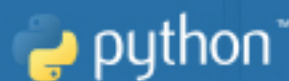


Marina Marjanović-Jakovljević
Vladimir Matić



Priručnik iz signala i sistema

(sa rešenim primerima u Pythonu)



UNIVERZITET SINGIDUNUM

Marina Marjanović-Jakovljević
Vladimir Matić

Priručnik iz
SIGNALA I SISTEMA
(sa rešenim primerima u Pythonu)

Prvo izdanje

Beograd, 2017.

Priručnik iz SIGNALA I SISTEMA (sa rešenim primerima u Pythonu)

Autor:

dr Marina Marjanović-Jakovljević
dr Vladimir Matić

Recenzenti:

dr Milan Milosavljević
dr Branko Kovačević

Izdavač:

UNIVERZITET SINGIDUNUM
Beograd, Danijelova 32
www.singidunum.ac.rs

Za izdavača:

dr Milovan Stanišić

Priprema za štampu:

Jelena Petrović

Dizajn korica:

Aleksandar Mihajlović

Godina izdanja:

2017.

Tiraž:

10 primeraka

Štampa:

Univerzitet Singidunum

ISBN 978-86-7912-664-1

Copyright:

© 2017. Univerzitet Singidunum

Izdavač zadržava sva prava.

Reprodukcija pojedinih delova ili celine ove publikacije nije dozvoljena.

Predgovor

Ovaj priručnik je nastao kao rezultat potrebe za odgovarajućim pisanim materijalom iz predmeta Signali i Sistemi koji autori drže na drugoj godini Fakulteta za Elektrotehniku i Računarstvo Univerziteta Singidunum u Beogradu. Pri pisanju je učinjen napor da priručnik bude prihvatljiv za čitaoce koji dobro vladaju znanjem iz matematike. Osnovni cilj autora bio da svi studenti koji slušaju predmet Signali i Sistemi mogu na razumljiv i lak način, uz razmatranje brojnih primera, da savladaju predviđeno gradivo, tj. steknu osnovno znanje iz teorije signala i sistema, kako u vremenskom, tako i u frekvencijskom domenu. Osim razumevanja osnovnih pojmova i uloge transformacija, student treba da se osposobi za korišćenje softvera za analizu signala i sistema.

Knjiga je podeljena u osam poglavlja: *Osobine kontinualnih signala, osobine diskretnih signala, kontinualni i diskretni sistemi, Furije-ovi redovi, Furijeova transformacija, Laplasova transformacija, Z transformacija, Filtriranje i **Signal zvuka***.

Prva glava knjige sadrži uvod u osnovne kontinualne signale, njihove osobine i operacije nad njima. Razmatrana je osobina periodičnosti, vremenskog skaliranja, inverzije, vremenskog pomeranja. Takođe, kao operacija nad kontinualnim signalima, posvećena je pažnja izračunavanju konvolucije.

Druga glava knjige je nalik prvoj, samo što je reč o diskretnim sistemima.

Treća glava detaljnije prezentuje osnovne osobine kontinualnih i diskretnih sistema i pojmove kao što su linearnost, kauzalnost, sistemi sa i bez memorije, BIBO stabilnost i invertibilnost. S obzirom da linearni vremenski invarijantni (LTI) sistemi imaju široku primenu u oblasti elektrotehnike, posebna pažnja biće posvećena njima.

Četvrta i peta glava sadrže frekvencijsku analizu periodičnih i aperiodičnih signala primenom Furije-ovih redova i Furijeove transformacije.

Šesta glava je posvećena analizi signala i sistema u kompleksnom domenu sa stanovišta Laplasove transformacije, a sedma glava analizi signala i sistema u kompleksnom domenu sa stanovišta z transformacije.

Poslednje poglavlje, posvećeno je obradi zvuka kao konkretnom primeru signala.

Sva teorijska razmatranja praćena su primerima u Python-u kako bi se studenti upoznali sa ovim programskim paketom.

Bićemo zahvalni na svakoj korisnoj sugestiji i savetu za buduće ispravke i dopune ovog materijala.

Beograd, juni 2017.god.

Autori



Sadržaj

Predgovor	III
1. Signali i Sistemi	1
1.1. Definicija i klasifikacija signala.....	1
1.1.1. Vremenski kontinualni i vremenski diskretni signali	1
1.1.2. Periodični i aperiodični signali	5
1.1.3. Parni i neparni signali	7
1.1.4. Realni i kompleksni signali	9
1.1.5. Deterministički i stohastički signali.....	9
1.1.5.1. Gausova (Normalna raspodela).....	10
1.1.6. Kauzalni i nekauzalni signali.....	19
1.1.7. Energija i snaga signala	19
1.2. Elementarni kontinualni signali	20
1.2.1. Odskočna ili Hevisajdova funkcija.....	20
1.2.2. Jedinična impulsna ili Dirakova funkcija	22
1.3 Transformacije u vremenu kontinualnih signala.....	26
1.3.1. Pomeranje signala u vremenu.....	27
1.3.2. Inverzija vremena	27
1.3.3. Skaliranje amplitude	28
1.3.4 Skaliranje vremena	28
1.3.5. Konvolucija kontinualnih vremenskih funkcija.....	29
1.4. Definicija i klasifikacija sistema	32
1.4.1. Kauzalni sistemi.....	33
1.4.2. Linearni sistemi	34
1.4.3. Vremenski invarijantni sistemi	35
1.4.4. Stabilni sistemi.....	36

2. Osobine diskretnih sistema	39
2.1. Podela diskretnih sistema	39
2.1.1. Sistemi sa i bez memorije	39
2.1.2. Kauzalni sistemi.....	40
2.1.3. Linearni sistemi	40
2.1.4. Vremenski invarijantni sistemi	42
2.1.5. Stabilni sistemi.....	43
2.2. Elementarni diskretni signali	44
2.2.1. Odskočna ili Hevisajdova funkcija.....	44
2.2.2. Jedinična impulsna ili Dirakova funkcija	45
2.2.3. Kompleksni eksponencijalni signal	48
2.2.4. Stohastički signali.....	49
2.2.5. Periodični signali	51
2.3. Transformacije u vremenu diskretnih signala.....	53
2.3.1. Pomeranje signala u vremenu.....	53
2.3.2. Inverzija vremena	53
2.3.3. Skaliranje amplitude	53
2.3.4. Skaliranje vremena	54
2.3.5. Simetričnost signala.....	55
2.3.6. Konvolucija diskretnih vremenskih funkcija.....	60
3. Furijeova analiza kontinualnih signala	67
3.1. Furijeovi redovi.....	67
3.2. Furijeova transformacija	69
3.3. Diskretna Furijeova Transformacija	70
3.4. Inverzna Diskretna Furijeova Transformacija	74
3.5. Teorema odabiranja.....	80

4. Laplasova transformacija	91
4.1. Definicija Laplasove teoreme	91
4.2. Inverzna Laplasova transformacija	92
4.2.1. Stukturni blok dijagrami sistema	96
5. z Transformacija	101
5.1. z Transformacija	101
5.1.1. Definicija z transformacije	101
5.1.2. Oblast konvergencije z transformacije	102
5.2. Inverzna z transformacija	103
5.2.1. Model nula i polova	104
5.2.2. Stukturni blok dijagrami sistema	105
6. Filtriranje signala	111
Zvučni signal	131
Literatura	142

1. Signali i Sistemi

Pojmovi signala i sistema, javljaju se u različitim granama nauke kao što su telekomunikacije, automatika, informatika i druge. U ovom poglavlju uvodimo osnovnu ideju o opisu i matematičkoj reprezentaciji signala i sistema. Takođe, biće predstavljeni osnovni tipovi kontinualnih signala, podela kontinualnih signala, kao i njihove transformacije u vremenu. Razmatrane su osobine periodičnosti, vremenskog skaliranja, inverzije, vremenskog pomeranja. Posebna pažnja posvećena je konvoluciji, važnoj matematičkoj operaciji, koja se prezentuje u kontinualnoj formi.

1.1. Definicija i klasifikacija signala

Signal, predstavlja fizičku promenljivu koja je funkcija raznih parametara, najčešće vremena. Obično sadrži informacije ili podatke o ponašanju, odnosno prirodi neke fizičke pojave. U ovom predmetu, pokazano je da se signali koji se javljaju u prirodi mogu predstaviti matematičkim izrazom kao funkcija više nezavisnih promenljivih kao što su: brzina vetra, vlažnost vazduha, vazdušni pritisak... U našoj analizi, obuhvatićemo samo signale koji su funkcija samo jedne promenljive, tj. vremena.

Budući da signali predstavljaju matematičke funkcije vremenskog argumenta, signali, kao i funkcije se mogu podeliti na više načina. U nastavku, biće reči o klasifikaciji signala po različitim kriterijumima.

1.1.1. Vremenski kontinualni i vremenski diskretni signali

Postoje dva osnovna tipa signala: kontinualni (ili analogni) i diskretni (ili digitalni) signali. O svakom od njih ćemo posebno govoriti. Signal $x(t)$ je kontinualan ukoliko je promenljiva t kontinualna promenljiva, tj. signal $x(t)$ je definisan u svakom vremenskom trenutku. Svi procesi u prirodi predstavljaju kontinualne funkcije. Za razliku od kontinualnih signala, kako bi se signali iz prirode prilagodili računarskoj obradi, uveden je pojam diskretnog signala $x[n]$, gde je n ceo broj. Diskretni signali dobijeni su odabiranjem kontinualnih signala $x(t)$ u odredjenim vremenskim trenucima. Budući da je signal diskretan u vremenu definisan u odredjenim trenucima, može se predstaviti kao sekvenca brojeva označena kao $\{x_n\}$ ili $x[n]$, gde je n ceo broj.

Zadatak 1.1.

U programskom jeziku Python nacrtati kontinualni signal $x(t) = t^2$ i diskretni signal $x[n] = n^2$

Rešenje:

Kontinualni signali se crtaju pomoću funkcije `plot()`, a diskretni pomoću funkcije `stem()`. Obe funkcije pripadaju biblioteci `matplotlib.pyplot`.

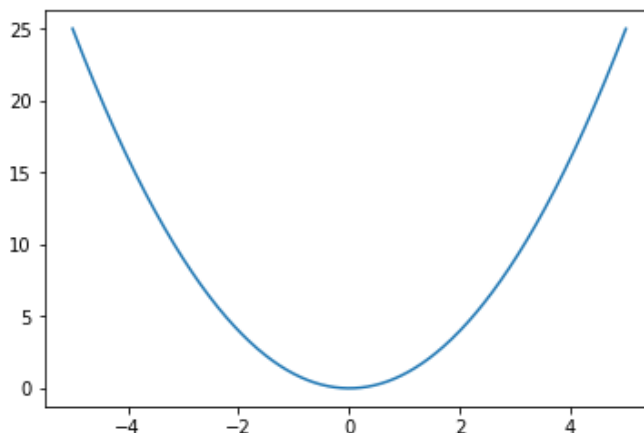
[Napomena: Iscrtavanje kontinualnih signala nije zapravo moguće, jer bi vremenska promenljiva t , i vrednosti funkcije $f(t)$ morala da ima beskonačno mnogo vremenskih trenutaka koje nije moguće smestiti u memoriju računara]

```
In [203]: import numpy as np  
import matplotlib.pyplot as plt
```

```
In [204]: n=np.arange(-5,5,1)  
t=np.arange(-5,5,0.001)
```

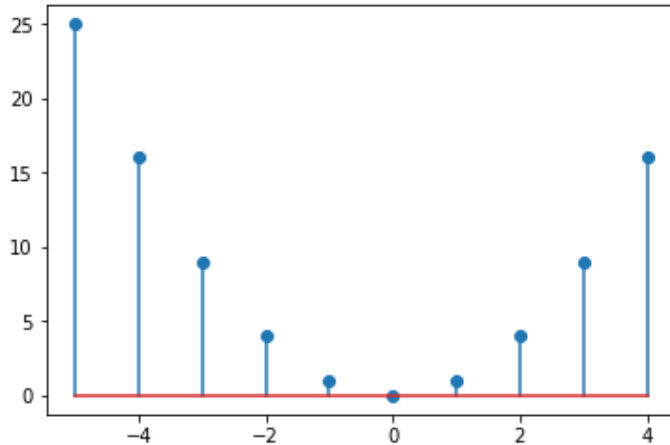
```
In [205]: x=t**2  
y=n**2
```

```
In [207]: plt.plot(t,x)  
plt.show()
```



Slika 1: Kontinualan signal $x(t) = t^2$

```
In [208]: plt.stem(n,y)
plt.show()
```



Slika 2: Diskretan signal $x[n] = n^2$

Signal $x(t)$ kontinualan u vremenu se može predstaviti preko svojih odbiraka x_n ili $x[n]$, čija je vrednost jednaka vrednosti signala u trenucima odabiranja, tj. $x(t_n)$. Kada su vremenski periodi između svih susednih odbiraka jednaki, tada je reč o uniformnom (ekvidistantnom) odabiranju i tada važi $x_n = x[n] = x(nT_s)$. Vremenski interval između dva uzastopna odabiranja, naziva se period odabiranja T_s i o njegovom određivanju biće reči kasnije.

Diskretni signal može biti označen na dva načina:

- 1) Pomoću pravila, na primer

$$x[n] = x_n = \begin{cases} (2)^n & n \geq 0 \\ 0 & n < 0 \end{cases}$$

- 2) Pomoću sekvence odbiraka, na primer

$$\{x_n\} = \{\dots, 0, 0, 1, 2, 4, 8, 16, \dots\}$$

↑

ili

$$\{x_n\} = \{0, 1, 2, 4, 8, 16\}$$

↑

Gde strelica označava vrednost niza za odбирак gde je $n=0$. U literaturi je usvojeno da ukoliko strelica u nizu nije označena, prvi element niza predstavlja odбирак za $n=0$.

Zbir i proizvod dva niza, definisan je kao

$$\{c_n\} = \{a_n\} + \{b_n\} \rightarrow c_n = a_n + b_n$$

$$\{c_n\} = \{a_n\} \{b_n\} \rightarrow c_n = a_n b_n$$

$$\{c_n\} = \alpha \{a_n\} \rightarrow c_n = \alpha a_n$$

Zadatak 1.2.

Koristeći programski jezik Python, formiraj niz

$$x[n] = x_n = \begin{cases} (2)^n & n \geq 0 \\ 0 & n < 0 \end{cases}$$

Rešenje:

```
In [24]: import numpy as np
import matplotlib.pyplot as plt

In [35]: n=np.arange(-5,5,1)

In [36]: x=np.zeros(10)

In [37]: x
Out[37]: array([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])

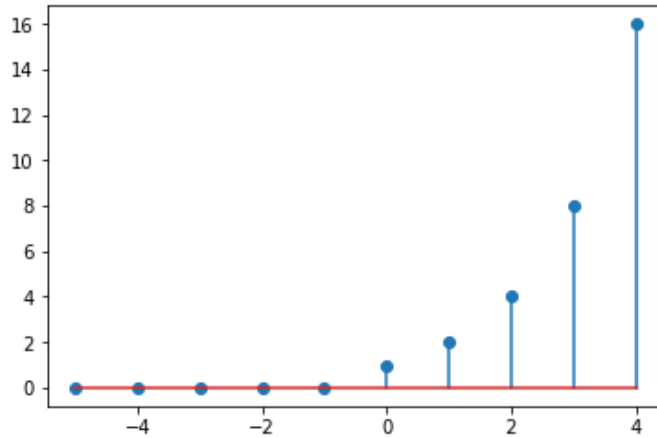
In [38]: n
Out[38]: array([-5, -4, -3, -2, -1,  0,  1,  2,  3,  4])

In [39]: x=2**n

In [40]: x
Out[40]: array([ 0,  0,  0,  0,  0,  1,  2,  4,  8, 16], dtype=int32)

In [41]: plt.stem(n,x)
plt.show()
```

```
In [41]: plt.stem(n,x)
plt.show()
```



Slika 3: Signal $x[n] = x_n = \begin{cases} (2)^n & n \geq 0 \\ 0 & n < 0 \end{cases}$

1.1.2. Periodični i aperiodični signali

Za kontinualni signal $x(t)$ se kaže da je *periodičan* ukoliko postoji pozitivna vrednost T za koju važi

$$x(t) = x(t \pm T) \text{ za svako } t$$

Takodje, periodični signali su signali predstavljeni trigonometrijskim funkcijama preko sinusa i kosinusa

$$x(t) = A \sin(\omega_0 t + \varphi) \text{ ili } x(t) = A \cos(\omega_0 t + \varphi),$$

gde A predstavlja amplitudu, ω_0 kružnu frekvenciju (rad/s), a φ fazni pomeraj signala $x(t)$.

