Universidade de Brasília

Departamento de Engenharia Elétrica

Tópicos em Engenharia - Processamento de Sinais Biomédicos

Lista de Exercícios 3

Autores:

Caio Luiz Candeias Flôres 190134283

Felipe Carneiro da Motta 200017616

João Pedro Daher Aranha 190109742

Brasília

01 de Julho de 2022

Contents

1 Exercícios

1.1 Exercício 2.31:

The file **bandwidths.mat** contains two signals having different bandwidths. The signal in x is narrowband while the signal in y is broadband. Plot and compare the autocorrelation functions of the two signals. Plot only lags between ±30 to show the initial decrease in correlation. Note the inverse relationship between bandwidth and width of the autocorrelation function.

The MATLAB's code:

```
clc; clear all; close all;

load bandwidths.mat; % loading the file
% there is two variable: signals x and y
% x: narrowband signal
% y: broadband signal

[rxx, lags_x] = xcorr(x, x, 'coeff', 30); % autocorrelation x
[ryy, lags_y] = xcorr(y, y, 'coeff', 30); % autocorrelation y

% plot
subplot(2, 1, 1);
plot(lags_x, rxx, 'k', 'DisplayName', 'Narrowband Signal');
ylabel('Autocorrelation value'); % x-axis name
xlabel('Lags'); hold on; % y-axis name; joint plot
axis([-30 30 -0.5 1.2]); % limits of the axis plot
legend('Orientation','horizontal','Box','on','Location','southoutside');

subplot(2, 1, 2);
plot(lags_y, ryy, 'b', 'DisplayName', 'Broadband Signal');
ylabel('Autocorrelation value');
xlabel('Lags');
axis([-30 30 -0.5 1.2]);
```

```
legend('Orientation','horizontal','Box','on','Location'
    ,'southoutside');
sgtitle('Autocorrelation of signals X and Y');
grid(); % insert lines
saveas(gcf, 'semmlow_2_31.png');
```

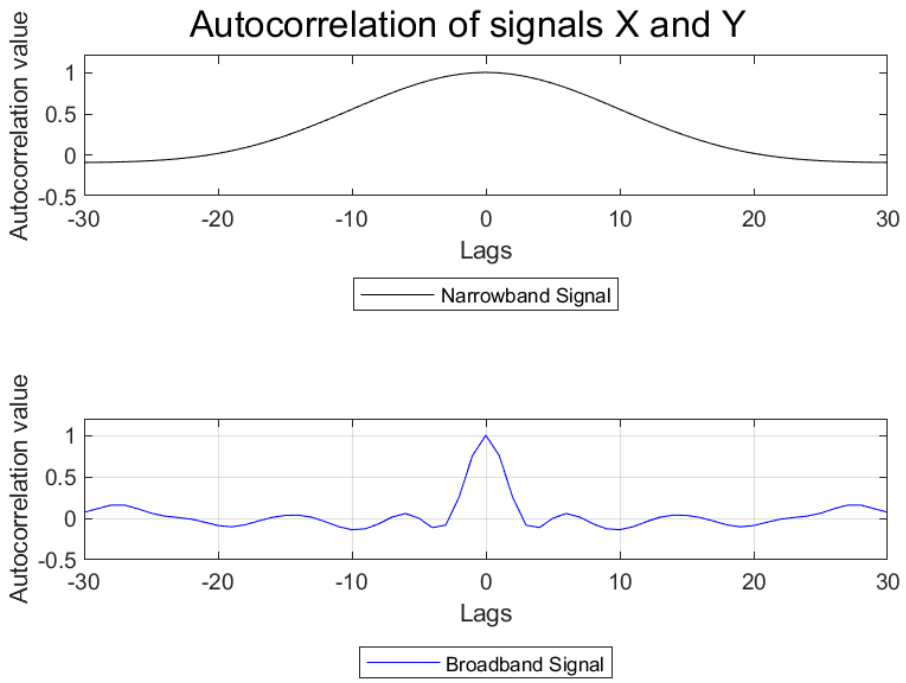

Figure 1: Exercise 2.31 - Autocorrelation plots of signals X and Y

Notably, from the Figure 1, the bandwidth and the width of the autocorrelation function are inversely proportional. The narrowband signal has a wide bandwidth between the lag in the ±30 range, and the broadband has an narrow bandwidht.

