LINEAR SYSTEMS
SECOND YEAR
LABORATORY SESSION 2 - ACADEMIC YEAR 2023/2024

1. Objectives

- Understand how continuous signals can be represented using Matlab.
- Representation of Fourier transforms of various continuous and discrete signals, paying special attention to the complex character of these transforms.
- Numerical evaluation of Fourier transforms of continuous and discrete signals.
- Understanding of the concept of frequency, its physical meaning, and the effects associated with elementary transformations of signals.

2. Representation of continuous lines in Matlab

Matlab uses *arrays* of numbers. In this way, the representation of a sequence such as

$$y[n] = \cos(\pi n/3)$$

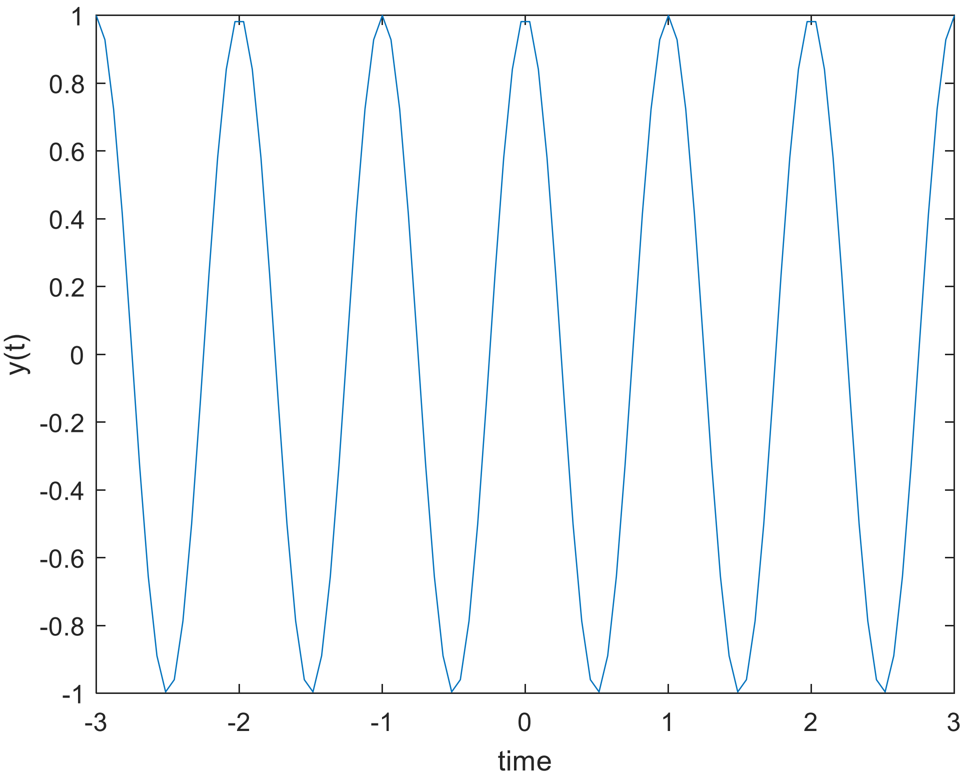
is immediate, while that of continuous signals, such as

$$y(t) = \cos(2\pi t)$$

it is not because of the argument *t* is continuous. To solve this problem, we represent the continuous signal by a sufficiently large number of samples of that signal. This procedure is called *sampling* and will be studied in greater detail later in the course. Consider the following example:

```
n_points = 100;
t = linspace(-3,3,n_points);
y = cos(2*pi*t);

figure(1)
plot(t,y)
xlabel('time'), ylabel('y(t)')
```



