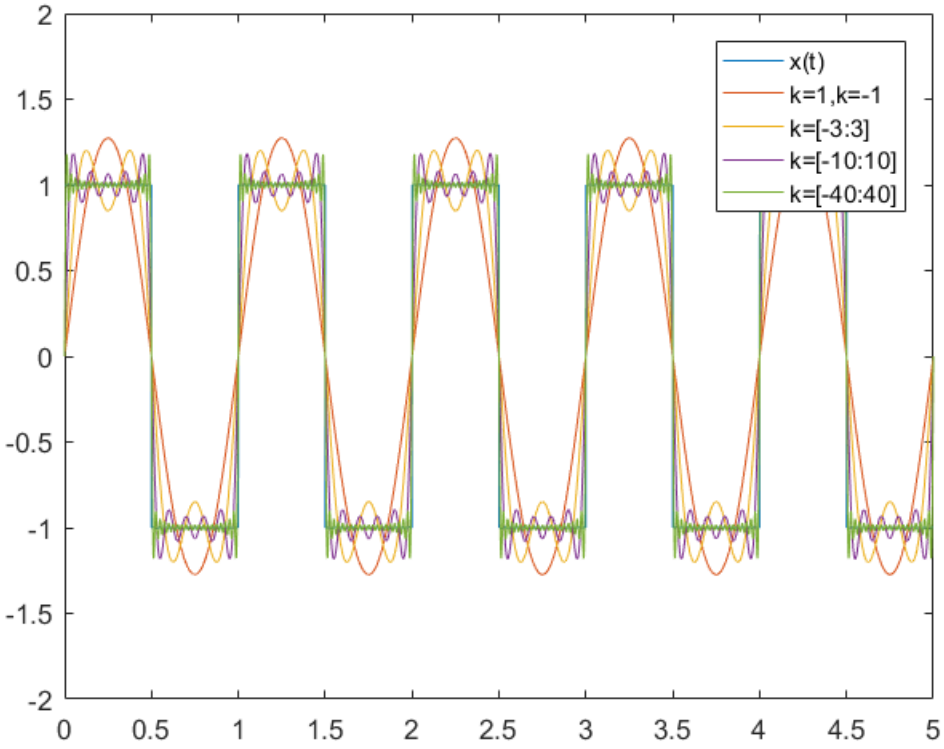


Published with MATLAB® R2016b

```
clear all;

T = 1;
x = linspace(0,5*T,10000);
y_val = [];
w_o = (2*pi)/T;
j = (-1)^(1/2);

for i = 1:length(x/2)
    if (x(i) < T/2)
        y_val(i) = 1;
    elseif (x(i) > T/2) && (x(i) < T)
        y_val(i) = -1;
    end
end
y_val = [y_val y_val y_val y_val y_val];

k = [-1:1];
k = k(k~=0);
values_synth = 0;
for i = 1:length(k)
    wet1 = @(x) exp(-j*k(i)*w_o*x);
    wet2 = @(x) -exp(-j*k(i)*w_o*x);
    p_1 = (1/T)*integral(wet1,0,T/2);
    p_2 = (1/T)*integral(wet2,T/2,T);
    ak = p_1 + p_2;
    values_synth = values_synth + ak*exp(j*k(i)*w_o*x);
end

k2 = [-3:3];
k2 = k2(k2~=0);
values_synth2 = 0;
for i = 1:length(k2)
    wet1 = @(x) exp(-j*k2(i)*w_o*x);
    wet2 = @(x) -exp(-j*k2(i)*w_o*x);
    p_1 = (1/T)*integral(wet1,0,T/2);
    p_2 = (1/T)*integral(wet2,T/2,T);
    ak = p_1 + p_2;
    values_synth2 = values_synth2 + ak*exp(j*k2(i)*w_o*x);
end

k3 = [-10:10];
k3 = k3(k3~=0);
values_synth3 = 0;
for i = 1:length(k3)
    wet1 = @(x) exp(-j*k3(i)*w_o*x);
    wet2 = @(x) -exp(-j*k3(i)*w_o*x);
    p_1 = (1/T)*integral(wet1,0,T/2);
    p_2 = (1/T)*integral(wet2,T/2,T);
    ak = p_1 + p_2;
    values_synth3 = values_synth3 + ak*exp(j*k3(i)*w_o*x);
end
```

```
k4 = [-40:40];
k4 = k4(k4~=0);
values_synth4 = 0;
for i = 1:length(k4)
    wet1 = @(x) exp(-j*k4(i)*w_o*x);
    wet2 = @(x) -exp(-j*k4(i)*w_o*x);
    p_1 = (1/T)*integral(wet1,0,T/2);
    p_2 = (1/T)*integral(wet2,T/2,T);
    ak = p_1 + p_2;
    values_synth4 = values_synth4 + ak*exp(j*k4(i)*w_o*x);
end

figure;
plot(x,y_val);
hold on
plot(x,values_synth)
hold on
plot(x,values_synth2)
hold on
plot(x,values_synth3)
hold on
plot(x,values_synth4)
ylim([-2 2])
legend('x(t)','k=1,k=-1','k=[-3:3]','k=[-10:10]','k=[-40:40]')
```

```
Warning: Imaginary parts of complex X and/or Y arguments ignored
Warning: Imaginary parts of complex X and/or Y arguments ignored
Warning: Imaginary parts of complex X and/or Y arguments ignored
```



Published with MATLAB® R2016b

Question 2 b:

i) $x(t) = \cos(8t) + \cos(20t)$

$$\cos(\omega_o t) = \frac{e^{j\omega_o t} + e^{-j\omega_o t}}{2} = \frac{1}{2}e^{j\omega_o t} + \frac{1}{2}e^{-j\omega_o t}$$

$$e^{j\omega_o t} \xrightarrow{F_\omega} 2\pi\delta(\omega - \omega_o)$$

$$e^{-j\omega_o t} \xrightarrow{F_\omega} 2\pi\delta(\omega + \omega_o)$$

$$X(\omega) = \underbrace{\frac{1}{2}(2\pi\delta(\omega-8)) + \frac{1}{2}(2\pi\delta(\omega+8))}_{\cos(8t)} + \underbrace{\frac{1}{2}(2\pi\delta(\omega-20)) + \frac{1}{2}(2\pi\delta(\omega+20))}_{\cos(20t)}$$

$$\boxed{X(\omega) = \pi[\delta(\omega-8) + \delta(\omega+8)] + \pi[\delta(\omega-20) + \delta(\omega+20)]}$$

ii) $y(t) = \begin{cases} 1 + \cos(4t) & , \ -T_1 < t < T_1 \\ 0 & , \ \text{otherwise} \end{cases}$

$y(t) = y_1(t) + y_2(t)$

$y_1(t) = \begin{cases} 1 & , \ -T_1 < t < T_1 \\ 0 & , \ \text{otherwise} \end{cases} = \text{rect}\left(\frac{t}{T}\right) = T\,\text{sinc}\left(\frac{\omega T}{2}\right); \ T = 2T_1$

$\text{rect}\left(\frac{t}{2T_1}\right) = 2T_1\,\text{sinc}(\omega T_1)$

$y_1(\omega) = 2T_1\,\text{sinc}(\omega T_1) \longrightarrow$

$y_2(t) = \begin{cases} \cos(4t) & , \ -T_1 < t < T_1 \\ 0 & , \ \text{otherwise} \end{cases}$

$y_2(t) = \cos(4t) \cdot \text{rect}\left(\frac{t}{2T_1}\right)$ $\quad \text{rect}\left(\frac{t}{2T_1}\right) = g(t)$

$y_2(t) = \frac{1}{2}\left(e^{j\omega_0 t} + e^{-j\omega_0 t}\right) g(t)$ $\quad \cos(\omega_0 t) = \frac{1}{2}\left(e^{j\omega_0 t} + e^{-j\omega_0 t}\right)$

$y_2(t) = \frac{1}{2} g(t) e^{j\omega_0 t} + \frac{1}{2} g(t) e^{-j\omega_0 t}$ $\quad f(t) e^{j\omega_0 t} \longrightarrow F(\omega - \omega_0)$

