Project 1

Matched Filters: Applications of Cross-Correlation

Prepared for:

ECGR 4124-001

Prepared by:

Jonathon Nguyen

Date submitted:

11/19/2021

MATLAB code for Task 1:

Task 1 Part 1

```
clc
close ALL
clearvars

% Task 1
freq = (pi/15);
phi = 0;

% x[n] = cos(pi/15 * n + phi), phi = 0
n = 1:200;
x_n = cos(freq*n + phi);

% The random noise.
noisevals = 30*(rand(1, length(x_n)) - 0.5);

%----------Matched Filters--------%
% H[n], K = 10
k_10 = 0:9;
h10_n = cos(freq*k_10);

% H[n], K = 20
k_20 = 0:19;
h20_n = cos(freq*k_20);

% H[n], K = 40
k_40 = 0:39;
h40_n = cos(freq*k_40);

% H[n], K = 80
k_80 = 0:79;
h80_n = cos(freq*k_80);
%----------Matched Filters--------%

% Create a noisy signal from the random values.
x_n_noisy = x_n(n) + noisevals(n);

% Time reverse h[n]
h10_n = fliplr(h10_n);
h20_n = fliplr(h20_n);
h40_n = fliplr(h40_n);
h80_n = fliplr(h80_n);

% Compute the cross correlation
h10x_n_conv = conv(x_n_noisy, h10_n);
h20x_n_conv = conv(x_n_noisy, h20_n);
h40x_n_conv = conv(x_n_noisy, h40_n);
h80x_n_conv = conv(x_n_noisy, h80_n);

% plot the cross correlation
figure(1)
subplot(4, 1, 1), plot(h10x_n_conv)
title("x[n] * h_{10}[-n]")
subplot(4, 1, 2), plot(h20x_n_conv)
title("x[n] * h_{20}[-n]")
subplot(4, 1, 3), plot(h40x_n_conv)
title("x[n] * h_{40}[-n]")
subplot(4, 1, 4), plot(h80x_n_conv)
title("x[n] * h_{80}[-n]")
sgtitle('Cross-Correlation Signals')
```

Task one allows for the detection of the phase and period of a sinusoid inside a noisy signal. A sinusoidal sequence was created from

$$x[n] = \cos\frac{\pi}{15}n + \Phi$$

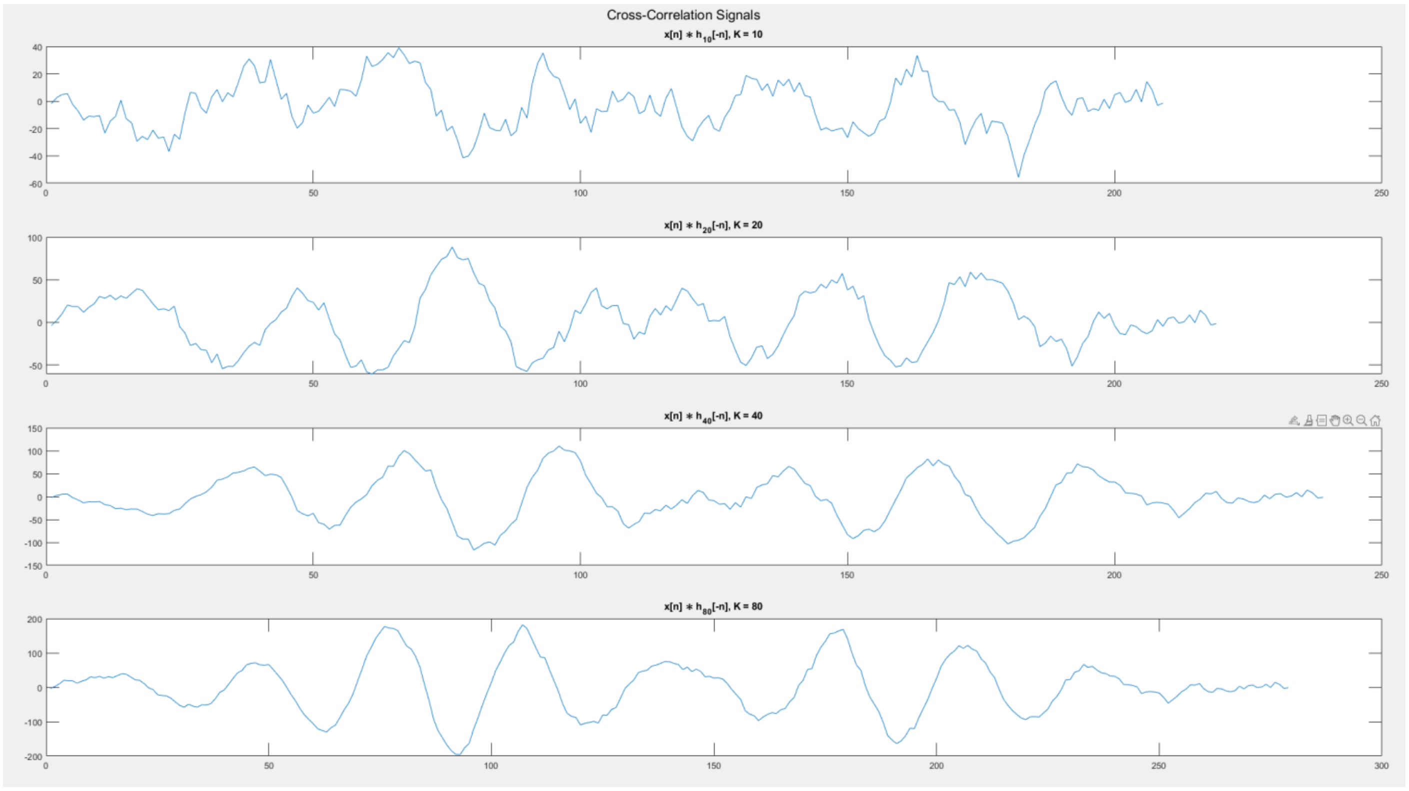
Then the noise was created by using the random number generator in MATLAB. The noise had a uniform distributed amplitude of +15 and -15. The noise was added to the x[n].

