Fourier Series
Laboratory II

Patrycja Nazim, Adrian Król, Kamil Chaj

1 Aim of the exercise

The purpose of exercise was to experimentally familiarize with the Fourier series - an operation that allows to represent any real periodic signal by the sum of sinusoidal signals. We measured the signals generated by the independent function generator with a vector spectrum analyzer.

2 What is Fourier Series

A Fourier series is an expansion of a periodic function $f(x)$ in terms of an infinite sum of sines and cosines. Fourier series make use of the orthogonal relationships of the sine and cosine functions. The computation and study of Fourier series is known as harmonic analysis and is extremely useful to break up an arbitrary periodic function into a set of simple terms that can be plugged in, solved individually, and then recombined to obtain the solution to the original problem or an approximation to it to whatever accuracy is desired or practical

3 Course of measurements

During our measurements we skipped first part of exercise and measured all signals in configuration for second part (Fig. 1).

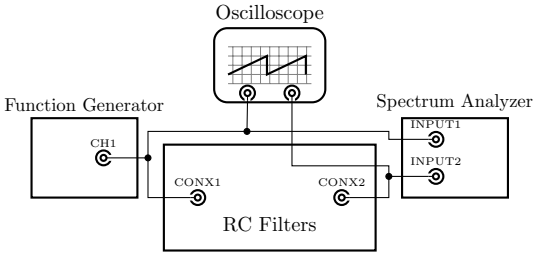


Figure 1: Measurements configuration

After connecting Function Generator, Spectrum Analyzer, Oscilloscope and board with RC filters according to above configuration (Fig. 1) we set Function Generator to Amplitude to $V_{pp} = 5$V and Frequency 1 kHz. With everything set up we proceeded with exercise and recording output of Oscilloscope and Spectrum Analyzer for signals:

- Sin wave
- Square wave 50% duty cycle
- Square wave 25% duty cycle
- Triangle wave 50% symmetry ratio
- Triangle wave 40% symmetry ratio

For Square wave 50% duty cycle and Triangle wave 40% symmetry ratio we also recorded Response for both RC Circuits (Fig. 2)

(a) Circuit A (b) Circuit B

Figure 2: Measured RC Circuits

4 Method for calculating Fourier Series coefficients

All calculations are made using Matlab with Signal Processing Toolbox, source code can be found in the Appendix A