1.2 Exercício 2.33:

The file **prob2-33-data.mat** contains a signal x (fs = 500 Hz). Determine if this signal contains a 50-Hz sine wave and if so over what time periods.

```
clc; clear all; close all;

load prob2_33_data.mat % loading the signal the file
% there is one variable:
% x -¿ fs = 500 Hz

% signal x
N = 1580                                % number of points
fs = 500;                               % plot frequency
t = (0:N-1)/fs;                         % time vector, same
    length as signal x

% sine wave of 50 Hz
f = 50;                                 % sine wave frequency
y = sin(2*pi*f*t);                      % the sine wave
[rxy, lags] = axcor(x, y);              % cross correlation x
    and sin_wave


%ploting the sinusoidal wave
subplot(3,1,1);
plot(t, y, 'b');
xlabel('Time (s)');
ylabel('$y(t)$','Interpreter', 'latex', 'Fontsize', 10)
    ;
title('\textbf{$sin(2\pi 50t)$}','Interpreter', 'latex'
    , 'Fontsize', 10');
axis([0 3.25 -1 1]);
grid();

%ploting the signal x
subplot(3,1,2);
plot(t, x, 'k');
xlabel('Time (s)');
ylabel('$x(t)$','Interpreter', 'latex', 'Fontsize', 10)
    ;
title('Signal x');
axis([0 3.25 -10 10]);
grid();

```

```
%ploting the cross correlation
subplot(3,1,3);
plot(lags/fs, rxy, 'r'); % plot cross correlation
    function
hold on;
[rxy_max, max_lags] = max(rxy); % find max correlation
max_lags = lags(max_lags)/fs; % convert max shift to
    sec
plot(max_lags, rxy_max, '*k'); % plot max correlation
xlabel('Lags (s)');
ylabel('$r_{xy}(t)$','Interpreter', 'latex', 'Fontsize'
    , 10);
title(sprintf('Max corr: %s | Delay: %s', string(
    rxy_max), string(max_lags)));
axis([-3.25 3.25 -0.1 0.1]);
grid();
saveas(gcf,sprintf('%s.png',mfilename));
```

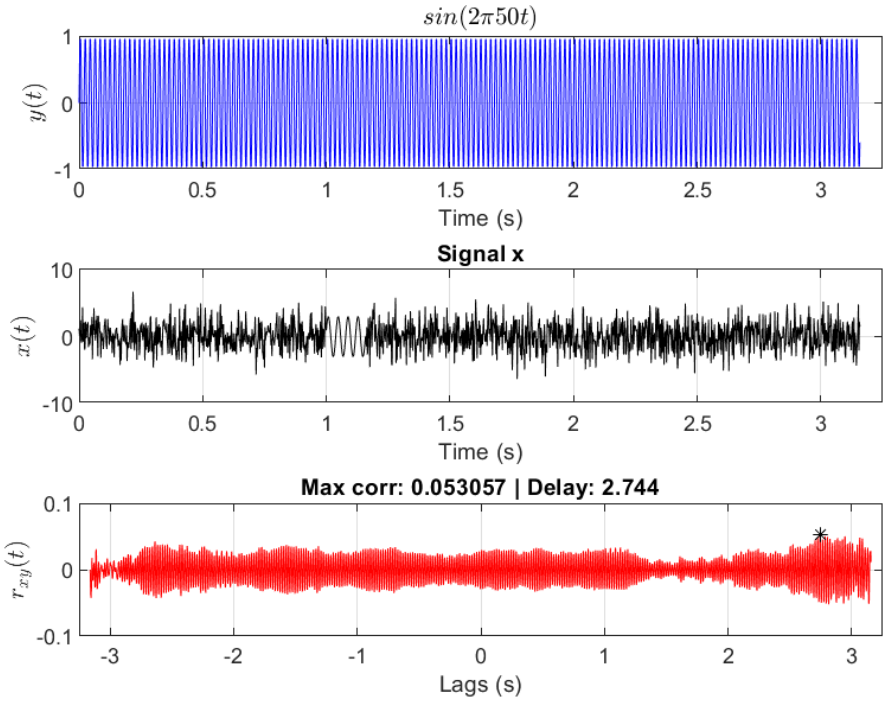

Figure 2: Exercise 2.33 - Cross correlation plots of signal X and the sine wave of 50 Hz