$\cos(4t) = \cos(\omega_0 t)$

$\omega_0 = 4$

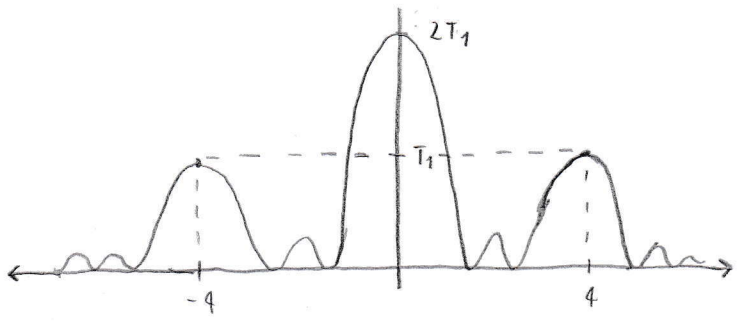
$y_2(\omega) = \frac{1}{2}\left[2T_1\,\text{sinc}((\omega-4)T_1)\right] + \frac{1}{2}\left[2T_1\,\text{sinc}((\omega+4)T_1)\right]$ For $y_2(\omega)$: $\begin{cases} \omega = \omega - \omega_0 \\ \omega = \omega - 4 \\ \text{rect}\left(\frac{t}{2T_1}\right) = 2T_1\,\text{sinc}(\omega-4)T \end{cases}$

$y_2(\omega) = T_1\,\text{sinc}[(\omega-4)T_1] + T_1\,\text{sinc}[(\omega+4)T_1]$

$y(\omega) = y_1(\omega) + y_2(\omega)$

$$\boxed{Y(\omega) = 2T_1\,\text{sinc}(\omega T_1) + T_1\,\text{sinc}[(\omega-4)T_1] + T_1\,\text{sinc}[(\omega+4)T_1]}$$

Magnitude Spectrum



```
clear all

load('C:\Users\Minh Quan Do\Desktop\GMU\year 3 contents\beng
 320\HW2\HW1_Q4_epilepsy_eeg.mat')

alpha = [0.25 0.5 0.75 1 1.25 1.5 2 2.5 3 4 6];

[labels] = eeg_epilepsy_detection_illustrator(eeg_sig1, alpha);

function [labels] = eeg_epilepsy_detection_illustrator(raw_sig,
 alphas)
```

inputs:

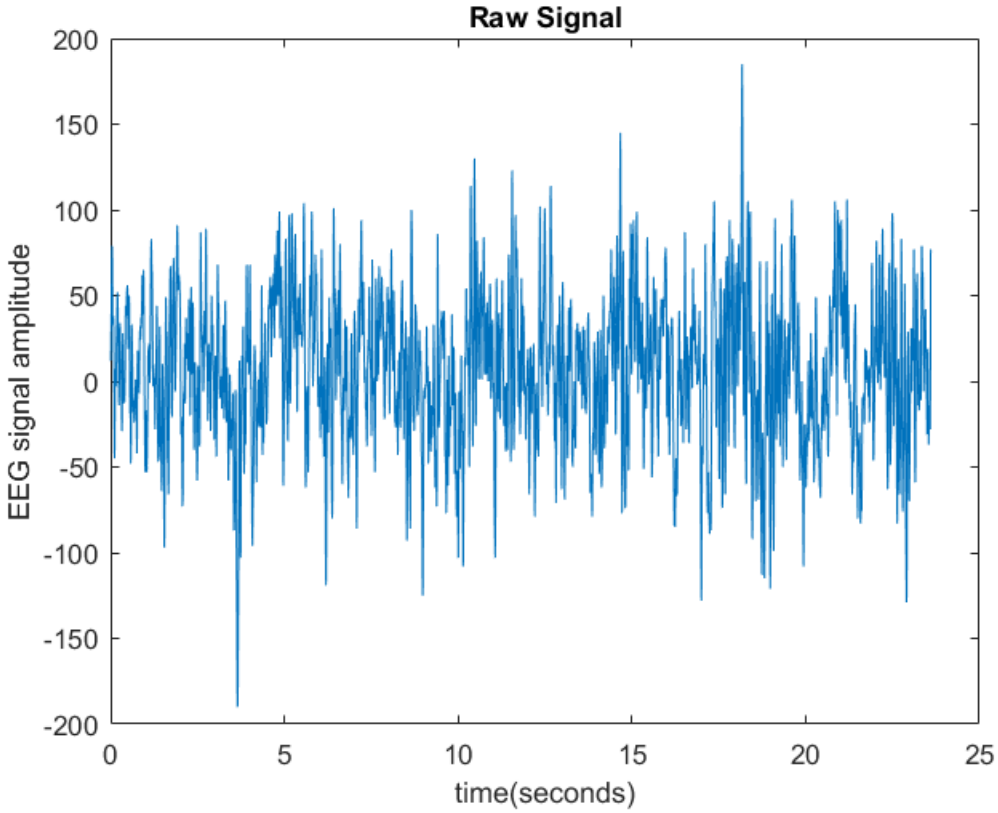
```
%the function takes two inputs
%raw_sig = input data as a time series, representing the eeg recording
 of
%size 1 x N (N is the number of sameples)
% alphas = an array containing the different scalar coefficients to
 set set
% different thresholds as 1 x Nb_thr (Nb_thr is the number of
 thresholds to
% test)
% thr = alphas*sigma; where sigma represents the standard deviation

%output
%labels: A matrix of 1s and 0s containing the thresholded data for
 each threshold with
%the size of output (labels) being Nb_thr x N

%%Complete your code as in the instructions in the homework, note your
 code
%%should work for any arbitrary 1-D signal (of any length) and any
 number
%%of threshold coefficients

    sigma = std(raw_sig);
    thr = alphas*sigma;
    labels = zeros(length(thr), length(raw_sig));

    for i = 1:length(raw_sig)
        for j = 1:length(thr)
            if abs(raw_sig(1,i)) > thr(1,j)
                labels(j,i) = 1;
            end
        end
    end

    neg_thresh = thr * -1;
    T = 1/173.61;
    t = T:T:length(raw_sig)*T;
```

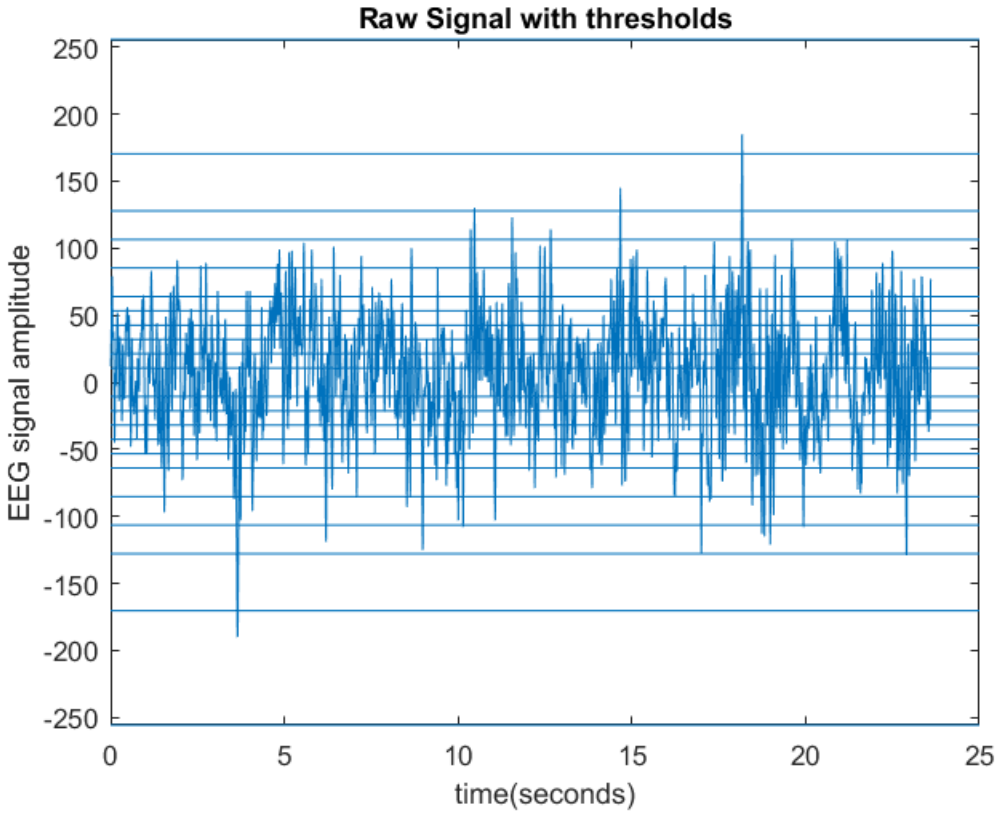
```
    figure(1)
    plot(t,raw_sig)
    title('Raw Signal')
    xlabel('time(seconds)')
    ylabel('EEG signal amplitude')

    figure(2)
    plot(t,raw_sig)
    title('Raw Signal with thresholds')
    xlabel('time(seconds)')
    ylabel('EEG signal amplitude')
    hold on
    for k = 1:length(thr)
        lines = refline(0,thr(k));
        lines.LineStyle(':')

        lines2 = refline(0,neg_thresh(k));
        lines2.LineStyle(':')
    end

    sum_matrix = sum(labels,2)

    figure(3)
    plot(thr,sum_matrix)
    title('threshold vs. number of epileptic seizure events')
    xlabel('threshold')
    ylabel('number of epileptic seizure events')
```

