SIGNALS AND SYSTEMS

DR. AMIRKHANI

SOLUTION OF SIMULATION 2

TA: HAMED HOSSEINNEJAD

iust.signal3992@gmail.com

2021

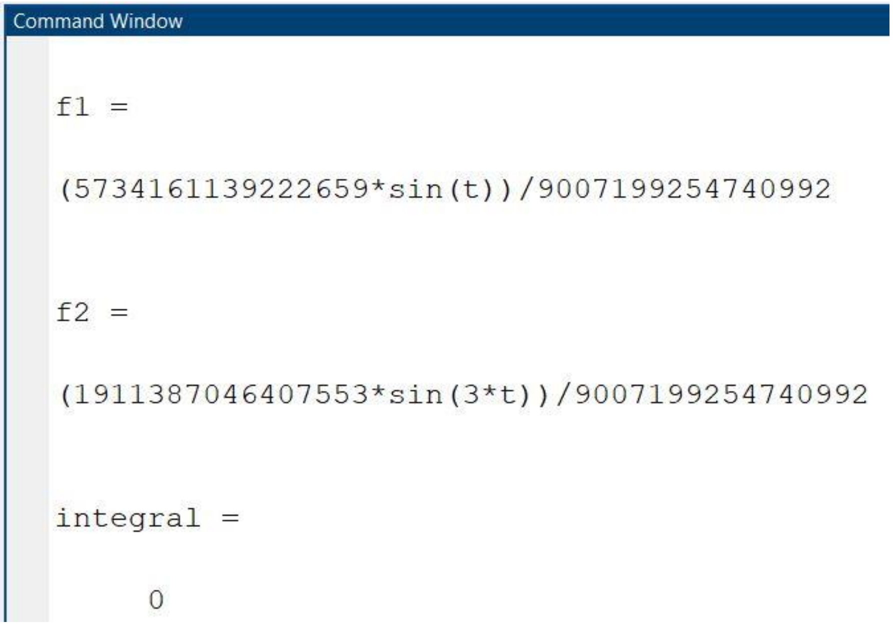
1- Consider the first two terms of a square wave that is shown below. Show that these two terms are orthogonal. Can we conclude all terms of a Fourier Series are orthogonal?

syms t
f1 = (2/pi)*sin(t)
f2 = (2/(pi*3))*sin(3*t)

Solution:

Orthogonality: $< f_1(t)\,, f_2(t) >= \int_0^T f_1(t)\,.\ f_2(t)\,dt\ =\ 0$

```
% This program shows that the terms of Fourier series are
% orthogonal. As a test case, the first two terms of a square wave are used.
clc
clear
close all
% Define variable t
syms t
% Define first term of Fourier series of square wave.
f1=2/pi*sin(t)
% Define second term of Fourier series of square wave.
f2=2/(pi*3)*sin(3*t)
% Take integral to check the first and second term of Fourier series
% of Square wave.
integral=double(int(f1*f2, t, 0, pi))
% We can Conclude that all of terms are orthogonal to each other
```

