Semnale continue și discrete

Ene Cristian Ștefan

1. Noțiuni teoretice

1.1 Semnale periodice și continue

Semnalele periodice se repetă la un interval fix de timp, acest interval fiind denumit drept perioadă.

Un semnal periodic se poate reprezenta prin următoarea formă matematică:

$$s(t) = Asin(\omega t + \varphi) \quad (1)$$

Unde:

- A = amplitudinea semnalului;
- ω = pulsația (frecvența unghiulară);
- φ = defazarea semnalului.

Frecvența unui semnal se poate scrie în două forme:

$$\omega\left[\frac{radiani}{s}\right] \quad \text{sau} \quad f\left[\frac{cicluri}{s} = Hz\right] \quad (2)$$

În funcție de aplicație se alege una din formele prezentate mai sus. În calcule este utilizată frecvența unghiulară, pentru că programele de calcul și funcțiile trigonometrice folosesc în mod standard unghiurile măsurate în radiani. Pe de altă parte, în aplicații practice și în laboratoare, se folosește cea de a doua formă, deoarece toate aparatele de măsură folosesc aceeași unitate de măsură.

Legătura dintre cele două forme este următoarea:

$$\omega = 2\pi f \quad (3)$$

Frecvența unghiulară se poate exprima ca fiind schimbarea valorii unui unghi într-un interval de timp:

$$\omega = \frac{d\theta}{dr} = \omega t \quad (4)$$

Orice semnal periodic s(t) se poate reprezenta ca fiind o sumă de semnale sinusoidale și cosinusoidale. Această reprezentare se numește *serie Fourier trigonometrică*.

$$s(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty}(a_n \cos(n\Omega t) + b_n \sin(n\Omega t)) \tag{5}$$

$$\Omega = 2\pi f_0 = \frac{2\pi}{T_0} \tag{6}$$

Unde:

- Ω = frevența fundamentală a semnalului;
- f_0 = frecvența fundamentală exprimată în hertzi;
- T_0 = perioada fundamentală a semnalului.

Totodată, semnalele periodice mai pot fi scrise și cu ajutorul *seriei Fourier armonice*:

$$s(t) = a_0 + \sum_{n=1}^{\infty} A_n \cos(n\Omega t + \varphi_n) \tag{7}$$

Unde:

- A_n = amplitutinea
- φ_n = defazarea componentei n

Amplitudinea și defazarea se pot calcula conform relațiilor de mai jos:

$$A_n = \sqrt{a_n^2 + b_n^2} \tag{8}$$

$$\varphi_n = -arctg\left(\frac{b_n}{a_n}\right) \tag{9}$$

1.2 Semnalele discrete

Aceste semnale se pot reprezenta prin secvențe ordonate de numere. Față de semnalele continue, semnalele discrete iau valori doar la anumite momente de timp.

Notarea semnalelor discrete:

$$\{x(nT) = x(n) = x(nT) = x[n]\} \tag{10}$$

1.3 Cuantificarea și eșantionarea semnalelor

Eșantionarea reprezintă evaloarea valorii unui semnal continuu la anumite intervale de timp. Pasul de eșantionare poartă numele de perioadă de eșantionare (T). Frecvența de eșantionare reprezintp inversul perioadei de eșantionare.

Pentru ca semnalul eșantionat să fie refăcut corect, frecvența de eșantionare trebuie să respecte teorema *Nyquist-Shannon*:

$$f_e \geq 2f_s \tag{11}$$

Unde:

- f_e = frecvența de eșantionare;
- f_s = frecvența semnalului.

Semnalul analogic poate să ia o infinitate de valori, pe când cel digital doar un număr finit, astfel semnalul analogic trebuie adaptat atunci când este digitalizat, această operașie poartă numele de cuantificare.

Variația minimă a semnalului ce poate să fie înregistrată de semnalul digital se numește rezoluție sau pas de cuantificare și are următoarea formulă:

$$Q_s = \frac{V_{MAX} - V_{min}}{2^N - 1} \tag{12}$$

Unde:

- V_{MAX} = valoarea maximă a semnalului;
- V_{min} = valoarea minimă a semnalului;
- N = numărul de biți pe care se face cuantificarea..