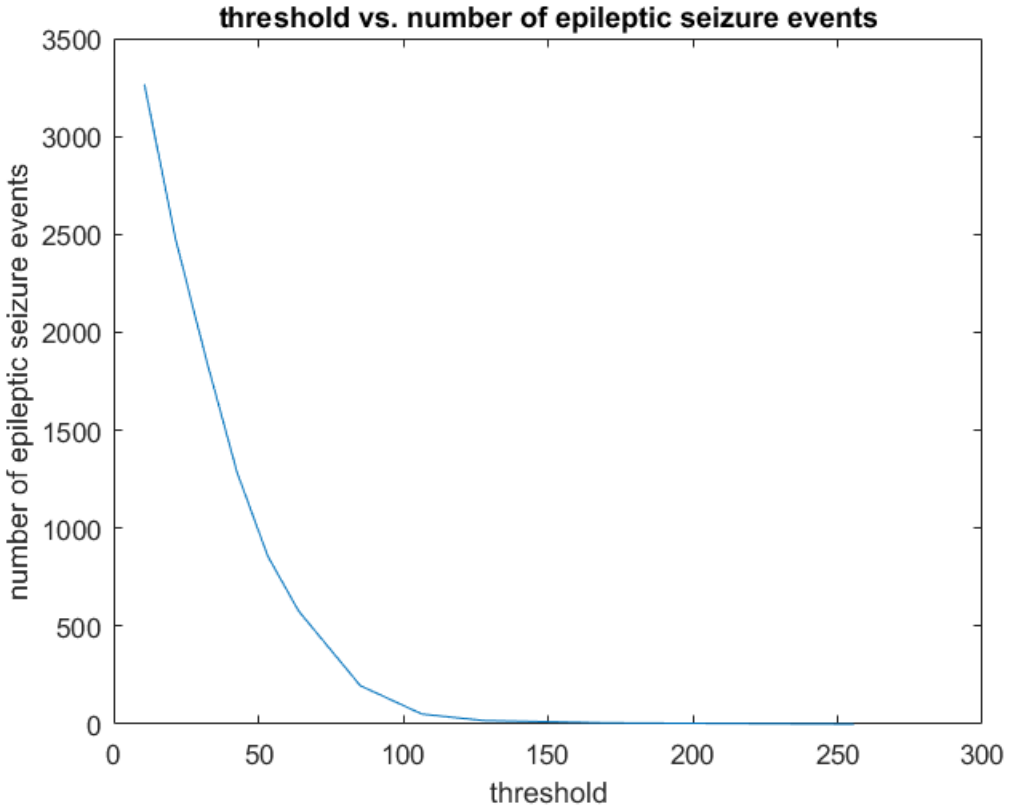
```
ans =

-


ans =

-


ans =

-


ans =

-


ans =

-
```

```
ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-

ans =

-
```

```
ans =

-


ans =

-


ans =

-


ans =

-


ans =

-


ans =

-


sum_matrix =

        3267
        2482
        1864
        1287
         857
         577
         196
          51
          18
           6
           0
```


Raw Signal
EEG signal amplitude
time(seconds)
200
150
100
50
0
-50
-100
-150
-200
0
5
10
15
20
25


Raw Signal with thresholds
EEG signal amplitude
time(seconds)
250
200
150
100
50
0
-50
-100
-150
-200
-250
0
5
10
15
20
25




```
end
```

Published with MATLAB® R2016b

```
% Question
 4:-------------------------------------------------------------
clear

A = load('C:\Users\Minh Quan Do\Desktop\GMU\year 3 contents\beng
 320\HW2\mr_lesions_sample.mat');

mr = A.mr;
mr_noise1 = A.mr_noise1;
mr_noise2 = A.mr_noise2;
mr_noise3 = A.mr_noise3;
tmp = A.tmp;

%
 a:-----------------------------------------------------------------------
figure(1)
imagesc(mr);
newmap = contrast(mr);
colormap(newmap)
title('mr')
% there is no noise in this image whatsoever

figure(2)
imagesc(mr_noise1);
newmap1 = contrast(mr_noise1);
colormap(newmap1)
title('mr noise 1')
% there is a little bit of noise in this image, but the image is still
% easily recognizable

figure(3)
imagesc(mr_noise2);
newmap2 = contrast(mr_noise2);
colormap(newmap2)
title('mr noise 2')
% there is a lot of noise in this image, but the image is somewhat
% recognizable

figure(4)
imagesc(mr_noise3);
newmap3 = contrast(mr_noise3);
colormap(newmap3)
title('mr noise 3')
% there is too much noise in this image, so much to the point where
 the
% image is unrecognizable, no features can be distiguished

%
 b:-----------------------------------------------------------------------
% in part b, i used the xcorr2() function to cross-correlate the two
% matrices
MR_x_tmp = xcorr2(mr,tmp);
```

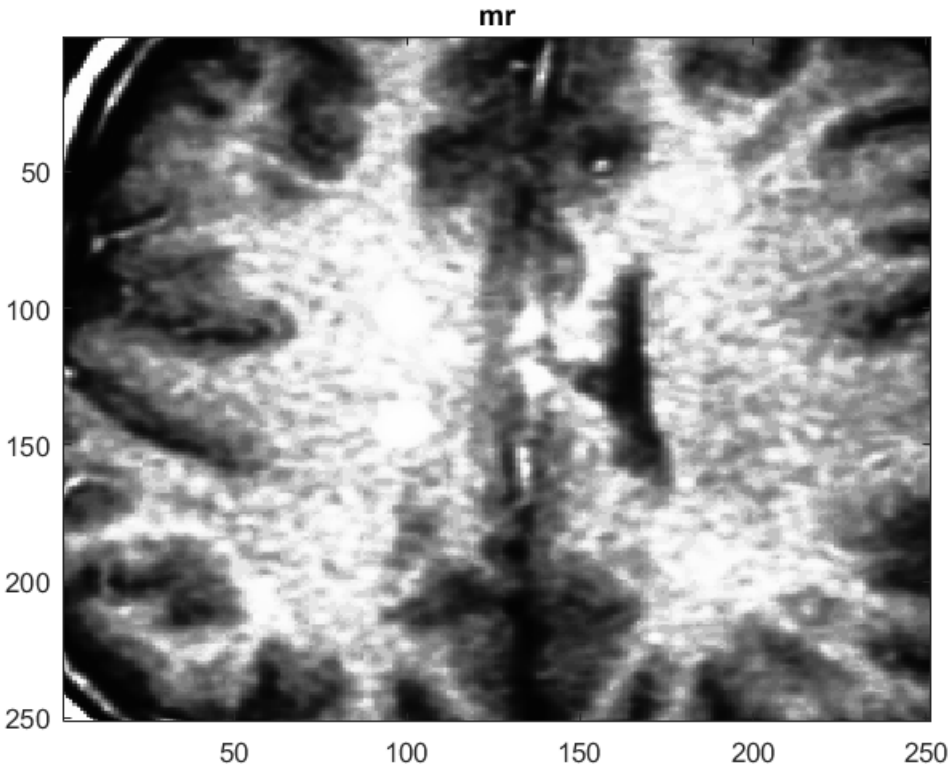
```
mr_noise1_x_tmp = xcorr2(mr_noise1,tmp);
mr_noise2_x_tmp = xcorr2(mr_noise2,tmp);
mr_noise3_x_tmp = xcorr2(mr_noise3,tmp);

%
 c:-----------------------------------------------------------------------
% MRnorm = (MR_x_tmp - min(MR_x_tmp(:)))/(max(MR_x_tmp(:)) -
 min(MR_x_tmp(:)));
% this is one way of normalizing the matrix,however, I'm going to use
 the mat2gray() function because it's shorter
% and easier to read and edit

MRnorm = mat2gray(MR_x_tmp);
mr_noise1_norm = mat2gray(mr_noise1_x_tmp);
mr_noise2_norm = mat2gray(mr_noise2_x_tmp);
mr_noise3_norm = mat2gray(mr_noise3_x_tmp);

figure(5)
imagesc(MRnorm);
newmap_MRnorm = contrast(MRnorm);
colormap(newmap_MRnorm)
title('Normalized MR image')

figure(6)
imagesc(mr_noise1_norm);
newmap_mr_noise1_norm = contrast(mr_noise1_norm);
colormap(newmap_mr_noise1_norm)
title('Normalized mr noise 1 image')

figure(7)
imagesc(mr_noise2_norm);
newmap_mr_noise2_norm = contrast(mr_noise2_norm);
colormap(newmap_mr_noise2_norm)
title('Normalized mr noise 2 image')

figure(8)
imagesc(mr_noise3_norm);
newmap_mr_noise3_norm = contrast(mr_noise3_norm);
colormap(newmap_mr_noise3_norm)
title('Normalized mr noise 3 image')

%
 d:-----------------------------------------------------------------------
thresh = 0.9;
binary_MRnorm = MRnorm < thresh;    %L = logical(MRnorm > 0.9)
binary_mr_noise1_norm = mr_noise1_norm < thresh;
binary_mr_noise2_norm = mr_noise2_norm < thresh;
binary_mr_noise3_norm = mr_noise3_norm < thresh;

figure(9)
imagesc(binary_MRnorm);
newmap_binary_MRnorm = contrast(binary_MRnorm);
colormap(newmap_binary_MRnorm)
title('binary mask applied to normalized mr image')
```

