浙江工業大學

MATLAB & Communication Simulations

Lab Report2

Name	凌智城
Teacher	张昱
Class	通信工程 1803 班
Student ID	201806061211
Department	信息工程学院

Date： Apr. 5th,2021

I. Task_1

A. Code

```
%% Generate the following signal
% where the duration is [0,5], sampling rate is 1000Hz.
% Plot the signal and its magnitude spectrum.
% x(t)=t*sin(100*pi*t), for t within [0,2]
% x(t)=(sin(pi*t)+2)*sin(100*pi*t), otherwise

clear;
clc;

Fs = 1000;                     % Sampling rate
Ts = 1/Fs;                     % Sampling interval
t = 0:Ts:5;                    % Time vector
x = zeros(1,length(t)); % Input signal initiation
for i = 1:length(t)
    if t(i)>=0&&t(i)<=2
        x(i) = t(i)*sin(100*pi*t(i));
    elseif t(i)>2&&t(i)<=5
        x(i) = (sin(pi*t(i))+2)*sin(100*pi*t(i));
    end
end

[f,sf] = T2F(t,x);          % FT

figure(1)
plot(t,x,'b-')
title('Orignal signal')
```

```
xlabel('t/s')
ylabel('x(t)')
grid on

figure(2)
plot(f,abs(sf),'R--')
title('Spectrum')
xlabel('f/Hz')
ylabel('sf')
grid on
```

