CAPITOLUL IV

Transformata Fourier și Laplace folosind Matlab/Octave/Python

A. Introducere în Transformata Fourier și Laplace

1. Transformata Fourier pentru semnale continue în timp

În Capitolul II s-a discutat despre tehnici de analiză a semnalelor periodice în timp folosind funcții de tip cosinus sau sinus pentru a exprima semnalele periodice ca o combinație liniară a acestor funcții. Trebuie să realizăm, însă, că majoritatea semnalelor cu care lucrăm sunt neperiodice. De aceea, pentru a analiza spectrul acestor semnale, domeniul frecvență, vom folosi Transformata Fourier.

Având un semnal $x(t)$ neperiodic ne întrebăm în ce domeniul de frecvență există acest semnal, sau, altfel spus, care este combinația de frecvențe necesare pentru a reconstrui semnalul folosind doar funcții standard de tip cosinus. Știind deja cum se reprezintă un semnal periodic, ca suma de funcții standard de tip cosinus, se va construi din semnalul neperiodic un semnal periodic prin repetarea semnalului neperiodic cu o perioada T. Folosind acest procedeu se va obține în final că transformata Fourier a unui semnal neperiodic $x(t)$ este:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\,dt$$

sau

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}\,dt$$

Transformata Fourier, este o funcție continuă de frecvență. Dacă știm expresia funcției $X(\omega)$, pentru a reconstrui semnalul inițial $x(t)$ vom folosi transformata Fourier inversă folosind formula

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{j\omega t}\,d\omega$$

Sau

$$x(t) = \int_{-\infty}^{\infty} X(f) e^{j2\pi f t}\, d\omega$$

Condițiile de existență a transformatei Fourier sunt

a. Semnalul $x(t)$ trebuie să fie integrabil în sens absolut
b. Dacă $x(t)$ este un semnal discontinuu, atunci trebuie să aibă un număr finit de discontinuități pe orice interval de timp
c. Semnalul $x(t)$ trebuie să aibă un număr finit de minime și maxime în orice interval de timp.

Pentru diferitele semnale uzuale, o lista a transformatelor Fourier[1] poate fi găsită aici http://www.thefouriertransform.com/pairs/fourier.php .

2. Transformata Fourier discretă

Având în vedere că transformata Fourier este rezultatul unei operații matematice de tip integrare care nu poate fi rezolvată analitic de un sistem de calcul, va trebui să folosim diferite metode numerice pentru a o calcula. Astfel, pentru a calcula transformata Fourier folosind Matlab/Octave/Python ne vom folosi de ceea ce se numește Transformata Fourier Discretă, care este o aproximare a transformatei Fourier.

Astfel formula pentru transformata Fourier se transformă în

$$X\left(\frac{kf_s}{N}\right) \cong T_s \sum_{n=0}^{N} x(nT_s) e^{-j2\pi kn/N}$$

Unde $T_s = \frac{1}{f_s}$ este perioada de eșantionare a semnalului $x(t)$, ales astfel încât în intervalul de timp T_s semnalul să nu aibă variații foarte mari, iar N reprezintă numărul de puncta de eșantionare, astfel încât durata semnalului să fie $N \cdot T_s$. k este indicele de sumare.

[1] https://allsignalprocessing.com/2019/10/22/fourier-methods-prominent/

În Matlab/Octave/Python, pentru a calcula transformata Fourier Discretă, se va folosi funcția *fft()* care este descrisă în Matlab aici: https://www.mathworks.com/help/matlab/ref/fft.html

Exemplu 4.1

Folosind Matlab/Octave/Python calculați și reprezentați modulul și faza transformatei Fourier Discrete pentru semnalul:

$$x(t) = \begin{cases} t(1-t), 0 < t < 1 \\ 0, \hat{i}n\ rest \end{cases}$$

```
%% MATLAB/Octave
%% Calculul transformatei Fourier folosind 32 de esantioane in intervalul
%% 0-2 secunde

clear all ; clf

N = 32 ;   % nr de esantioane pt esantionarea semnalului pt cele 2 secunde
Ts = 2/N ; % perioada de esantionare raportata la 2 secunde

fs = 1/Ts ; % frecventa de esantionare
df = fs/N ; % rezolutia în domeniul frecventa
n = [0:N-1]' ; % vectorul coloana indicilor esantionati de timp
t = Ts*n ; % vectorul valorilor timpului la momentele de esantionare
x = t.*(1-t).*(sign(t)-sign(t-1))/2; % vectorul valorilor functiei în
momentele de esantionare.
X = Ts*fft(x) ; % transformata fourieri discret
k = [0:N/2-1]' ; % valorile indicelui freceventelor
%X = fftshift(Ts*fft(x)) ; %folosit pentru a afisa tot spectrul semnalului
%k = [-N/2:N/2-1]'; % folosit impreuna cu fftshift

%Graficele
subplot(3,1,1) ;
p = plot(t,x,"k") ; set(p,"LineWidth",2) ; grid on ;
xlabel("Timpul, t (s)") ; ylabel("x(t)") ;
subplot(3,1,2) ;
p = plot(k*df,abs(X(1:N/2)),"k") ; set(p, "LineWidth",2) ; grid on;
%p = plot(k*df,abs(X(1:N/2)),"k") ; set(p, "LineWidth",2) ; grid on; %
folosit impreuna cu fftshift
xlabel("Freceventa, f (Hz)") ; ylabel("|X(f)|") ;
subplot(3,1,3) ;
p = plot(k*df,angle(X(1:N/2)),"k") ; set(p,"LineWidth",2) ; grid on ;
%p = plot(k*df,angle(X(1:N/2)),"k") ; set(p,"LineWidth",2) ; grid on ; %
folosit impreuna cu fftshift
xlabel( "Frecventa, f (Hz)") ; ylabel("Faza X(f)") ;
```

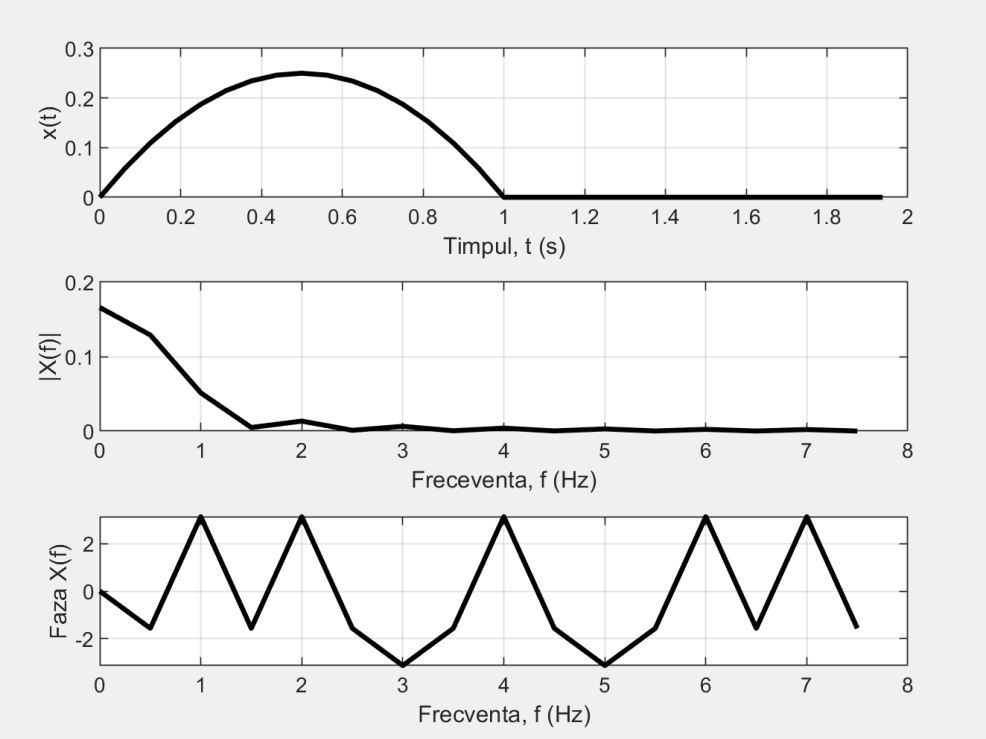


Fig.4.1. Graficele Exemplului 4.1 folosind fft()

Pentru a putea vizualiza tot spectrul semnalului se va folosi în locul funcției fft(), funcția fftshift(). În figura 4.2 se pot observa graficele rezultate obținute prin comentarea liniilor 14,15,24,28 și de-comentarea liniilor 16,17,25,29.