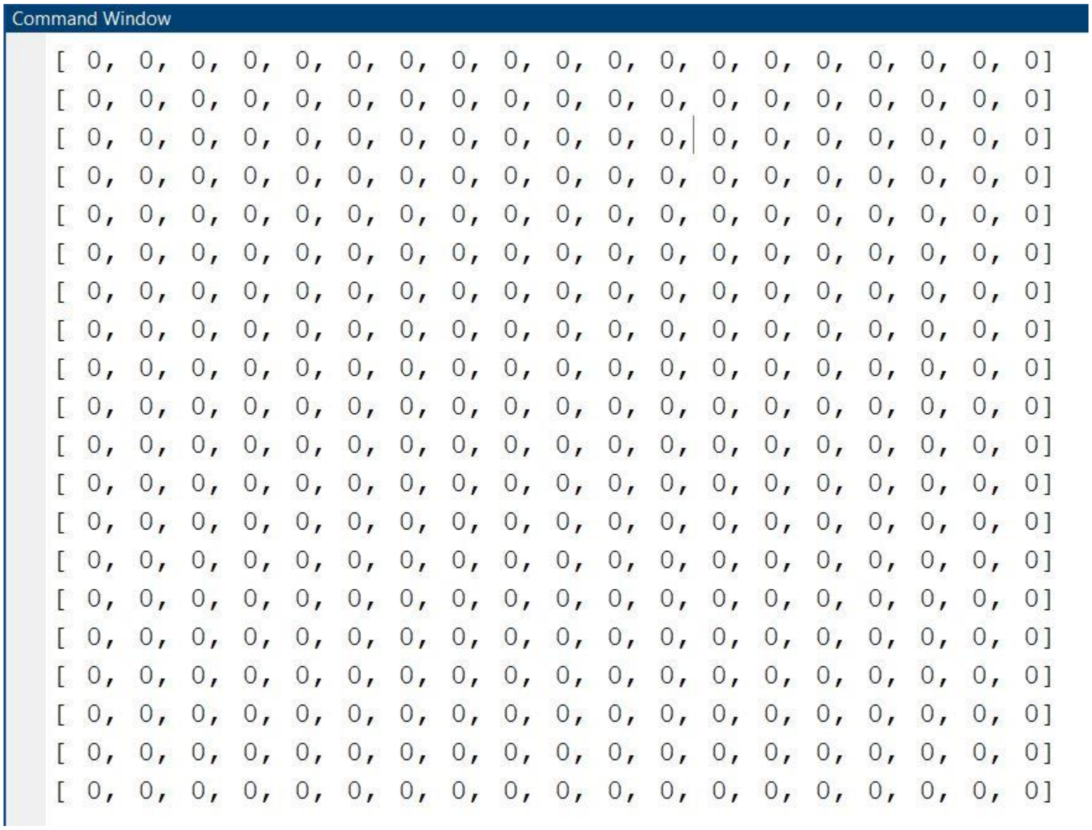


Command Window

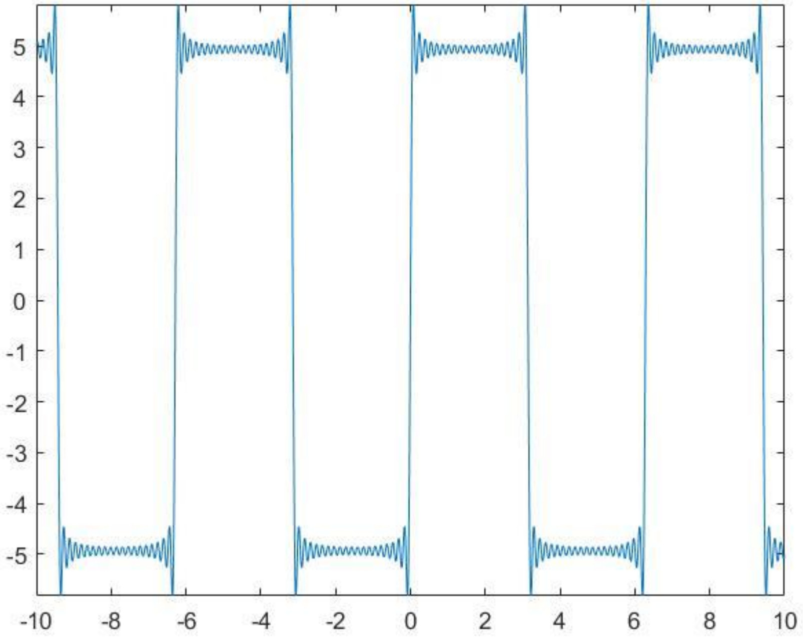
```
f1 =

(5734161139222659*sin(t))/9007199254740992


f2 =

(1911387046407553*sin(3*t))/9007199254740992


integral =

     0
```

```
clc
clear
close all
syms t %defining a symbolic var
fn = 0; %initial value for signal
N = 20; %Number of terms
for n = 1:N %a loop to quantify terms
f(n) = (2/(2.*n-1).*pi)*sin((2.*n-1).*t);
fn = fn + f(n);
```

```
end
fplot(fn,[-10 10]);
for n=1:N
for i = 1: N
if(i==n) break;end %check if 2 terms are equal
F(n,i) = int(f(n).*f(i),[-pi +pi]);
end
end
disp(F);
```