According to the Figure 2, more specifically in the black asterisk, the signal x doesn't have a frequency of 50 Hz. The Pearson Cross Correlation is a good statistic tool to verify if there is some specific frequency in a comparative signal from a base signal (in this case $\sin(2\pi 50t)$. The maximum Pearson Cross Correlation value is ≈ 0.053057, which is evidently a very low cross correlation value and quite close to zero.

1.3 Exercício 2.35:

Develop a program along the lines of Example 2.11 to determine the correlation in heart rate variability during meditation. Load file **Hr-med.mat** which contains the heart rate in variable **hr-med** and the time vector in variable **t-med**. Calculate and plot the autocovariance. The result will show that the heart rate under meditative conditions contains some periodic elements. Can you determine the frequency of these periodic elements? A more definitive approach is presented in the next chapter which shows how to identify the frequency characteristics of any waveform.

The MATLAB's code:

```
clc; clear all; close all;
% attention: insert the file 'axcor'

load Hr_med.mat % loading the file
% there is two variables:
% hr_med: heart rate
% t_med: time vector

[cov_hrm, lags_hrm] = axcor(hr_med - mean(hr_med)); % normal autocovariance
plot(lags_hrm, cov_hrm, 'k'); hold on;
plot([lags_hrm(1) lags_hrm(end)], [0 0], 'r--');
xlabel('Lags');
ylabel('$c_{xx}$', 'Interpreter', 'latex', 'Fontsize', 20); % latex symbol
title('Heart Rate Autocovariance');
axis([-30 30 -0.2 1.2]); % x-axis [-30, 30] beats
grid();
saveas(gcf,sprintf('%s.png',mfilename));

%% 1) simple form to extract the frequency of periodic elements
% based on 2 samples
f1 = 1/( t_med(2) - t_med(1) );

% based on all the "t_med" samples
for i=1:(length(t_med) - 1)
    f_vec(i) = 1/( t_med(i+1) - t_med(i) );
end
f_med = mean(f_vec);

out = sprintf('2-samples Frequency: %.4f | Average Frequency: %.4f', f1, f_med);
disp(out);

%% 2) using the cross correlation in with f = 0.1 to 1 Hz
```

```
% this range [0.1, 1] is usually the heart rate range

cnt = 1;
for i=(0.1:0.1:2)
    f_axcorr_vec(cnt) = max(axcor(hr_med, cos(2*pi*i*
        t_med)));
    freqs(cnt) = i;
    out = sprintf('Frequency: %.2f Hz --¿ Max cross
        correlation: %.4f', freqs(cnt), f_axcorr_vec(cnt
        ));
    disp(out);
    cnt = cnt+1;
end
```