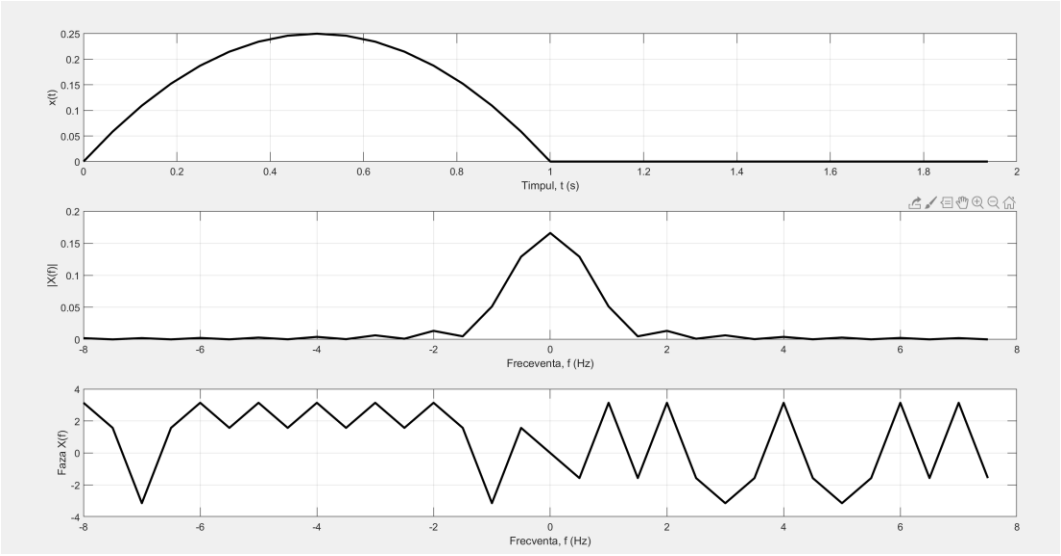


Fig.4.2. Graficele Exemplului 4.1 folosind fft()

Varianta Python

```
import numpy as np
import matplotlib.pyplot as plt
from mpldatacursor import datacursor

#importuri necesare  pentru ca graficele sa se deschida
#in fereastra separata si sa putem folosi zoom si datacursor
from IPython import get_ipython
get_ipython().run_line_magic('matplotlib', 'qt')


N = 32 ; # nr de esantionae
Ts = 2/N ; # perioada de esantionare

fs = 1/Ts ; # frecventa de esantionare
df = fs/N ; # rezolu?ia în domeniul frecventa
n = np.arange(0,N-1); # vectorul indicilor esantionati de timp
t = Ts*n ; # vectorul valorilor timpului la momentele de esantionare
x = t*(1-t)*(np.sign(t)-np.sign(t-1))/2; # vectorul valorilor func?iei în
momentele de esantionare
X = Ts*np.fft.fft(x) ; # transformata fourieri discret
k = np.arange(0,N/2) ; # valorile indicelui freceventelor
#X = fftshift(Ts*fft(x)) ; #folosit pentru a afisa tot spectrul semnalului
#k = [-N/2:N/2-1]'; # folosit impreuna cu fftshift


f,(ax1,ax2,ax3)=plt.subplots(3,1);
ax1.plot(t,x) ;
ax1.set_xlabel("Timpul, t (s)") ;
ax1.set_ylabel("x(t)") ;
ax1.set_xlim(0, 2)

ax2.plot(k*df,np.abs(X[0:int(N/2)])) ;
#p = plot(k*df,abs(X(1:N/2)),"k") ; set(p, "LineWidth",2) ; grid on; %
folosit impreuna cu fftshift
ax2.set_xlabel("Freceventa, f (Hz)") ;
ax2.set_ylabel("|X(f)|") ;
ax2.set_xlim(0, 8)

ax3.plot(k*df,np.angle(X[0:int(N/2)]));
#p = plot(k*df,angle(X(1:N/2)),"k") ; set(p,"LineWidth",2) ; grid on ; %
folosit impreuna cu fftshift
ax3.set_xlabel( "Frecventa, f (Hz)") ;
ax3.set_ylabel("Faza X(f)") ;
ax2.set_xlim(0, 8)
```