Four matched filter with different lengths was created from $h[n] = \cos\frac{\pi}{15}n$ and was cross correlated with the x[n].

The length of the filter was 10, 20, 40, and 80.

Task 1 Part 2:

Here are the subplots for the cross-correlation signals for each of the match filters with varying length.

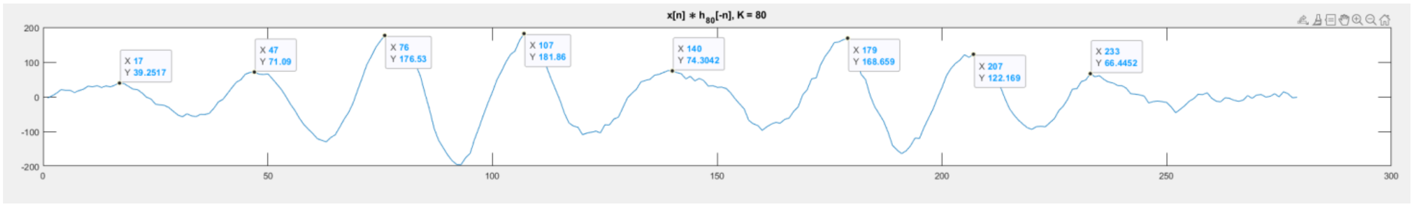


Task 1 Part 3:

The number of samples between each peak is about 30 samples. The matched filter signal is sampled at 1 second/sample.

$$30\ seconds = 30\ samples\,\frac{1\ second}{1\ sample}$$

The estimated period of the noisy sinusoidal input is **30 seconds**.



Task 1 Part 4:

The variable K impacts the estimation of the period by adjusting the length of the filtering signal. When the variable K is small, the matched filter is short. The cross-correlation output will have more of the noise from the input function, making it harder to detect and determine the period of the signal. When the variable K is large, the matched filter is longer and can filter the sinusoid signal much better, and calculating the period becomes easier.

Task 1 Part 5:

If the noise amplitude were increased, the length of the h[n] would need to be increased to get a better estimation of the sinusoid signal. To increase the length, K would need to be increase as well to increase the length of h[n].

Task 1 Part 6:

It is possible to estimate the phase of the function. The index where the peaks are will be shifted.

A method to estimate the phase of a function is to see how many indices the cross-correlated signal has shifted compared to the match filter signal.