B. Figure

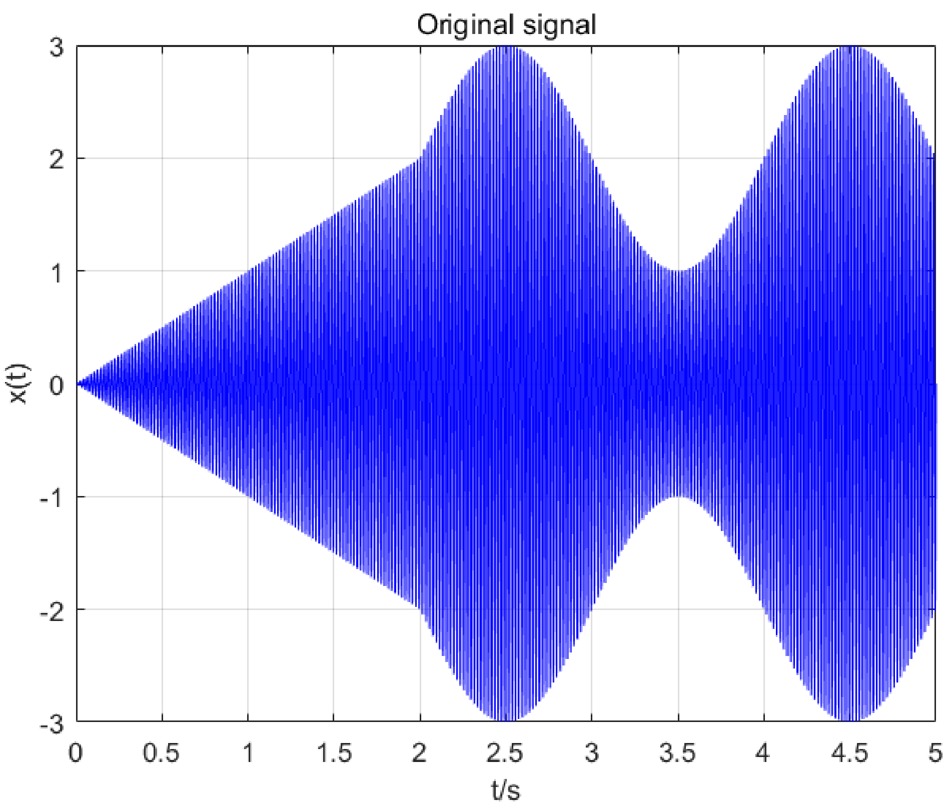


Figure1：Original signal

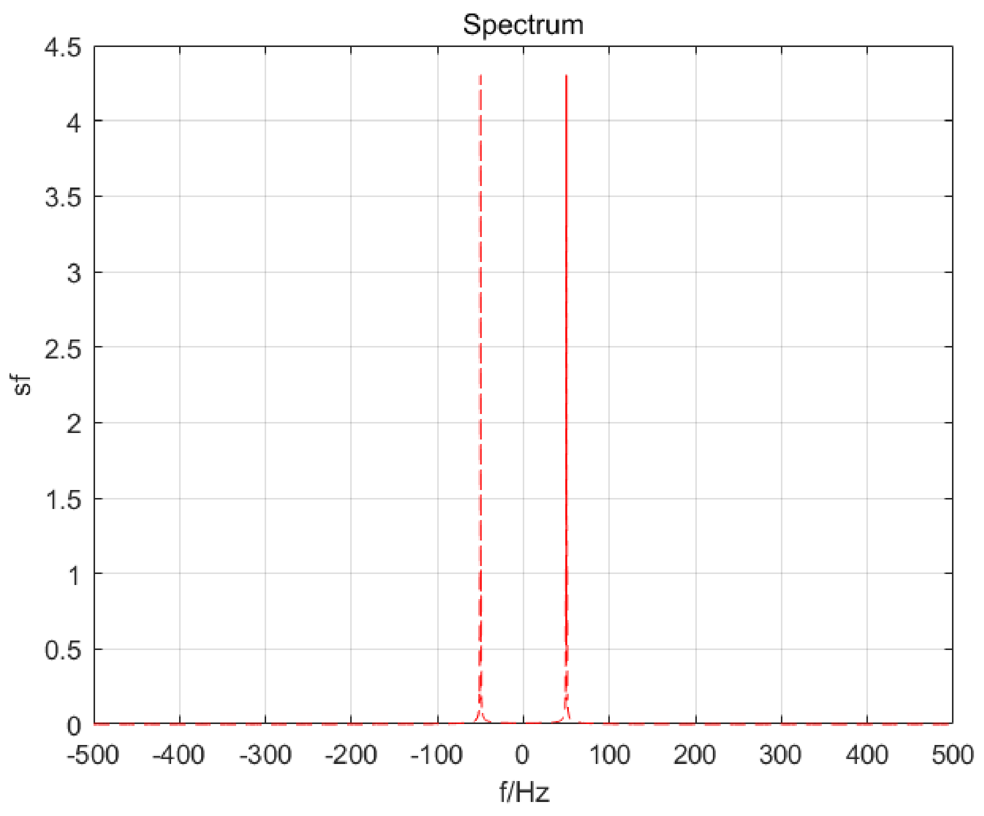


Figure2：Spectrum of x(t)

C. Discussion

For the initiation of input signal, we have two choices, the one is using ‘ts’ and length of x to initial it, while another way is to use ‘while’ or ‘for’ loop to initial it. In this code above, we use the second way. To plot the spectrum of this signal, we can use the F2T function and then plot it.

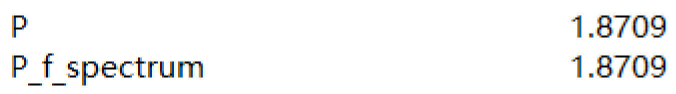
II. Task_2

A. Code

```
%% Calculate the energy and power of x(t)
% in both time domain and frequency domain, and verify the equality.

clear;
clc;
```

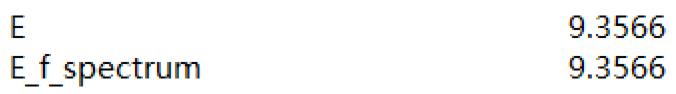
```
Fs = 1000;                    % Sampling rate
Ts = 1/Fs;                    % Sampling interval
t = 0:Ts:5;                   % Time vector
x = zeros(1,length(t)); % Input signal initiation
for i = 1:length(t)
    if t(i)>=0&&t(i)<=2
        x(i) = t(i)*sin(100*pi*t(i));
    elseif t(i)>2&&t(i)<=5
        x(i) = (sin(pi*t(i))+2)*sin(100*pi*t(i));
    end
end
[f,sf] = T2F(t,x);            % FT
dt = t(2)-t(1);               % Sampling period
df = f(2)-f(1);               % Frequency accuracy
N = length(t);                % Length of t
T = t(end)-t(1)+dt;           % Signal duration

E = sum(abs(x).^2)*dt;                    %Energy of x(t) in time domain
P = sum(abs(x).^2)/N;                     %Power of x(t) in frequency domain
E_f_spectrum = sum(abs(sf).^2)*df;    %Energy of x(t) in time domain
P_f_spectrum = sum(abs(sf).^2)*df/T;%Power of x(t) in frequency domain
```