1.4 Transformata Fourier Discretă și Rapidă

Transformata Fourier discretă (TFD) este utilizată în analiza semnalelor discrete. Această transformată este esențială în prelucrarea digitală a semnalelor, iar formula este cea de mai jos:

$$TFD\{x(n)\} = X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi kn}{N}} = \sum_{n=0}^{N-1} x(n) W_N^{-nk} \tag{13}$$

Transformata Fourier Rapidă (TFR) reprezintă orice algoritm de reducere a numărului de calcule necesar aflării TFD.

2. Rezolvarea cerințelor

1. Definiți și afișați un semnal dreptunghiular de amplitudine 5 și frecvență 2 Hz.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

#Declarare semnale
t=np.linspace(0,1,1000) #definire timp
f1=2
p=5*signal.square(2*np.pi*f1*t,0.5) #semnal dreptunghiular

#Afisare
plt.subplot(4,1,1)
plt.plot(t,p,linewidth=2)
plt.title("Semnal dreptunghiular")
plt.xlabel("timp [s]")
plt.ylabel("p(t)")
plt.show()
```

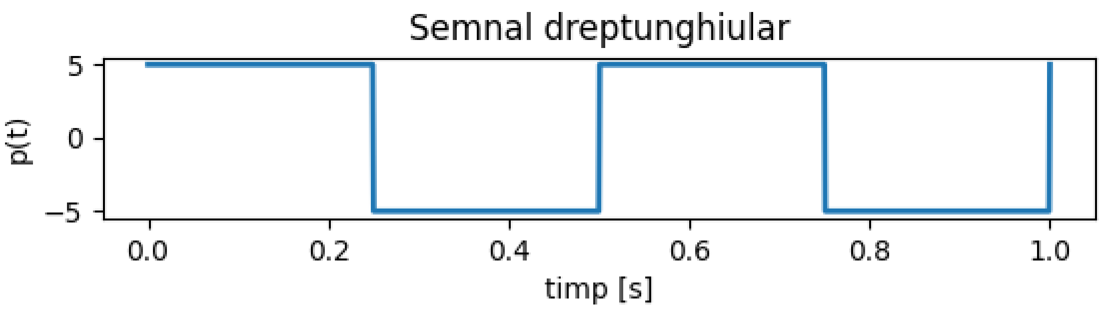

Figura 1. Semnalul dreptunghiular

2. Definiți și afișați un semnal triunghiular de amplitudine 1,5 și frecvență 4 Hz.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

#Declarare semnale
t=np.linspace(0,1,1000) #definire timp
f2=4
v=1.5*signal.sawtooth(2*np.pi*f2*t,0.5) #semnal triunghiular

#Afisare
plt.subplot(4,1,1)
plt.plot(t,v,linewidth=2)
plt.title("Semnal triunghiular")
```

```
plt.xlabel("timp [s]")
plt.ylabel("v(t)")
plt.show()
```

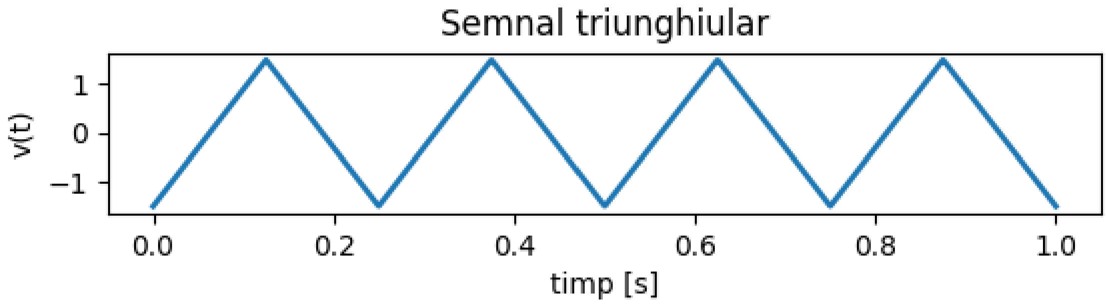

Figura 2. Semnalul triunghiular

3. Definiți și afișați un semnal dinte de fierăstrău de o amplitduine 3 și frecvență 3 Hz.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