Tema 4.1 Scrieți un program Matlab/Octave/Python care să afișeze modulul și faza transformatei Fourier pentru semnalul :

$$x(t) = \begin{cases} 9 - (t-3)^2, 0 < t < 3 \\ 0, \hat{i}n\ rest \end{cases}$$

3. Transformata Fourier în Timp Discret

Pentru a calcula transformata Fourier în timp discret se apelează la o metoda similar cu transformata Fourier pentru semnale continue. Astfel plecând de la seria Fourier în timp discret se obține următoarea formă a transformatei Fourier în timp discret

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$$

iar pentru refacerea semnalului în timp discret, cunoscându-se expresia transformatei Fourier în timp discret folosim următoarea formulă

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega)e^{j\Omega n}\, d\Omega$$

Pentru ca transformata Fourier în timp discret să existe trebuie ca semnalul $x[n]$ să fie absolut sumabil.

Exemplul 4.2

Folosind Matlab/Octave/Python calculați și reprezentați modulul și faza Transformata Fourier în timp discret pentru semnalul

$$x[n] = \begin{cases} 1, n = \overline{0,9} \\ 0, rest \end{cases}$$

```
%% MATLAB/Octave
%% Calculul transformatei Fourier in timp discret
clear all;

xn = ones(1,10);          % semnalul x[n]
puteri=linspace(0,9,10); %puterile exponentialei din formula transformatei
Fourier in timp discret (n)
Omega = [-0.1:0.01:1.1]*2*pi; %vectorul  'Omega' necesar calculului
transformatei Fourier în timp discret
TFTD= sum(exp(-1i*puteri'*Omega),1); %transformata fourier în timp discret
%TFTD = sin(5*Omega)./sin(0.5*Omega).*exp(-1i*4.5*Omega); %transformata
%fourier în timp discret metoda alternativa de calcul folosind
%proprietatile seriilor in progresie geometrica

% Grafic
figure(1)
subplot(211);
plot(Omega,abs(TFTD)); grid;
axis([-0.2*pi,2.2*pi,-1,11]);
xlabel('\Omega (rad)');
ylabel('Amplitudine')

subplot(212)
plot(Omega,angle(TFTD)); grid;
axis([-0.2*pi,2.2*pi,-pi,pi]);
xlabel('\Omega (rad)');
ylabel('Faza (rad)');
```