B. Figure



```
P                           1.8709
P_f_spectrum                1.8709
```

Figure3：Power in time and frequency domain



```
E                           9.3566
E_f_spectrum                9.3566
```

Figure4：Energy in time and frequency domain

C. Discussion

In time domain, ‘E=sum(x.*conj(x))*dt’ or ‘E=sum(abs(x).^2)*dt’, ‘P=sum(x.*conj(x))/N’ or ‘P=sum(abs(x).^2)/N’ ; while in frequency domain, ‘E=sum(x.*conj(x))*df’ or ‘E=sum(abs(x).^2)*df’, ‘P=sum(x.*conj(x))*df/T’ or ‘P=sum(abs(x).^2)*df/T’. ‘dt’ is sampling period, ‘df’ is frequency accuracy, ‘N’ is length of t, and the ‘T’ is the duration of this signal. In this task, we can see that both in the time domain and frequency domain, energy and power of x(t) is equal, so we verify the Parseval's theorem.

III. Task_3

A. Code

```
%% Plot the time-averaged autocorrelation and the power spectral density of x(t)
% verify the Wiener-Khinchin theorem.

clear;
clc;

Fs = 1000;                          % Sampling rate
Ts = 1/Fs;                          % Sampling interval
t = 0:Ts:5;                         % Sampling period
x = zeros(1,length(t)); % Input signal initiation
for i = 1:length(t)
    if t(i)>=0&&t(i)<=2
        x(i) = t(i)*sin(100*pi*t(i));
    elseif t(i)>2&&t(i)<=5
        x(i) = (sin(pi*t(i))+2)*sin(100*pi*t(i));
        end
end
N = length(t);                    % Length of t
dt = t(2)-t(1);                 % Sampling period
T = t(end)-t(1)+dt;          % Signal duration

R = xcorr(x);                       % average autocorrelation
R = R*Ts/T;
tau = -5:Ts:5;

figure(1);
plot(tau,R);
title('Time-averaged autocorrelation')
xlabel('t/s')
ylabel('R(t)')
grid on

[f,sf] = T2F(t,x);                          % FT
P_f_spectrum = abs(sf).^2/T;        % Power spectral density(psd)
[f1,R_f] = T2F(tau,R);                    % FT of the autocorrelation

figure(2)
plot(f,P_f_spectrum)
%semilogy(f,P_f_spectrum)
title('Power spectral density of x(t)')
xlabel('f/Hz')
ylabel('P\_f\_spectrum')
grid on

figure(3)
```

```
%semilogy(f1,abs(R_f))
plot(f1,abs(R_f))
title('PSD of x(t) thorw Wiener-Khinchin theorem')
xlabel('f/Hz')
ylabel('abs(R\_f)')
grid on
```