MATLAB code for Task 2:

```
clc
close ALL
clearvars

red_pixels_file = load('red_pixel_values.mat');
red_pixels = red_pixels_file.red_pixel_values;

% Frequency ranging from 1hz to 2hz
freq = (1:0.2:2)*2*pi;
discrete_freq = freq*(1/30);

% The matched filters
n = -50:50;
h_0 = sin(discrete_freq(1)*n);
h_1 = sin(discrete_freq(2)*n);
h_2 = sin(discrete_freq(3)*n);
h_3 = sin(discrete_freq(4)*n);
h_4 = sin(discrete_freq(5)*n);
h_5 = sin(discrete_freq(6)*n);

% Plotting the matched filters
figure(1)
hold on
plot(h_0);
plot(h_1);
plot(h_2);
plot(h_3);
plot(h_4);
plot(h_5);
title("Matched Filters")
xlabel("Input: n")
ylabel("Output Values")
legend("h_0[n] 1 Hz","h_1[n] 1.2 Hz","h_2[n] 1.4 Hz" ...
,"h_3[n] 1.6 Hz","h_4[n] 1.8 Hz","h_5[n] 2 Hz")
hold off

% Compute the cross correlation
h0red_conv = conv(red_pixels, fliplr(h_0), 'valid');
h1red_conv = conv(red_pixels, fliplr(h_1), 'valid');
h2red_conv = conv(red_pixels, fliplr(h_2), 'valid');
h3red_conv = conv(red_pixels, fliplr(h_3), 'valid');
h4red_conv = conv(red_pixels, fliplr(h_4), 'valid');
h5red_conv = conv(red_pixels, fliplr(h_5), 'valid');