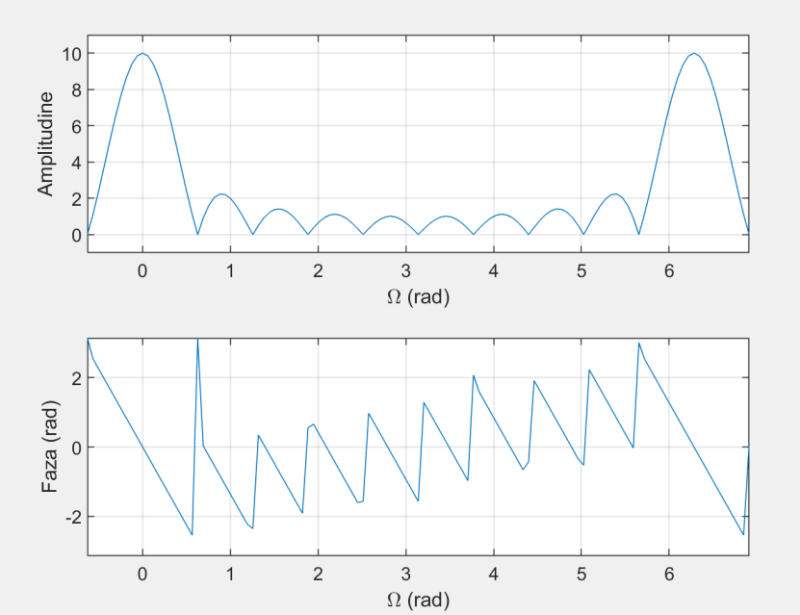


Fig.4.3. Graficele Exemplului 4.2

Varianta python

```
# Python
# Calculul transformatei Fourier in timp discret
import numpy as np
import matplotlib.pyplot as plt
from mpldatacursor import datacursor

#importuri necesare  pentru ca graficele sa se deschida
#in fereastra separata si sa putem folosi zoom si datacursor
from IPython import get_ipython
get_ipython().run_line_magic('matplotlib', 'qt')

xn = np.ones(10);         # semnalul x[n]
puteri=np.linspace(0,9,10); #puterile exponentialei din formula transformatei
fourier in timp discret
puteri.shape=(puteri.size,1); #transpusa unui vector
Omega = np.arange(-0.1,1.1,0.01)*2*np.pi;# vectorul  'Omega' nnecesar
calcului transformatei fourier în timp discret
TFTD= np.sum(np.exp(1j*puteri*Omega),0); #transformata fourier în timp
discret
#TFTD = sin(5*Omega)./sin(0.5*Omega).*exp(-1i*4.5*Omega); %transformata
#fourier în timp discret metoda alternativa de calcul folosind
#proprietatile seriilor in progresie geometrica

# Grafic

f,(ax1,ax2)=plt.subplots(2,1);

ax1.plot(Omega,np.abs(TFTD));
ax1.set_xlim(-0.2*np.pi,2.2*np.pi);
ax1.set_ylim(-1,11);
ax1.set_xlabel("Omega (rad)");
ax1.set_ylabel("Amplitudine");

ax2.plot(Omega,np.angle(TFTD));
ax2.set_xlim(-0.2*np.pi,2.2*np.pi);
ax2.set_ylim(-np.pi,np.pi);
ax2.set_xlabel("Omega (rad)");
ax2.set_xlabel("Faza (rad)");
```

Tema 4.2 Scrieți un program Matlab/Octave/Python care să afișeze modulul și faza transformatei Fourier în timp discret pentru semnalul :

$$x[n] = \begin{cases} ln(n+1), 0 \leq n < 10 \\ -ln(-n+1), -10 < n < 0 \\ 0, in\ rest \end{cases}$$

4. Transforma Laplace a semnalelor continue

Anterior, atât în capitolul (Laboratorul) 2 cât și în acest capitol (Laborator) am folosit seria și transformata Fourier pentru a determina caracteristicile, în domeniul frecvență, pentru anumite tipuri de semnale continue în timp. Pentru ca un semnal să aibă serie și transformată Fourier trebuie să fie absolut integrabil. Astfel, o funcție de tipul $t \cdot \sigma(t)$ nu este absolut integrabil și, deci, nu se poate folosi transformata Fourier dar se poate aplica transformata Laplace. Astfel transformata Laplace poate fi considerată ca o generalizare a transformatei Fourier.

Transformata Laplace (numită și transformată Laplace bilaterală) pentru un semnal $x(t)$ este

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st}\,dt$$

,unde s este un număr complex.

Transformată Laplace unilaterală este definită

$$X(s) = \int_{0}^{\infty} x(t)e^{-st}\,dt$$

Astfel, având un semnal $x(t)$, mulțimea tuturor numerelor complexe s pentru care integrala există se numește Regiune de Convergență. De exemplu pentru funcție treaptă, $\sigma(t)$, regiunea de convergentă este dată de $Real(s) > 0$.

Ecuația folosită pentru a reconstrui semnalul $x(t)$, știind transformata acestuia Laplace, $X(s)$, este:

$$x(t) = \frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} X(s)e^{st}\,ds$$

Integrala este evaluate pentru $s = c + j\omega$ în planul complex între limitele $c + j\infty$ *și* $c - j\infty$, unde c este un număr real pentru care s aparține regiunii de convergență a lui $X(s)$. Din punct de vedere practic, integrala este destul de dificil de rezolvat astfel că se folosesc metode algebrice, cum ar fi descompunerea în fracții

simple, folosirea de perechi cunoscute $x(t) \underset{L}{\leftrightarrow} X(s)$ sau determinarea reziduurilor și polilor lui $X(s)$. O serie de perechi Laplace se pot găsi aici http://ece-research.unm.edu/bsanthan/ece541/table_ME.pdf

Ecuația pentru reziduri este data de ecuația, pentru un semnal $x(t)$,

$cu\ x(t) = 0, t < 0$:

$$x(t) = \sum_{\substack{toți\ polii\ lui\ X(s) \\ din\ semiplanul\ stâng}} Rez[X(s)e^{-st}]$$

Exemplul 4.3

Folosind scrierea simbolică, să se determine transformata Laplace pentru semnalul

$$x(t) = \begin{cases} e^{-at}, t \geq 0 \\ 0, în\ rest \end{cases}$$

```
%% MATLAB/Octave
%% Calculul transformatei Fourier in timp discret
%% pkg load symbolic %pentru octave decomentati aceasta linie
syms x a t;
x = exp(-a*t);
X = laplace(x) %funcție care enerează transformata laplace pentru scrierea
simbolica
```

Rezultatul obținut :

```
X = (sym)

    1
  -----
  a + s
```

Nota: Pentru Octave vedeți capitolul (laboratorul) 1, despre cum să se instaleze scrierea simbolică.