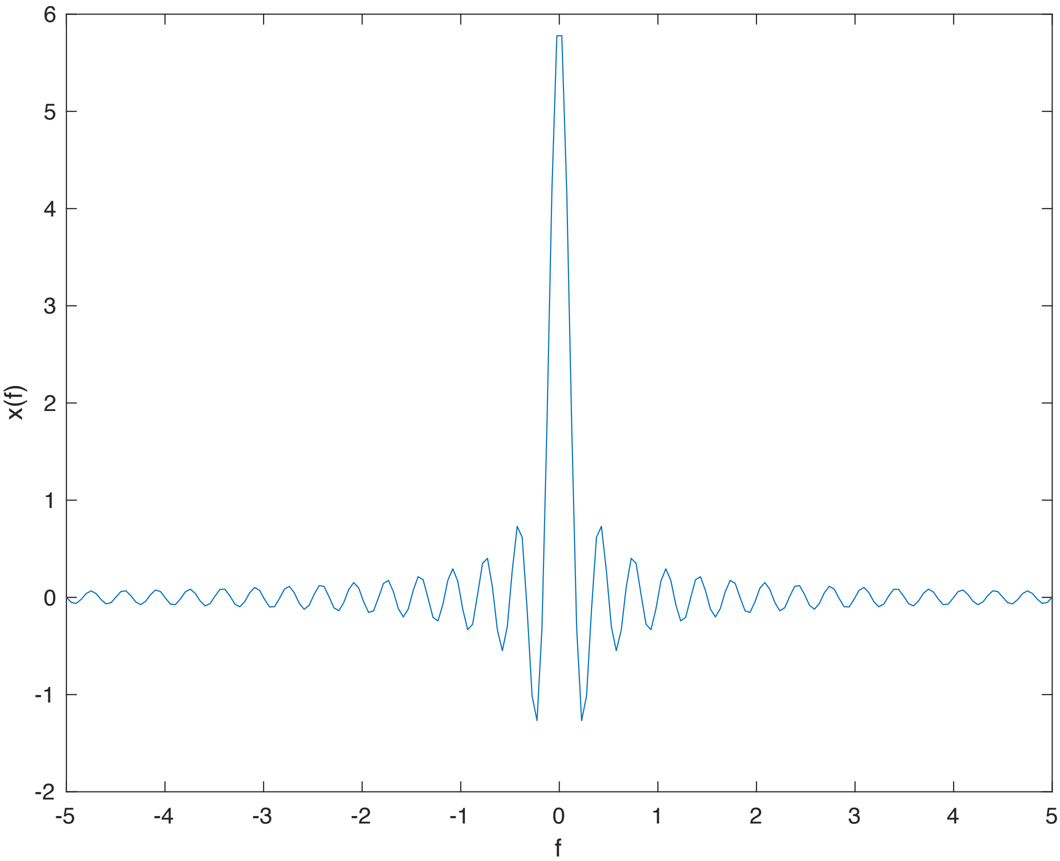
Observe how the array *y* is sufficiently detailed to represent the continuous signal in the interval [-3,3]. Nevertheless, the above plot still shows a sequence approximating a continuous signal and not a continuous signal, as it has been obtained by linearly connecting the pairs of points in the array *y*. You can see this by zooming in on the figure, or by varying the number of points of the array *y*.

Obviously, a similar approximation can be used in the case of Fourier transforms of continuous and discrete signals, $f \mapsto \hat{x}(f)$, since again the argument of the function is continuous. The following code represents the Fourier Transform of the rectangular pulse function:

$$x(t) = \begin{cases} 1, & \text{if } |t| \leq T_1 \\ 0, & \text{if } |t| > T_1 \end{cases} \quad \circ\!\!-\!\!\bullet \quad 2T_1 \mathrm{sinc}(2T_1 f)$$

```
n_points = 200;
T1 = 3;
f = linspace(-5,5,n_points);
x_hat = 2.*T1.*sinc(2.*T1.*f);

figure(2)
plot(f,x_hat);
xlabel('f'), ylabel('x(f)')
```



Exercise 1

Graphically represent the Fourier transform of the signal

$$x(t) = \begin{cases} e^{-at}, & \text{if } t \geq 0 \\ 0, & \text{if } t < 0 \end{cases}$$

where a is a positive real constant. Since in this case the Fourier Transform is complex, please represent the real and imaginary part separately. Study the effect that the parameter a has on the spectrum of the signal. To this end, represent in the same graph the real and imaginary part of the spectrum and play with the Matlab commands ***hold on*** and *hold off* to represent in the same plot the spectra for different values of a. We recommend to use a frequency space from -5 to 5 with sufficient points to represent the spectrum accurately.

3. Numerical calculation of Discrete Fourier Transforms

It is known that the Fourier Transform of the sequence $x[\cdot]$ can be obtained as

$$\hat{x}(f) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi fn}$$

When $x[\cdot]$ is a sequence of finite duration, the above sum has a finite number of nonzero terms, so it is easy to compute the Fourier Transform for any particular value of frequency f_k. Note that you are not required to obtain a closed-form expression of the Fourier Transform.