% Plot the cross correlation
figure(2)
hold on
plot(h0red_conv), plot(h1red_conv), plot(h2red_conv), plot(h3red_conv)
plot(h4red_conv)
plot(h5red_conv)
title("Cross Correleation of the red pixel signal and H_i[n]")
legend("Pixel * h_0[-n]", "Pixel * h_1[-n]", "Pixel * h_2[-n]"...
,"Pixel * h_3[-n]", "Pixel* h_4[-n]", "Pixel * h_5[-n]");
hold off
```

```
% Finding the best guess.
for freq = 0:0.001:2

    discrete_freq = freq*(1/30)*2*pi;
    h = sin(discrete_freq*n);
    red_conv = conv(red_pixels, fliplr(h), 'valid');

    figure(3)
    plot(red_conv)
    ylim([-80 80])
    title("Frequency: " + string(freq) + " Hz")


    pause(.01)

end
```

Task two allows for the detection of the frequency of a sinusoid inside a noisy signal.

There will be 6 filter signals with frequencies ranging from 1 Hz to 2 Hz in increments of 0.2 Hz
The input signal is sampled at a rate of 30 samples per second. That means that the filtered signal will be sampled at the same rate.

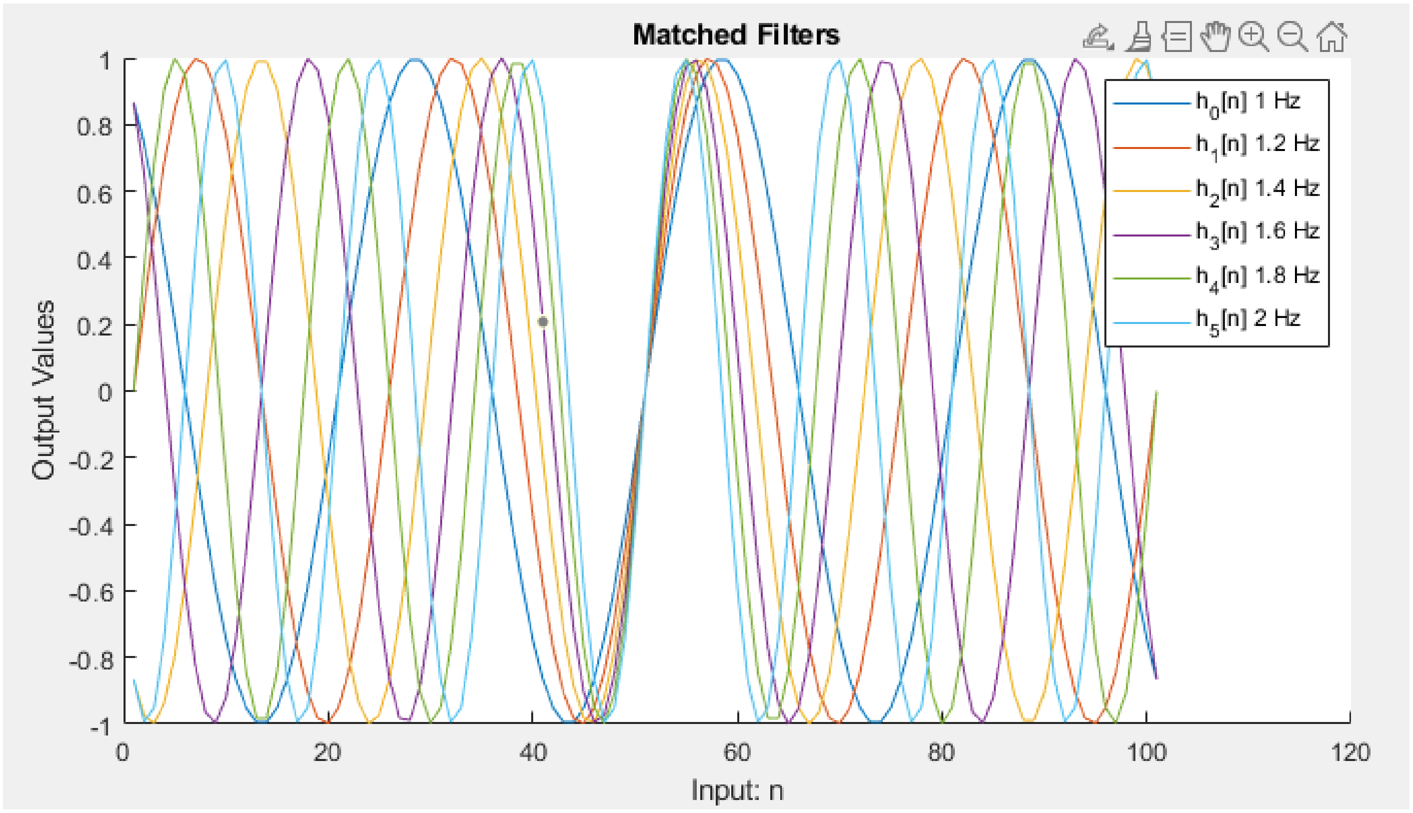
These are the equation for the six filter signals with frequencies ranging from 1 Hz to 2 Hz.