Varianta python:

```
# Python
# Calculul transformatei Laplace simbolica

import sympy as syms;

t = syms.Symbol('t'); # definirea variabilelor ca simboluri
x = syms.Symbol('x'); # definirea variabilelor ca simboluri
a = syms.Symbol('a'); # definirea variabilelor ca simboluri
X = syms.Symbol('X'); # definirea variabilelor ca simboluri
s = syms.Symbol('s'); # definirea variabilelor ca simboluri

x=syms.exp(-a*t);

X=syms.laplace_transform(x,t,s);
```

Teama 4.3

Folosind scrierea simbolică, să se determine transformata Laplace pentru semnalul

$$x(t) = \begin{cases} \cos(\omega t), t \geq 0 \\ 0, \hat{\imath}n\ rest \end{cases}$$

Exemplul 4.4

Folosind scrierea simbolică, să se determine transformata Laplace inversă pentru semnalul

$$X(s) = \frac{s+2}{s^3 + 4s^2 + 3s}$$

```
%% MATLAB/Octave
%% Calculul transformatei Laplace inverse
%% pkg load symbolic %pentru octave decomentati aceasta linie

syms X s x ; %simbolurile
X = (s+2)/(s^3+4*s^2+3*s); %transformata laplace
x = ilaplace(X) %transformata inversa
```

Rezultatul obținut :

```
x = (sym)

       -t    -3*t
  2    e     e
  - - --- - -----
  3    2       6
```

Nota: Pentru Octave vedeți capitolul (laboratorul) 1, despre cum să se instaleze scrierea simbolică.

Varianta python:

```
# Python
# Calculul transformatei Laplace simbolica

import sympy as syms;

t = syms.Symbol('t'); # definirea variabilelor ca simboluri
x = syms.Symbol('x'); # definirea variabilelor ca simboluri
X = syms.Symbol('X'); # definirea variabilelor ca simboluri
s = syms.Symbol('s'); # definirea variabilelor ca simboluri


X=(s+2)/(s**3+4*s**2+3*s); #expresia Laplace

x=syms.inverse_laplace_transform(X,s,t) #transformata laplace inversa
```

Exemplu 4.4

Determinați parametrii semnalului x(t) știind că transformata laplace este

$$X(s) = \frac{s+2}{s^3 + 4s^2 + 3s}$$

```
%% Calculul transformatei Laplace simbolica
numaratorul = [1 2]; %coeficientii numaratorului
numitorul = [1 4 3 0]; %coeficientii numitorului
[r,p] = residue(numaratorul,numitorul); %calculul reziduurilor (r) si
polilor(p)
```

Rezultatul este

```
>> r
r =

   0.66667
  -0.50000
  -0.16667

>> p
p =

   0
  -1
  -3
```

Folosind rezultatele obținute pentru r si p rezultă semnalul invers $x(t)$

$$x(t) = 0.66667 - 0{,}5 \cdot e^{-t} - 0{,}16667 \cdot e^{-3t}, t \geq 0$$

Varianta python

```
# Python
# Calculul transformatei Laplace simbolica
import numpy as np
from scipy import signal


numaratorul=np.array([1,2]); # coefienții numaratorului
numitorul =np.array([1,4,3,0]); #coeficientii numitorului
r,p,q =signal.residue(numaratorul,numitorul); # reziduurile (r), polii(p).
parametrul q nu este de interes
```