B. Figure

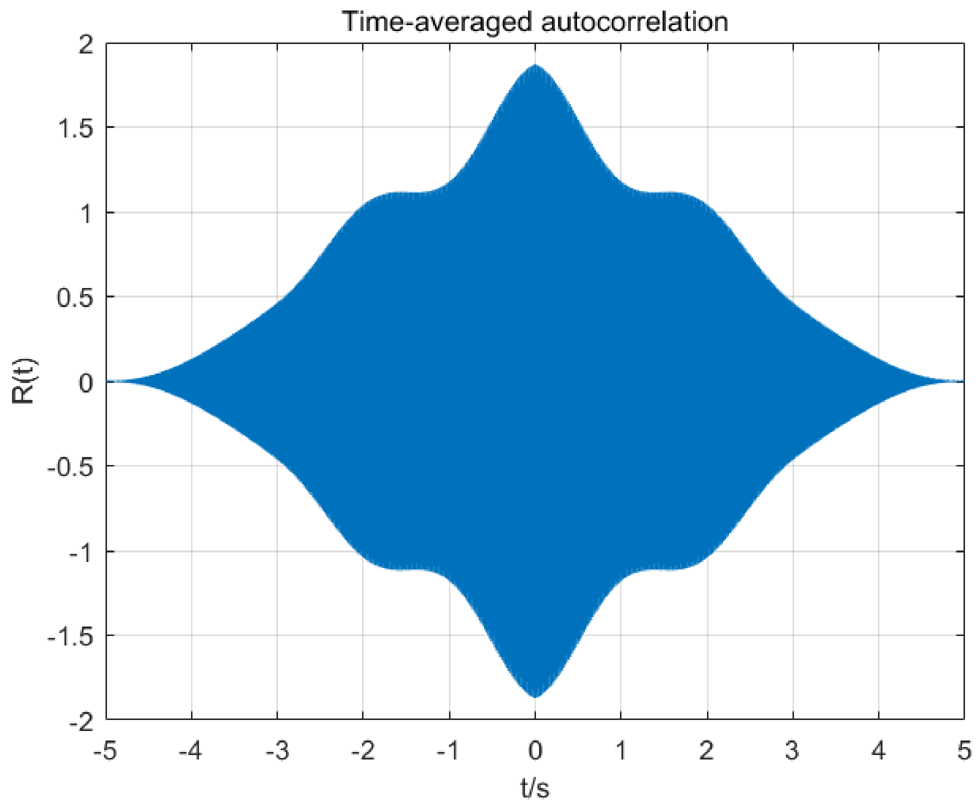


Figure5：Time-averaged autocorrelation

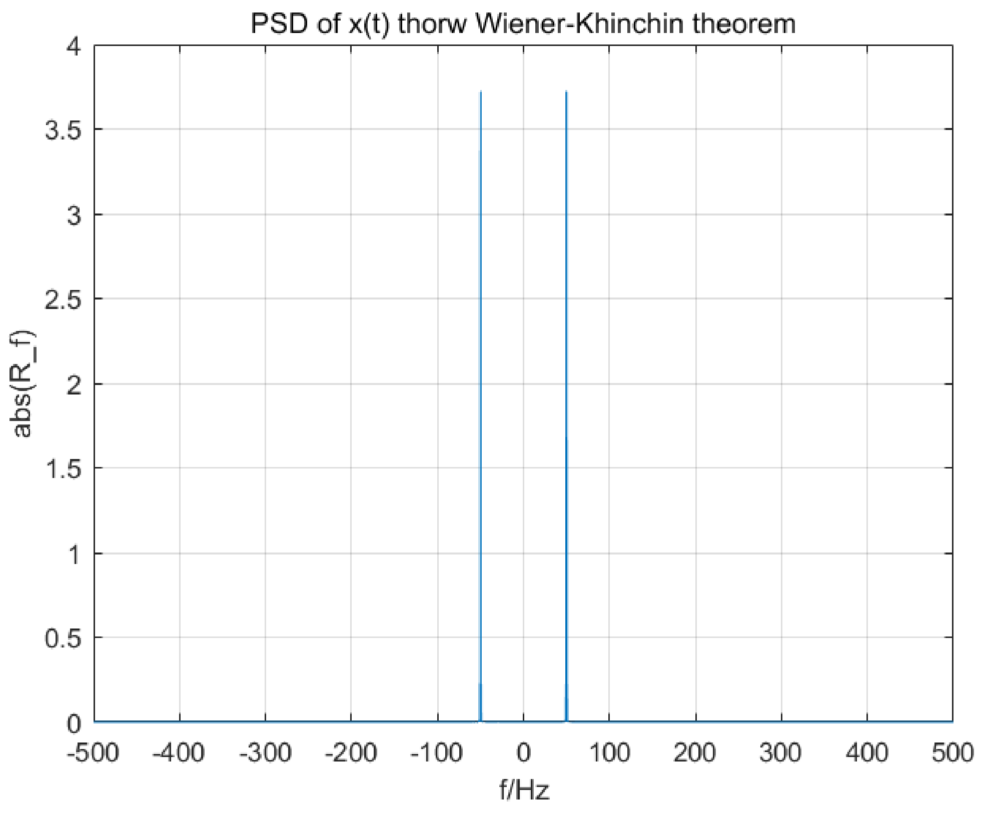


Figure6：PSD of x(t) throw Wiener-Khinchin theorem

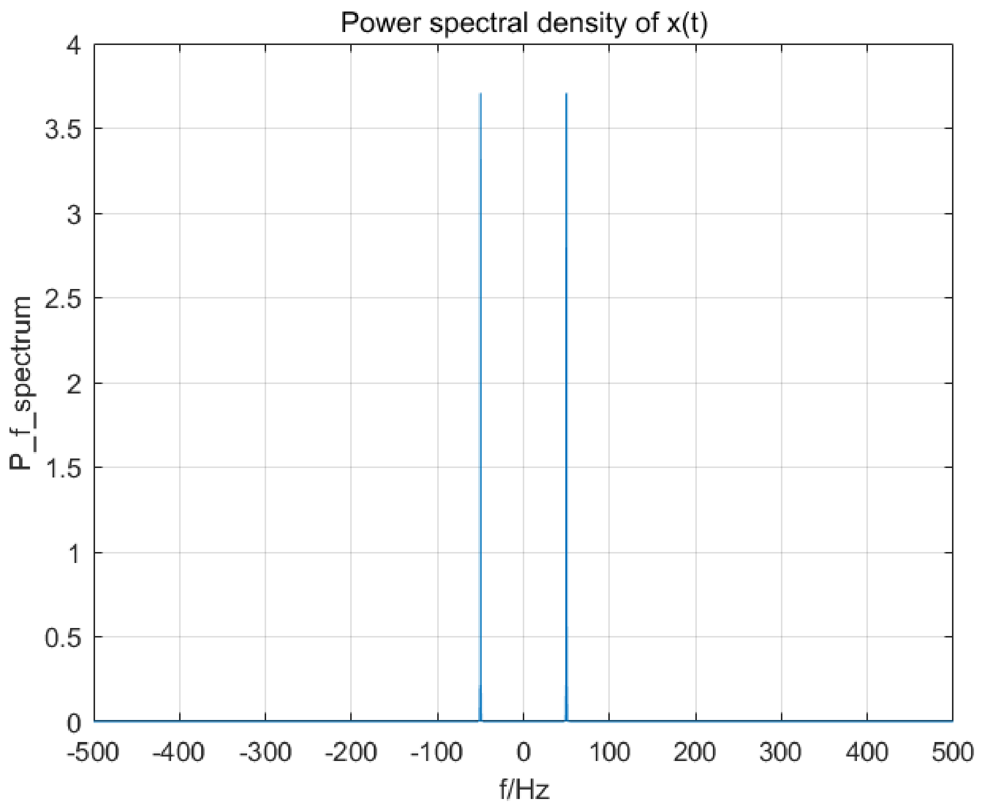


Figure7：Power spectral density of x(t)

C. Discussion

Wiener-Khinchin theorem: The power spectral density of any generalized stationary random process with a constant mean value is the Fourier transform of its autocorrelation function. ’tau’ of ‘xcorr’ is the length of autocorrelation, of course, ‘tau’ is double of ‘t’. The application of the ‘xcorr’ command in engineering is usually to process the sampled data sequence x in time. When the data points are collected and handed over to Matlab for processing, Matlab does not know your sampling time interval, it only inputs according to the calculation process The data sequence x is calculated, but we can define the time interval ourselves, for example, dt=0.01, at this time t=dt*b represents the time delay in the correlation calculation, the first half is leading, and the second half is lagging.