#Declarare semnale
t=np.linspace(0,1,1000) #definire timp
f3=3
r=3*signal.sawtooth(2*np.pi*f3*t) #semnal dinte de fierastrau

#Afisare
plt.subplot(4,1,1)
plt.plot(t,r,linewidth=2)
plt.title("Semnal dinte de fierastrau")
plt.xlabel("timp [s]")
plt.ylabel("r(t)")
plt.show()
```

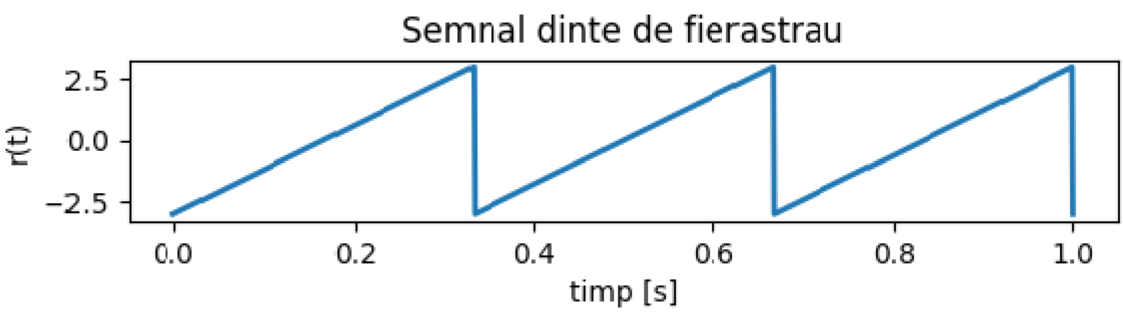

Figura 3. Semnalul dinte de fierăstrău

4. Afișați cele trei semnale definite anterior într-o singură figură (unul sub celălalt).

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

#Declarare semnale
t=np.linspace(0,1,1000) #definire timp
f1=2
f2=4
f3=3
p=5*signal.square(2*np.pi*f1*t,0.5)
v=1.5*signal.sawtooth(2*np.pi*f2*t,0.5)
s=3*signal.sawtooth(2*np.pi*f3*t)

#Afisare
plt.subplot(4,1,1)
plt.plot(t,p,linewidth=2)
plt.title("Semnal dreptunghiular")
plt.xlabel("timp [s]")
plt.ylabel("p(t)")
plt.subplot(4,1,2)
plt.plot(t,v,linewidth=2)
plt.title("Semnal triunghiular")
plt.xlabel("timp [s]")
plt.ylabel("v(t)")
plt.tight_layout()
plt.subplot(4,1,3)
plt.plot(t,s,linewidth=2)
plt.title("Semnal dinte de fierastrau")
plt.xlabel("timp [s]")
plt.ylabel("s(t)")
plt.tight_layout()
plt.show()
```

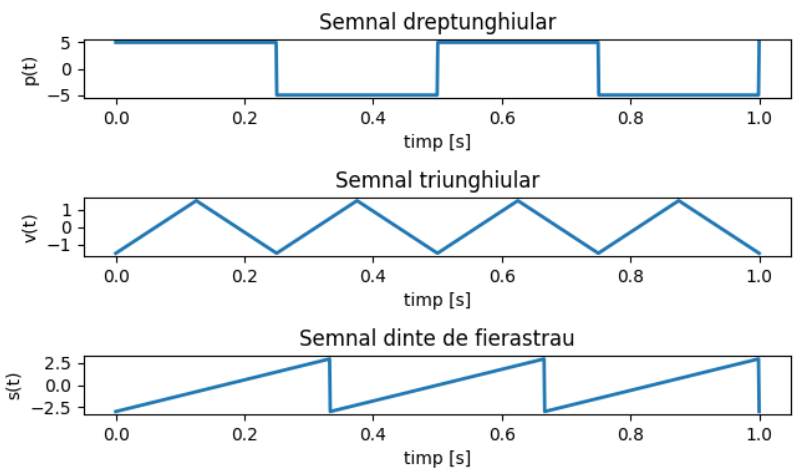

Figura 4. Afișarea semnalelor

5. Reprezentați spectrele de amplitduine și fază pentru semnalul dreptunghiular.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
dt=0.001
t=np.arange(0+dt,2+dt,dt)
s=5*signal.square(2*np.pi*2*t,0.5) #semnal dreptunghiular cerinta 5