- $h_0[n] = (1 * (1/30) * 2 * pi * n)$
- $h_1[n] = (1.2 * (1/30) * 2 * pi * n)$
- $h_2[n] = (1.4 * (1/30) * 2 * pi * n)$
- $h_3[n] = (1.6 * (1/30) * 2 * pi * n)$
- $h_4[n] = (1.8 * (1/30) * 2 * pi * n)$
- $h_5[n] = (2 * (1/30) * 2 * pi * n)$

The filter was sampled from n = -50 to 50.

Task 2 Part 1

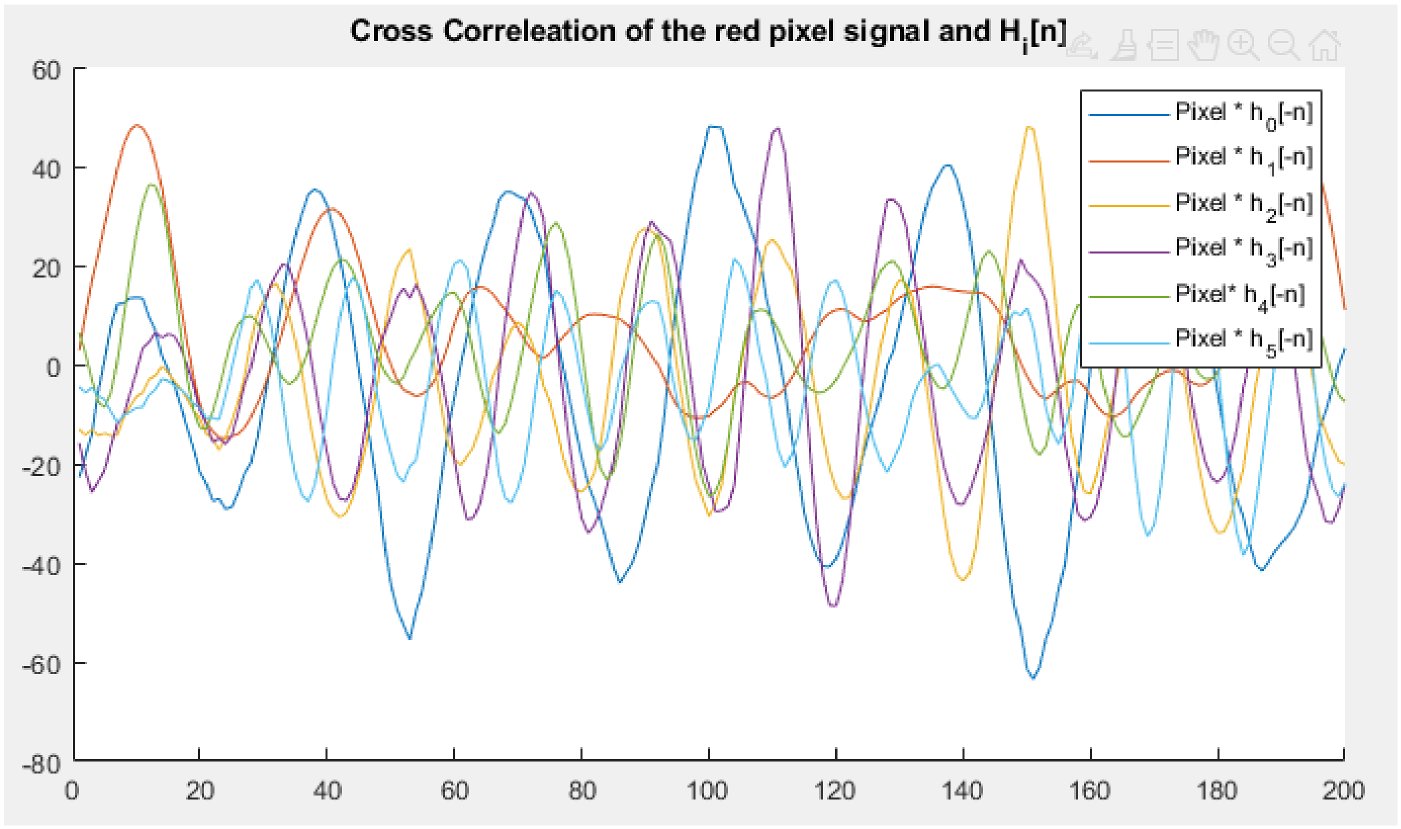
This plot shows all 6 of the filters representing the hypothesized frequencies.