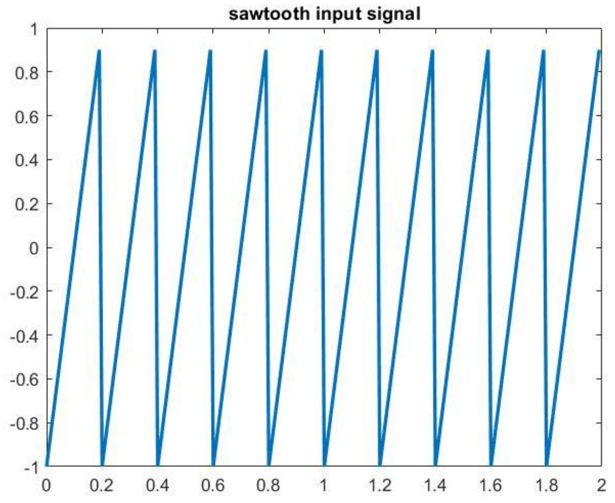


```
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

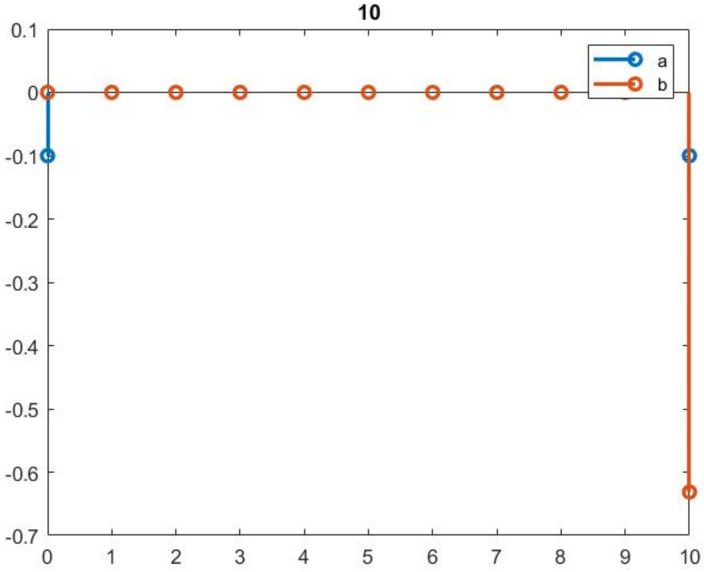


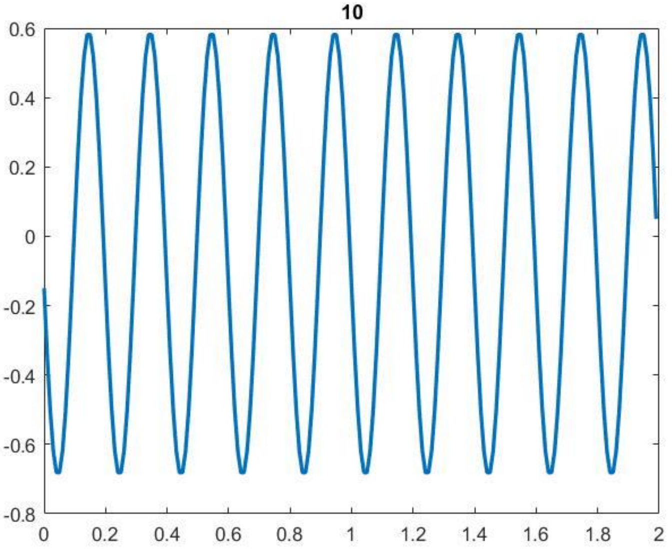
2- Generate a sawtooth signal and plot it. Then, calculate Fourier Series of that and plot coefficients and Fourier series with different amounts of N including 10, 50, 100, 10000