#Reprezentarea in timp a semnalului
plt.subplot(1,2,1)
plt.plot(t,s,linewidth=2)
plt.title("Semnalul periodic")
plt.xlabel("timp [s]")
plt.ylabel("s(t)")

#Definire vectori Fourier
n=30 #Numarul de frecvente analizate
A=np.zeros((n))
B=np.zeros((n))
M=np.zeros((n))
P=np.zeros((n))

#Calcularea coeficientilor Fourier
for i in range(1,n):
 A[i]=sum(s*np.cos(2*np.pi*i*t))*dt
 B[i]=sum(s*np.sin(2*np.pi*i*t))*dt
 M[i]=np.sqrt(np.power(A[i],2)+np.power(B[i],2))
 P[i]=-np.arctan(B[i]/A[i])

#Reprezentarea spectrelor
plt.subplot(2,2,2)
plt.stem(M)
plt.title("Spectru amplitudini")
plt.xlabel("Frecventa [Hz]")
plt.ylabel("Amplitudine [V]")
plt.subplot(2,2,4)
plt.stem(P)
plt.title("Spectru faze")
plt.xlabel("Frecventa [Hz]")
plt.ylabel("Faza [rad]")
plt.tight_layout()
plt.show()
```

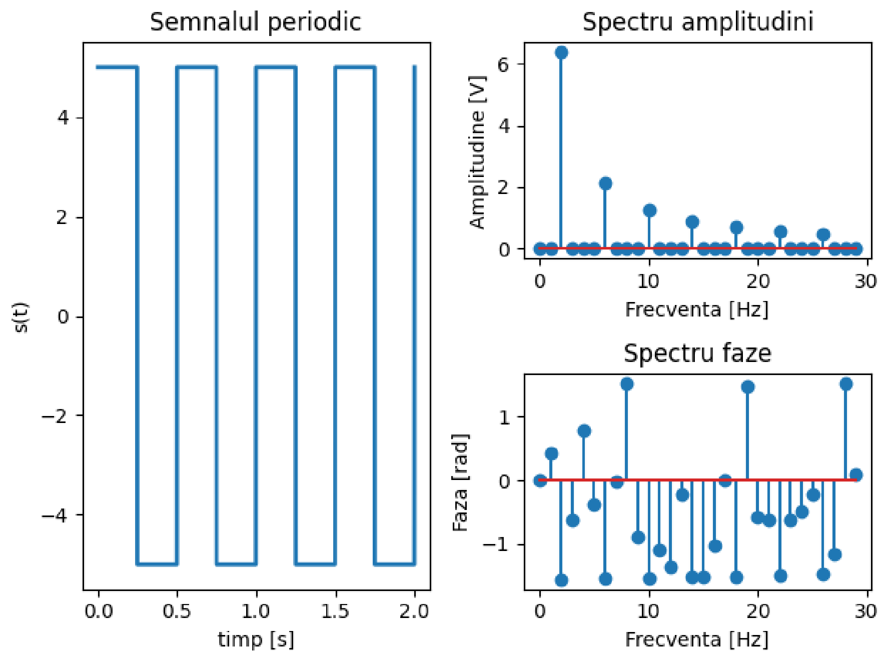

Figura 5. Afișarea spectrelor semnalului dreptunghiular

6. Reprezentați spectrele de amplitudine și fază pentru semnalul triunghiular.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
dt=0.001
t=np.arange(0+dt,2+dt,dt)
s=1.5*signal.sawtooth(2*np.pi*4*t,0.5)

#Reprezentarea in timp a semnalului
plt.subplot(1,2,1)
plt.plot(t,s,linewidth=2)
plt.title("Semnalul periodic")
plt.xlabel("timp [s]")
plt.ylabel("s(t)")

#Definire vectori Fourier
n=30 #Numarul de frecvente analizate
A=np.zeros((n))
B=np.zeros((n))
M=np.zeros((n))
P=np.zeros((n))

#Calcularea coeficientilor Fourier
for i in range(1,n):
 A[i]=sum(s*np.cos(2*np.pi*i*t))*dt
 B[i]=sum(s*np.sin(2*np.pi*i*t))*dt
 M[i]=np.sqrt(np.power(A[i],2)+np.power(B[i],2))
 P[i]=-np.arctan(B[i]/A[i])
```

```
#Reprezentarea spectrelor
plt.subplot(2,2,2)
plt.stem(M)
plt.title("Spectru amplitudini")
plt.xlabel("Frecventa [Hz]")
plt.ylabel("Amplitudine [V]")
plt.subplot(2,2,4)
plt.stem(P)
plt.title("Spectru faze")
plt.xlabel("Frecventa [Hz]")
plt.ylabel("Faza [rad]")
plt.tight_layout()
plt.show()
```