Znajući da je sinusoidalna funkcija periodična, lako se može izračunati njen osnovni period iz izraza:

$$x(t) = A \sin(\omega_0 t + \varphi) = \sin(\omega_0 t + \varphi + T\omega_0) = \sin(\omega_0(t + T) + \varphi)$$

Ukoliko je $\omega_0 T = 2\pi m$ tj. $T = \frac{2\pi m}{\omega_0}$, gde je m ceo broj, tada je osnovni period

funkcije $x(t)$, $T_0 = \frac{2\pi}{\omega_0}$.

Zadatak 1.3.

Koristeći programski jezik Python, nacrtati sinusoidu i kosinusoidu čija je učestanost 3Hz.

Rešenje:

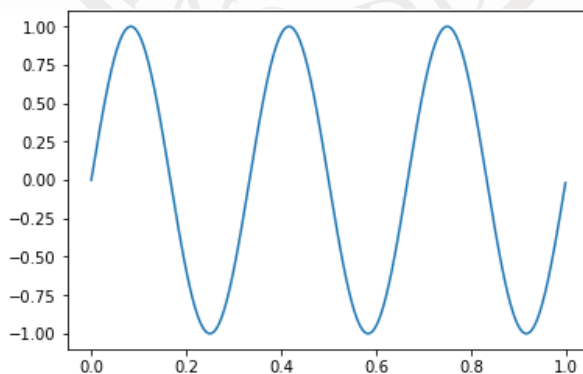
```
In [65]: import numpy as np
import matplotlib.pyplot as plt

In [66]: t=np.arange(1000,dtype=float)

In [67]: t=t/1000

In [68]: x=np.sin(2*np.pi*3*t)

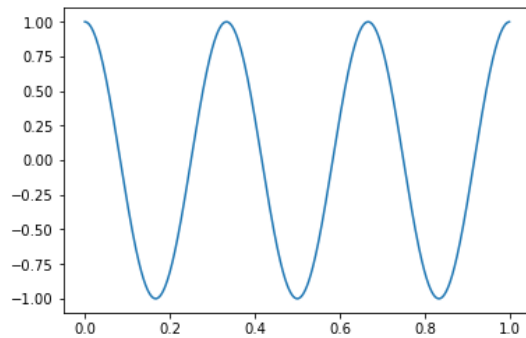
In [69]: plt.plot(t,x)
plt.show()
```



Slika 4: Sinusni signal učestanosti 3kHz

```
In [70]: y=np.cos(2*np.pi*3*t)
```

```
In [71]: plt.plot(t,y)  
plt.show()
```



Slika 5: Kosinusni signal učestanosti 3kHz

1.1.3. Parni i neparni signali

Signal $x(t)$ može biti *paran* $x(t)=x(-t)$ i *neparan* $x(t)=-x(-t)$. Ukoliko je signal paran, on je *simetričan* u odnosu na y osu, a ukoliko je neparan, simetičan je u odnosu na koordinatni početak.

Zadatak 1.4.

Koristeći programski jezik Python, nacrtati neparnu funkciju $x(t) = -x(t) = t$ i parnu funkciju $y(t) = y(-t) = |t|$

Rešenje:

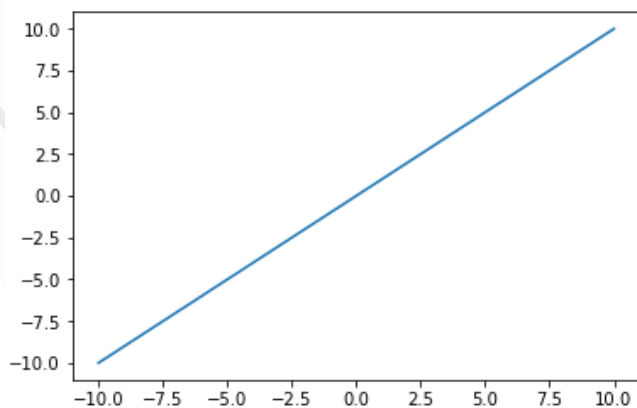
```
In [100]: import numpy as np  
import matplotlib.pyplot as plt
```

```
In [101]: t=np.arange(-10,10,0.001)
```

```
In [102]: x=t
```

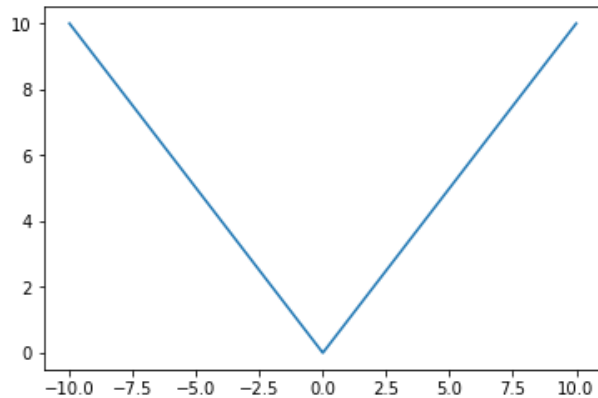
```
In [103]: y=np.abs(t)
```

```
In [105]: plt.plot(t,x)  
plt.show()
```



Slika 6: Neparan signal $x(t) = t$

```
In [107]: plt.plot(t,y)
plt.show()
```



Slika 7: Paran signal $y(t) = y(-t) = |t|$

1.1.4. Realni i kompleksni signali

Realni signal je realna matematička funkcija vremenskog argumenta, dok je kompleksni signal kompleksna funkcija vremena (za svako t , vrednost signala je jednoznačno definisan kompleksni broj, koji je opisan svojim modulom i argumentom).

1.1.5. Deterministički i stohastički signali

Kao što je na početku uvodnog poglavlja bilo reči, signali se mogu deliti na determinističke i stohastičke. Deterministički signal je realna matematička funkcija nezavisnog vremenskog argumenta, koja je jednoznačno određena za svaku vrednost argumenta t . Sa druge strane, stohastički signal je funkcija dva nezavisna argumenta; prvi od njih je vreme koji pripada skupu realnih brojeva, dok je drugi argument slučajni ishod. Primer stohastičkog signala je slučaj termičkog šuma koji zavisi od vremena i od frekvencije.

1.1.5.1. Gausova (Normalna raspodela)

U teoriji verovatnoće i statistici poznata je tzv. normalna ili Gausova raspodela. Funkcija gustine raspodele ima sledeći oblik:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, x \in R, \sigma \in R, \mu \in R$$

Tada se kaže da X ima normalnu

raspodelu sa parametrima σ^2 i μ što se označava sa $N(\mu, \sigma^2)$, gde promenljiva σ^2 predstavlja standardnu devijaciju, a μ predstavlja matematičko očekivanje. Gustina raspodele slučajne promenljive X ima sledeće osobine:

- Pozitivna je funkcija, tj. Za svako $x \in R$ važi $g(x) > 0$
- Gustina raspodele je simetrična u odnosu na pravu $x = \mu$, tj. za svako $x \in R$ važi $g(\mu + x) = g(\mu - x)$
- Gustina raspodele dostiže maksimum u $x = \mu$ i on iznosi

$$g(\mu) = \frac{1}{\sqrt{2\pi\sigma^2}}$$

- Gustina raspodele konvergira ka 0 kada $x \rightarrow \pm\infty$ (**plus/minus beskonacno**)

Zadatak 1.5.

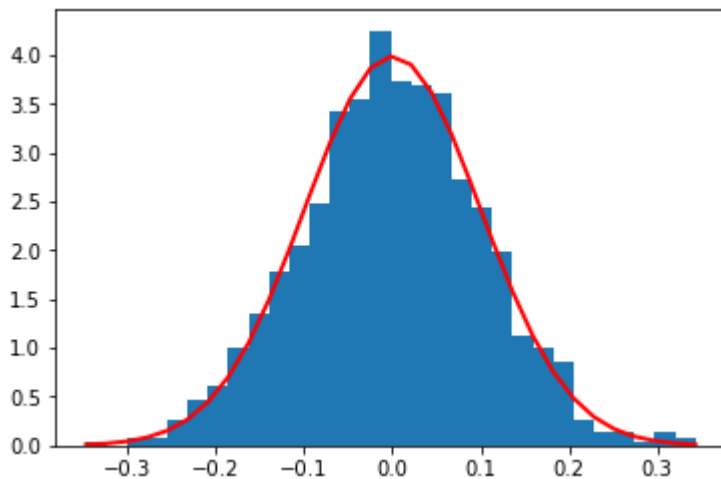
Nacrtati histogram (od pseudoslučajnih brojeva) i Gausovu Gustinu Raspodele za 'Probability Density Function' (pdf).

In [4]:

```
import numpy as np
import matplotlib.pyplot as plt
mu, sigma = 0, 0.1 # srednja vrednost i standardna devijacija
s = np.random.normal(mu, sigma, 1000)
```

In [5]:

```
count, bins, ignored = plt.hist(s, 30, normed=True)
plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) *
np.exp( - (bins - mu)**2 / (2 * sigma**2) ),
linewidth=2, color='r')
plt.show()
```



Slika 8: Histogram i Gausova raspodela srednja vrednosti $\mu = 0$ i standardne varijacije $\sigma = 0.1$ (dužine 1000)

Zadatak 1.6.

Koristeći programski jezik Python, nacrtati kontinualni signal Gausovog Šuma trajanja 100s. Uzeti vrednosti $\mu = 0$ i $\sigma^2 = 1$.

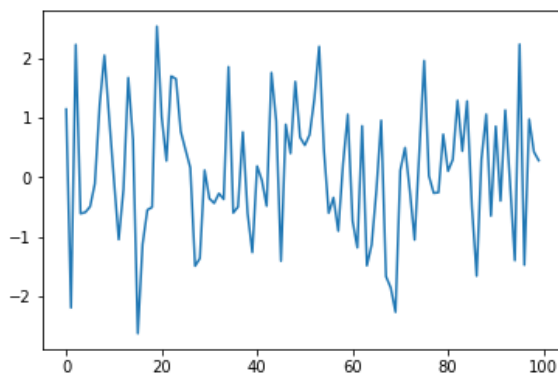
```
In [41]: import numpy as np

In [42]: import matplotlib.pyplot as plt

In [43]: noise = np.random.normal(0,1,100)

In [44]: plt.plot(noise)
Out[44]: [<matplotlib.lines.Line2D at 0x1fb63eabf28>]

In [45]: plt.show()
```



Slika 9: Slučajan signal (kontinualan u vremenu) Belog Gausovog Šuma-Aditive White Gaussian Noise (AWGN) srednje vrednosti $\mu = 0$ i standardne varijacije $\sigma = 1$ (dužine 100)

Veoma često nam se u praksi može desiti da imamo i šum merenja prilikom merenja neke fizičke veličine. Ovaj šum se često modeluje kao aditivni Gausovski šum. Postavlja se pitanje na koji način možemo umanjiti dejstvo ovakvog šuma. Jedna od najjednostavnijih metoda sastoji se od toga da se izvrši filtracija signala. Ukoliko znamo u kom delu spektra se nalazi koristan deo signala sasvim slobodno možemo izvršiti filtraciju u ostalom delu spektra. Npr. vidimo da su dve učestanosti kosinusnih funkcija podešene na relativno niskim učenostima, stoga možemo na dati signal primeniti filter propusnik niskih učestanosti.

Zadatak 1.7.

U ovom primeru, u programkom jeiku Python, videćemo efekat filtriranja Aditivnog Belog Gausovog Šuma dodatok na signal

$$x(t) = \cos(2\pi 3t) + 0.5 \cos(2\pi 11t)$$

```
In [284]: import scipy as sp
```

```
In [285]: import numpy as np
```

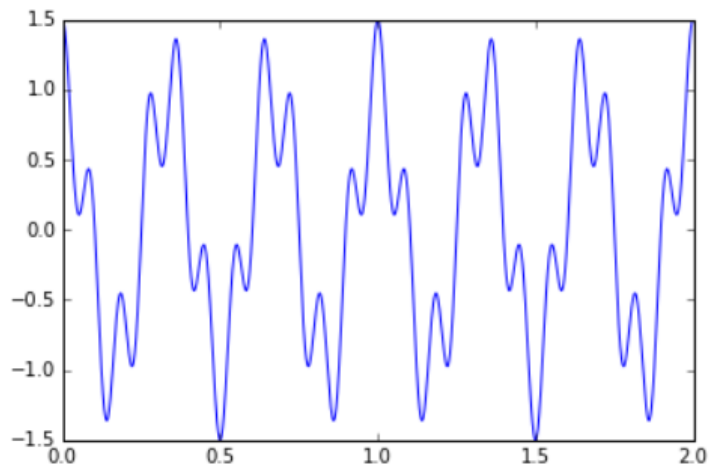
```
In [286]: import matplotlib.pyplot as plt
```

```
In [287]: Fs = 1000
```

```
In [288]: from scipy import signal
```

```
In [301]: plt.plot(t, x1)
```

```
Out[301]: [<matplotlib.lines.Line2D at 0x3bf4e0f0>]
```



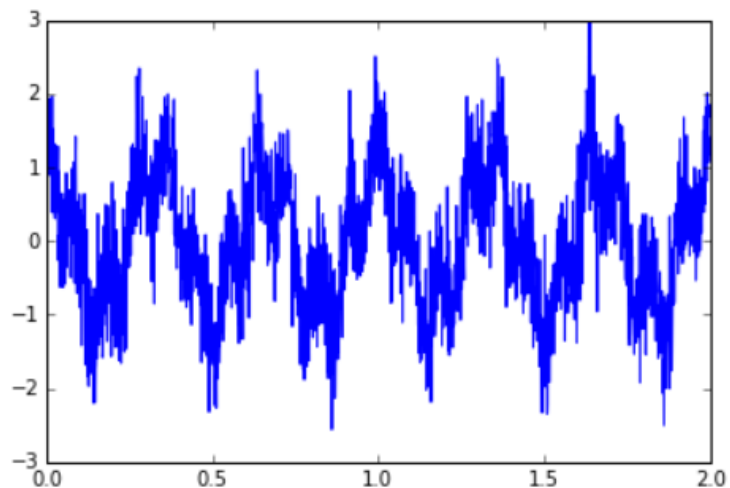
Slika 10: Signal $x(t) = \cos(2\pi 3t) + 0.5 \cos(2\pi 11t)$

U sledećem koraku, dodavajući Beli Gausov Šum na signal $x(t)$, dobićemo signal $x_1(t)$

```
In [304]: x = np.cos(2*np.pi*t*3) + 0.5*np.cos(2*np.pi*t * 11)
          x1 = x + np.random.randn(len(x1))*0.5
          # koeficijent kojim mnozimo izraz np.random.randn(len(x1)) utice
          # na amplitudu suma koji dodajemo deterministickom signalu x
```

```
In [306]: plt.plot(t, x1)
```

```
Out[306]: [<matplotlib.lines.Line2D at 0x3c540860>]
```



Slika 11: Signal $x_1(t) = \cos(2\pi 3t) + 0.5 \cos(2\pi 11t) + n(t)$

```
In [307]: x1 = sp.fft(x1)
```

```
In [308]: len(x1)
```

```
Out[308]: 2000
```