IV. Task_4(Optional)

A. Code

```
%% Plot the quadrature component of x(t) where the carrier fc=50Hz
% plot the corresponding spectrum.

clear;
clc;

Fs = 1000;                    % Sampling rate
Ts = 1/Fs;                    % Sampling interval
t = 0:Ts:5;                   % Time vector
x = zeros(1,length(t)); % Input signal initiation
for i = 1:length(t)
    if t(i)>=0&&t(i)<=2
        x(i) = t(i)*sin(100*pi*t(i));
    elseif t(i)>2&&t(i)<=5
```

```
        x(i) = (sin(pi*t(i))+2)*sin(100*pi*t(i));
    end
end
[f,sf] = T2F(t,x);            % FT
dt = t(2)-t(1);               % Sampling period
df = f(2)-f(1);               % Frequency accuracy
N = length(t);                  % Length of t
T = t(end)-t(1)+dt;           % Signal duration

% Quadrature component
x_a = hilbert(x);
fc = 50;                                       % Carrier fc=50Hz
x_l = x_a.*exp(-1i*2*pi*fc*t);    % Lowpass equivalent
x_i = real(x_l);                          % In-phase component
x_q = imag(x_l);                          % Quadrature component
[f_i,sf_i] = T2F(t,x_i);              % FT of in-phase component
[f_q,sf_q] = T2F(t,x_q);              % FT of quadrature component

figure(1)
plot(t,x_q)
title('Quadrature component of x(t)')
xlabel('t/s')
ylabel('x\_q(t)')
grid on

figure(2)
plot(f_q,abs(sf_q),'R-')
title('Spectrum of quadrature component')
xlabel('f/Hz')
ylabel('sf\_q')
```

```
grid on

figure(3)
plot(t,x_i)
title('In-phase component of x(t)')
xlabel('t/s')
ylabel('x\_i(t)')
grid on

figure(4)
plot(f_i,abs(sf_i),'R-')
title('Spectrum of in-phase component')
xlabel('f/Hz')
ylabel('sf\_i')
grid on
```

B. Figure

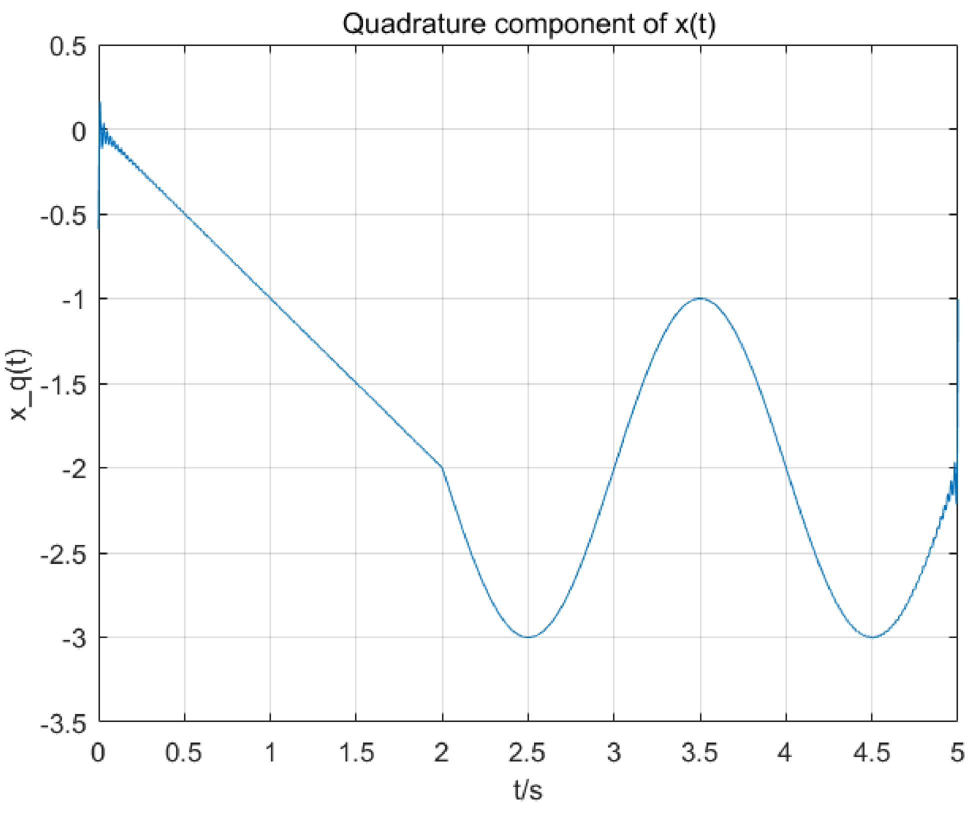


Figure8：Quadrature component of x(t)