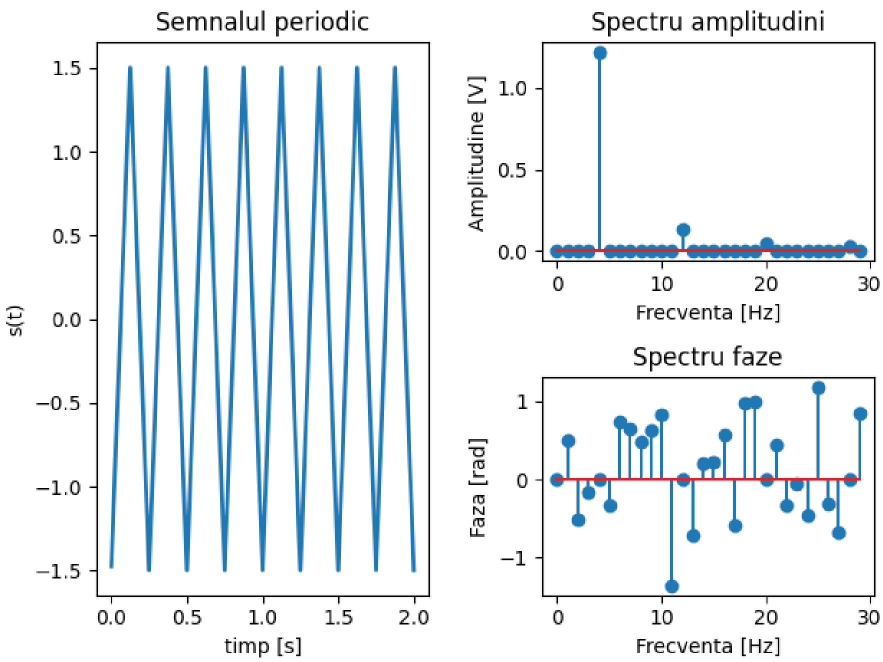

Figura 6. Afișarea spectrelor semnalului triunghiular

7. Eșantionați semnalul dreptunghiular și reprezentați grafic această operație. – Țineți cont de frecvența minimă de eșantionare necesară.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
t=np.linspace(0,2,2000)
s=5*signal.square(2*np.pi*2*t,0.5)