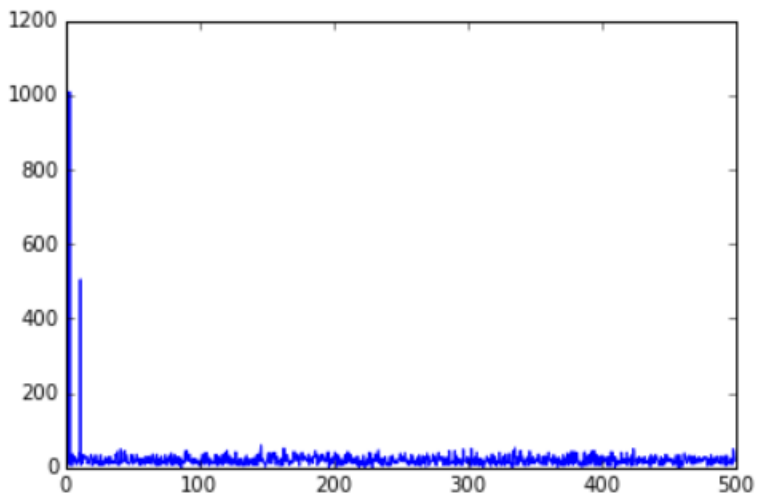
```
In [309]: f = np.arange(0, 1, 1./len(x1))  
f = f * Fs
```

```
In [310]: len(f)
```

```
Out[310]: 2000
```

```
In [311]: #Spektar signala kada je u signalu prisutan i Gausovski sum  
plt.plot(f[0:len(x1)/2], abs(x1[0:len(x1)/2]))
```

```
Out[311]: [<matplotlib.lines.Line2D at 0x3c5e5be0>]
```



Slika 12: Spektar signala $x_1(t) = \cos(2\pi 3t) + 0.5 \cos(2\pi 11t) + n(t)$

U sledećim koracima primenićemo filtriranje signala $x_1(t)$ primenom Batervortovog filtra

```
In [314]: b, a = signal.butter(4, 50.0/500 )
```

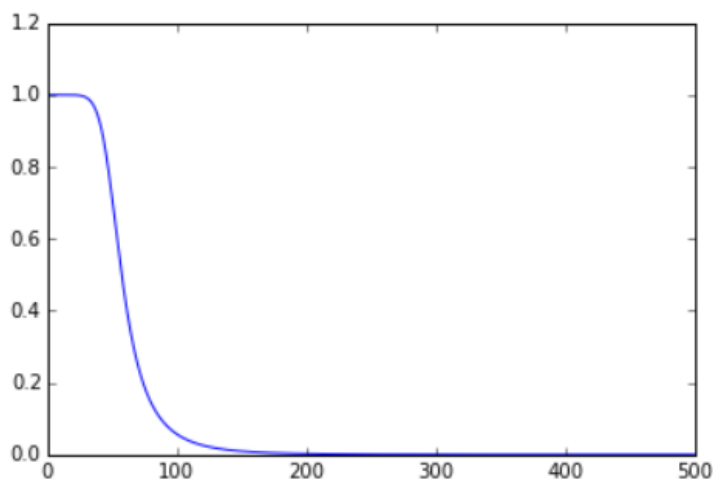
```
In [316]: w, h = signal.freqz(b, a, len(X1)/2)
```

```
In [317]: f1 = w / sp.pi * Fs/2
```

```
In [318]: plt.plot(f1, abs(h))
```

```
Out[318]: [<matplotlib.lines.Line2D at 0x3cdf0080>]
```

```
Out[318]: [<matplotlib.lines.Line2D at 0x3cdf0080>]
```



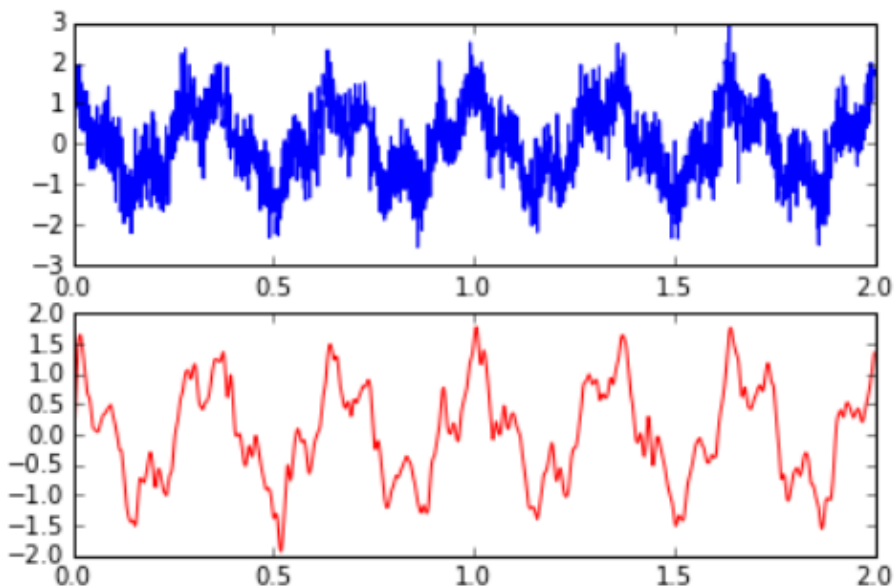
Slika 13: Frekvencijski odziv Batervortovog filtra

```
In [319]: y = signal.lfilter(b, a, x1)
```

```
In [320]: plt.subplot(211)
plt.plot(t, x1)
plt.subplot(212)
plt.plot(t, y, c='r')
# Na dve donje slike su prikazani signali
# prvi originalan signal
# a drugi koji je izfiltriran
# --> vidimo da je drugi signal zadržao originalni korisni deo signala,
# dok je sum uglavnom potisnut

# Treba obratiti pažnju da sum cijeli je frekvencijski deo smesten do 50Hz
# nije ovim procesom uklonjen.
```

```
Out[320]: [<matplotlib.lines.Line2D at 0x3d039358>]
```



Slika 14: Signal $x_1(t) = \cos(2\pi 3t) + 0.5 \cos(2\pi 11t) + n(t)$ pre (slika gore) i nakon filtriranja Batervortovim filtrom (slika dole)

Zadatak 1.8.

U ovom primeru treba promeniti vrednosti F_s tako da je Nikvistova teorema zadovoljena.

probati $F_s = 200, 100, 70, 65$ u ovim slučajevima će se na spektru tačno prikazati frekvenije odgovarajućih kosinusnih funkcija za vrednosti manje od dvostruke najveće učestanosti ($2 \cdot 30 \text{ Hz} = 60 \text{ Hz}$) $F_s = 55, 45$, itd..

```
Fs1 = 65

t1 = np.arange(0, 5, 1./Fs1)
```

```
In [322]: x3 = np.cos(2*np.pi*t1* 3) + np.cos(2*np.pi*t1* 30)
```

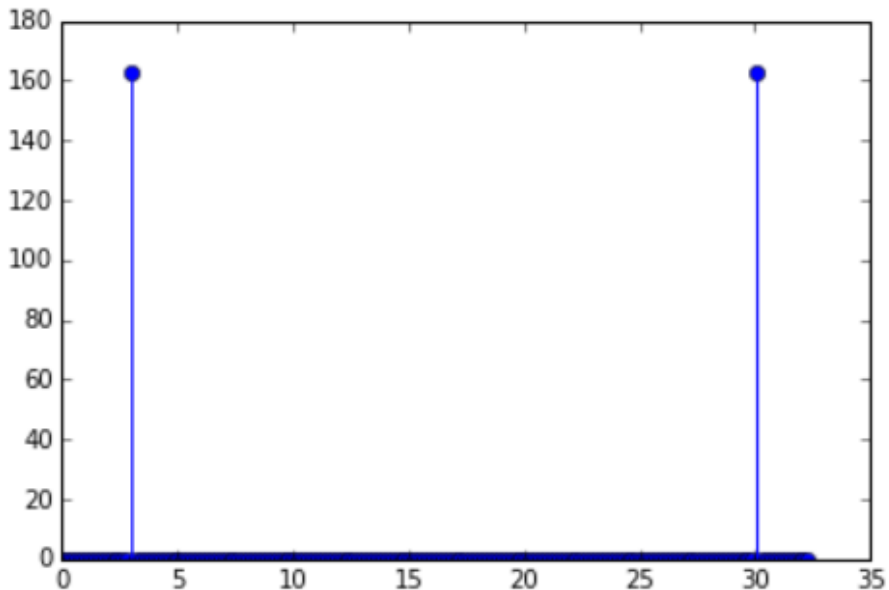
```
In [323]: X3 = sp.fft(x3)
           f = np.linspace(0, 1, len(X3)) * Fs1
```

```
In [324]: len(x3)
```

```
Out[324]: 325
```

```
In [325]: plt.stem(f[0:len(x3)/2], abs(X3[0:len(x3)/2]))
```

```
Out[325]: <Container object of 3 artists>
```



1.1.6. Kauzalni i nekauzalni signali

Ukoliko je ispunjen uslov

$f(t) = 0$ za $t < 0$, za signal se kaže da je kauzalan

U suprotnom slučaju, ukoliko je $f(t) \neq 0$ za neko $t < 0$, za signal se kaže da je nekauzalan. Očigledno je da kada govorimo o signalima koji se javljaju u prirodi, govorimo o kauzalnim signalima.

1.1.7. Energija i snaga signala

Rad ili energija vremenski-kontinualnog signala $f(t)$ na interval vremena $[t_1, t_2]$ dužine $L = t_2 - t_1$, definisana je izrazom:

$$E_L = \int_{t_1}^{t_2} f^2(t) dt$$

Ukoliko je $f(t)$ kompleksna, a ne realna funkcija, tada je energija signala na datom intervalu:

$$E_L = \int_{t_1}^{t_2} |f(t)|^2 dt, \text{ gde je } |f(t)| = f(t)f^*(t), \text{ gde simbol } (*) \text{ označava konjugovano}$$

kompleksnu vrednost broja $f(t)$.

Budući da snaga predstavlja brzinu vršenja rada, srednja snaga ima oblik dat izrazom:

$$P_L = \frac{1}{L} \int_{-L/2}^{L/2} |f(t)|^2 dt$$

1.2. Elementarni kontinualni signali

U ovoj sekciji upoznaćemo nekoliko važnih kontinualnih signala. Ovi signali, ne samo da predstavljaju elementarne signale, već služe da se iz njih izvedu ostali složeniji signali iz prirode.

1.2.1. Odskočna ili Hevisajdova funkcija

Odskočna ili *Hevisajdova* funkcija ili jedinični odskočni signal koji se uobičajeno obeležava i definiše na sledeći način:

$$u(t) = \begin{cases} 1, & t > 0 \\ 0, & t < 0 \end{cases}$$

Hevisajdova funkcija, prikazana je na slici 15.

Može se primetiti da ova funkcija ima prekid za vrednost $t=0$ jer je u toj tački nedefinisana. Ukoliko ova funkcija ima kašnjenje t_0 , obeležava i definiše se na sledeći način:

$$u(t-t_0) = \begin{cases} 1, & t > t_0 \\ 0, & t < t_0 \end{cases}$$

Zadatak 1.9.

Koristeći programski jezik Python, nacrtati Hevisajdovu funkciju $u(t)$ i funkciju $u(t-2)$.

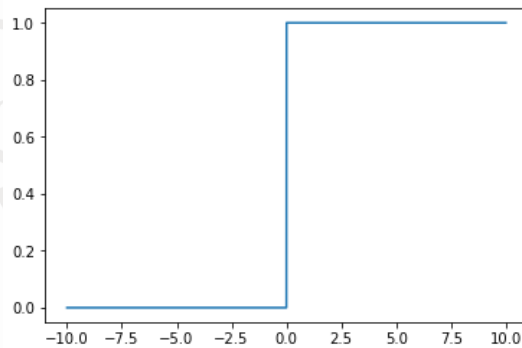
Rešenje:

```
In [112]: import numpy as np
import matplotlib.pyplot as plt

In [113]: t=np.arange(-10,10,0.001)

In [114]: x=1.*(t>=0)

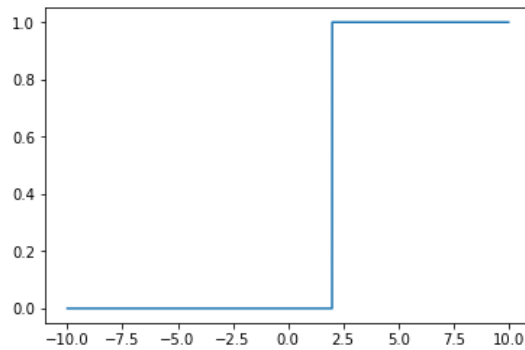
In [115]: plt.plot(t,x)
plt.show()
```



Slika 15: Hevisajdova funkcija $u(t)$

```
In [116]: y=1.*(t>=2)
```

```
In [117]: plt.plot(t,y)
plt.show()
```



Slika 16: Hevisajdova funkcija $u(t-2)$

1.2.2. Jedinična impulsna ili Dirakova funkcija

Jedinična ili *Dirakova* funkcija predstavlja “brzinu promene” Hevisajdove funkcije u trenutku $t=0$, tj.:

$$\delta(t) = \left. \frac{du(t)}{dt} \right|_{t=0}$$

Uobičajeno se definiše na sledeći način:

$$\delta(t) = \begin{cases} 1, & t = 0 \\ 0, & t \neq 0 \end{cases}$$

Teoretski, Dirakova funkcija predstavlja impuls čija amplituda teži beskonačnosti, a trajanje teži nuli.

Zadatak 1.10.

Koristeći programski jezik Python, nacrtati Dirakov impuls $\delta(t)$ i $\delta(t-2)$.

Rešenje:

```
In [157]: import numpy as np  
import matplotlib.pyplot as plt
```

```
In [158]: t=np.arange(-10,10,1)
```

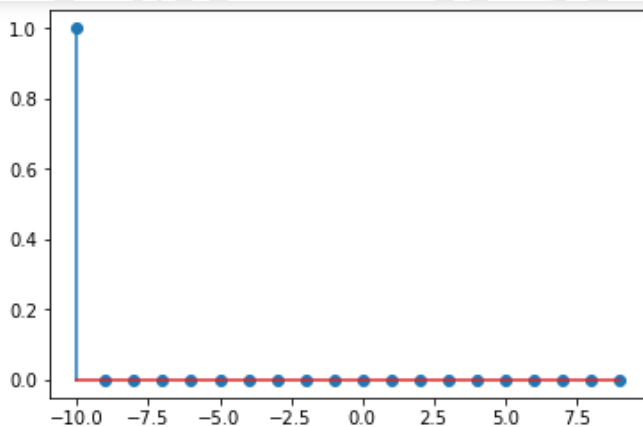
```
In [159]: len(t)
```

```
Out[159]: 20
```

```
In [160]: x=np.zeros(20)  
y=np.zeros(20)
```

```
In [161]: x[0]=1
```

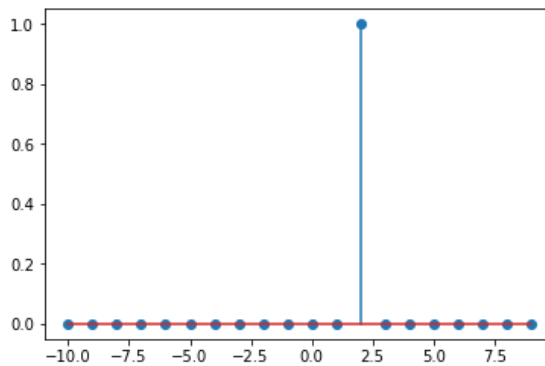
```
In [162]: plt.stem(t,x)  
plt.show()
```



Slika 17: Dirakov impuls $\delta(t)$

```
In [163]: y[12]=1
```

```
In [164]: plt.stem(t,y)  
plt.show()
```

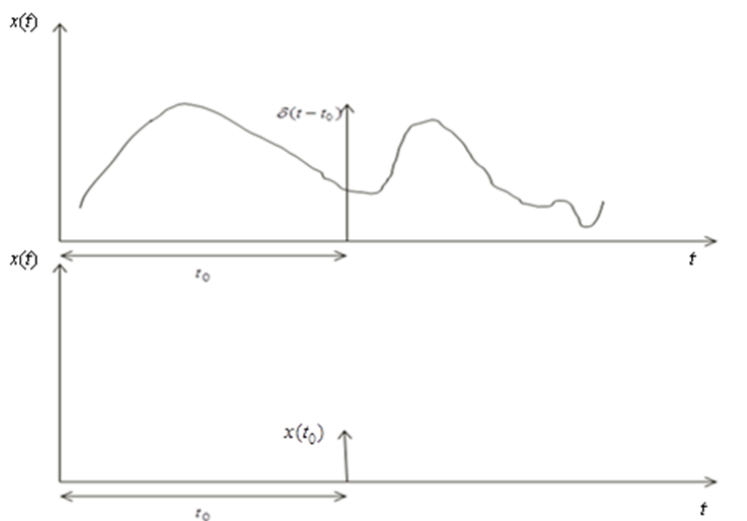


Slika 18: Dirakov impuls $\delta(t-2)$

Može se primetiti da proizvod nekog signala $x(t)$ i dirakovog impulsa $\delta(t)$ u određenom trenutku t_0 , predstavlja vrednost signala u tom trenutku, tj.

$$x(t) \cdot \delta(t - t_0) = x(t_0)$$

Dati proizvod, prikazan je na slici 14.