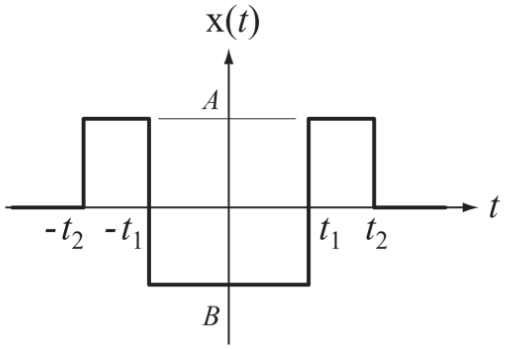
Tema 4.4

Determinați parametrii semnalului $x(t)$ (simbolic și numeric) știind că transformata Laplace este :

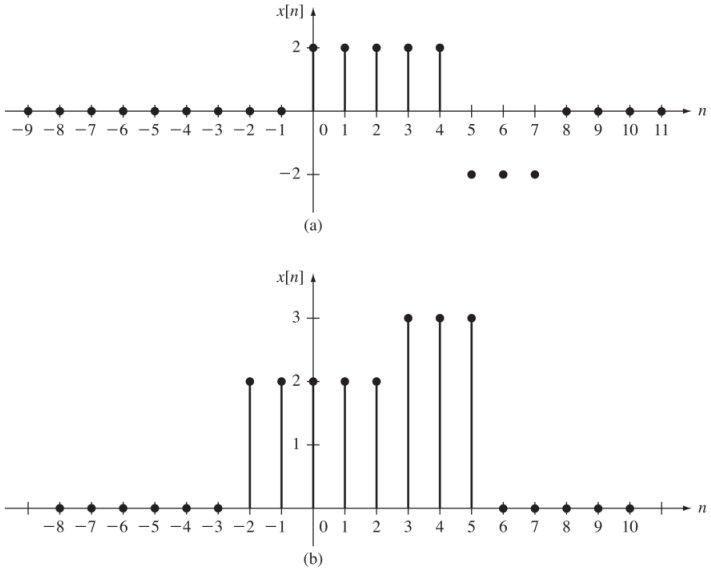
$$X(s) = \frac{5s - 1}{s^3 - 3s - 2}$$

Exerciții suplimentare

- Transformata Fourier pentru semnale continue neperiodice
  A. Calculați și reprezentați amplitudinea și faza transformatei Fourier pentru semnalele
     1. $x(t) = 6e^{-4|t|}$
     2. $x(t) = e^{-\pi t^2} sin(20\pi t)$
  B. Calculați și reprezentați amplitudinea și faza transformatei Fourier pentru semnalul din Figura de mai jos unde A=-B=1, t1=1 și t2=2



- Transformata Fourier în timp discret
  C. Calculați și reprezentați modulul și faza transformatei fourier în timp discret pentru semnalele din figurile de mai jos



D. Calculați și reprezentați modulul și faza transformatei fourier în timp discret pentru semnalele

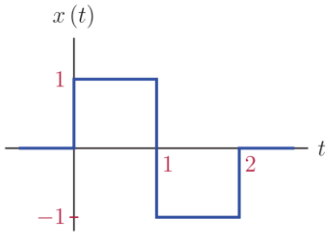
1. $x[n] = \begin{cases} 0{,}8^n, n \geq 0 \\ 0, \hat{i}n\ rest \end{cases}$
2. $x[n] = \begin{cases} n \cdot 0{,}5^n, n \geq 0 \\ 0, \hat{i}n\ rest \end{cases}$

- Transformata Laplace

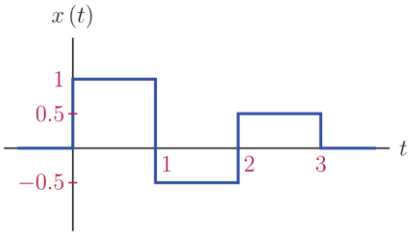
E. Determinați transformata Laplace pentru următoarele semnale :

1. $x(t) = \begin{cases} e^{-2t}, t \geq 0 \\ 0, \hat{i}n\ rest \end{cases}$
2. $x(t) = \begin{cases} 1, 0 < t < 1 \\ -1, 1 < t < 2 \\ 0, \hat{i}n\ rest \end{cases}$

F. Determinați transformata laplace pentru următoarele semnale :



(a)



(b)

G. Determinați semnalul care are transformata laplace data de următoarele expresii . Considerați semnalul ca fiind cauzal.

1. $X(s) = \frac{1-e^{-s}}{s+1}$
2. $X(s) = \frac{s+1}{(s+2)(s+3)}$