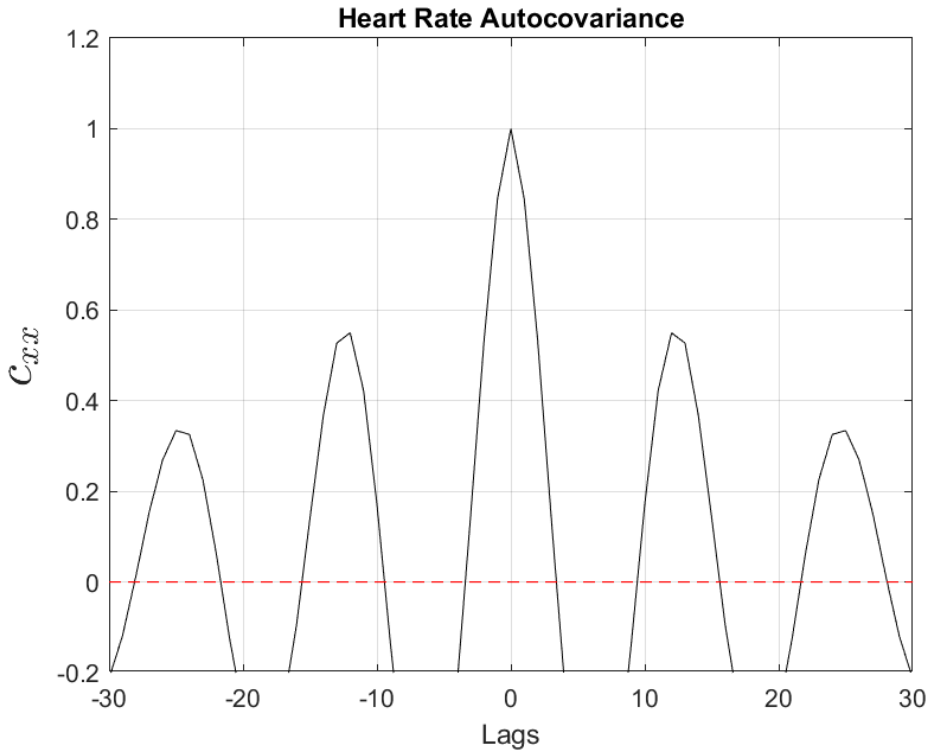

Figure 3: Exercise 2.35 - Heart Rate Autocovariance

According to the figure 3, it's notorious that the autocovariance represents the memory of the system. As the graph shows, there's a pattern in the value of autocovariance. This behavior is essential to the control and maintenance

of the functions that the heart rate represents directly or indirectly. Plenty of parts in the human's body are controlled by signals in many different and specifics frequencies specters which convey a lot of information.

```
2-samples Frequency: 1.5060 | Average Frequency: 1.3554
Frequency: 0.10 Hz --> Max cross correlation: 0.1483
Frequency: 0.20 Hz --> Max cross correlation: 0.0621
Frequency: 0.30 Hz --> Max cross correlation: 0.0390
Frequency: 0.40 Hz --> Max cross correlation: 0.0190
Frequency: 0.50 Hz --> Max cross correlation: 0.0353
Frequency: 0.60 Hz --> Max cross correlation: 0.0070
Frequency: 0.70 Hz --> Max cross correlation: 0.0449
Frequency: 0.80 Hz --> Max cross correlation: 0.0099
Frequency: 0.90 Hz --> Max cross correlation: 0.0320
Frequency: 1.00 Hz --> Max cross correlation: 0.0905
Frequency: 1.10 Hz --> Max cross correlation: 0.5066
Frequency: 1.20 Hz --> Max cross correlation: 0.7015
Frequency: 1.30 Hz --> Max cross correlation: 0.0665
Frequency: 1.40 Hz --> Max cross correlation: 0.2520
Frequency: 1.50 Hz --> Max cross correlation: 0.0802
Frequency: 1.60 Hz --> Max cross correlation: 0.0449
Frequency: 1.70 Hz --> Max cross correlation: 0.0818
Frequency: 1.80 Hz --> Max cross correlation: 0.0218
Frequency: 1.90 Hz --> Max cross correlation: 0.0462
Frequency: 2.00 Hz --> Max cross correlation: 0.0422
```

Figure 4: Exercise 2.35 - Frequency of periodic elements

In the Figure 4, in the first line, a quite simple approach to extract the frequency was done. Instants of time in a row were subtracted $(t2 - t1)$, and the frequency was $fs = \frac{1}{(t2 - t1)}$. In a attempt to make this result better approximate, a for loop was created to do this process iteratively such as, in a generic representation, f $= \frac{1}{(t_{n+1} - t_n)}$. But, in sequence, became obvious that this approach was a bad marker if more than one frequency is in this instant of time, what is likely because the heart rate is a non-linear process.

In a attempt to get a better information about the frequency, a new

approach, now using the cross correlation, was followed. The Pearson Cross Correlation is a good parameter to verify if there is some frequency in a comparative signal from a base function (in this case, the $cos(2\pi t + \theta)$). In a for loop, values in the range $[0.1, 2]$ Hz were tested and, in the frequencies of 1.1 and 1.2 Hz, high values of Pearson Cross Correlation were obtained, ≈ 0.5066 and ≈ 0.7015 respectively. Due these results, it's possible to infer that this file of heart rate has some frequency around this range $[1.2, 1.1]$. More precisely conclusion would be drawn with a direct analysis of the frequency domain with the Fourier Transform or some computational version of this algorithm.