Task 2 Part 2

Each filter was time reversed and cross-correlated with the red pixel signal.

This plot shows the valid portion of the cross-correlation for each filter and the red pixel signal.



Task 2 Part 3

The best filter was found by looking at Task 2 part 2's graph and finding the largest peak response.

Filter peak responses:

- $h_0[n]$: 49.856
- $h_1[n]$: 48.4132
- $h_2[n]$: 47.9501
- $h_3[n]$: 47.8345
- $h_4[n]$: 36.3109
- $h_5[n]$: 34.0398

The filter that has the frequency closest to the unknown frequency is first filter, **h_0[n]** and it has a frequency of **1 Hz**.

$h_0[n] = (1 * (1/30) * 2 * pi * n)$

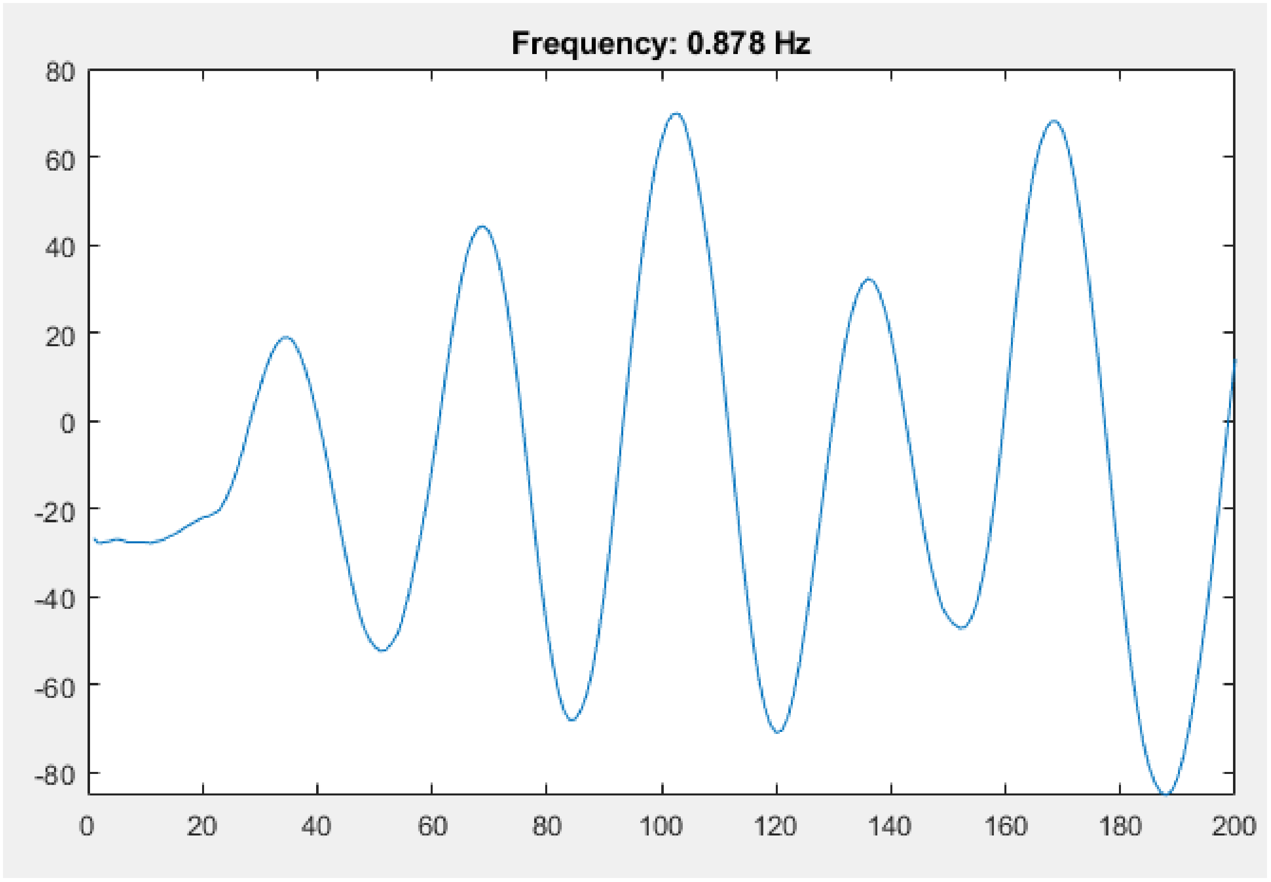
Task 2 Part 4

The guess was refined by looping through frequencies from 0 to 2 in increments of 0.001 and plotting the cross-correlation of the red pixel signal and the new filter on an updating figure. This allowed the waveform to be viewed in real-time to find the best peak response.

The new best guess for the frequency is **0.878 Hz**, and the new best guess filter equation is

$$\boldsymbol{h[n] = \sin\left(0.878 * 2\pi * \frac{1}{30} * n\right)}$$

This plot is the cross-correlation of the new filter equation and the red pixel signal.