Slika 19: Proizvod funkcije $x(t)$ i dirakovog impulsa $\delta(t)$ u trenutku $t=t_0$

Zadatak 1.11.

U programskom jeziku Python, nacrtati funkciju $x(t) = \sin(t)/t$, poznatu kao $\text{sinc}(t)$

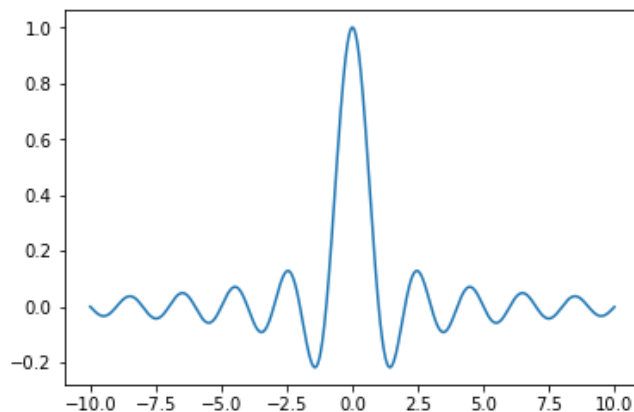
Rešenje:

```
In [83]: import numpy as np
import matplotlib.pyplot
```

```
In [84]: t=np.arange(-10,10,0.01)
```

```
In [85]: x=np.sinc(t)
```

```
In [86]: plt.plot(t,x)
plt.show()
```



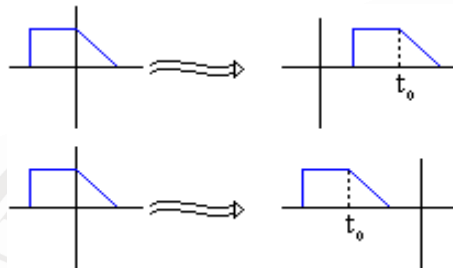
Slika 20: Signal $\text{sinc}(t)$

1.3 Transformacije u vremenu kontinualnih signala

U ovoj sekciji, predstavimo grafički i analitički kontinualne signale koji su nastali nekom modifikacijom promenljive t , kao što su: pomeranje signala u vremenu, inverzija vremena, skaliranje vremena, simetričnost, periodičnost i konvolucija signala.

1.3.1. Pomeranje signala u vremenu

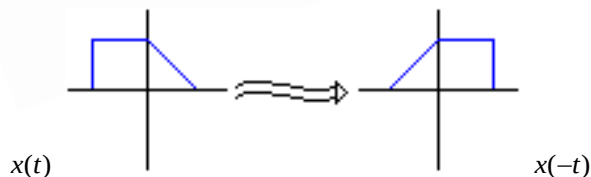
Pomeranje signala $x(t)$ u vremenu, analitički predstavljeno kao $x(t-t_0)$ predstavlja pomeranje signala $x(t)$ za vremenski interval t_0 . Ukoliko je $t_0 > 0$, reč je o kašnjenju signala (pomeranje u desno), dok je za $t_0 < 0$, reč je o žurenju tj. prednjačenju signala (pomeranje u levo). Na slici 21, grafički je predstavljena ova modifikacija.



Slika 21: Pomeranje signala u vremenu

1.3.2. Inverzija vremena

Inverzija signala $x(t)$ u vremenu, analitički predstavljeno kao $x(-t)$, predstavlja “sliku u ogledalu” signala $x(t)$. Na slici 22, grafički je predstavljena ova modifikacija.



Slika 22: Inverzija signala

1.3.3. Skaliranje amplitude

Skaliranje signala po amplitudi se postiže tako što se amplituda signala pomnoži konstantom A , tako da se za signal kontinualan u vremenu dobija

$$x(t) = Ax(t)$$

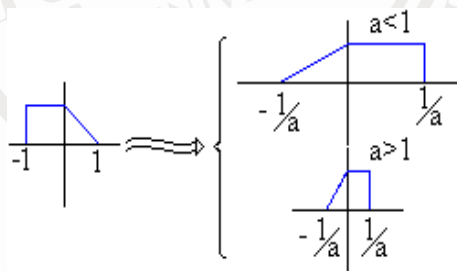
- Skaliranje sa $A=-1$ dovodi do rotacije funkcije oko y ose.
- Ova modifikacija se se najčešće koristi u slučaju modeliranja pojačanja ili slabljenja signala

1.3.4 Skaliranje vremena

Skaliranje signala $x(t)$ u vremenu, analitički predstavljeno kao $x(at)$ predstavlja skaliranje signala $x(t)$ vremenskom konstantom a .

- za vrednosti parametra $a>1$ funkcija se sužava duž vremenske ose (ubrzavanje)
- za vrednosti parametra $0<a<1$ funkcija se širi duž vremenske ose (usporavanje)
- za vrednosti parametra $a<0$ funkcija se rotira oko y-ose (vreme "teče unazad")

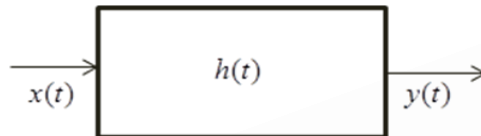
Na slici 23, grafički je predstavljena funkcija $x(-at)$ za slučajeve $0<a<1$ i $a>1$.



Slika 23: Skaliranje signala u vremenu

1.3.5. Konvolucija kontinualnih vremenskih funkcija

Poznavajući impulsni odziv $h(t)$ LTI sistema (slika 24), matematičkom operacijom poznatom kao *konvolucija*, moguće je odrediti njegov odziv $y(t)$ na pobudni signal $x(t)$ proizvoljnog oblika.



Slika 24: LTI sistem

Analitički, konvolucija signala $x(t)$ i $h(t)$ se predstavlja kao:

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t - \tau)d\tau$$

Integral se tumači kao površina ispod rezultujuće krive između signala $x(\tau)$ i $h(t - \tau)$.

Za konvoluciju važe sledeće osobine:

- Komutacija

$$x(t) * h(t) = h(t) * x(t)$$

- Asocijacija

$$(x(t) * h(t)) * g(t) = x(t) * (h(t) * g(t))$$

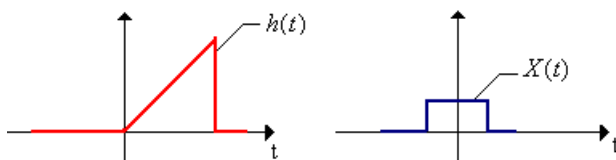
- Distribucija sa operacijom sabiranja

$$x(t) * (h_1(t) + h_2(t)) = x(t) * h_1(t) + x(t) * h_2(t)$$

Najveća primena konvolucije je u telekomunikacijama, u izračunavanju odziva kontinualnih sistema i harmonijskim analizama. Rezultat se može dobiti analitički, rešavanjem integrala i grafički, izračunavanjem površine ispod rezultujuće krive.

Koraci u konvoluciji

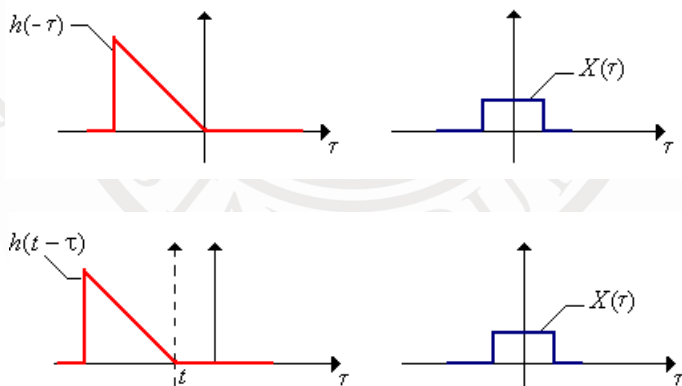
Kako bismo bolja razumeli pojam konvolucije, posmatrajmo signale $x(t)$ i $h(t)$ sa slike 25. čiju konvoluciju želimo da izračunamo.



Slika 25: Primeri signala za koje želimo izračunati konvoluciju

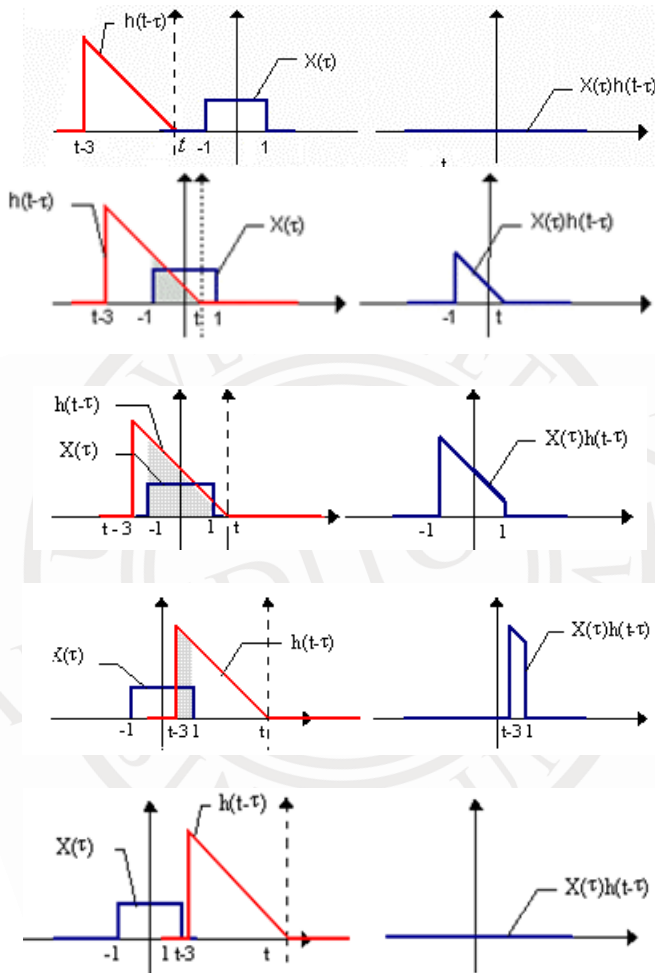
Proces konvolucije se može podeliti u 4 koraka:

1. Signal $h(\tau)$ se prvo invertuje i pomeri u vremenu (slika 26) da bi se dobio signal $h(t - \tau)$, tako da se signal $h(t - \tau)$ posmatra kao funkcija od τ , gde t predstavlja parametar fiksne vrednosti.



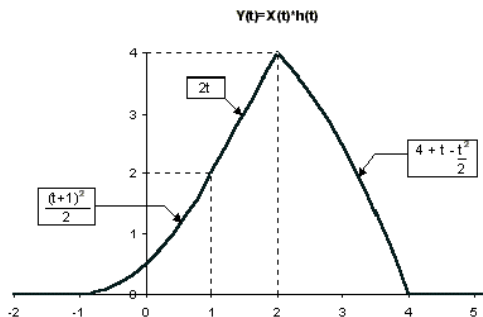
Slika 26: Invertovan signal $h(\tau)$ i pomeren u vremenu

2. Signali $x(\tau)$ i $h(t-\tau)$ se izmnože za sve moguće vrednosti τ za neko fiksno t (slika 27)
- 3.



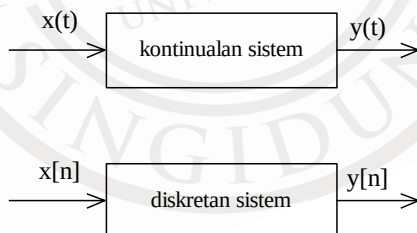
Slika 27: Računanje funkcije konvolucije za sve vrednosti parametra t

3. Proizvod $x(\tau)h(t-\tau)$ se integriše po celom skupu vrednosti τ i tako se dobija vrednost $y(t)$ za neko fiksno t .
4. Ponove se koraci 1, 2, 3 za sve vrednosti t iz skupa $(-\infty, \infty)$ kako bi se dobila kompletna funkcija konvolucije signala $x(t)$ i $h(t)$, $y(t)$ (slika 28).

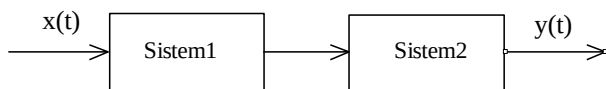
Slika 28: Funkcija konvolucije signala $x(t)$ i $h(t)$

1.4. Definicija i klasifikacija sistema

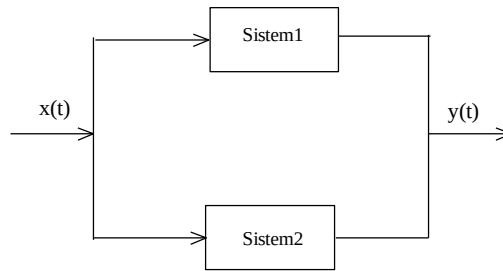
Sistem je matematički model fizičkog procesa koji predstavlja odnos odziva $y(t)$ i pobude $x(t)$, tj. $y(t)=Tx(t)$ za kontinualan ili $y[n]=Tx[n]$ za diskretan sistem. Sistemi mogu imati veći broj ulaza i veći broj izlaza. Više sistema može biti povezano kaskadno (slika 30), paralelno (slika 31) ili može formirati povratnu spregu (slika 32).



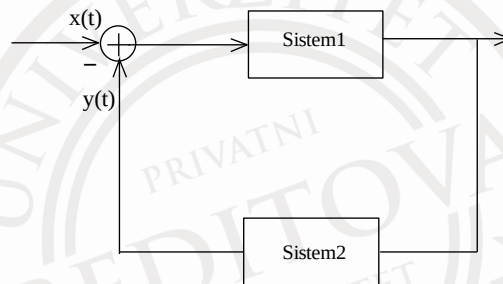
Slika 29: Blok dijagram kontinualnog i diskretnog sistema



Slika 30: Kaskadna veza sistema



Slika 31: Paralelna veza sistema



Slika 32: Sistem sa povratnom spregom

1.4.1. Kauzalni sistemi

Za sistem kažemo da je *kauzalan* ukoliko njegov izlaz $y(t)$ u slučajnom trenutku t_0 zavisi samo od ulaza $x(t)$ za $t \leq t_0$. Tako da u kauzalnim sistemima izlaz zavisi samo od sadašnjeg ili prošlog stanja vrednosti ulaza. Zbog toga, u kauzalnim sistemima je nemoguće izračunati vrednost izlaza ako pre toga nismo imali vrednost ulaza. Svi sistemi bez memorije su kauzalni, dok svi kauzalni nisu bez memorije.

Primeri za nekauzalni sistem:

$$y(t) = x(t + 1)$$

$$y(t) = x(t) + x(t + 3)$$

Primeri za kauzalni sistem:

$$y(t) = x(t-3)e^{(t-2)}$$

$$y(t) = x(t) + x(t-3)$$

1.4.2. Linearni sistemi

Za sistem se kaže da je *linearan* ako su ispunjeni uslovi *homogenosti* i *aditivnosti*, tj. da bi bila ispunjena osobina linearnosti, potrebno je da se odziv na pobudu $x_1(t) + x_2(t)$ može predstaviti izrazom $y_1(t) + y_2(t)$.