For example, consider the pulse signal $x_{\text{pulse}}[\cdot]$, defined as

$$x_{\text{pulse}}[n] = \begin{cases} 1, & \text{if } |n| \leq N_1 \\ 0, & \text{if } |n| > N_1 \end{cases}$$

with N_1=5. We can compute the Fourier Transform of this signal by executing the following Matlab instructions:

```
%Representation of the signal in the time domain
N1 = 5;
n = -10:10;
x_pulse = zeros(size(n));
x_pulse(find(abs(n)<=N1)) = 1;
subplot(2,1,1);
stem(n,x_pulse);
xlabel('n'); ylabel('x_{pulse}[n]');title('Time Domain')

%Representation of the signal in the frequency domain
n_points = 200;
f = linspace(-0.5,0.5,n_points);

for k = 1:length(f)
   X_pulse(k) = sum(x_pulse.*exp(-j*n*2*pi*f(k)));
end
subplot(2,1,2)
plot(f,real(X_pulse));
xlabel('f'); ylabel('x_{pulse}(f)');title('Frequency domain')
```

If you carefully examine the values of the variable *X_pulse* you will see that, due to numerical reasons, some of the entries in the array have an imaginary part that is not null (although it is very small). It is important to be aware of the symmetry of the signal to eliminate these effects.

In order to validate the Fourier Transform computed in the above procedure, plot in the same subplot the Fourier Transform of the discrete pulse obtained by analytical evaluation.

Exercise 2

Graphically represent the Fourier Transform $\hat{x}_{\text{pulse}}(\cdot)$ of the rectangular pulse sequence

$$x_{\text{pulse}}[n] = \begin{cases} 1, & \text{if } |n| \leq N_1 \\ 0, & \text{if } |n| > N_1 \end{cases}$$

where N_1 is an integer to be chosen by you. Due to the periodic nature of the Discrete Fourier Transform, it is sufficient to represent one period of that transform. In addition, given the real and even character of the sequence in the time domain, it is known that its Fourier Transform signal $\hat{x}_{\text{pulse}}(\cdot)$ is real-valued, so it is enough to only draw the real part.

Vary the value of N_1 in the range {1,...,9} and study how the parameter affects the spectrum of the signal.

Exercise 3

In this exercise, we consider the Fourier Transform of periodic sequences. To this end, analyze the behavior of the following function, which performs an extension of the sequence *x_pulse* by repeating it with period 21:

```
function [n_ext,x_ext] = repite_pulso(N1,n_times)

n = -10:10;
x = zeros(size(n));
x(find(abs(n)<=N1)) = 1;

x_ext = x;
for k = 1:n_times
   x_ext = [x x_ext x];
End

max_n = (length(x_ext)-1)/2;
n_ext = -max_n:max_n;
```

Draw the Fourier Transforms of the sequences produced by the above function as the number of repetitions increases. How does the spacing between the peaks of the Fourier Transforms depend on the number of repetitions? Can you explain this behavior?

4. Numerical calculation of the Fourier Transform of continuous signals

The integral in the analysis equation of the Fourier Transform can be approximated numerically when the signal is time limited. To this end, we resort to the following approximation:

$$\hat{x}(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi f t} \mathrm{d}t = \lim_{\Delta_t \to 0} \sum_{n=-\infty}^{\infty} x(n\Delta_t) e^{-j2\pi f n \Delta_t} \Delta_t$$

The Fourier Transform can thus be approximated by evaluating the above sum for a sufficiently small Δ_t.

As an example, we will consider again the rectangular pulse function

$$x(t) = \begin{cases} 1, & \text{if } |t| \leq T_1 \\ 0, & \text{if } |t| > T_1 \end{cases}$$

Using the following code, approximate the Fourier Transform of $x_{\text{pulse}}[\cdot]$ for different values of Δ_t (in the program *delta_t*). Compare this approximation with the exact Fourier Transform shown in Section 2.

```
n_points = 200;
T1 = 3;

% Explore different values for delta_t
delta_t = XXXX;
t = -5:delta_t:5;
x = zeros(size(t));
x(find(abs(t)<=T1)) = 1;

f = linspace(-5,5,n_points);
for k = 1:length(f)
   X_num(k) = sum(x.*exp(-j*2*pi*f(k)*t)*delta_t);
end

% X_num contains the numerical evaluation of the Fourier analysis integral
```