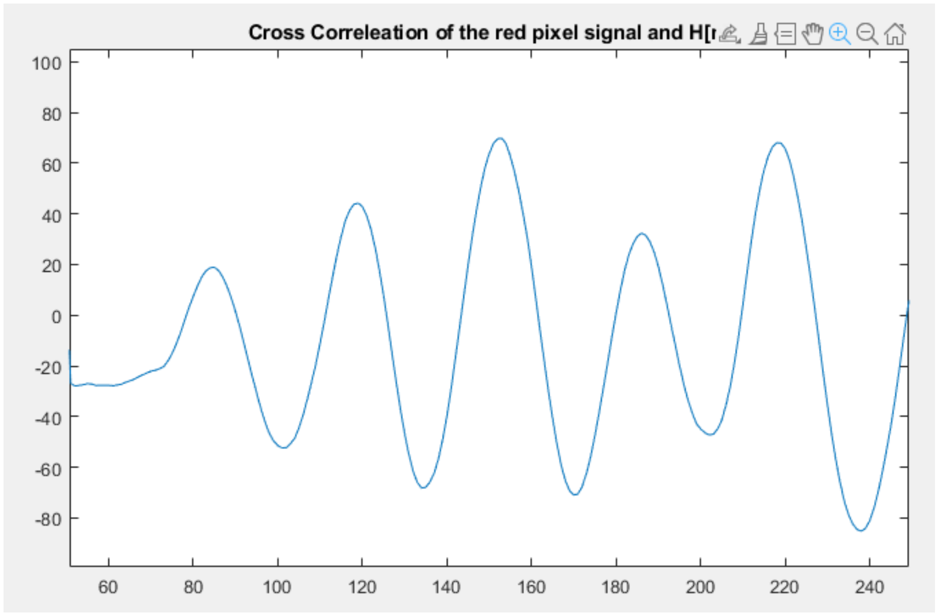


MATLAB code for Task 3:

```
clc
close ALL
clearvars

video = load('rgb_video_values.mat');
red_pixels_file = load('red_pixel_values.mat');
x_n = video.imageStack_uint8;

% The matched filter
n = -50:50;
discrete_freq = 0.878*(1/30)*2*pi;
h_n = sin(discrete_freq*n);

% Time Reverse h[n]
h_n = fliplr(h_n);

% Plot the cross-correlation
figure(1)
plot(conv(red_pixels_file.red_pixel_values, h_n, "same"))
title("Cross Correleation of the red pixel signal and H[n]")

% Compute the cross correlation
y_n = conv2(x_n, h_n, 'same');

alpha = 1.8;

% Code provided to reshape the matrix into a image.
rgb_images = reshape(x_n, [video.rows, video.cols, 3, video.num_frames]);
rgb_processed = reshape(y_n, [video.rows, video.cols, 3, video.num_frames]);
rgb_processed = rgb_images + uint8(alpha * rgb_processed);

% Code provided to see the output video.
for frame = 1:video.num_frames
    figure(2)
    subplot(1,2,1), imshow(rgb_images(:,:, :,frame),'InitialMagnification', 'fit')
    xlabel('original image');
    framestr = sprintf('processed image %d',frame);
    subplot(1,2,2), imshow(rgb_processed(:,:,:,frame),'InitialMagnification', 'fit')
    xlabel(framestr);
    pause(1/30);
end

minframe = 238;
maxframe = 153;
figure(3), subplot(2,2,1), imshow(rgb_images(:,:,:,minframe));
xlabel('Original min image');
figure(3), subplot(2,2,2), imshow(rgb_processed(:,:,:,minframe))
xlabel('Magnified min image');
figure(3), subplot(2,2,3), imshow(rgb_images(:,:,:,maxframe))
xlabel('Original max image');
figure(3), subplot(2,2,4), imshow(rgb_processed(:,:,:,maxframe))
xlabel('Magnified max image');
```

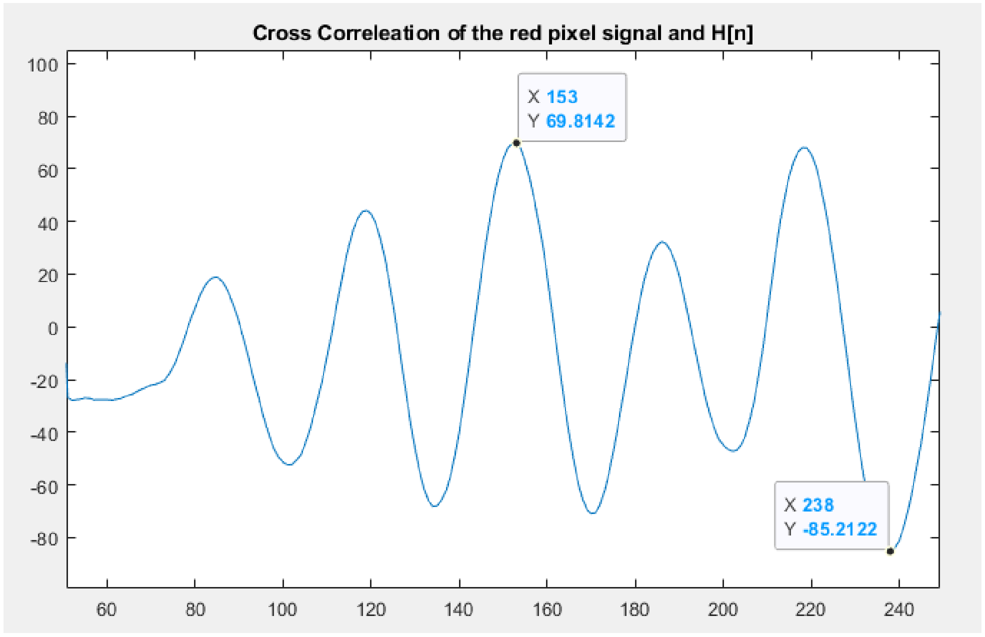
The best guess of the filter signal is used to process the video. This will magnify the normally undetectable signal.

Task 3 Part 1
This is the plot of the cross-correlation between the red pixel signal and the best guess of the filter signal.



Task 3 Part 2
The max value of the plot is 69.8142 at index 153. The minimum value of the plot is -85.2122 at index 238.