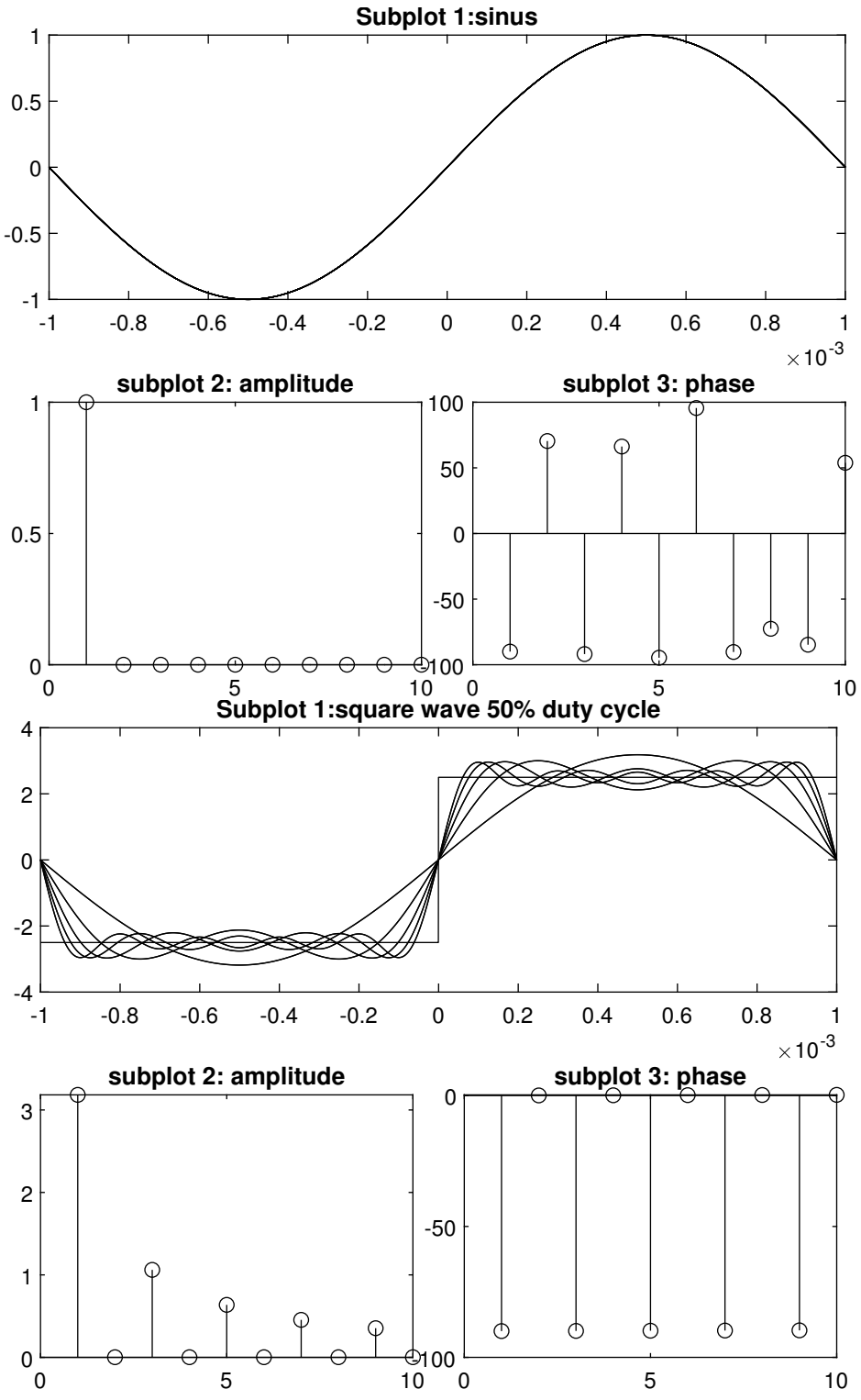
5 Comparison

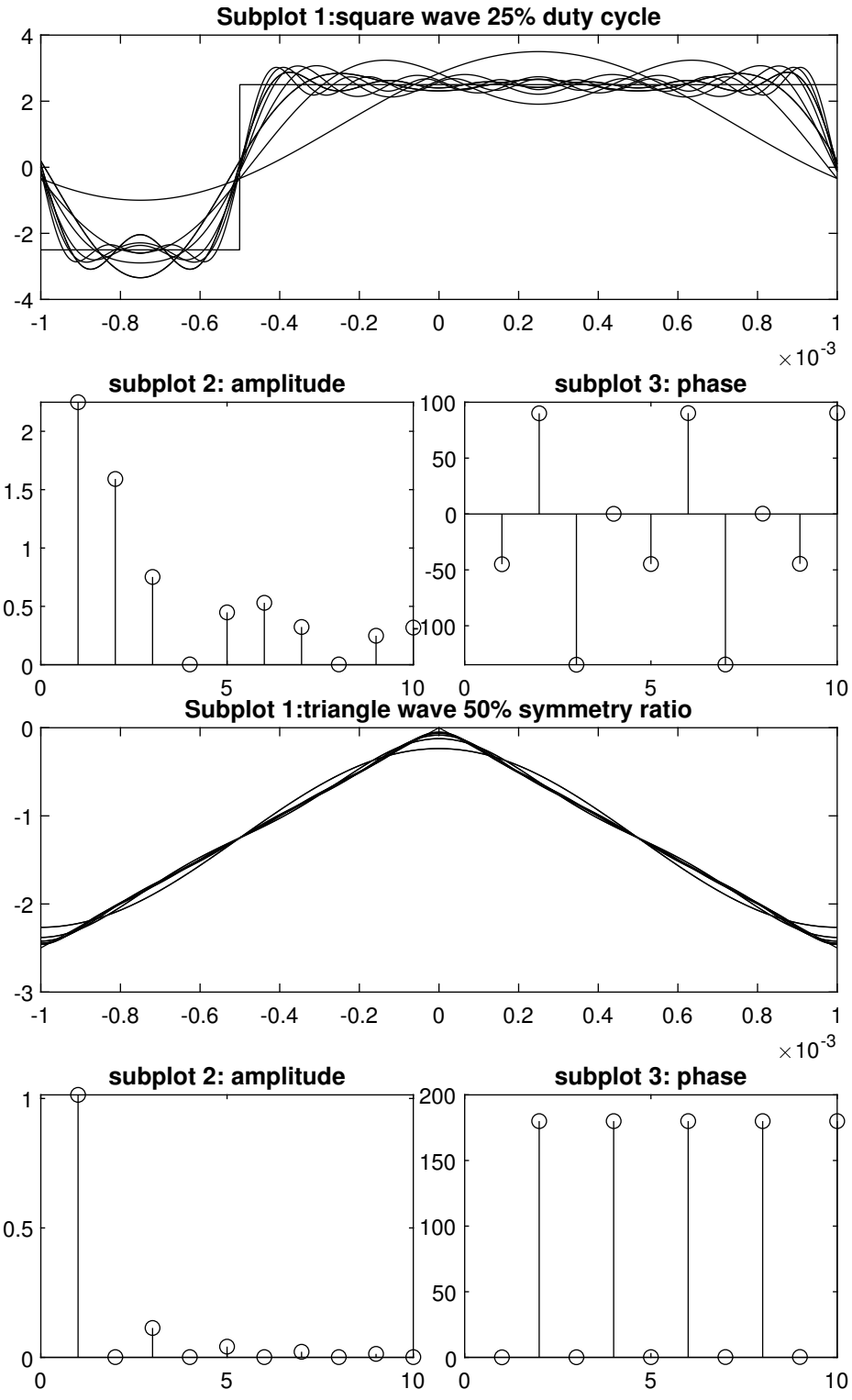
5.1 Pure signals

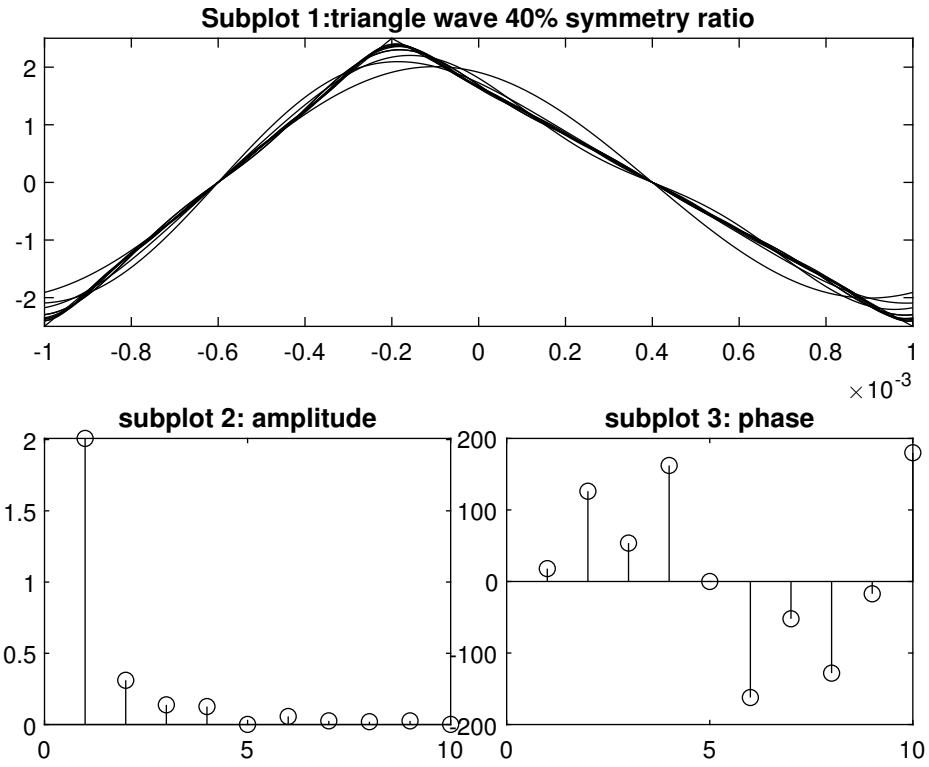
5.2 RC filters

5.2.1 Low-pass filter

5.2.2 High-pass filter


Subplot 1:sinus
subplot 2: amplitude
subplot 3: phase
Subplot 1:square wave 50% duty cycle
subplot 2: amplitude
subplot 3: phase


Subplot 1:square wave 25% duty cycle
4
2
0
-2
-4
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
×10-3
subplot 2: amplitude
2
1.5
1
0.5
0
0
5
10
subplot 3: phase
100
50
0
-50
-100
0
5
10
Subplot 1:triangle wave 50% symmetry ratio
0
-1
-2
-3
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
×10-3
subplot 2: amplitude
1
0.5
0
0
5
10
subplot 3: phase
200
150
100
50
0
0
5
10


Subplot 1:triangle wave 40% symmetry ratio
2
1
0
-1
-2
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
×10-3
subplot 2: amplitude
2
1.5
1
0.5
0
0
5
10
subplot 3: phase
200
100
0
-100
-200
0
5
10