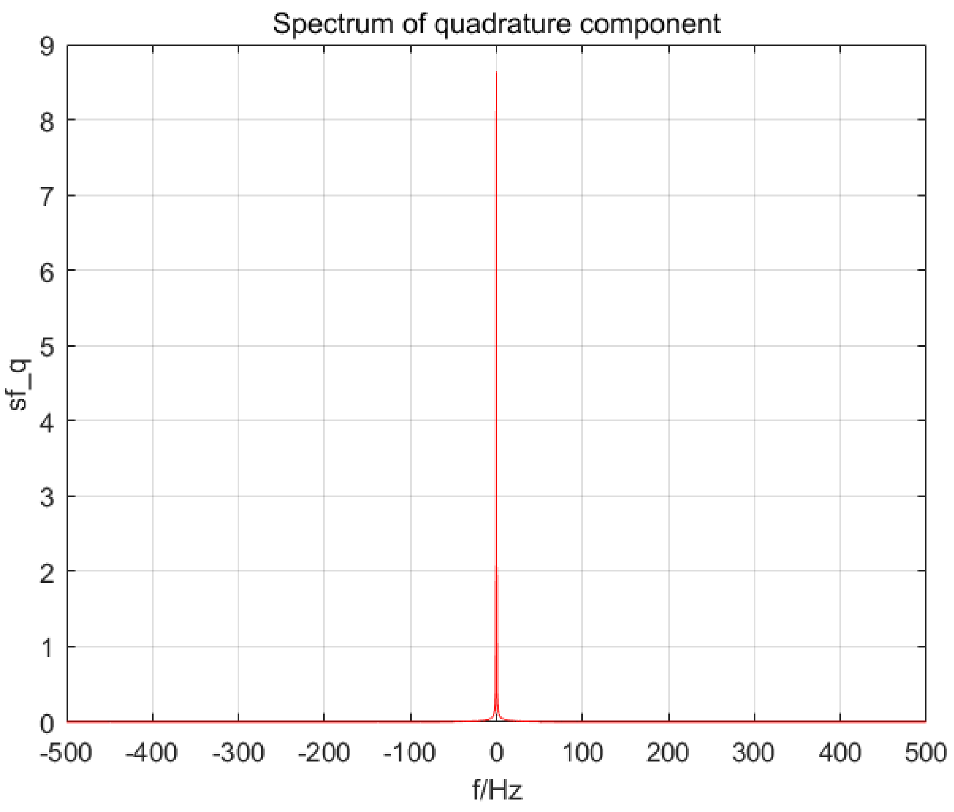


Figure9：Quadrature component of x(t)