```
clc
clear
close all
%% function Definition
Ts = 0.01; %seconds/sample
T = 2;
t = 0:Ts:T-Ts;
f = sawtooth(2*pi*5*t);
figure(1)
plot(t,f,'LineWidth',2)
title('f')
title('sawtooth input signal')
%% an & bn
for N = [10 50 100 10000]
    a = zeros(1,N+1);
    b = zeros(1,N+1);
    for n = 0:N
        a(n+1) = (2*Ts/T)*sum(f.*cos(2*pi*n*t/T));
        b(n+1) = (2*Ts/T)*sum(f.*sin(2*pi*n*t/T));
    end
    figure
    stem(0:N,a,'LineWidth',2)
    hold on
    stem(0:N,b,'LineWidth',2)
    legend('a','b')
    title(N)
    %% caculate Fourier Series
    t2 = 0:Ts:T-Ts; % t synthesis
    fs = (a(1)/2)*ones(size(t2));
    for n = 1:N
        fs = fs +(a(n+1)*cos(2*pi*n*t2/T)+b(n+1)*sin(2*pi*n*t2/T));
    end
    figure
    plot(t2,fs,'LineWidth',2)
    title(N)
end
```

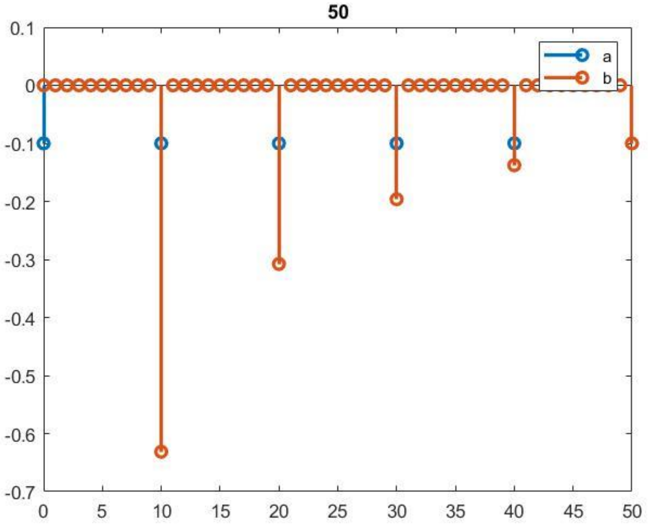


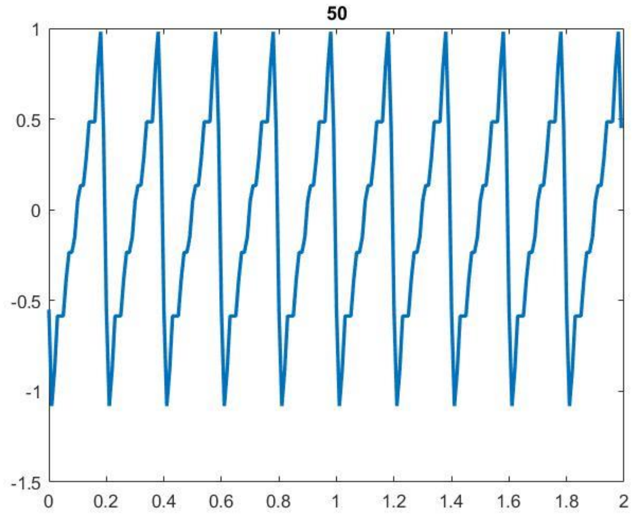
N = 10


10
a
b
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
0
1
2
3
4
5
6
7
8
9
10

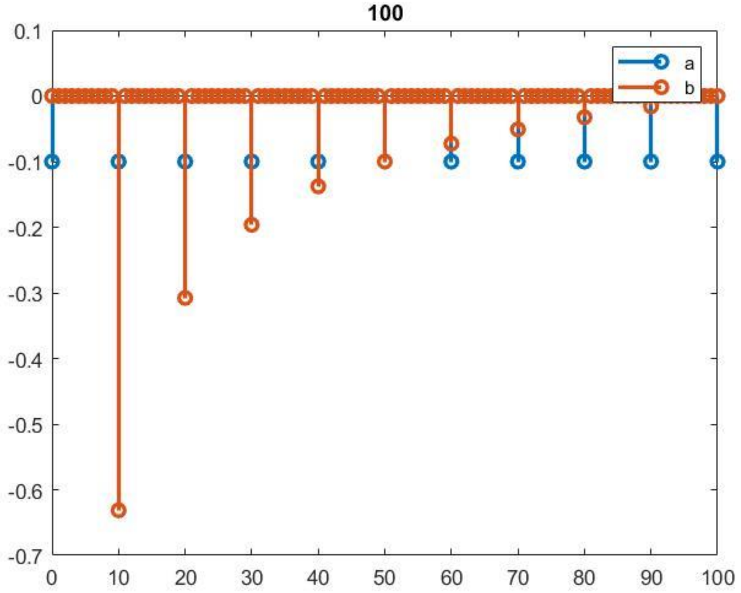

10
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

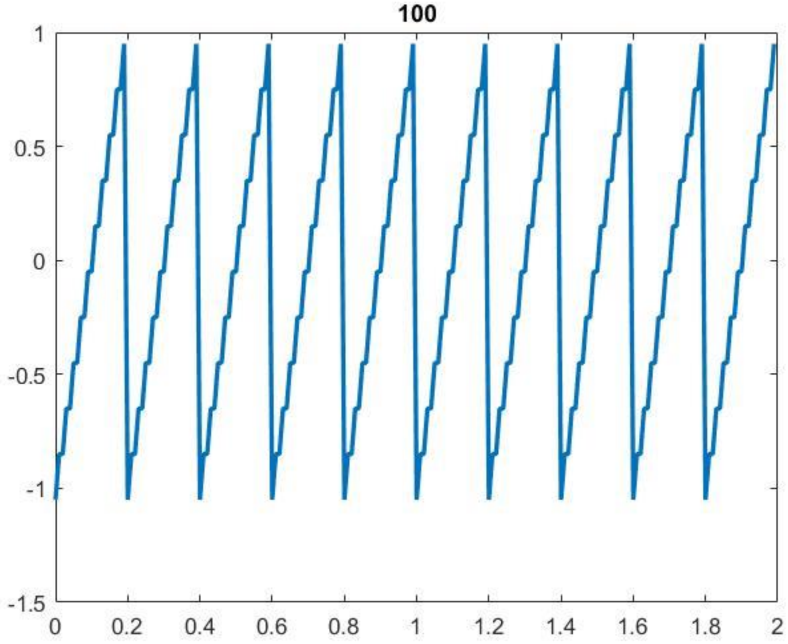
N = 50


50
a
b
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
0
5
10
15
20
25
30
35
40
45
50

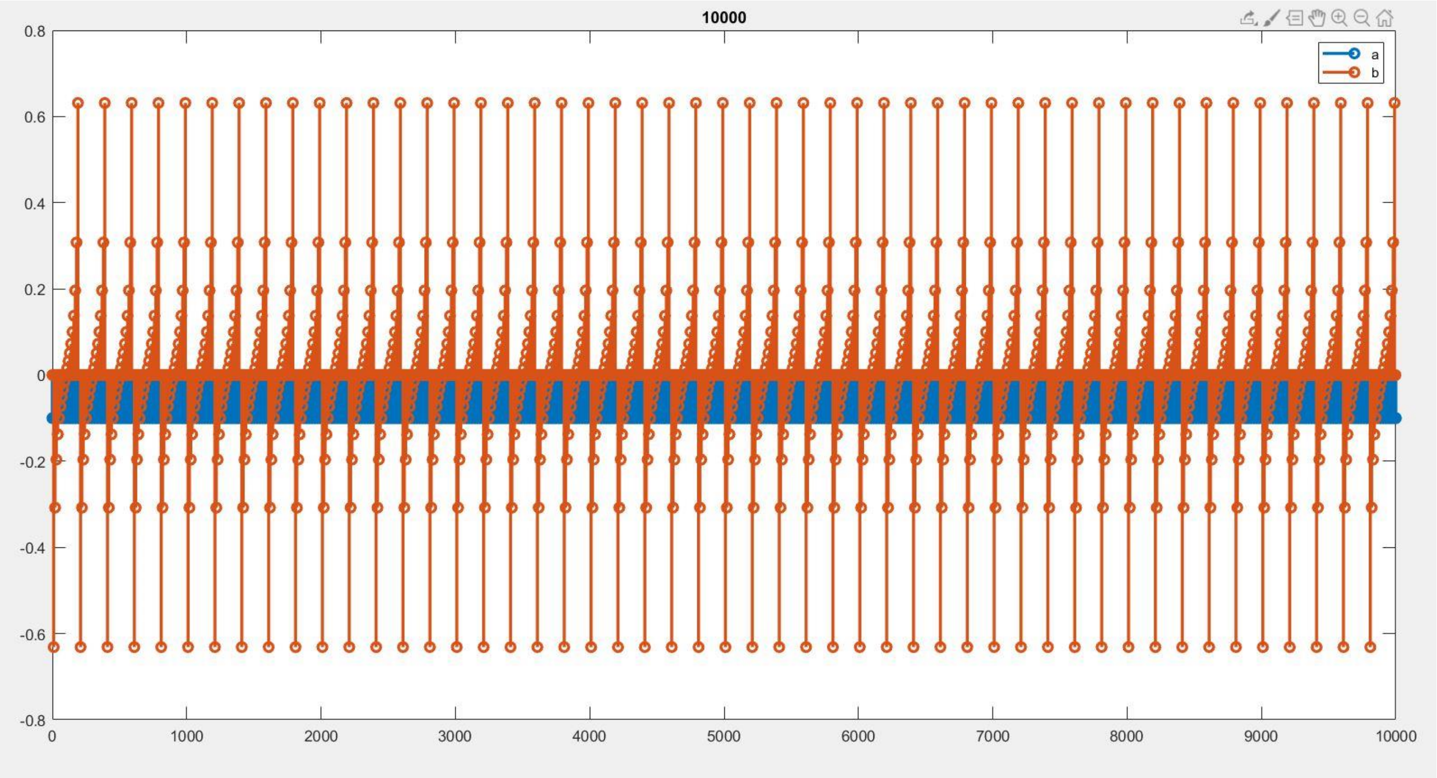

50
1
0.5
0
-0.5
-1
-1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

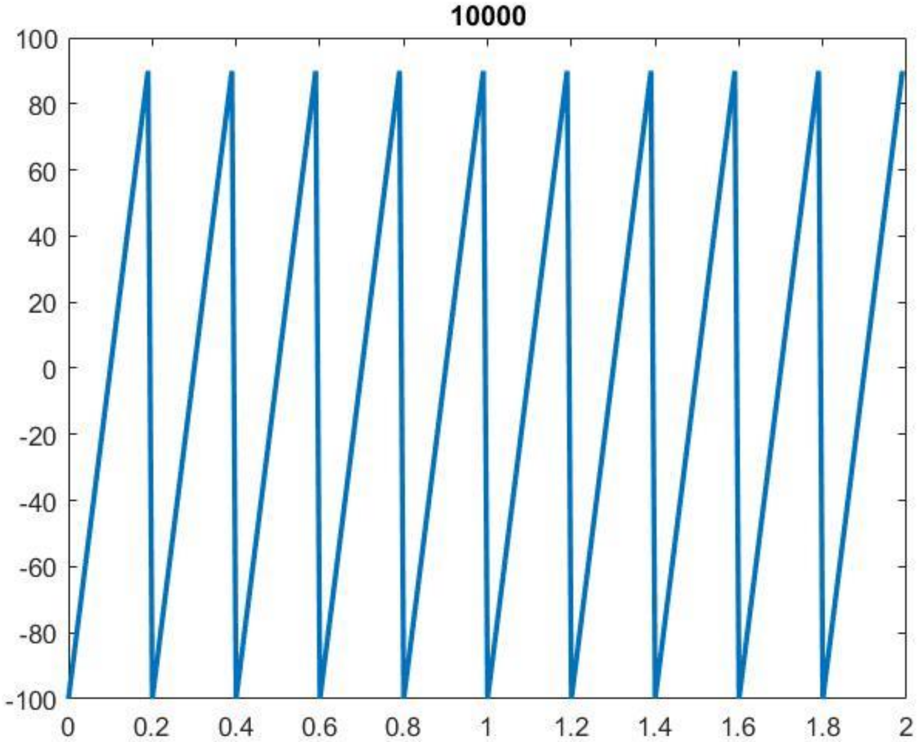