Da bi bila ispunjena osobina homogenosti, za bilo koje vrednosti konstanti α i β , potrebno je da bude zadovoljen sledeći izraz:

$$y_1(t) = f(x_1(t)); y_2(t) = f(x_2(t)) \Rightarrow \alpha y_1(t) + \beta y_2(t) = f(\alpha x_1(t) + \beta x_2(t))$$

Primer za linearan sistem:

$$y = 5x$$

Dokaz:

$$\alpha x(t) \Rightarrow y(t) = 5\alpha x(t) = \alpha y(t), \forall \alpha \in C, \text{ što znači da važi uslov homogenosti}$$

$$x(t) = x_1(t) + x_2(t) \Rightarrow y(t) = 5(x_1(t) + x_2(t)), \text{ što znači da važi uslov aditivnosti}$$

Primer za nelinearan sistem:

$$y = x^2$$

Dokaz:

$$x_1 \rightarrow y_1 = x_1^2$$

$$x_2 \rightarrow y_2 = x_2^2$$

$$(x_1 + x_2)^2 = x_1^2 + 2x_1x_2 + x_2^2 \neq x_1^2 + x_2^2$$

1.4.3. Vremenski invarijantni sistemi

Za sistem se kaže da je *vremenski invarijantan* (*time invariant*) ako vremenski pomeraj u pobudi prouzrokuje isti vremenski pomeraj u odzivu, tj. za bilo koju realnu vrednost t_0 važi:

$$y(t) = f(x(t)) \Rightarrow y(t - t_0) = f(x(t - t_0))$$

Primeri za vremenski invarijantan sistem:

$$1. \quad y(t) = x(t) \cos \omega_c t$$

Dokaz:

$$y(t - t_0) = x(t - t_0) \cos \omega_c (t - t_0) \neq x(t - t_0) \cos \omega_c t;$$

$$2. \quad y(t) = tx(t)$$

Dokaz:

$$y(t - t_0) = (t - t_0)x(t - t_0) \neq tx(t - t_0);$$

Primer za vremenski invarijantan sistem:

$$y(t) = ax(t)$$

Dokaz:

$$y(t - t_0) = ax(t - t_0);$$

Na osnovu dobijene jednakosti, vidi se da se izlaz sistema ne menja u zavisnosti od trenutka pojave ulaznog signala. Drugim rečima, vremenski nezavisan system, ukoliko njegov ulazni signal $x(t)$, prouzrokuje izlazni signal $y(t)$, tada će i ulazni signal $x(t - t_0)$, prouzrokovati izlaz $y(t - t_0)$.

1.4.4. Stabilni sistemi

Za sistem se kaže da je *stabilan* ili u literaturi poznat kao *bounded-input/bounded-output (BIBO)*, ako za bilo koju vrednost ulaza x , definisanu sa

$$x < |k_1|$$

Odgovarajuća vrednost izlaza je takodje ograničena, tj.

$$y < |k_2|$$

$$k_1, k_2 \in R$$

Primer za stabilan sistem:

$$y(t) = x(t) \cos \omega_c t$$

Dokaz:

$$|\cos \omega_c t| \leq 1 \Rightarrow |y(t)| = |x(t) \cos \omega_c t| \leq |x(t)|$$

Što znači da ako je signal $x(t)$ ograničen, sledi da je i signal $y(t)$ ograničen.

Primer za nestabilan sistem:

$$y(t) = tx(t)$$

Dokaz:

Ako je signal $x(t)$ ograničen, tj.

$$x(t) \leq |k|$$

Ukoliko vremenska promenljiva teži beskonačnosti, tj.

$$t \rightarrow \infty$$

Sledi i da promenljiva $y(t)$ teži beskonačnosti, tj.

$$y(t) \rightarrow \infty$$

Zadatak 1.12.

U programskom jeziku Python, napisati program koji iscrtava zbir sinusoida na 10, 30 i 50 Hz. Odrediti osnovnu učestanost signala. Odrediti osnovni period i kružnu učestanost za ovu funkciju:

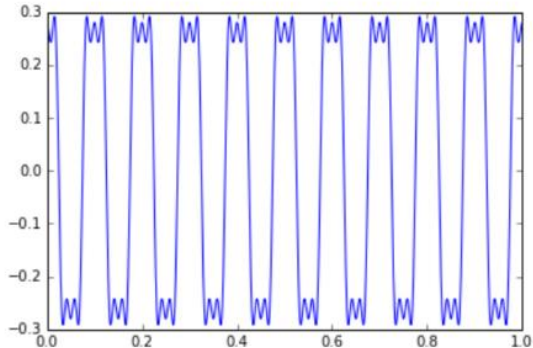
Rešenje:

```
In [31]: import scipy as sp
import matplotlib.pyplot as plt

In [32]: t = sp.arange(0,1,0.001)

In [33]: x = 0.33*sp.cos(2 * sp.pi * 10 * t) - 0.1 * sp.cos(2 * sp.pi * 3 * 10 * t) + 0.05 * sp.cos(2 * sp.pi * 5 * 10 * t)

In [34]: plt.plot(t,x)
Out[34]: [<matplotlib.lines.Line2D at 0xe32c358>]
```



$$\omega_0 = \text{NZD}(10\pi, 30\pi, 50\pi) = 5\pi \text{ rad/s} \quad T_0 = \frac{2\pi}{5\pi} = 0.4\text{s}$$

2. Osobine diskretnih sistema

Diskretni signal, za razliku od kontinualnih signala, da bi mogao da se “obrađuje” u računarskim analizima, jednoznačno je definisan u diskretnim vremenskim trenucima. Ovi trenuci, nastali su odabiranjem analognih signala u jednakim vremenskim intervalima. U slučaju da je amplituda diskretnog signala kvantovana, možemo reći da je reč o *digitalnim signalima*. Iste osobine koje važe za analogne, važe i za digitalne, sa razlikom da digitalne signale označavamo sa $x[n]$, gde $n \in \mathbb{Z}$ i zapisujemo kao sekvence brojeva, odnosno funkcije diskretne promenljive.

Na način na koji smo u predhodnom poglavlju analizirali analogne, u ovom poglavlju analiziraćemo digitalne signale.

2.1. Podela diskretnih sistema

2.1.1. Sistemi sa i bez memorije

Za diskretni sistem se kaže da je *bez memorije* ukoliko samo zavisi od ulaza u istom trenutku n . U suprotnom, kaže se da je sistem sa memorijom. Primer sistema bez memorije je otpornost R sa ulazom $x[n]$ i izlazom $y[n]$ koji mogu biti napon ili struja. Na osnovu Omovog zakona važi jednakost:

$$y[n] = Rx[n]$$

Primer sistema sa memorijom je kondenzator C sa strujom na ulazu $x[n]$ i naponom na izlazu $y[n]$;

$$y[n] = \frac{1}{C} \sum_{n=-\infty}^N x[n]$$

Primeri sistema bez memorije:

$$y[n] = x[n]^2 + 5 * x[n] + 2$$

$$y[n/5] = 6 * x[n/5]$$

Primeri sistema sa memorijom:

$$y[n] = 2 * x[n - 7]$$

$$y[n] = 6 * x[n/8]$$

2.1.2. Kauzalni sistemi

Za diskretni sistem možemo reći da je *kauzalan* ukoliko njegov izlaz $y[n]$ u slučajnom trenutku N zavisi samo od ulaza $x[n]$ za $n \leq N$. Ovaj uslov znači da u kauzalnim sistemima izlaz zavisi samo od sadašnjeg ili prošlog stanja vrednosti ulaza. Zbog toga, u kauzalnim sistemima je nemoguće izračunati vrednost izlaza ako pre toga nismo imali vrednost ulaza.

Primeri za diskretnan nekauzalan sistem:

$$y[n] = x[n + 1]$$

$$y[n] = x[n] + x[n + 3]$$

Primeri za kauzalni sistem:

$$y[n] = e^{[n-3]} + x[n]e^{n/2}$$

$$y[n] = (x[n] + x[n-3])^2$$

Kao i kod kontinualnih signala, svi sistemi bez memorije su kauzalni, dok svi kauzalni nisu bez memorije.

2.1.3. Linearni sistemi

Za diskretnan sistem se kaže da je *linearan* ako važe principi *homogenosti* i *aditivnosti*.

Osobina aditivnosti je ispunjena ukoliko se odziv na pobudu $x_1[n] + x_2[n]$ može predstaviti izrazom $y_1[n] + y_2[n]$.

Osobina *homogenosti* je ispunjena ako važi sledeći izraz

$$y_1[n] = f(x_1[n]); y_2[n] = f(x_2[n])$$

$$\Rightarrow \alpha y_1[n] + \beta y_2[n] = f(\alpha x_1[n] + \beta x_2[n])$$

gde su α i β konstante.

Primer za linearan sistem:

$$y[n] = 5x[n] + 3$$

Dokaz:

Uslov homogenosti je ispunjen jer važi sledeća jednakost:

$$\alpha x[n] \Rightarrow y[n] = 5\alpha x[n] + 3\alpha = \alpha y[n], \forall \alpha \in C,$$

Takodje, ispunjen je i uslov aditivnosti

$$x[n] = x_1[n] + x_2[n] \Rightarrow y[n] = 5(x_1[n] + x_2[n])$$

Iz ovih osobina, vidimo da je ovaj sistem linearan.

Primer za nelinearan sistem:

$$y[n] = \sin(x[n])$$

Dokaz:

Osobina homogenosti je ispunjena jer ako na ulaze dovedemo signale $x_1[n]$ i $x_2[n]$, na izlazima će se pojaviti signali $y_1[n]$ i $y_2[n]$, respektivno.

$$x_1[n] \rightarrow y_1[n] = \sin(x_1[n])$$

$$x_2[n] \rightarrow y_2[n] = \sin(x_2[n])$$

Osobina aditivnosti nije ispunjena jer ako se na ulaz dovede zbir

$$x[n] = x_1[n] + x_2[n]$$

Na izlazu se dobija signal

$$y[n] = \sin(x_1[n] + x_2[n])$$

Što je različito od

$$\sin(x_1[n]) + \sin(x_2[n]) = y_1[n] + y_2[n]$$

2.1.4. Vremenski invarijantni sistemi

Za diskretan sistem se kaže da je *vremenski nezavisan* (*time invariant*) ako vremenski pomeraj u pobudi prouzrokuje isti vremenski pomeraj u odzivu, tj. za bilo koju realnu vrednost N važi:

$$y[n] = f(x[n]) \Rightarrow y[n - N] = f(x[n - N])$$

Primer za vremenski zavisani sistem:

$$y[n] = nx[n]$$

Dokaz:

Ako se na ulaz sistema dovede signal $x[n - N]$, na izlazu će se pojaviti signal

$$y[n - N] = nx[n - N]$$

Što je različito od vrednosti $[n - N]x[n - N]$, potrebno da bi se reklo da je sistem nezavisan.

Primer za vremenski nezavisan sistem:

$$y[n] = \sin(x[n])$$

Dokaz:

Ako se na ulaz dovede signal $x[n - N]$, na izlazu se dobija vrednost

$$y[n - N] = \sin(x[n - N])$$

Što znači da je sistem vremenski nezavisan.

2.1.5. Stabilni sistemi

Za sistem se kaže da je *stabilan* ili *bounded-input/bounded-output (BIBO)* ako za bilo koju vrednost ulaza $x[n]$, definisanu sa

$$|x[n]| < |k_1|$$

Odgovarajuća vrednost izlaza je takodje ograničena sa

$$|y[n]| < |k_2|$$

gde

$$k_1, k_2 \in \mathbb{R}$$

Primer za stabilan sistem:

$$y[n] = x[n] \cos(x[n])$$

Dokaz:

Budući da je signal $|\cos x[n]|$ ograničen, tj.

$$|\cos x[n]| \leq 1$$

Znači i da je signal $|y[n]|$ ograničen, ukoliko je signal $|x[n]|$ ograničen, tj.

$$|y[n]| = |x[n] \cos(x[n])| \leq |x[n]|$$

Primer za nestabilan sistem:

Ako je dat signal na izlazu $y[n]$

$$y[n] = \sum_{n=-\infty}^{+\infty} x[n] \text{ gde je } x[n] = u[n]$$

Tada se dobija da je signal na izlazu

$$y[n] = (n+1)u[n]$$

Dokaz:

Zamenom vrednosti promenljive n ($n=1, 2, 3...$) u izlazni signal $y[n]$, dobija se

$$y[1] = 2, y[2] = 3 \dots y[n] = n + 1$$

Tako da u slučaju da n raste do beskonačnosti, sistem je nestabilan.

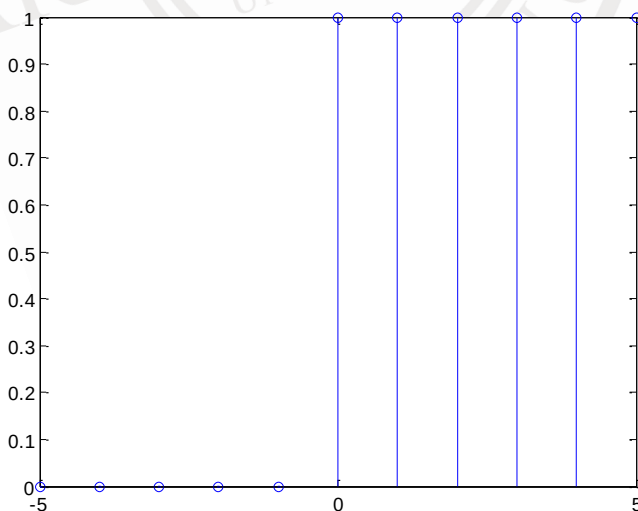
2.2. Elementarni diskretni signali

2.2.1. Odskočna ili Hevisajdova funkcija

Odskočna ili *Hevisajdova* funkcija uobičajeno se obeležava i definiše na sledeći način:

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}$$

Diskretna Hevisajdova funkcija, prikazana je na slici 33.



Slika 33: Diskretna Hevisajdova funkcija.

Zadatak 2.1.

Koristeći programski jezik Python, nacrtati diskretnu Hevisajdovu funkciju na vremenskoj osi na intervalu $(-10,10)$.

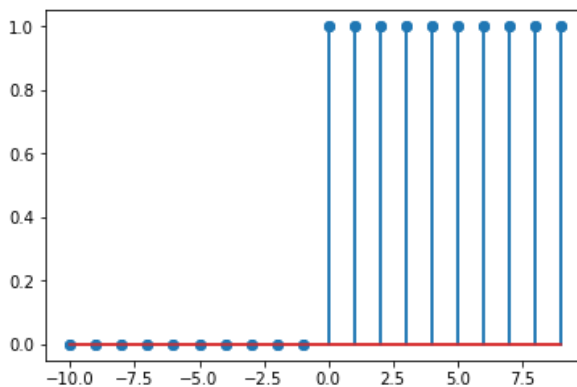
Rešenje:

```
In [36]: import numpy as np
import matplotlib.pyplot as plt
```

```
In [37]: n=np.arange(-10,10,1)
```

```
In [39]: x=1.*(n>=0)
```

```
In [41]: plt.stem(n,x)
plt.show()
```

**2.2.2. Jedinična impulsna ili Dirakova funkcija**

Jedinična ili Dirakova funkcija uobičajeno se definiše na sledeći način:

$$\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

Zadatak 2.2.

Koristeći programski jezik Python, kreirati i nacrtati jedinični Dirakov impuls $\delta[n-4]$ na vremenskoj osi u rasponu (0,10).

Rešenje:

```
In [73]: import numpy as np  
import matplotlib.pyplot as plt
```

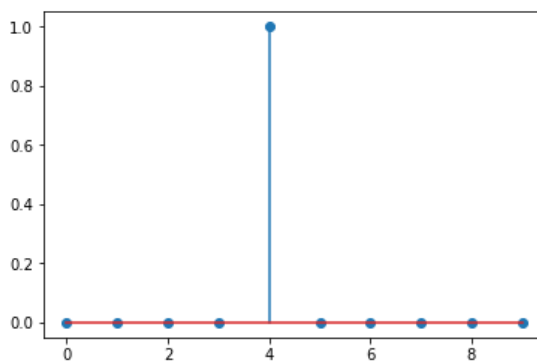
```
In [75]: x=np.zeros(10)
```

```
In [76]: x[4]=1
```

```
In [77]: plt.stem(x)
```

```
Out[77]: <Container object of 3 artists>
```

```
In [78]: plt.show()
```



Slika 34: Dirakov impuls $\delta[n-4]$

Zadatak 2.3.