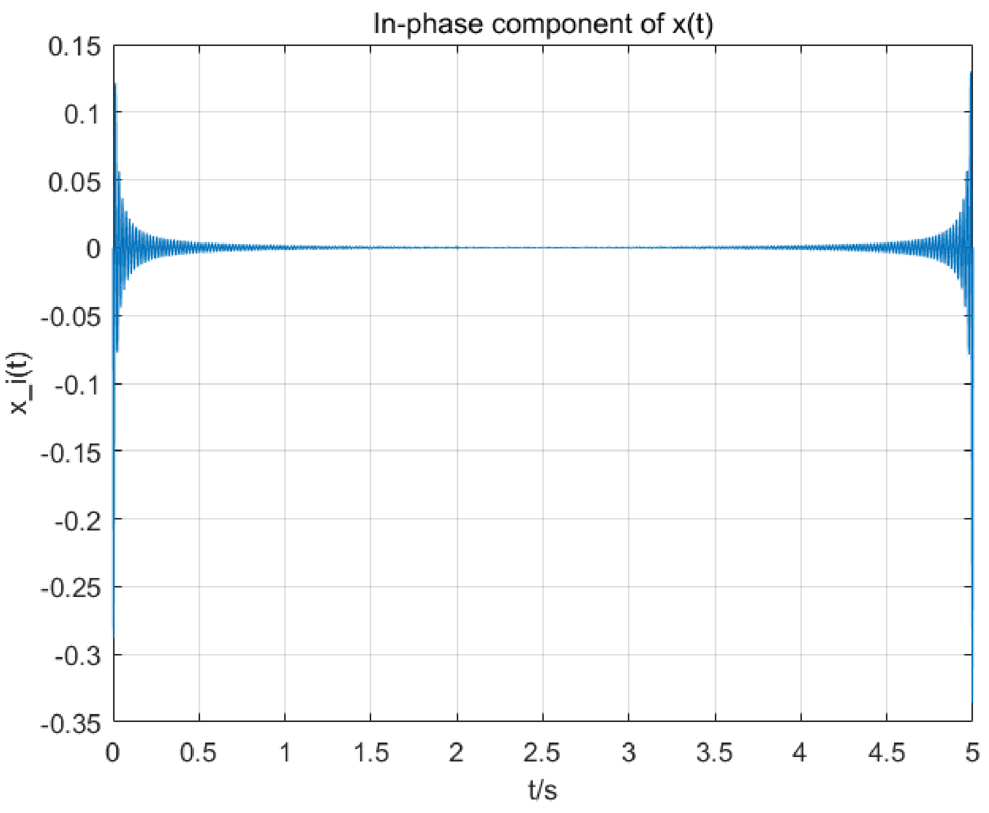


Figure10：In-phase component of x(t)

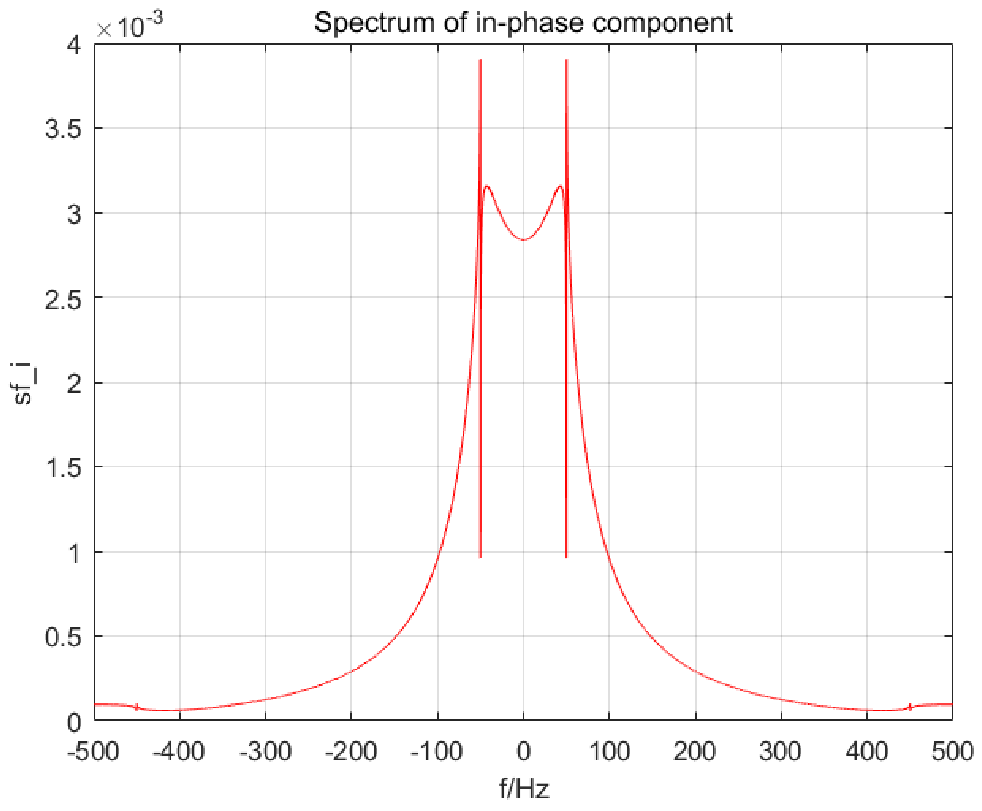


Figure11：Spectrum of in-phase component

C. Discussion

As we can see, there is almost no in-phase component. Every sinusoid can be expressed as the sum of a sine function (phase zero) and a cosine function (phase is pi/2). If the sine part is called the `in-phase` component, the cosine part can be called the `phase-quadrature` component. In general, `phase quadrature` means `90 degrees out of phase`. z = hilbert(x) ,Hilbert transform the real signal x(n), and get the analytic signal z(n). After this transform,$z(t)=s(t)+j\breve{s}(t)$, $s_L(t)=\left(s(t)+js(t)\right)e^{-j2\pi f_c t}$, $s_L(t)=s_r(t)+js_c(t)$, $s_r(t)$ is the in-phase component, $s_c(t)$ is the quadrature component.