N = 100


100
a
b
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
0
10
20
30
40
50
60
70
80
90
100


100
1
0.5
0
-0.5
-1
-1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

N = 10000


10000
a
b
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000

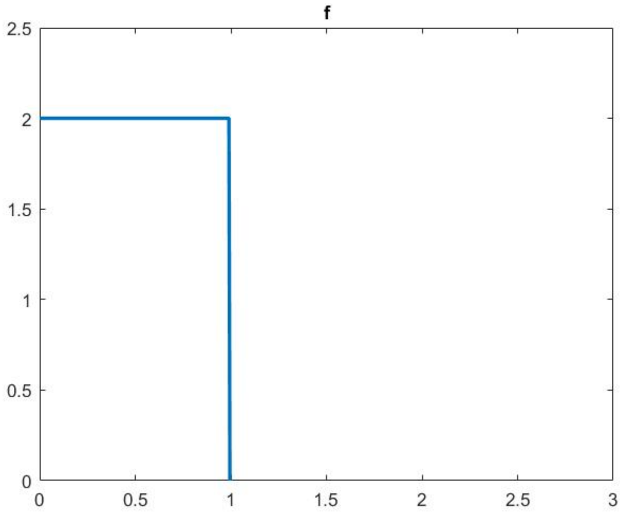

10000
100
80
60
40
20
0
-20
-40
-60
-80
-100
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

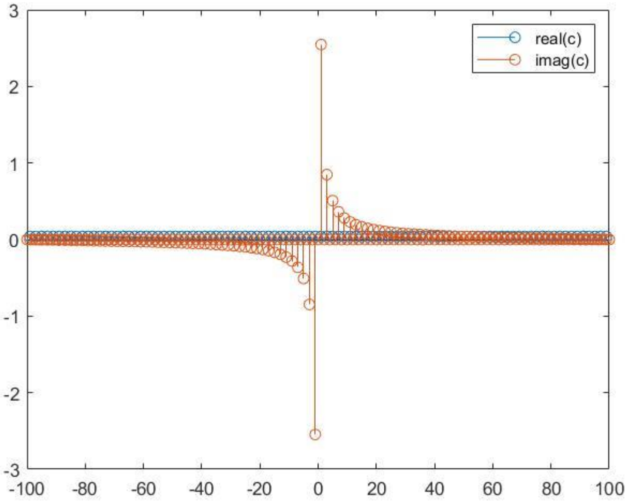
Extra:

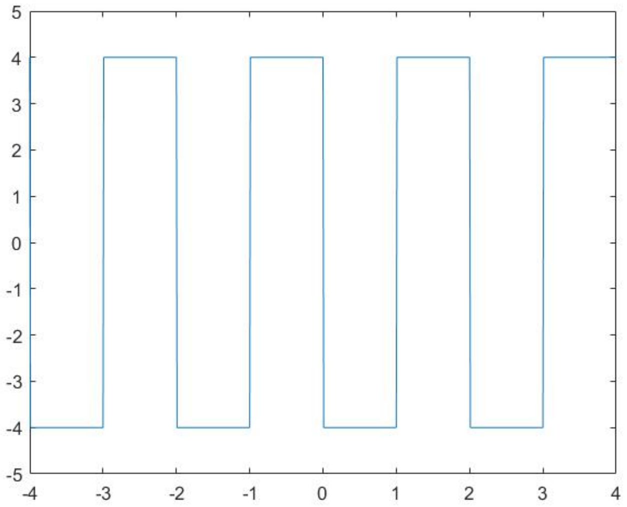
First, calculate complex-valued coefficients, second, plot Fourier series complex exponential form. (consider desired pulse input and other proper parameters)