1.4 Exercício 2.36:

Construct a 512-point Gaussian noise array, then filter it by averaging segments of three consecutive samples. In other words, construct a new array in which every point is the average of the preceding three points in the noise array: $y[n] = \frac{x[n]}{3} + \frac{x[n-1]}{3} + \frac{x[n-2]}{3}$. You could write the code to do this, but an easier way is to convolve the original data with a function consisting of three equal coefficients having a value of 1/3, in MATLAB: h(n) = $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$. Construct, plot, and compare the autocorrelation of the original and filtered waveform. Limit the x-axis to ± 20 lags to emphasize the difference. Note that while the original Gaussian data were uncorrelated point to point, the filtering process imposed some correlation on the signal.

```
clc; clear all; close all;
N = 512; % Number of data points
nu_bins = 40;
rng('default'); %defines random number generator to
% Mersenne Twister generator with seed 0
s = rng; %saves to variable s
x= randn(1,N); %creates normal distribution array with
   N points
[ht,xout] = hist(x,nu_bins); % Calculate histogram
ht = ht/max(ht); % Normalize histogram to 1.0
bar(xout, ht,'DisplayName', 'Random array with normal
   distribution'); % Plot as bar graph
ylabel('Frequency in range'); % y-axis name
xlabel('Value'); hold on; % x-axis name; joint plot
```

```
13 legend('Orientation','horizontal','Box','on','Location'
       ,'southoutside');
14 grid(); % insert lines
15
16 hold on %
17 y = -4:0.1:4; %range obtained by test
18 mu = 0; %mean 0
19 sigma = 1; %variance 1
20 f = exp(-(y-mu).^2./(2*sigma^2))./(sigma*sqrt(2*pi)); %
       expression for normal dstribution
21 f_norm=f/max(f); %Normalization by 1
22 plot(y,f_norm,'LineWidth',1.5,'DisplayName', 'Normal
       distribution generated mathematically') %plot of
       normal distribution
23
24 saveas(gcf, 'ex2_36_1.png');
25
26 %Autocorrelation of non-filtered signal
27 [rxx,lags_x]=xcorr(x,'coeff',20);
28
29 %filtered signal
30 h_n = [1/3,1/3,1/3]; %filter that performs moving
       average of every 3 points
31 x_filtered = conv(x,h_n);
32
33 [rxx_filtered,lags_x_filtered]=xcorr(x_filtered,'coeff'
       ,20);
34
35 % plot
36 figure;
37 subplot(2, 1, 1);
38 plot(lags_x ,rxx, 'k', 'DisplayName', 'Non-filtered
       signal');
39 ylabel('Autocorrelation value'); % y-axis name
40 xlabel('Lags'); hold on; % x-axis name; joint plot
41 legend('Orientation','horizontal','Box','on','Location'
       ,'southoutside');
42 grid(); % insert lines
43
44 subplot(2, 1, 2);
45 plot(lags_x_filtered,rxx_filtered, 'b', 'DisplayName',
```

```
    'Filtered signal');
ylabel('Autocorrelation value');
xlabel('Lags');
legend('Orientation','horizontal','Box','on','Location'
    ,'southoutside');
sgtitle('Autocorrelation of signals X and X filtered');
grid(); % insert lines
saveas(gcf, 'ex2_36_2.png');
```

A Gaussian noise array follows a normal distribution with the parameters 0 mean and 1 variance. The array was created using the *randn* function, which was made for this purpose. The histogram plot was inspired by example 2.3 of Semmlow's book. It can be noticed that 512 points provides a resemblance of the normal distribution, although more points would provide a more accurate distribution. In order to have reproducibility of the results of the code, a random number generator default was set. Otherwise, every execution of the code would generate a different array.