Koristeći programski jezik Python, nacrtati funkciju $y[n] = \delta[n] + 3\delta[n-2]$ na vremenskoj osi u rasponu $(-10,10)$.

Rešenje:

Signal $y[n] = \delta[n] + 3\delta[n-2]$, u intervalu $n \in [-10,10]$ se sastoji od dve komponente, tj. dva dirakova impulsa amplituda 1 i 3 (slika 35)

```
In [13]: import numpy as np

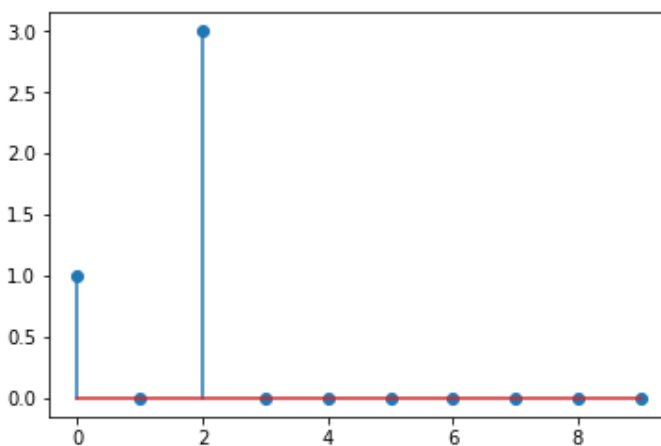
In [14]: import matplotlib.pyplot as plt

In [21]: n=np.arange(-10,10,1)

In [25]: y=np.zeros(10)

In [26]: y[0]=1
          y[2]=3

In [27]: plt.stem(y)
          plt.show()
```



Slika 35: Predstava signala $y[n] = \delta[n] + 3\delta[n-2]$

2.2.3. Kompleksni eksponencijalni signal

Kompleksni eksponencijalni signal $x[n]$, kontinualan u vremenu se može predstaviti u sledećoj formi

$$x[n] = Ca^{j\omega_0 n}$$

Zadatak 2.4.

Koristeći programski jezik Python, nacrtati eksponencijalnu funkciju za slučaj $C>0$ i $n>0$. Ovde je nacrtano i za n manje od nula.

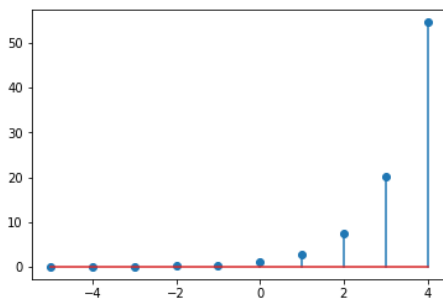
Rešenje:

```
In [137]: import numpy as np
import matplotlib.pyplot as plt

In [138]: n=np.arange(-5,5,1)

In [140]: plt.stem(n,np.exp(n)) #u ovom slučaju, crtamo funkciju y=np.exp(n) (C=1, n=1)
Out[140]: <Container object of 3 artists>

In [141]: plt.show()
```



Slika 36: Eksponencijalna funkcija

2.2.4. Stohastički signali

Naredba 'random.random_int(a,b,c)'

Ovaj modul implementira generatore pseudo-slučajnih brojeva za različite raspodele.

Za integers, uniforman je izbor iz opsega. Postoje funkcije za izračunavanje uniformnih, normalnih (Gaussian), logormalnih, negativnih eksponencijalnih, gama i beta distribucija.

Skoro sve funkcije modula zavise od osnovne funkcije slučajno (), koja generiše slučajni float ravnomerno u polu otvorenom opsegu [0.0, 1.0).

Zadatak 2.5.

Koristeći programski jezik Python, formirati slučajan signal $N=10$. tj. Signal čiji odbirci predstavljaju slučajno odabrane cele brojeve iz intervala $(-5,10)$. Koristiti naredbu, random.random_int(a,b,c)

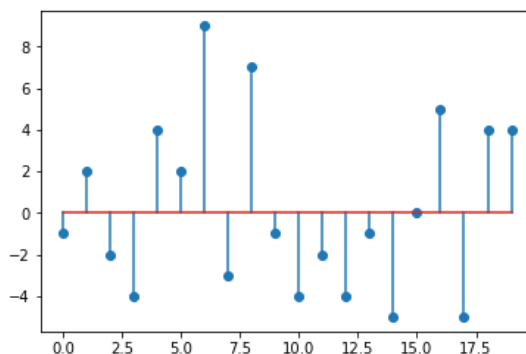
Rešenje:

```
In [28]: import numpy as np
```

```
In [30]: import matplotlib.pyplot as plt
```

```
In [33]: x=np.random.random_integers(-5,10,20)
```

```
In [34]: plt.stem(x)
plt.show()
```



Slika 37: Slučajan signal u intervalu od -5 do 10 (dužine 20)

Zadatak 2.6.

Koristeći programski jezik Python nacrtati diskretni signal Belog Gausovog Šuma-Additive White Gaussian Noise (AWGN) srednje vrednosti $\mu=0$ i standardne varijacije $\sigma^2=1$

Rešenje:

```
In [35]: import numpy as np
```

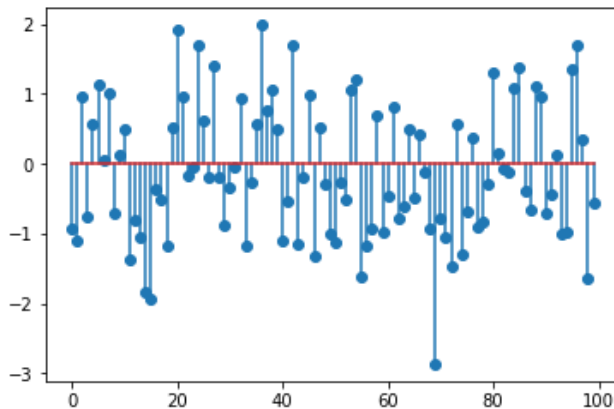
```
In [36]: import matplotlib.pyplot as plt
```

```
In [38]: noise = np.random.normal(0,1,100)
```

```
In [39]: plt.stem(noise)
```

```
Out[39]: <Container object of 3 artists>
```

```
In [40]: plt.show()
```



Slika 38: Slučajan signal (diskretan u vremenu) Belog Gausovog Šuma-Aditive White Gaussian Noise (AWGN) srednje vrednosti $\mu = 0$ i standardne varijacije $\sigma = 1$ (dužine 100)

2.2.5. Periodični signali

Za vremenski diskretan signal $x[n]$ se kaže da je *periodičan* ukoliko postoji pozitivna vrednost $N \neq 0$ za koju važi

$$x[n] = x[n \pm kN] \text{ za svako } k \in \mathbb{Z}$$

Najmanja pozitivna vrednost N za koju je uslov periodičnosti ispunjen naziva se osnovni period. Međutim, u slučaju konstantnog diskretnog signala, osnovni period nije definisan jer je konstantni signal periodičan za bilo koji izbor vrednosti N .

Zadatak 2.7.

Koristeći programski jezik Python, nacrtati diskretni sinusoidalni signal sa periodom odabiranja 0.001s čija je frekvencija 5Hz.

Rešenje:

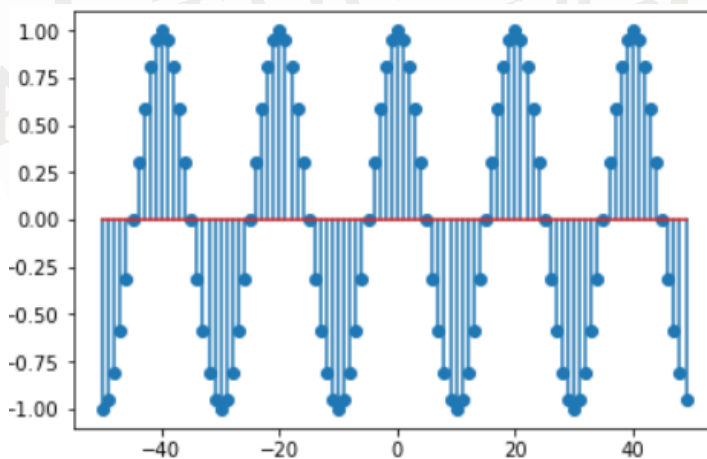
```
In [7]: import numpy as np
```

```
In [8]: import matplotlib.pyplot as plt
```

```
In [9]: t=np.arange(-50,50,1)
```

```
In [10]: x=np.cos(2*np.pi*t*0.05)
```

```
In [13]: plt.stem(t,x)  
plt.show()
```



Slika 39: Sinusoidalni signal sa periodom odabiranja 0.001s čija je frekvencija 5Hz

2.3. Transformacije u vremenu diskretnih signala

U ovoj sekciji, biće grafički i analitički predstavljeni diskretni signali koji su nastali nekom modifikacijom promenljive n , kao što su: pomeranje signala u vremenu, inverzija vremena, skaliranje vremena, simetričnost, periodičnost i konvolucija signala.

2.3.1. Pomeranje signala u vremenu

Kao i kod kontinualnih signala, mnoge operacije nad signalima se mogu predstaviti jednostavnom modifikacijom vremenske promenljive u analitičkim izrazima za signale. Već smo videli da signal $x[n-n_0]$ nije ništa drugo nego pomeranje signala $x[n]$ za n_0 odbiraka. Ukoliko je $n_0 > 0$, to je zapravo kašnjenje signala (pomeranje u desno), a ukoliko je $n_0 < 0$, to je takozvano prednjačenje signala (pomeranje u levo).

Zadatak za vežbu 2.8.

Koristeći programski jezik Python:

- nacrtati signal $x[n] = \begin{cases} 1 - e^{-5n}, & -5 \leq n < 1 \\ e^{-5(n-1)} [1 - e^{-5}], & n \geq 1 \end{cases}$
- zakasniti originalni signal za 3 odbirka
- uprednjačiti originalni signal za 2 odbirka

2.3.2. Inverzija vremena

Inverzija signala $x[n]$ u vremenu, analitički predstavljeno kao $x[-n]$, predstavlja “sliku u ogledalu” signala $x[n]$.

2.3.3. Skaliranje amplitude

Skaliranje signala po amplitudi se postiže tako što se amplituda signala pomnoži konstantom A , tako da se za signal diskretan u vremenu dobija $x[n] = Ax[n]$.

2.3.4. Skaliranje vremena

Skaliranje signala $x[n]$ u vremenu, analitički predstavljeno kao $x[an]$ predstavlja skaliranje signala $x[n]$ vremenskom konstantom a .

- za $a > 1$ funkcija se sužava duž vremenske ose (ubrzavanje)
- za $0 < a < 1$ funkcija se širi duž vremenske ose (usporavanje)
- za $a < 0$ funkcija se rotira oko y -ose (vreme "teče unazad")

Zadatak 2.9.

Nacrtati signal $x[n] = n$, a zatim nacrtati signal $x[3n]$

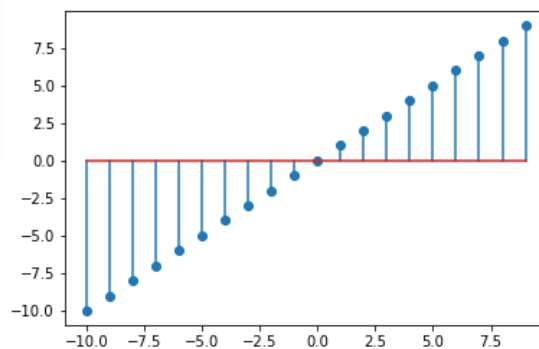
Rešenje:

```
In [187]: import numpy as np
import matplotlib.pyplot as plt

In [188]: n=np.arange(-10,10,1)
n1=3*n

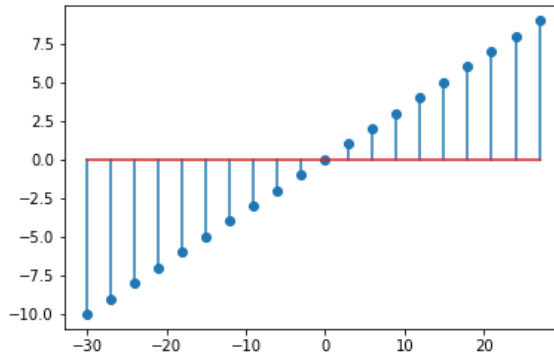
In [189]: x=n
y=n

In [190]: plt.stem(n,x)
plt.show()
```



Slika 40: Signal $x[n] = n$

```
In [191]: plt.stem(n1,y)
plt.show()
```



Slika 41: Ubrzani signal $x[3n]$

Poredeći dobijene slike, primećuje se da signal $x[3n]$ dostiže iste vrednosti na y osi za 3 puta kraće vreme.

2.3.5. Simetričnost signala

Signal $x[n]$ može biti *paran*, za koji važi $x[n] = x[-n]$ i *neparan*, za koji važi $x[n] = -x[-n]$. Ukoliko je signal paran, on je *simetričan* u odnosu na y osu, a ukoliko je neparan, simetičan je u odnosu na koordinatni početak.

Zadatak 2.10.

Koristeći programski jezik Python, nacrtati neparnu funkciju $x[n] = -x[n] = n$
i parnu funkciju

$$y[n] = y[-n] = |n|$$

Rešenje:

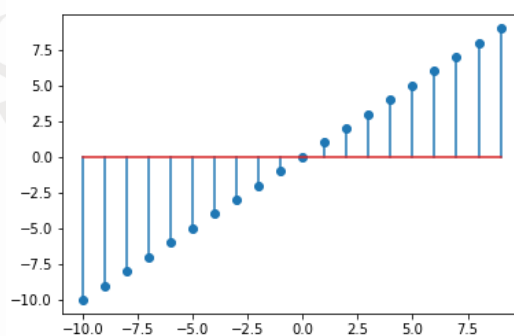
```
In [165]: import numpy as np
import matplotlib.pyplot as plt
```

```
In [166]: n=np.arange(-10,10,1)
```

```
In [167]: x=n
```

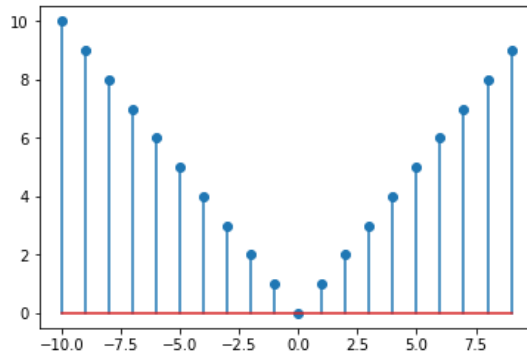
```
In [168]: y=np.abs(n)
```

```
In [170]: plt.stem(n,x)
plt.show()
```



Slika 42: Neparan signal $x[n] = n$

```
In [171]: plt.stem(n,y)
plt.show()
```



Slika 43: Paran signal $x[n] = |n|$

Važno je uočiti da se signal $x[n]$ može predstaviti pomoću svog parnog $x_p[n]$ i neparnog oblika $x_n[n]$, tj.

$$x[n] = x_p[n] + x_n[n]$$

Na osnovu osobina parnih i neparnih signala, važi da je

$$x[-n] = x_p[-n] + x_n[-n] = x_p[n] - x_n[n]$$

Dakle, parna i neparna komponenta signala $x[n]$ se određuje na sledeći način:

$$x_p[n] = \frac{x[n] + x[-n]}{2}$$

$$x_n[n] = \frac{x[n] - x[-n]}{2}$$

Zadatak 2.11.