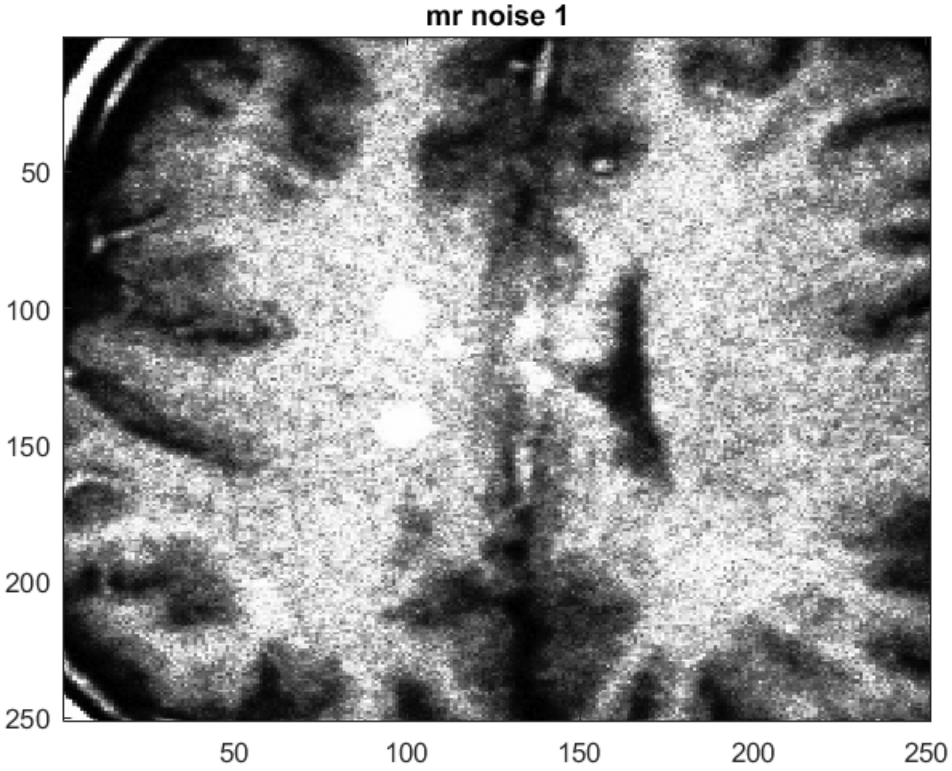
```
figure(10)
imagesc(binary_mr_noise1_norm);
newmap_binary_mr_noise1_norm = contrast(binary_mr_noise1_norm);
colormap(newmap_binary_mr_noise1_norm)
title('binary mask applied to normalized mr noise1 image')

figure(11)
imagesc(binary_mr_noise2_norm);
newmap_binary_mr_noise2_norm = contrast(binary_mr_noise2_norm);
colormap(newmap_binary_mr_noise2_norm)
title('binary mask applied to normalized mr noise2 image')

figure(12)
imagesc(binary_mr_noise3_norm);
newmap_binary_mr_noise3_norm = contrast(binary_mr_noise3_norm);
colormap(newmap_binary_mr_noise3_norm)
title('binary mask applied to normalized mr noise3 image')

% by applying the threshold, i can reveal the location of the lesions
 at
% (100,150) and (100,100) because those are the zero values
% although the instructions said to look for values in matrix MR that
 are
% larger than 0.9, I found that finding values below 0.9 actually
 works.
% the reason for this working is that most of the values of MR is
 actually
% lower than 0.9.
% in a binary colormap, values that are 1 are white, whereas values
 that
% are 0 are black.

%
 e:-----------------------------------------------------------------------
stats = regionprops(binary_MRnorm == 0,'Centroid');
stats1 = regionprops(binary_mr_noise1_norm == 0,'Centroid');
stats2 = regionprops(binary_mr_noise2_norm == 0,'Centroid');
stats3 = regionprops(binary_mr_noise3_norm == 0,'Centroid');
% i set binary_MRnorm == 0 because the black pixels have values of 0.
% therefore, by setting binary_MRnorm == 0, i can tell regionprops()
 to
% find the centroid of the black regions, otherwise, regionprops()
 will
% find the center of the entire image

%
 f:-----------------------------------------------------------------------
num_objects = length(stats); % 2 objects found
num_objects1 = length(stats1);
num_objects2 = length(stats2);
num_objects3 = length(stats3);
%objects_matrix =
 [num_objects,num_objects1,num_objects2,num_objects3];
```

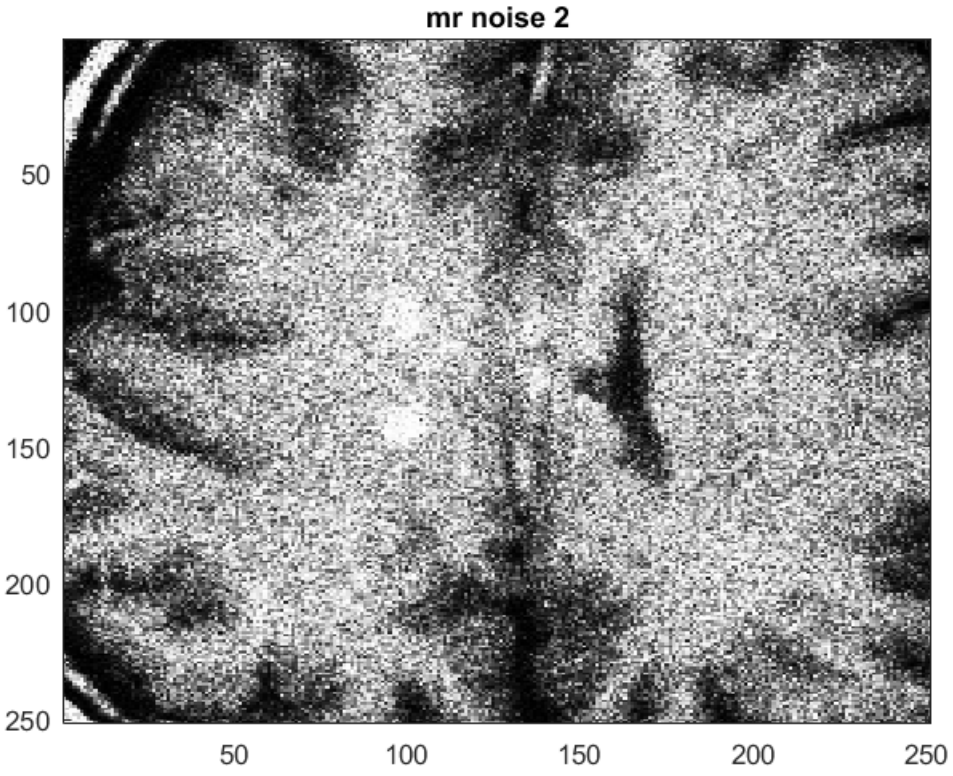
```
%
 g:-----------------------------------------------------------------------
if num_objects > 2
    disp('Input image (mr original) may be too noisy')
    figure(13)
    imagesc(mr)
else
%
 h:-----------------------------------------------------------------------
    figure(13)
    imagesc(mr);
    hold on

    for i = 1:num_objects
        object = stats(i).Centroid;
        objectx = object(1,1) - (0.5*length(tmp));
        objecty = object(1,2) - (0.5*length(tmp));
        plot(objectx,objecty,'r+')
        hold on
    end
    title('mr image with markers')
    legend('= Location of lesions')
end

if num_objects1 > 2
    disp('Input image (mr noise2) may be too noisy')
    figure(14)
    imagesc(mr_noise1)
else
    figure(14)
    imagesc(mr_noise1);
    hold on

    for i = 1:num_objects1
        object = stats1(i).Centroid;
        objectx = object(1,1) - (0.5*length(tmp));
        objecty = object(1,2) - (0.5*length(tmp));
        plot(objectx,objecty,'r+')
        hold on
    end

    title('mr noise1 image with markers')
    legend('= Location of lesions')
end

if num_objects2 > 2
    disp('Input image may (mr noise2) be too noisy')
    figure(15)
    imagesc(mr_noise2)
else
    figure(15)
    imagesc(mr_noise2);
    hold on
```