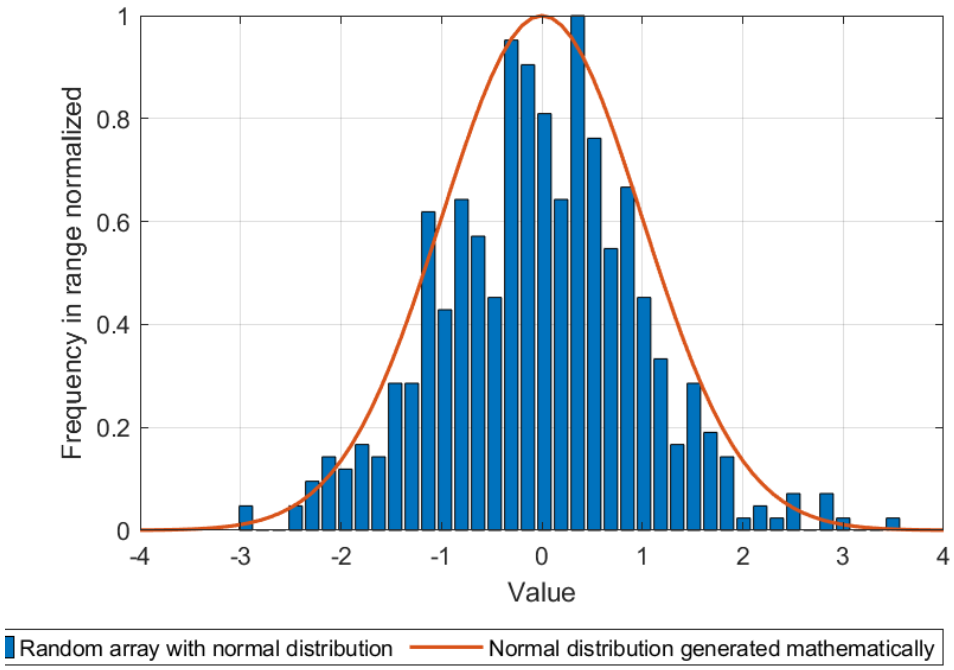

Figure 5: Exercise 2.36 - Normal distribution

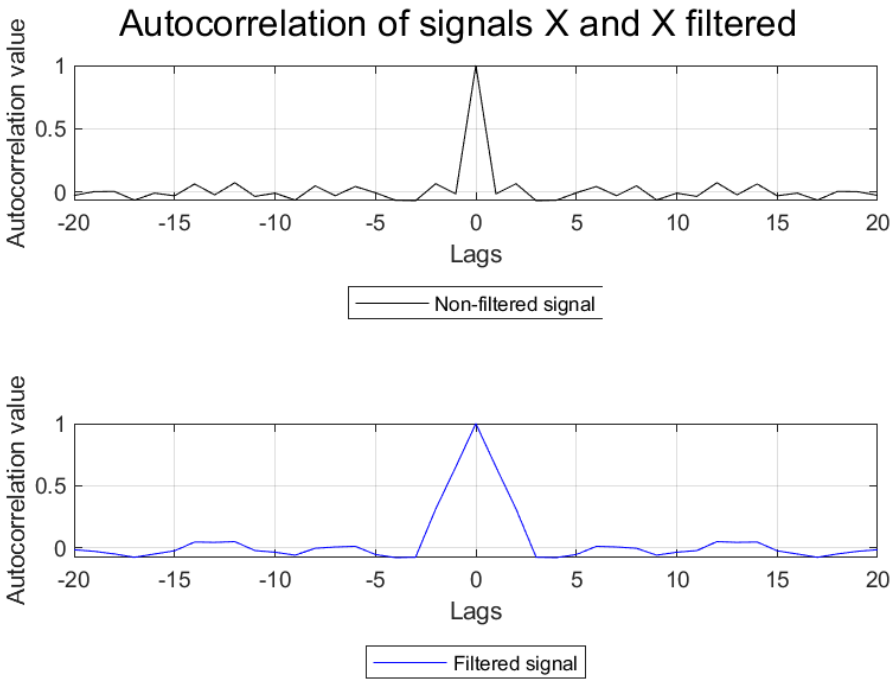

Figure 6: Exercise 2.36 - Correlation

Comparing the subplots from Figure 6, it is evident that between -3 to 3 lags the auto correlation increased. The statement of the textbook of this course could be experimentally proven. The digital filter indeed imposed some correlation between lagged versions of the signal to a signal that is intrinsically low auto-correlated.