Koristeći programski jezik Python:

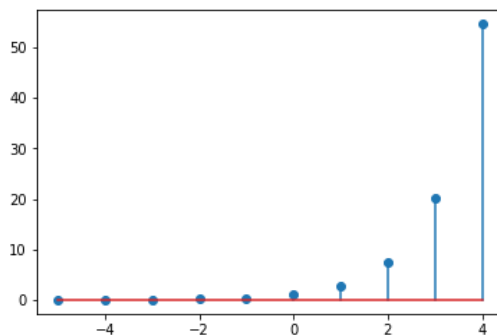
- a) nacrtati signal $x[n] = e^n$
- b) invertovati signal pod a)
- c) zakasniti originalni signal za 2 odbirka
- d) upednjačiti originalni signal za 2 odbirka

Rešenje:

a)

```
In [2]: import numpy as np  
import matplotlib.pyplot as plt
```

```
In [3]: n=np.arange(-5,5,1)
```

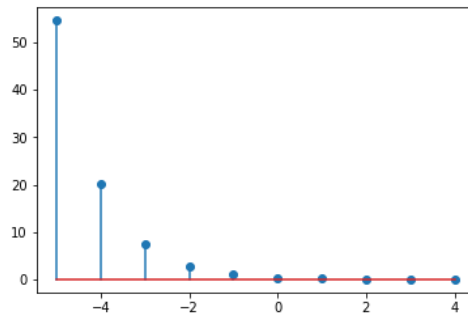


Slika 44: Signal $x[n]$

b)

```
In [6]: n_invert=n[::-1] #invrtovanje niza n
```

```
In [8]: plt.stem(n,np.exp(n_invert))
plt.show()
```

Slika 45: Signal $x[-n]$

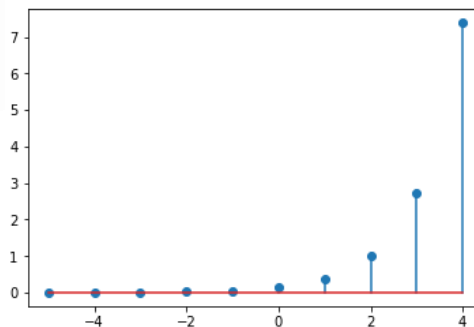
c)

```
In [20]: num=2 #pomeranje niza unazad num>0-kašnjenje, tj. time delay
```

```
In [21]: x=np.exp(n)
```

```
In [22]: x_shifted=np.roll(x,num)
x_shifted[0:num]=0
```

```
In [23]: plt.stem(n,x_shifted)
plt.show()
```

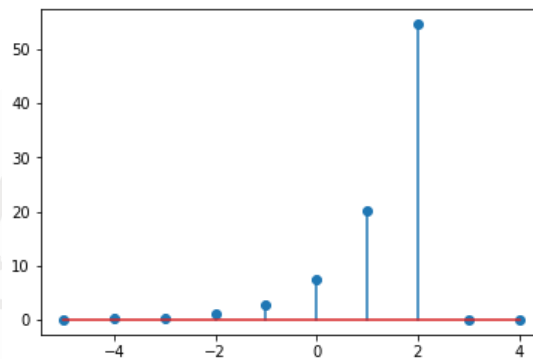
Slika 46: Signal $x[n-2]$

d)

```

In [24]: num=-2 #pomeranje niza unazad num>0-kašnjenje, tj. time delay
In [25]: x=np.exp(n)
In [26]: x_shifted=np.roll(x,num)
In [27]: plt.stem(n,x_shifted)
          plt.show()

```

Slika 47: Signal $x[n+2]$

2.3.6. Konvolucija diskretnih vremenskih funkcija

Konvolucija dve funkcije $x[n]$ i $h[n]$ analitički se označava kao $y[n]=x[n]*h[n]$, gde $x[n]$ predstavlja pobudu, a $y[n]$ odziv signala

Definicija konvolucije za diskretni signal, predstavljena je sledećim izrazom

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]h[n-k]$$

Ukoliko se uvede smena $m=n-k$, dobija se izraz

$$y[n] = - \sum_{\infty}^{-\infty} x[m-n]h[m] = \sum_{-\infty}^{\infty} h[m]x[m-n] = h[n] * x[n]$$

Što znači da za konvoluciju važe sledeće osobine:

Komutacija, tj:

$$x[n]*h[n]=h[n]*x[n]$$

Asocijacija

$$(x[n]*h[n])*g[n]=x[n]*(h[n]*g[n])$$

Distributivna je sa operacijom sabiranja signala

$$x[n]*(h_1[n]+h_2[n])=x[n]*h_1[n]+x[n]*h_2[n]$$

Konvolucije ima najveću primenu u telekomunikacijama i harmonijskim analizama. Integral se tumači kao suma zajedničkih odbiraka između proizvoda signala $x[n]$ i $h[n-k]$.

Rezultat se može dobiti analitički, rešavanjem integrala, i grafički, izračunavanjem površine ispod rezultujuće krive.

Koraci u konvoluciji

1. Signal $h[n]$ se prvo invertuje (slika 2.15) i pomeri u vremenu da bi se dobio signal $h[n-k]$ što postaje funkcija od k gde je n konkretan parametar
2. Signali $x[k]$ i $h[n-k]$ se izmnože za sve moguće vrednosti k za neko fiksno n
3. Proizvod $x[k] \cdot h[n-k]$ se sumira za sve vrednosti k i tako se dobija vrednost $y[n]$ za neko fiksno n .
4. Ponove se koraci 1, 2, 3 za sve vrednosti n iz skupa $(-\infty, \infty)$ kako bi se dobila kompletna funkcija $y[n]$.

Zadatak 2.12.

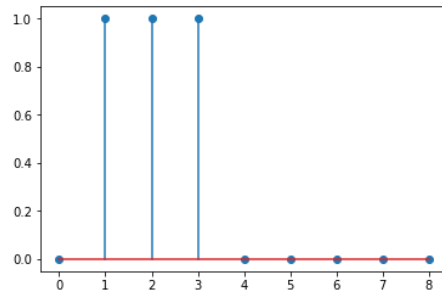
Koristeći programski jezik Python, izračunati konvoluciju dva pravougaona impulsa (signal $x[n]$ neka bude trajanja 3 odbirka, a signal h trajanja 2 odbirka).

Rešenje:

```
In [33]: import numpy as np  
import matplotlib.pyplot as plt
```

```
In [34]: x=np.array([0,1,1,1,0])
```

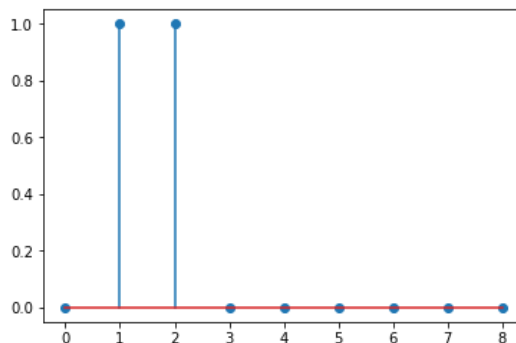
```
In [51]: plt.stem(x)  
plt.show()
```



Slika 48: Pravougaoni signal $x[n]$ trajanja 3 odbirka

```
In [37]: h=np.array([0, 1, 1,0])
```

```
In [52]: plt.stem(h)  
plt.show()
```



Slika 49: Pravougaoni signal $h[n]$ trajanja 2 odbirka

```

In [53]: lx=len(x)

In [54]: lh=len(h)

In [55]: zerosh=np.zeros(lh)

In [56]: x=np.concatenate((x, zerosh), axis=0)

In [57]: zerosx=np.zeros(lx)

In [58]: h=np.concatenate((h, zerosx), axis=0)

In [59]: len(x)
Out[59]: 18

In [60]: len(h)
Out[60]: 18

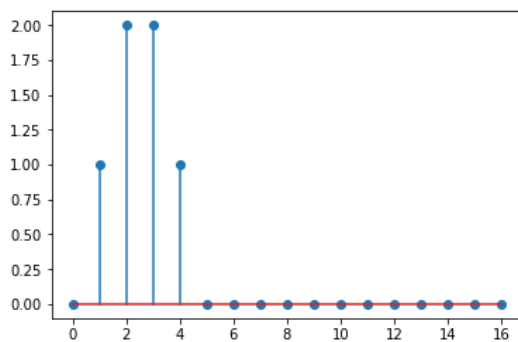
In [61]: y=np.zeros(lx+lh-1)

In [62]: # računanje konvolucije korak po korak
        for n in range(1, lx+lh-1):
            y[n]=0
            for k in range(1, lx):
                if (n-k+1>0):
                    y[n]=y[n]+x[k]*h[n-k+1]

In [63]: plt.stem(y)
Out[63]: <Container object of 3 artists>

In [64]: plt.show()

```



Slika 50: Signal $y[n] = x[n] * h[n]$

Zadatak 2.13.

Koristeći programski jezik Python i funkciju *convolve* za računanje konvolucije, nacrtati sledeće signale dužine $N=10$:

$x_1[n]$ -jedinični impuls

$x_2[n]$ -Hevisajdova funkcija, odskočni signal

$h[n] = e^{0.5n}$ -deo eksponencijalnog signala

Nacrtati signal $y_2 = (x_1 * x_2) * h$, gde '*' predstavlja konvoluciju signala.

Rešenje:

```
In [59]: import scipy as sp
         from scipy import signal
         import matplotlib.pyplot as plt
```

```
In [60]: n=sp.arange(-5,5,1)
```

```
In [61]: x1=np.zeros(10)
         x1[4]=1 #dirakov impuls
```

```
In [62]: x2=1.*(n>=0) #hevisajdova funkcija
```

```
In [63]: h=np.exp(0.5*n)
```

```
In [64]: len(x1)
```

```
Out[64]: 10
```

```
In [65]: len(x2)
```

```
Out[65]: 10
```

```
In [66]: len(h)
```

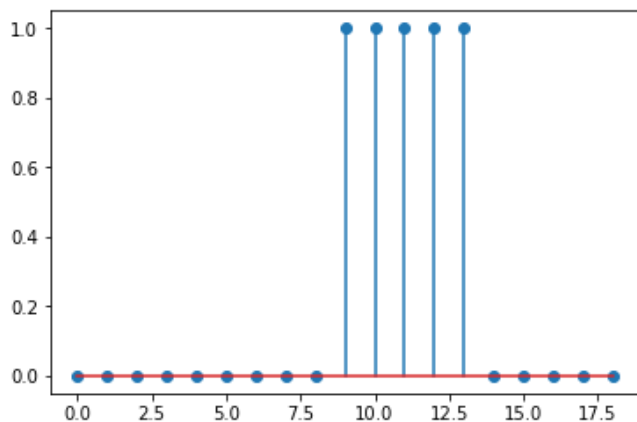
```
Out[66]: 10
```

```
In [67]: y1=signal.convolve(x1,x2)
```

```
In [68]: plt.stem(y1)
```

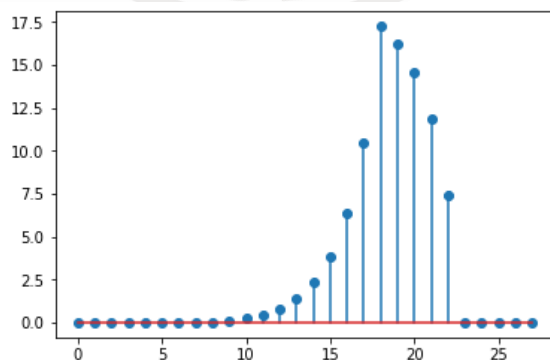
```
Out[68]: <Container object of 3 artists>
```

```
In [69]: plt.show()
```



```
In [70]: y2=signal.convolve(y1,h)
```

```
In [71]: plt.stem(y2)
plt.show()
```



Slika 51: Diskretan signal $y_2 = (x_1 * x_2) * h$ -konvolucija tri signala

3. Furijeova analiza kontinualnih signala

Francuski matematičar Josef Furije, početkom XIX veka dokazao je da se sve funkcije mogu predstaviti kao periodične. Ukoliko je funkcija $f(t)$ periodična, ona se može predstaviti kao beskonačna suma sinusnih i kosinusnih signala (Furijeov red). Takodje, dokazao je da ukoliko funkcija $f(t)$ nije periodična, moguće je predstaviti kao periodičan signal preko integrala u kome period teži beskonačnosti (Furijeova transformacija). Krajem druge polovine XIX veka, ova teorija nalazi primenu u teoriji obrade zvuka, slike i kvantne mehanike. Kao što smo do sada predstavljali signal vremenskom domenu, koristeći Furijeovu analizu, signal možemo predstavljati i u frekvencijskom domenu. Predstavljanje signala u frekvencijskom domenu nam omogućava interpretaciju mnogih važnih osobina signala koje ne možemo sagledati u vremenskom domenu. Opšti termin Furijeova analiza se odnosi na obradu:

- kontinualnih i periodičnih signala (Furijeov red)
- kontinualnih i aperiodičnih signala (Furijeova transformacija)
- diskretnih signala (Diskretna Furijeova transformacija) U ovom poglavlju biće opisana primena Furijeovih redova i Furijeove transformacije.

3.1. Furijeovi redovi

Kao što smo u poglavljima 1 i 2 definisali, funkcija $f(t)$ je periodična sa periodom $T > 0$ ukoliko je ispunjen sledeći uslov

$$x(t) = x(t \pm T) \text{ za svako } t$$

Predpostavka od koje je pošao Furije, bila je da se svaki periodičan signal može predstaviti kao zbir sinusoida i kosinusoida, odnosno može se predstaviti harmoničnim ili trigonometrijskim redom u formi koja je identična ili slična relaciji:

$$x(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk\omega_0 t}$$

a_k redsstavlja kompleksni koeficijent, a ω_0 osnovnu kružnu učestanost $\omega_0 = \frac{2\pi}{T}$.

Za vrednosti $k = \pm 1$, komponente Furijeovog reda nazivaju se osnovnim komponentama sa periodom

$$T_1 = \frac{2\pi}{\omega_0},$$

dok za vrednosti $|k| \geq 2$, komponente se nazivaju harmonicima sa periodom

$$T_k = \frac{2\pi}{|k|\omega_0}.$$

Dakle, ceo Furijeov red periodičan je sa periodom T_1 . Nakon izvodjenja, koje neće biti izvedeno u ovoj knjizi, dobijaju se kompleksni koeficijenti koji predstavljaju srednju vrednost signala u toku jedne njegove periode.

$$a_0 = \frac{1}{T} \int_T x(t) dt$$

$$a_k = \frac{1}{T} \int_T x(t) e^{-jk\omega_0 t} dt$$

Signal $x(t)$ i funkcija a_k se nazivaju *Furijeov transformacioni par*, a uslovi pod kojima ova jednakost važi, nazivaju se *Dirichlet-ovim uslovima*. Korišćenjem Euler-ove formule

$$e^{j\omega_0 t} = \cos \omega_0 t + j \sin \omega_0 t$$

Furije-ov red se može predstaviti i na drugi način

$$x(t) = \frac{a_0}{2} + \sum_{k=1}^{+\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] = \frac{a_0}{2} + \sum_{k=1}^{+\infty} c_k \cos(k\omega_0 t + \theta_k)$$

Gde je su a_k , b_k i c_k koeficijenti Furije-ovog reda.

$$a_k = \frac{2}{T} \int_{-T/2}^{T/2} x(t) \cos(k\omega_0 t) dt$$

$$b_k = \frac{2}{T} \int_{-T/2}^{T/2} x(t) \sin(k\omega_0 t) dt$$

$$c_k = \sqrt{a_k^2 + b_k^2}$$

Koeficijent c_k predstavlja amplitudu k -tog harmonika signala, a

$$\theta_k = \arctg\left(-\frac{b_k}{a_k}\right)$$

predstavlja njegovu fazu. U frekvencijskom domenu, signal je predstavljen pomoću njegovog *spektra*. Amplitude harmonika predstavljaju se njegovim *amplitudskim*, a faze se predstavljaju njegovim *faznim spektrom*. Ovi spektri su diskretni jer su definisani samo za diskretne vrednosti učestanosti.