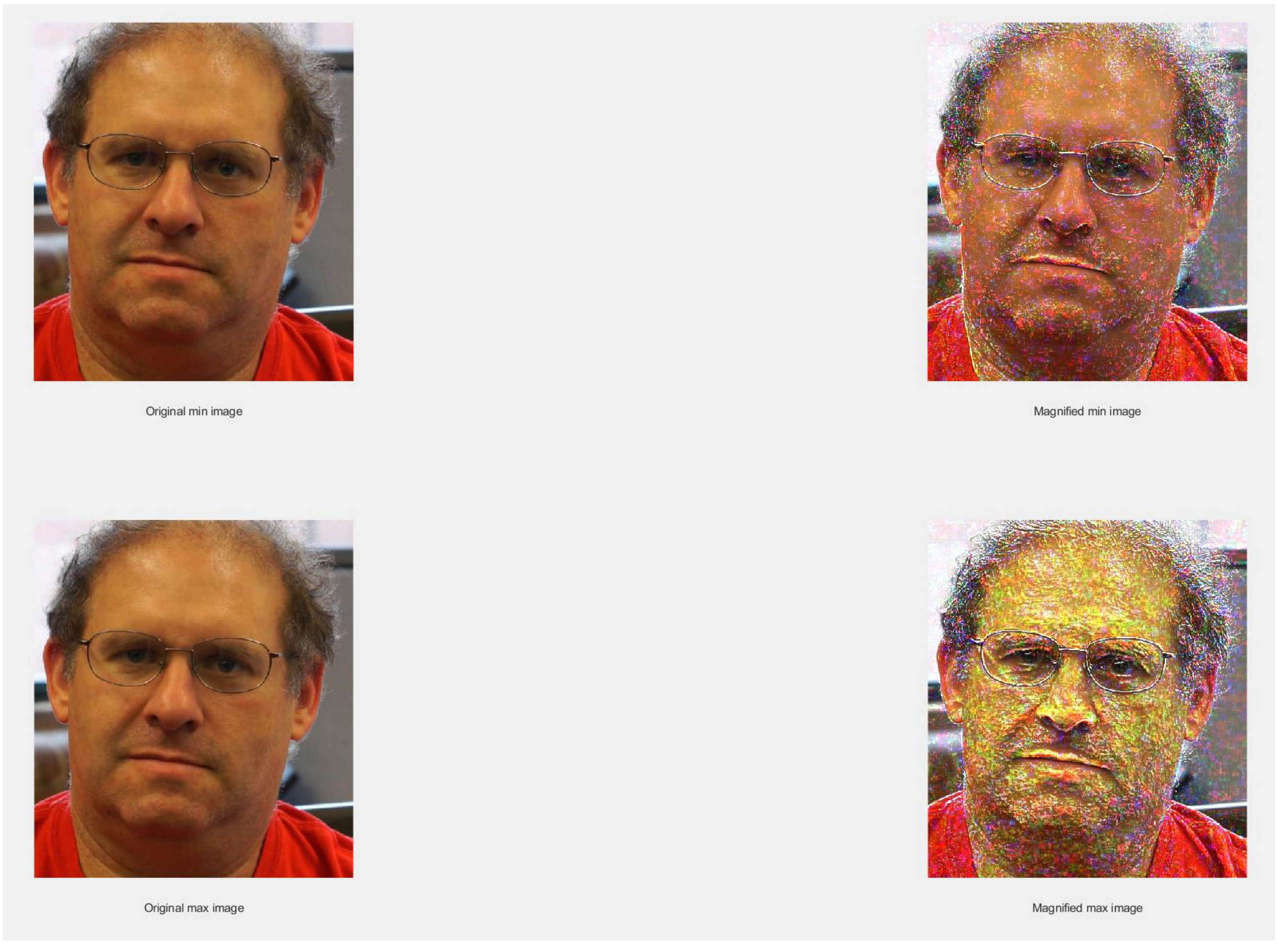
The max index is: **153**
The minimum index is: **238**



Task 3 Part 3

Using the indices selected in task 3, part 2, a two-by-two matrix of image plots is created. The index found selects the frames where the cross-correlation values are at their maximum and minimum.

This is the 2x2 matrix of image plots showing the original image on the left and the magnified effect image on the right.



Task 3 Part 4

The physical effect that is being magnified by the filter in the image is the heart pumping blood into the face. This effect is not observable in the original image because the physical effect of the blood entering the face and changing the observed color is very tiny and not visible to the naked eye. The matched filter can filter the changing color on the face, and from there, the effect is able to be magnified.

The reason that magnification is specific to only the heart pumping is because the frequency of the filter only matches or correlates to the frequency it is filtering out. The only thing that is at the filter frequency is the heart pumping or the heart rate.

Task 3 Part 5

The heart rate of the subject in the video is **53 BPM**.

The heart rate can be computed from the magnified video images by either watching the magnified video at the correct playback rate and then time the difference between when the face is red or use the peaks of the values from the magnified video or the cross-correlated signal to analytically solve for the BPM.

The sample difference between each of the peak of the cross-correlation signal of the red pixel is about 34 samples. The video was sampled at 30 sample/second. The 34 samples are divided by 30 sample/second is equal 1.11333 seconds.

$$1.133333\ seconds = 34\ sample \frac{1\ second}{30\ sample}$$

The 1.11333 seconds is converted to minutes.

$$0.01888888\ minutes = 1.133333\ \frac{1\ minute}{60\ second}$$

Then the 0.01888888 minutes is inverted to get the beats per minute of the heart.

$$53 \frac{beats}{minutes} = (0.01888888\ minutes)^{-1}$$