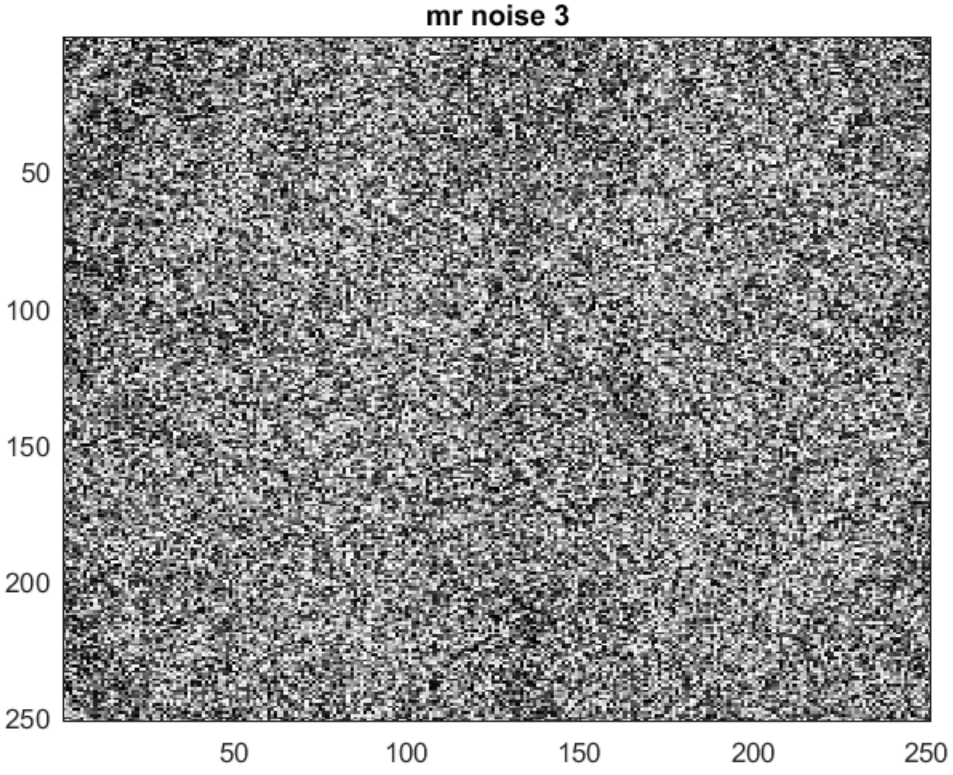
```
    for i = 1:num_objects2
        object = stats2(i).Centroid;
        objectx = object(1,1) - (0.5*length(tmp));
        objecty = object(1,2) - (0.5*length(tmp));
        plot(objectx,objecty,'r+')
        hold on
    end

    title('mr noise2 image with markers')
    legend('= Location of lesions')
end

if num_objects3 > 2
    disp('Input image (mr noise3) may be too noisy')
    figure(16)
    imagesc(mr_noise3)
else
    figure(16)
    imagesc(mr_noise3);
    hold on

    for i = 1:num_objects3
        object = stats3(i).Centroid;
        objectx = object(1,1) - (0.5*length(tmp));
        objecty = object(1,2) - (0.5*length(tmp));
        plot(objectx,objecty,'r+')
        hold on
    end

    title('mr noise3 image with markers')
    legend('= Location of lesions')
end
```

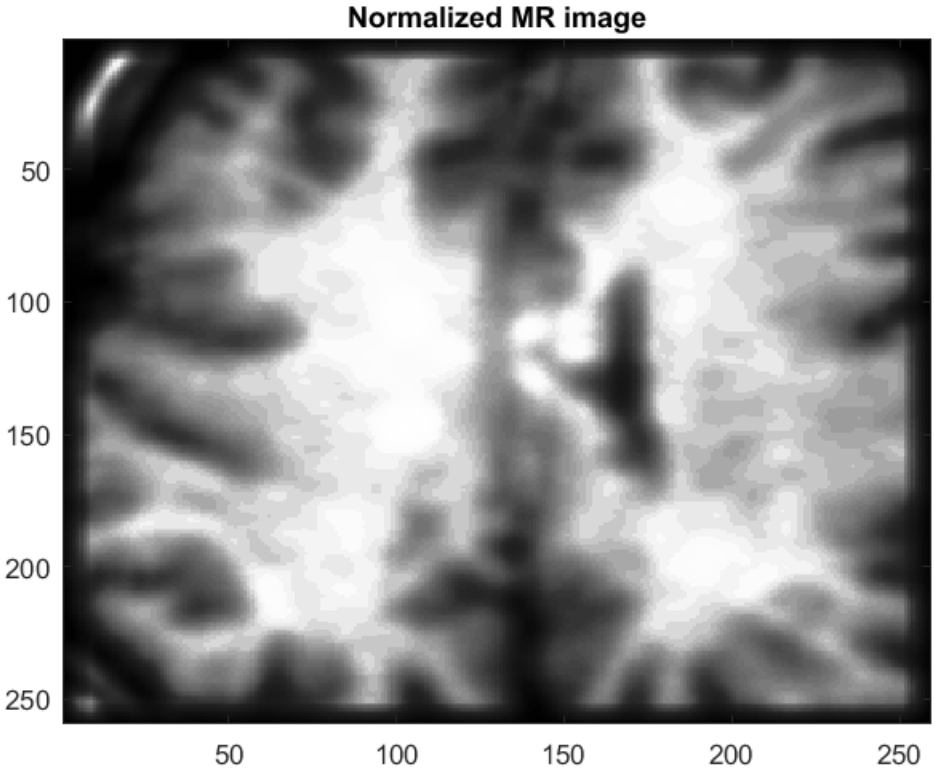
```
Input image (mr noise3) may be too noisy
```