```
clc
clear
close all
%% function Definition
Ts = 0.01; %seconds/sample
T = 2;
t = 0:Ts:T-Ts;
f(t < T/2)=2;
f((t>=T/2)&(t<T)) = -2;
figure(1)
plot(t,f,'LineWidth',2)
xlim([0 3])
ylim([0 2.5])
title('f')
% %% an & bn
% N = 100;
% a = zeros(1,N+1);
% b = zeros(1,N+1);
% for n = 0:N
%   a(n+1)= (2*Ts/T)*sum(f.*cos(2*pi*n*t/T));
%   b(n+1) = (2*Ts/T)*sum(f.*sin(2*pi*n*t/T));
% end
% figure(2)
% stem(0:N,a)
% hold on
% stem(0:N,b)
% legend('a','b')
% %% caculate Fourier Series
% t2 = -2*T:Ts:2*T; % t synthesis
% fs = (a(1)/2)*ones(size(t2));
% for n = 1:N
%  fs = fs + (a(n+1)*cos(2*pi*n*(t2)/T) + b(n+1)*sin(2*pi*n*(t2)/T));
% end
% figure(3)
% plot(t2,fs)
%% Calculate complex valued coefficients
N = 100;
c = zeros(1,2*N+1);
for n = -N:N
    c(n+1+N) = (2*Ts/T)*sum(f.*exp(1i*2*pi*n*t/T));

end
figure(4)
stem(-N:N,real(c))
hold on
stem(-N:N,imag(c))
legend('real(c)','imag(c)')
%% Fourier Series Complex Exponential Form
```

```
t2 = -2*T:Ts:2*T;
fs = zeros(size(t2));
for n = -N:N
   fs = fs + (c(n+N+1)*exp(1i*2*pi*n*t2/T));
end
figure(5)
plot(t2,real(fs))
```


f
2.5
2
1.5
1
0.5
0
0
0.5
1
1.5
2
2.5
3


real(c)
imag(c)


5
4
3
2
1
0
-1
-2
-3
-4
-5
-4
-3
-2
-1
0
1
2
3
4