#Definire perioade de esantionare (ms)
T1=250

#Definire semnale esantionare
u1=np.zeros((len(s)))
```

```
for i in range(1,len(s)//T1):
 u1[i*T1]=1

e1=u1*s
plt.subplot(3,1,1)
plt.plot(t,s,'--')
plt.title("$T_e$=400ms")
plt.xlabel("timp [s]")
plt.ylabel("s(t)")
plt.xticks(np.arange(0, 2.1, 0.1))
plt.plot(t,e1)
plt.tight_layout()
plt.show()
```

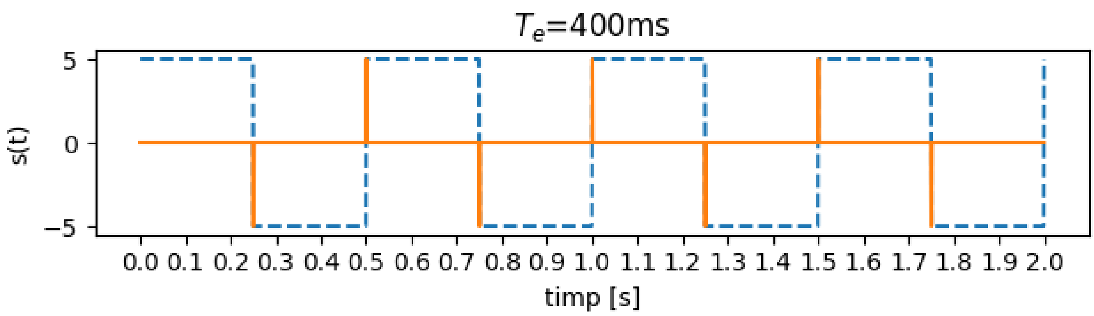

Figura 7. Eșantionarea semnalului dreptunghiular

8. Calculați și reprezentați grafic spectrul de amplitudini al semnalului triunghiular folosind transformata Fourier discretă.

```
from scipy.fft import fft
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal

Fe=1000
dx=1/Fe
t=np.arange(0,2+dx,dx)
s=1.5*signal.sawtooth(2*np.pi*4*t,0.5)

F=fft(s,1024)
M=abs(F)*2*dx
f=Fe*(np.arange(0,31,1))/1024
plt.plot(f,M[0:31],color='red',linewidth='3')
plt.title("Spectrul de Amplitudini calculat cu FFT")
plt.xlabel("Frecventa [Hz]")
plt.ylabel("Amplitudinea")
plt.show()
```

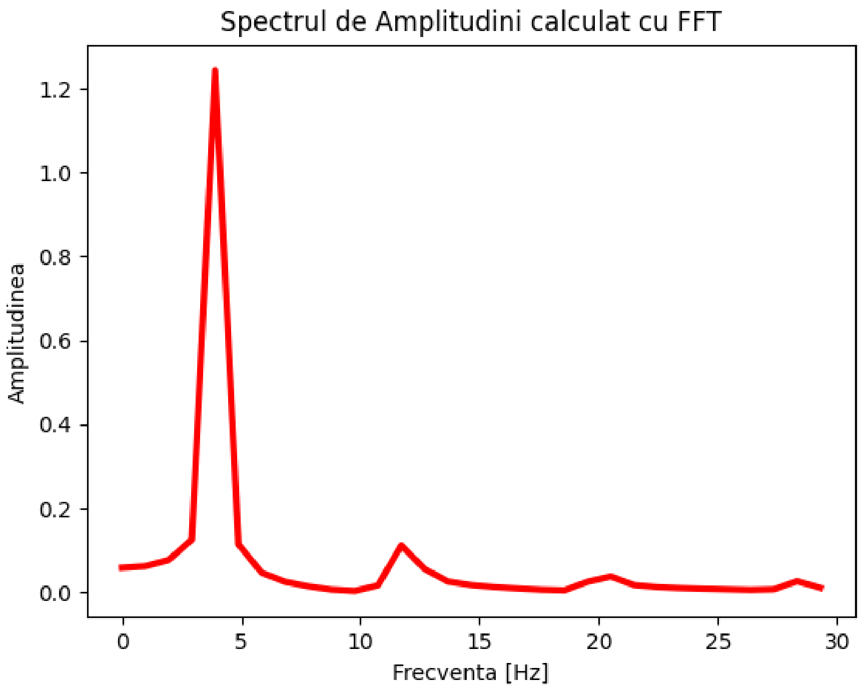

Figura 8. Spectrul de amplitudini al semnalului triunghiular