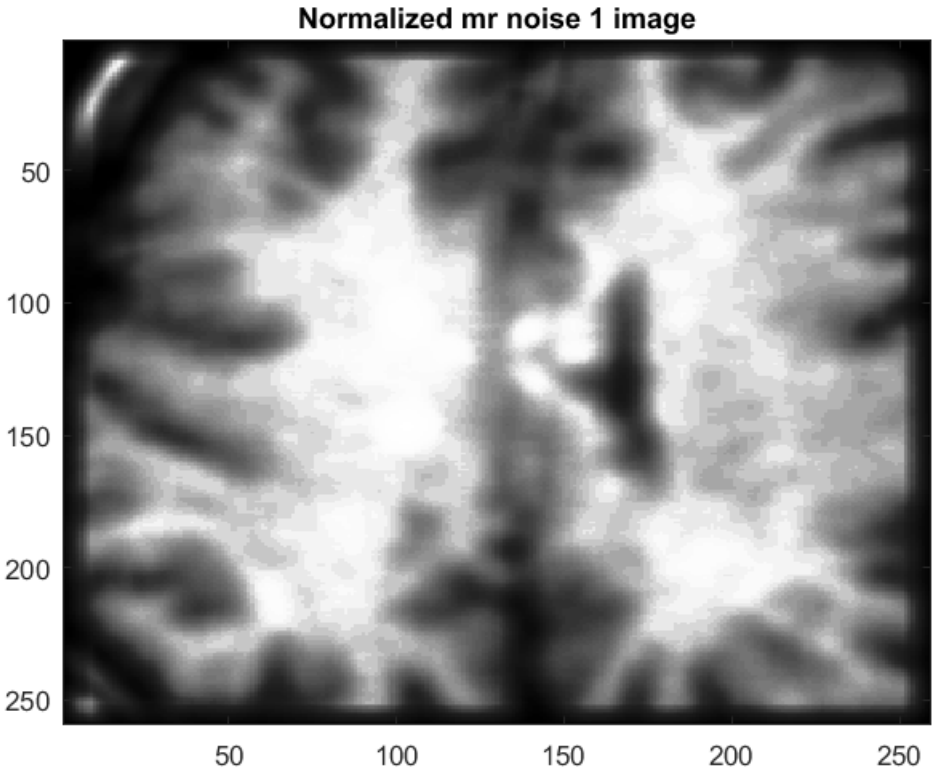

mr
50
100
150
200
250
50
100
150
200
250

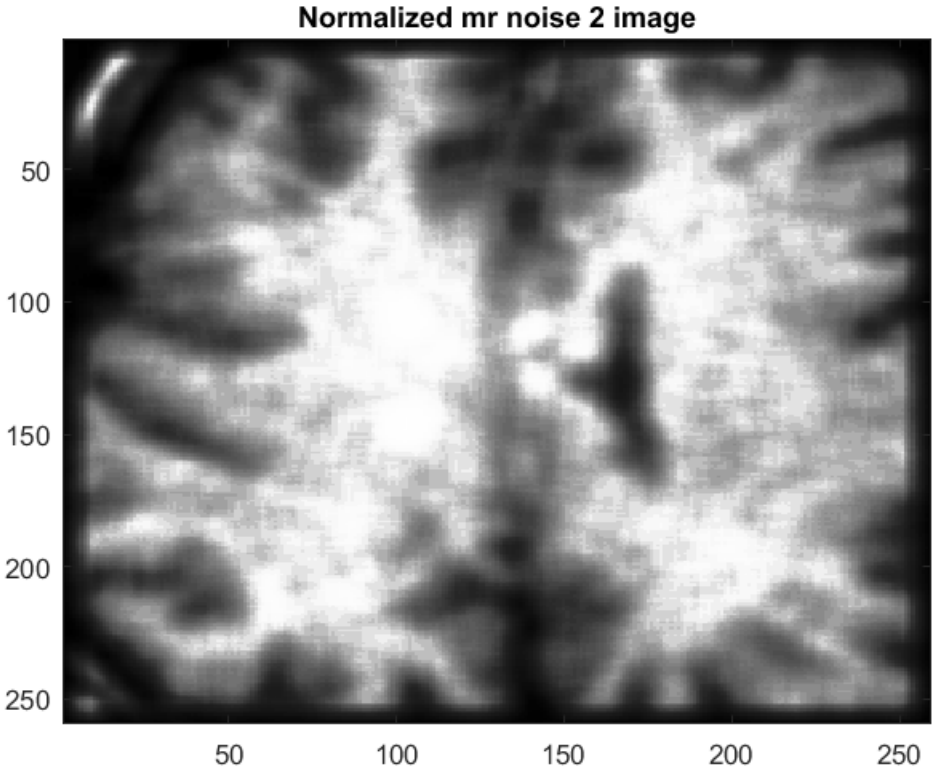

mr noise 1
50
100
150
200
250
50
100
150
200
250

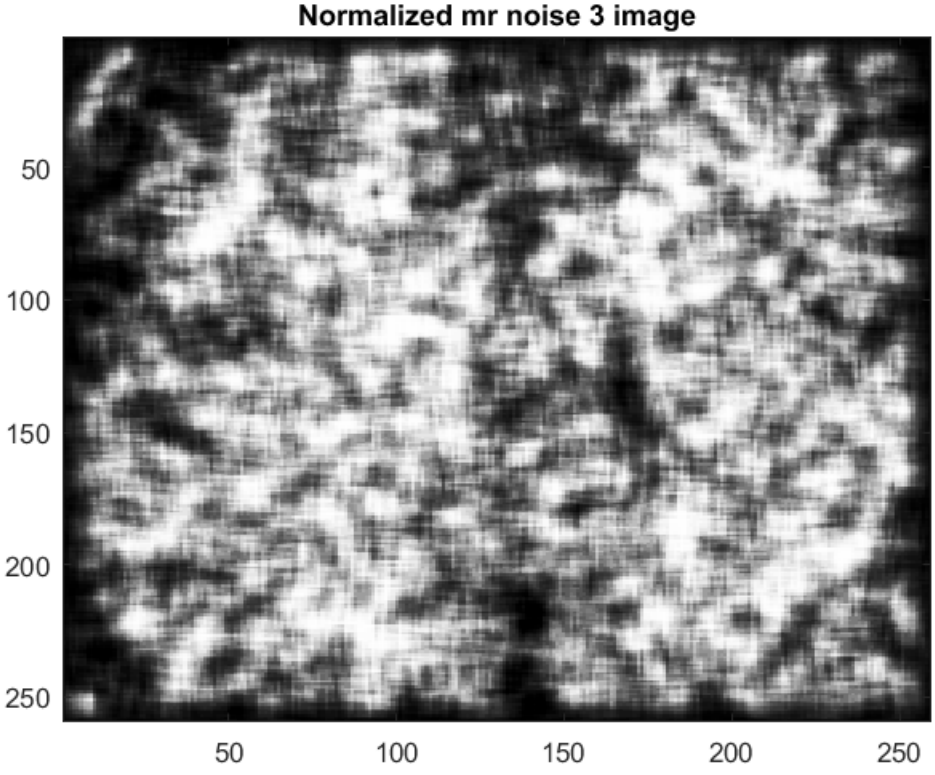

mr noise 2
50
100
150
200
250
50
100
150
200
250

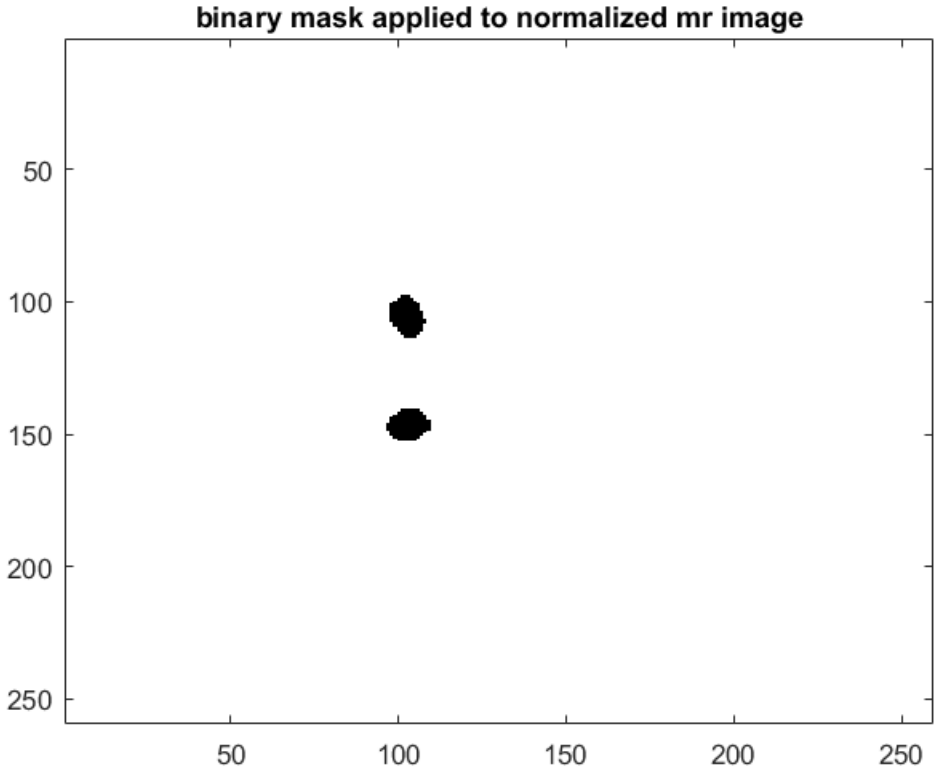

mr noise 3
50
100
150
200
250
50
100
150
200
250

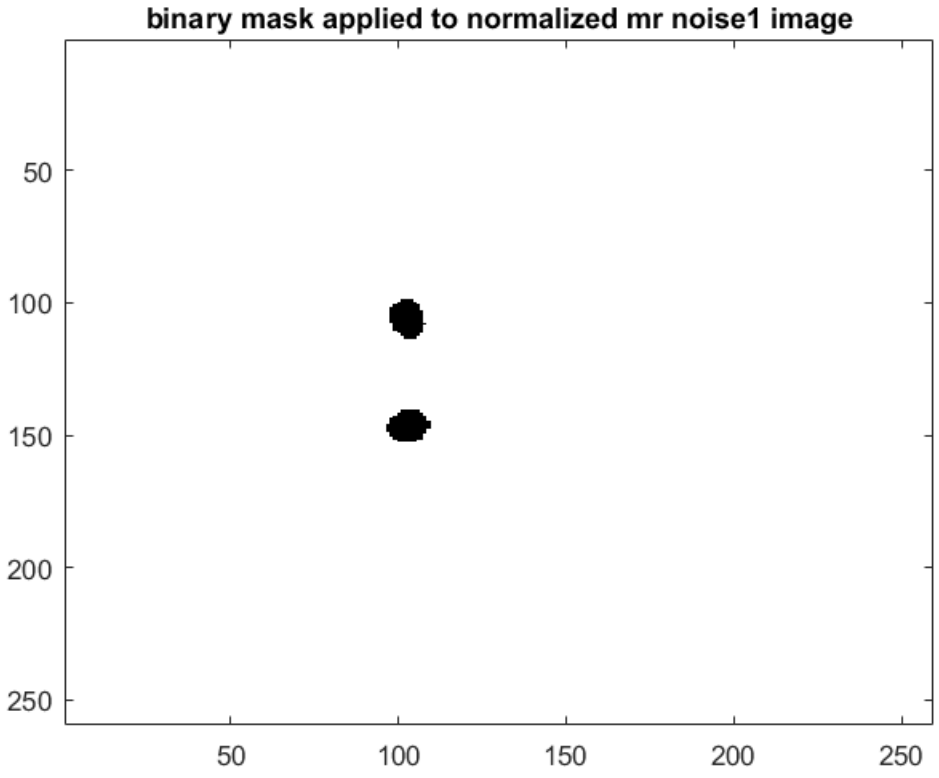

Normalized MR image
50
100
150
200
250
50
100
150
200
250

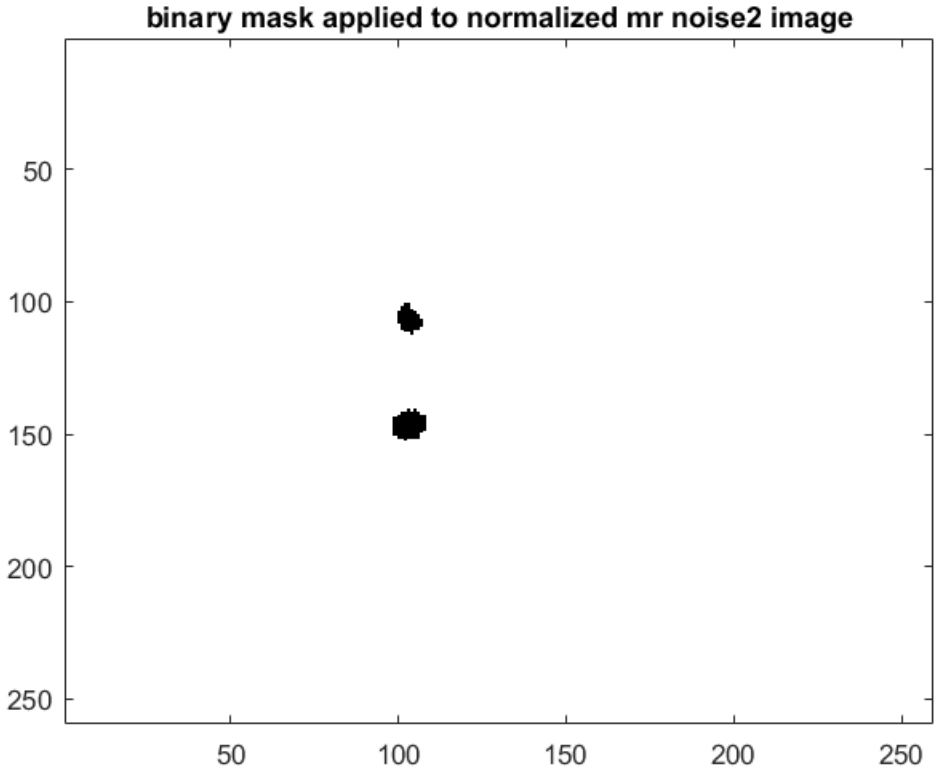

Normalized mr noise 1 image
50