1.5 Exercício 2.38:

Modify Example 2.12 to use a 2500-points uniformly distributed random array as the input. Use convolution to find the response of this system to a random input. Note the similarity between the response to a random input and that of a step.

```
clc; clear all; close all;

fs = 500;
TT = 5;
```

```
N = fs*TT;
t = (0:N-1)/fs;
tau = 1;
h = exp(-t./tau); % impulse response

random = rand(1, N); % uniform random array
step = ones(1, N); % step array

y_r = conv(random, h); % random convolution
y_s = conv(step, h); % step convolution

subplot(1,2,1);
plot(t, h, 'k', 'DisplayName', 'Impulse Response'); % impulse response plot
axis([0 5 0 1]);
xlabel('Time (s)'); % x-axis name
ylabel('$h(t)$', 'Interpreter', 'latex', 'Fontsize', 16); hold on;
legend();
grid();
title('Impulse response');

subplot(1,2,2);
plot(t, y_r(1:N), 'b', 'DisplayName', 'Random Response');
hold on;
plot(t, y_s(1:N), 'r', 'DisplayName', 'Step Response');
axis([0 5 0 500]);
xlabel('Time (s)'); % x-axis name
ylabel('$y_{r}(t)$, $y_{s}(t)$', 'Interpreter', 'latex', 'Fontsize', 16);

legend();
grid();
title('Random and Step Responses');
sgtitle('Convolution');

saveas(gcf,sprintf('%s.png',mfilename));
```

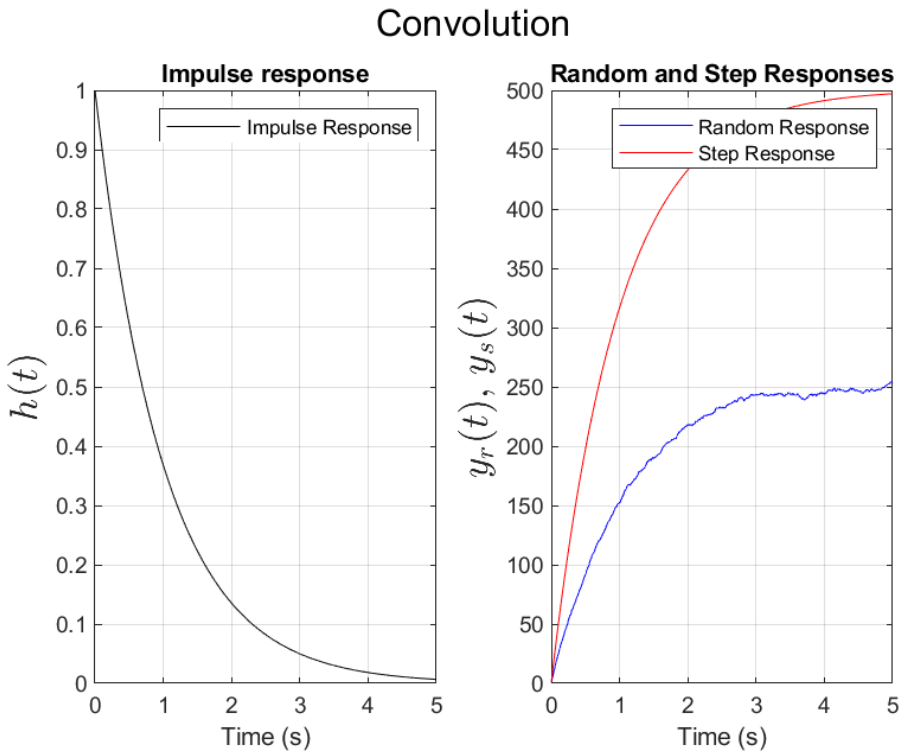

Figure 7: Exercise 2.38 - Random and Step Responses

According to the figure 7, the response to a random input has a significant similarity with the unit step response. The shape of the curve is the same, but the step response behaves in a less oscillatory way, mainly, when close to the saturation, and, in addition, has a higher response than the random response.