6 Summary

A Source Code

GITHUB repository

Fourier.m

```
clear all; close all; clc;

%domain
frequency = 1000;
Vpp = 5;
L = 1/frequency;
N = 5.*1024;
dx = 2*L/(N-1);
x = -L:dx:L;

%%signals
%sin
Sinus = 0*x;
Sinus(1:N) = sin(pi*frequency*x);
%square duty cycle 50
Square50 = 0*x;
Square50(1:N/2-1) = -Vpp/2;
Square50(N/2:N) = Vpp/2;
%square duty cycle 25
Square25 = 0*x;
Square25(1:N/4-1) = -Vpp/2;
Square25(N/4:N) = Vpp/2;
%triangle symetry ratio 50
Triangle50 = 0*x;
Triangle50(1:N/2-1) = Vpp*(1:N/2-1)/N-(Vpp/2);
Triangle50(N/2:N) = -Vpp*(N/2:N)/N+(Vpp/2);
%triangle symetry ratio 40
Triangle40 = 0*x;
Triangle40(1:(4*N/10-1)) = (Vpp/0.4*(1:4*N/10-1))/N-(Vpp*0.5);
Triangle40((4*N/10):N) = -(Vpp/0.6*(4*N/10:N))/N+(Vpp*1.167);
%%RC filters responses
% [S5I, S5D, T4I, T4D, time] = Import_csv('csv/RC_filters_1.csv');
% figure(1);
% plot(time, S5I);
% figure(2)
% plot(time, S5D);
% figure(3)
% plot(time, T4I);
% figure(4)
% plot(time, T4D);

pause(1);
%fourier coefficient
fourier_coefficient(Sinus, frequency, 'sinus', 'sin');
fourier_coefficient(Square50, frequency, 'square wave 50% duty cycle', 'sqr50');
fourier_coefficient(Square25, frequency, 'square wave 25% duty cycle', 'sqr25');
fourier_coefficient(Triangle50, frequency, 'triangle wave 50% symmetry ratio', 'tri50');
fourier_coefficient(Triangle40, frequency, 'triangle wave 40% symmetry ratio', 'tri40');
```

```
% [test] = RC_circuit(frequency, N);
% fourier_coefficient(test,frequency, N, 'test', 'test');
```

fourier coefficient.m

```
function [amplitude, phase, T] = fourier_coefficient(f, frequency, name, filename)
figure('Name', name);
%domain
N = size(f,2);
L = 1/frequency;
dx = 2*L/(N-1);
x = -L:dx:L;

%function
subplot(2,4,[1,4]);
plot(x, f); hold on;
title(join(['Subplot 1:',name]));

%fourier coefficient
A0 = (1/L)*sum(f.*ones(size(x)))*dx;
fFS = A0/2;
for k=1:10
    A(k) = (1/L)*sum(f.*cos(pi*k*x*frequency))*dx;
    B(k) = (1/L)*sum(f.*sin(pi*k*x*frequency))*dx;
    AB(k) = sqrt(A(k).^2.+B(k).^2);
    % phi = angle(conj(complex(A(k), B(k))));
    fFS = fFS + A(k)*cos(k*pi*x*frequency) + B(k)*sin(k*pi*x*frequency);
    % fFS = fFS + AB(k)*cos((k*pi*x*frequency)+phi);
    plot(x, fFS);
    pause(.05);
end
% amplitude = 20 .* log10(AB/AB(1));
amplitude = AB;
phase = rad2deg(angle(conj(complex(A, B))));
% phase = atand(-B/A);
subplot(2,4,[5,6]);
stem(amplitude);
title('subplot 2: amplitude');
subplot(2,4,[7,8]);
stem(phase);
title('subplot 3: phase')
%T = table((1:10)', amplitude', phase', 'VariableNames', {'N', 'Amplitude', 'Phase'});
print(join(['img/',filename]), '-deps');
end
```

RC circuit.m

```
function [Vout] = RC_circuit(frequency, N)
figure('Name', 'RC circuit');
Vpp = 5;
frequency = 1000;
T = 1/frequency;
R = 1.5e3;
```

```
C1 = 47e-4;
C2 = 10;
dt = T/(N-1);
t = 0:dt:T;
Vout = 0*t;
Vout(N/2+1:N) = Vout(N/2+1:N) + Vpp;
Vout(N/2+1:N) = Vout(N/2+1:N) + (-Vpp)*exp(-(1:N/2)/(R*C1));
Vout(1:N) = Vout(1:N) + (Vpp)*exp(-(1:N)/(R*C1));
Vout(1:N) = Vout(1:N) - Vpp/2;
plot(t, Vout);
pause(.1)
end
```