100
150
200
250
50
100
150
200
250

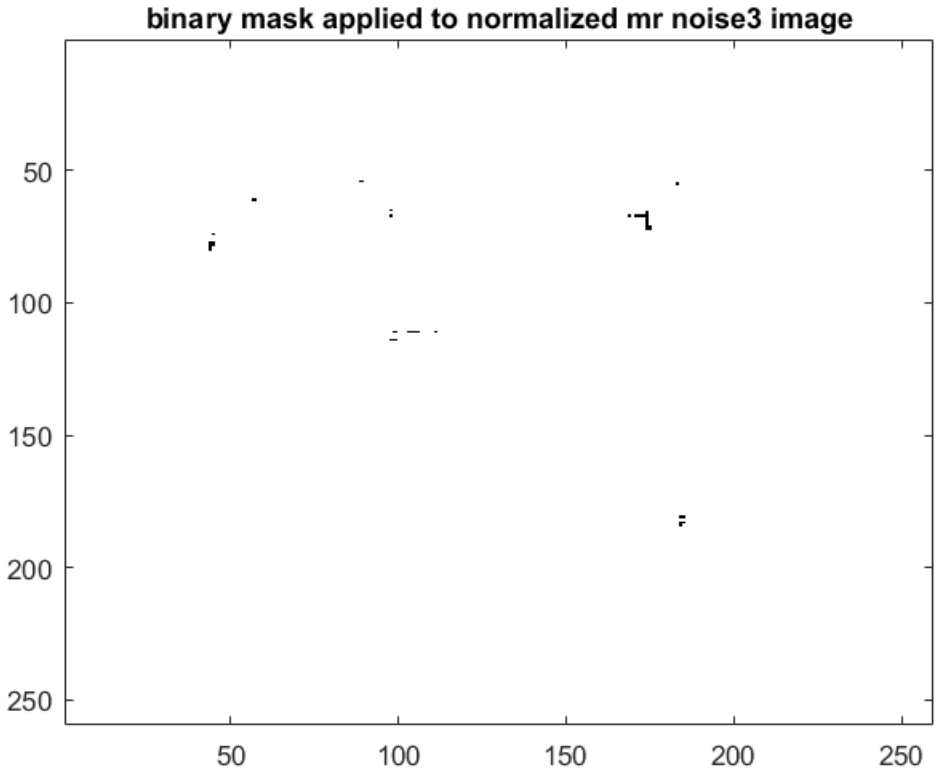

Normalized mr noise 2 image
50
100
150
200
250
50
100
150
200
250

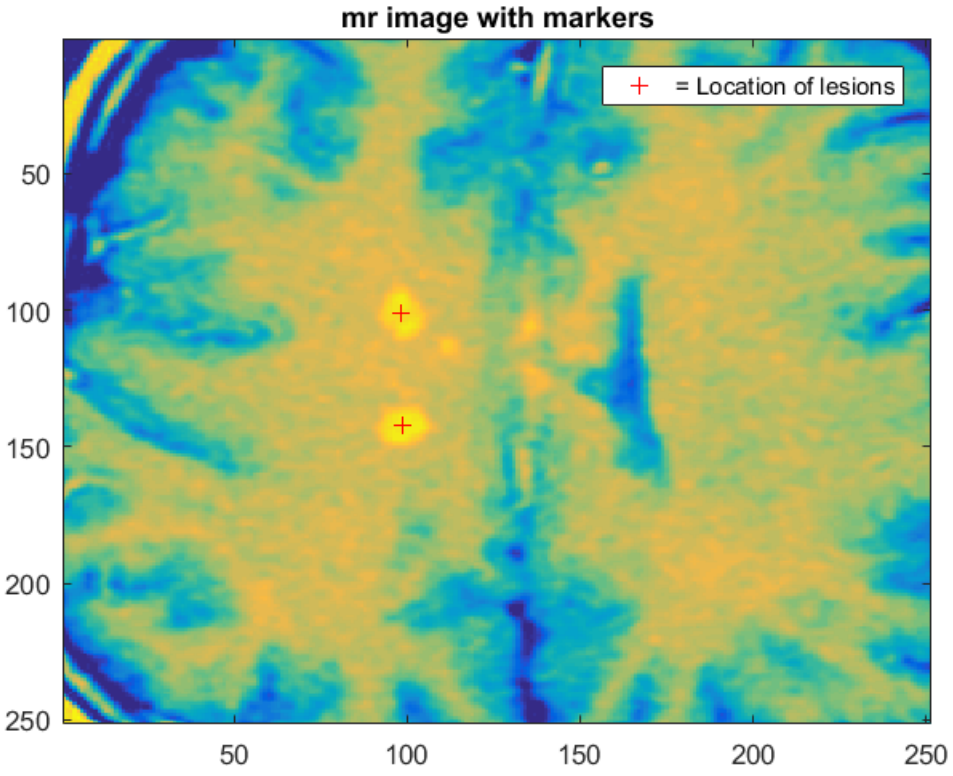

Normalized mr noise 3 image
50
100
150
200
250
50
100
150
200
250

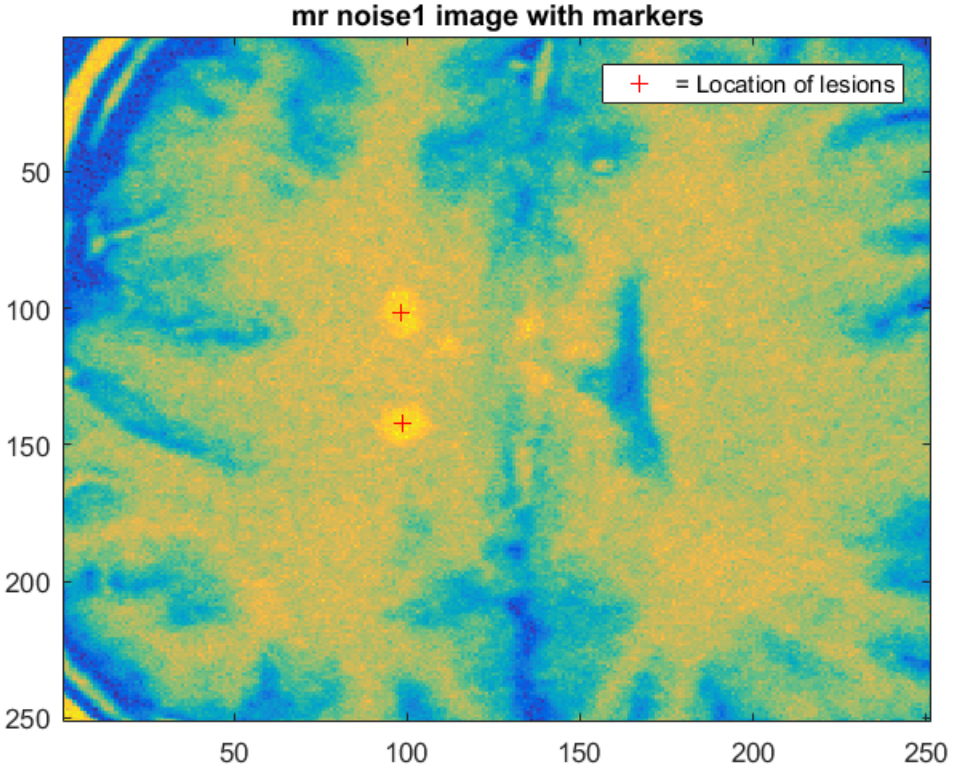

binary mask applied to normalized mr image
50
100
150
200
250
50
100
150
200
250

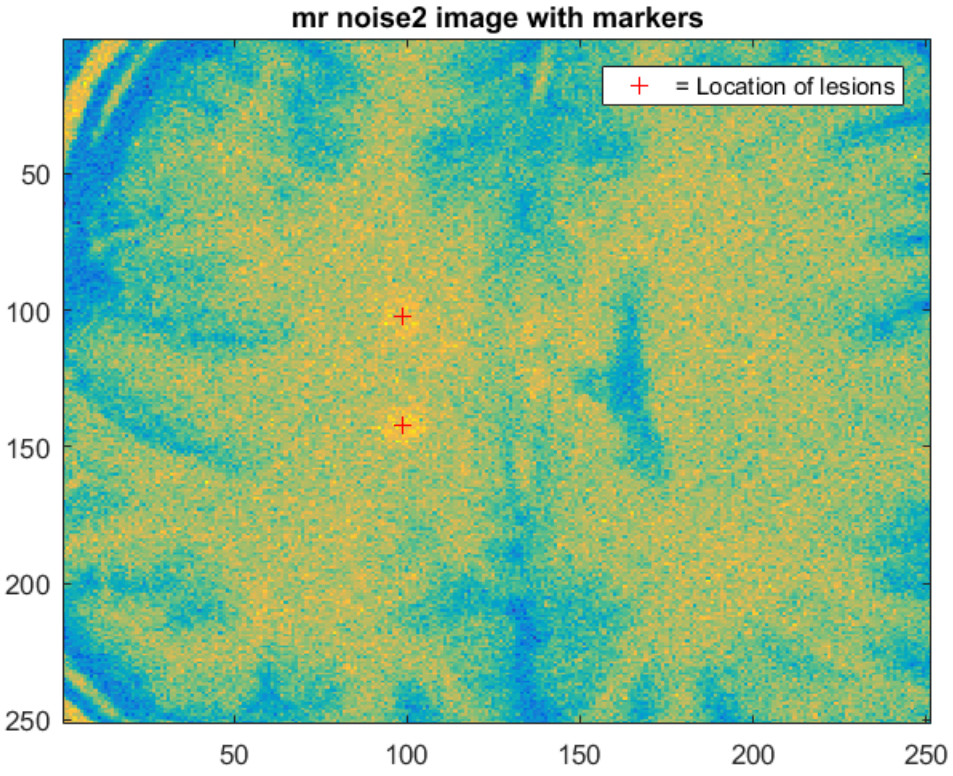

binary mask applied to normalized mr noise1 image
50
100
150
200
250
50
100
150
200
250

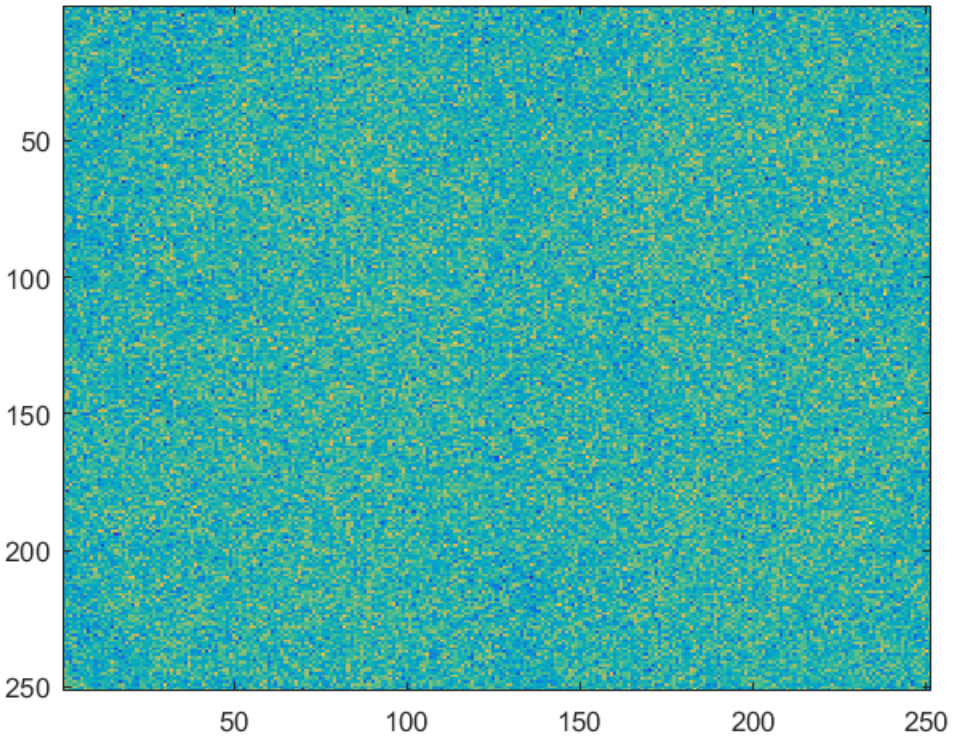

binary mask applied to normalized mr noise2 image
50
100
150
200
250
50
100
150
200
250


binary mask applied to normalized mr noise3 image
50
100
150
200
250
50
100
150
200
250


mr image with markers
= Location of lesions
50
100
150
200
250
50
100
150
200
250


mr noise1 image with markers
= Location of lesions
50
100
150
200
250
50
100
150
200
250


mr noise2 image with markers
= Location of lesions
50
100
150
200
250
50
100
150
200
250


50
100
150
200
250
50
100
150
200
250

Published with MATLAB® R2016b