3.2. Furijeova transformacija

Kao što smo videli u predhodnoj sekciji, svaki periodičan signal se može predstaviti zbirom beskonačno mnogo sinusnih i kosinusnih funkcija. Ukoliko se neperiodičan signal posmatra kao periodičan signal sa beskonačno velikim periodom, Furijeova transformacija se može primeniti i na neperiodične signale. Primenjujući ovu teoriju i na neperiodične signale, uvođenjem beskonačno velikog perioda, osnovna učestanost signala postaje beskonačno mala, a samim tim se i razlike između susednih učestanosti beskonačno smanjuju, tako da frekvencijski spektar postaje kontinualna funkcija učestanosti. Furijeova transformacija ovog signala, data je izrazom:

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

Dok je inverzni Furije-ov transformacioni par definisan je kao

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

Signal $x(t)$ i funkcija $X(j\omega)$ se nazivaju *Furijev transformacioni par*. $X(j\omega)$ predstavlja kompleksni spektar signala koji može da se predstavi kao funkcija svoje spektralne gustine amplituda $|X(j\omega)|$ i spektralne gustine faze $\theta(j\omega)$

$$X(j\omega) = |X(j\omega)|e^{-j\theta(j\omega)}$$

3.3. Diskretna Furijeova Transformacija

Diskretna Furijeova Transformacija (DFT) je ekvivalentna Furijeovoj transformaciji samo za diskretne signale $x[n]$

$$X(n) = \sum_{k=0}^{N-1} x[k]e^{-j\omega kT}$$

Ovaj izraz može da se izračuna za bilo koje ω , ali samo sa N odbiraka, od interesa je posma ω trati N odbiraka izlaza.

Zadatak 3.1.

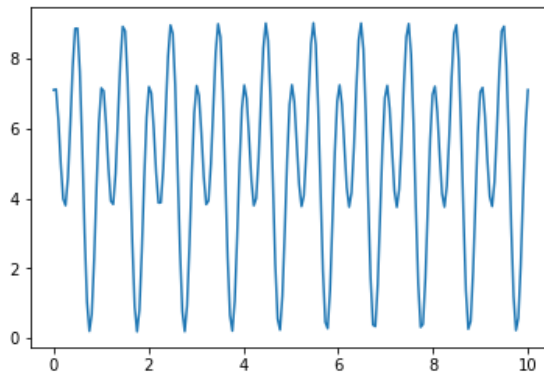
Ukoliko imamo signal $x(t) = \underbrace{5}_{dc} + \underbrace{2\cos(2\pi t - 90^\circ)}_{1\text{Hz}} + \underbrace{3\cos 4\pi t}_{2\text{Hz}}$

```
In [49]: import numpy as np
import matplotlib.pyplot as plt

In [50]: t=np.linspace(0,10,200)

In [51]: x=5+2*np.cos(2*np.pi*t-90)+3*np.cos(4*np.pi*t)

In [52]: plt.plot(t,x)
plt.show()
```



Slika 52: Kontinualan signal u vremenskom domenu

Ako ovaj signal odabiramo 4 puta u sekundi ($F_s = 4\text{Hz}$) od $t=0$ do $t = \frac{3}{4}$.

Odbirci diskretnog signala se mogu predstaviti kao

$$x[k] = 5 + 2 \cos\left(\frac{\pi}{2}k - 90^\circ\right) + 3 \cos \pi k \text{ stavljajući da je } t = kT_s = \frac{k}{4}$$

Tj. $x[0]=8$, $x[1]=4$, $x[2]=8$, $x[3]=0$, $N=4$

Tada je

$$X[n] = \sum_0^3 x[k] e^{-j\frac{\pi}{2}nk} = \sum_{k=0}^3 x[k] e^{-j\frac{\pi}{2}nk} = \sum_{k=0}^3 x[k] (-j)^{nk}$$

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & W & W^2 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W^{N-1} & W^{N-2} & \dots & W \end{bmatrix} \begin{bmatrix} 0 \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}$$

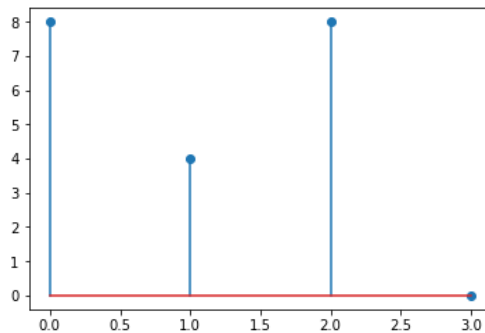
Gde je $W = e^{-j2\frac{\pi}{N}}$

```
In [134]: import scipy as sp
import matplotlib.pyplot as plt
```

```
In [135]: k=sp.arange(0,4,1)
```

```
In [136]: x=5+2*sp.cos(0.5*np.pi*k-0.5*sp.pi)+3*np.cos(sp.pi*k)
```

```
In [137]: plt.stem(k,x)
plt.show()
```



Slika 53: Odabrani signal puta u sekundi ($F_s = 4\text{Hz}$) od $t=0$ do $t = \frac{3}{4}$.

U nastavku računamo DFT

```
In [138]: X=sp.fft(x)

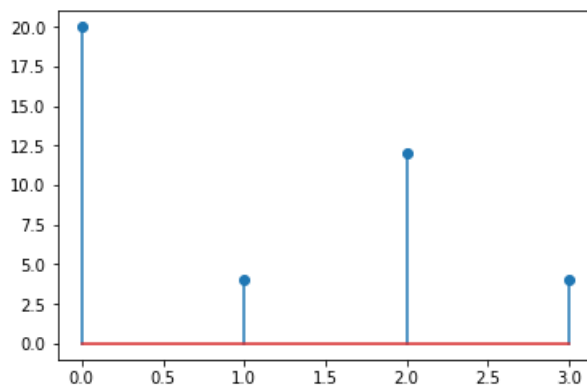
In [139]: X
Out[139]: array([ 20.+0.j,   0.-4.j,  12.+0.j,   0.+4.j])

In [140]: F=sp.linspace(0,Fs,len(X)+1)

In [141]: F=F[:-1]

In [142]: N=len(X)
           plt.stem(F[0:N],sp.absolute(X[0:N]))
Out[142]: <Container object of 3 artists>

In [143]: plt.show()
```



Slika 54: DFT odabranog signala (4 odbirka)

U ovom konkretnom slučaju, dobili smo da su amplitude koeficijentat DFT

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ x[1] \\ \cdot \\ \cdot \\ x[N-1] \end{bmatrix}$$

3.4. Inverzna Diskretna Furijeova Transformacija

Inverzna Diskretna Furijeova transformacija od

$$X[n] = \sum_{k=0}^{N-1} x[k] e^{-j\frac{2\pi}{N}nk}$$

, može se napisati u sledećoj formi

$$x[k] = \frac{1}{N} \sum_{n=0}^{N-1} X[n] e^{+j\frac{2\pi}{N}nk}$$

Inverzna matrica je $\frac{1}{N}$ puta konjugovano kompleksna (simetrična)

matrica. Primetimo da su koeficijenti $x[n]$ kompleksni. Možemo pretpostaviti da su vrednosti $x[k]$ realne. U procesu računanja inverzne transformacije preko $X[n]$ i $X[N-n]$ (spektar je simetričan za $N/2$), može se zapisati

$$x_n[k] = \frac{1}{N} \left\{ X[n] e^{j\frac{2\pi}{N}nk} + X[N-n] e^{j\frac{2\pi}{N}(N-n)k} \right\}$$

Za sve realne vrednosti $x[k]$,

$$X[N-n] = \sum_{k=0}^{N-1} x[k] e^{-j\frac{2\pi}{N}(N-n)k}$$

Ali

$$e^{-j\frac{2\pi}{N}(N-n)k} = \underbrace{e^{-j2\pi k}}_{1 \quad \forall k} e^{\frac{j2\pi n}{N}k} = e^{+j\frac{2\pi}{N}nk}$$

tj.

$X[N-n] = X^*[n]$ su konjugovano kompleksne vrednosti
Menjajući date vrednosti u $x_n[k]$, dobijamo

$$x_n[k] = \frac{1}{N} \left\{ X[n] e^{j\frac{2\pi}{N}nk} + X^*[n] e^{-j\frac{2\pi}{N}nk} \right\}$$

Kako je $e^{j2\pi k} = 1$, dobija se da je

$$x_n[k] = \frac{2}{N} \left\{ \operatorname{Re}\{X[n]\} \cos \frac{2\pi}{N}nk - \operatorname{Im}\{X[n]\} \sin \frac{2\pi}{N}nk \right\} = \frac{2}{N} |X[n]| \cos \left\{ \left(\frac{2\pi}{N}n \right) kT + \arg(F[n]) \right\}$$

Za slučaj $n=0$, $x[0] = \sum x[k]$ predstavlja sumu odbiraka.

```

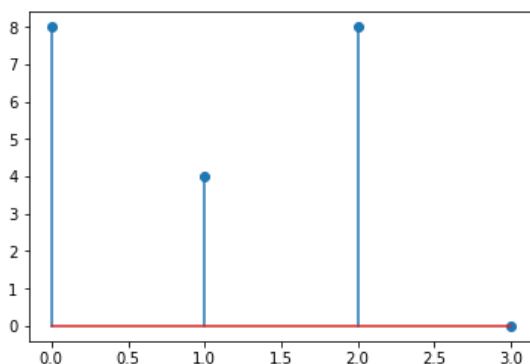
In [144]: k=sp.arange(0,4,1)

In [145]: x=sp.ifft(X)

In [146]: plt.stem(k,x)
plt.show()

```

C:\Users\mmarjanovic\Anaconda3\lib\site-packages\numpy\fft\real.py:152: RuntimeWarning: real discards the imaginary part
 return array(a, dtype, copy=False, order=order)



Slika 55: Rekonstruisani signal nakon IDFT (4 odbirka)

Spektralna Gustina Srednje Snage Signala (SGSS) diskretnog u vremenu, računa se po formuli

$$\frac{1}{N} \left| \sum_{n=1}^N x[n] e^{-j \frac{2\pi k n}{N}} \right|^2$$

Zadatak 3.2.

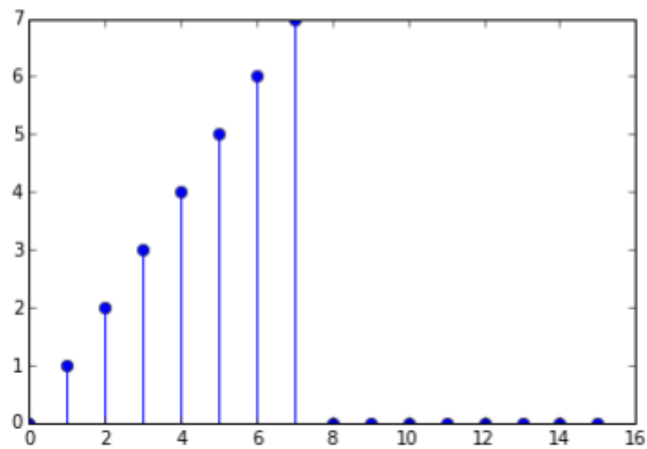
Izračunati DFT i nacrtati spektar signala $x[n] = n \cdot (u[n] - u[n-7])$

```
In [7]: y=sp.zeros(16)
```

```
In [8]: y[0:len(x)]=x
```

```
In [9]: plt.stem(y)
```

```
Out[9]: <Container object of 3 artists>
```



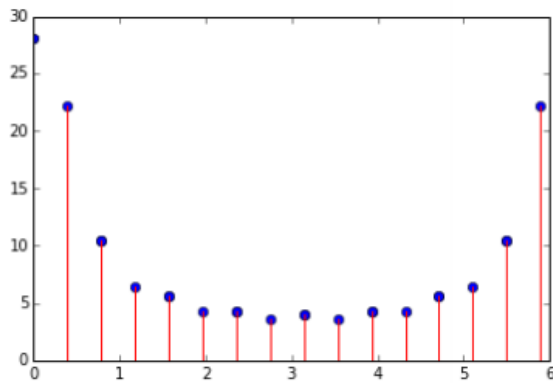
Slika 56: Signal $x[n] = n \cdot (u[n] - u[n-7])$

```

In [10]: plt.clf
X = sp.fft(x)
w = sp.linspace(0,sp.pi*2, len(x)+1)
w=w[:-1]
plt.stem(w,sp.absolute(X))
Y = sp.fft(y)
w = sp.linspace(0,sp.pi*2, len(y) +1 )
w = w[:-1]
plt.stem(w, sp.absolute(Y),'r')

```

Out[10]: <Container object of 3 artists>



Slika 57: Spektralna Gustina Snage Signala $x[n] = n \cdot (u[n] - u[n-7])$

Zadatak 3.3.

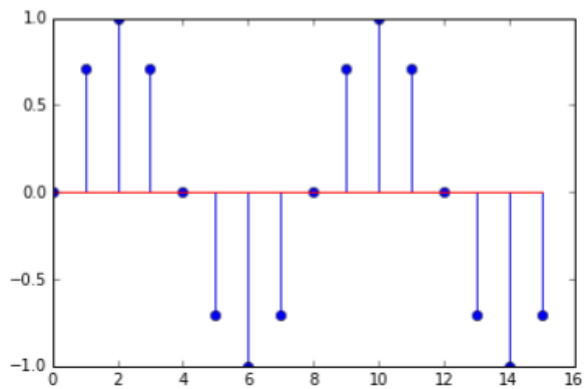
Neka je dat signal

$$x[n] = \sin(2\pi 0.125n)$$

Nacrtati njegov spektar

```
In [11]: n=sp.arange(16)
x = sin (2 * sp.pi * n * 0.125)
# dodati DC komponentu
#x = sin (2 * sp.pi * n * 0.125) +1
plt.stem(n,x)
```

Out[11]: <Container object of 3 artists>

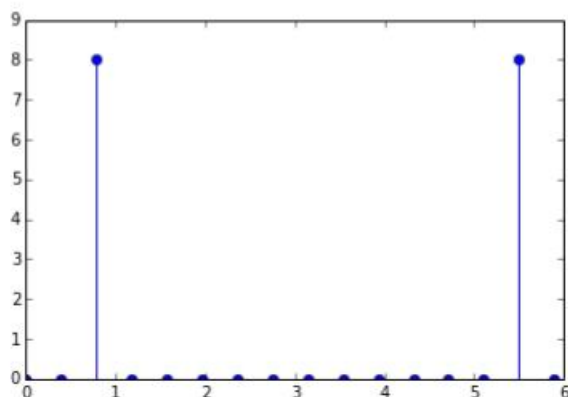


Slika 58: Signal $x[n] = \sin(2\pi 0.125n)$

```
In [13]: X = sp.fft(x)
```

```
In [14]: plt.clf
w = sp.linspace(0,sp.pi*2, len(x)+1)
w=w[:-1]
plt.stem(w,sp.absolute(X))
```

```
Out[14]: <Container object of 3 artists>
```



Slika 59: Spektar signala $x[n] = \sin(2\pi 0.125n)$

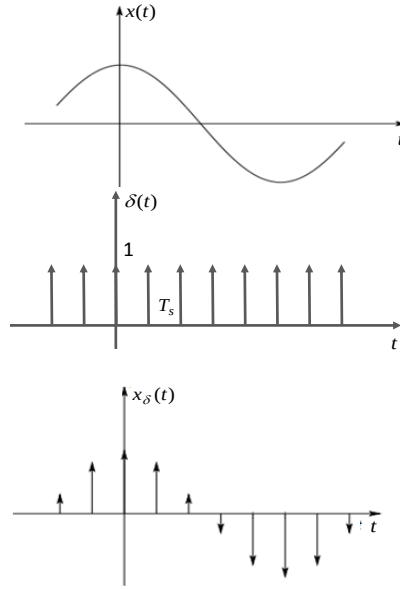
3.5. Teorema odabiranja

Osnovna operacija za formiranje sistema za modulaciju impulsa je proces odabiranja gde se analogni signal kovertuje u sekvencu brojeva koja je obično odabrana u jednakim vremenskim intervalima. Kako bi signal $x_\delta(t)$ nastao odabiranjem analognog signala $x(t)$ pravilno prenosio informacije o datom analognom signalu, potrebno je pravilno odrediti učestanost odabiranja

$f_s = \frac{1}{T_s}$, gde je T_s period odabiranja.

$$x_\delta(t) = \sum_{n=-\infty}^{+\infty} x(nT_s) \delta(t - nT_s) = x(t) \sum_{n=-\infty}^{+\infty} \delta(t - nT_s), \text{ gde je } x(nT_s) \text{ idealno}$$

odabrani signal. Na slici 60, prikazana je ilustracija odabiranja signala u vremenskom domenu.



Slika 60: Ilustracija odabiranja signala u vremenskom domenu

S obzirom da se signali $x(t)$ i $\delta(t)$ množe, koristeći se osobinama Furijeove transformacije, za frekvencijski domen ovih signala važe sledeće jednakosti:

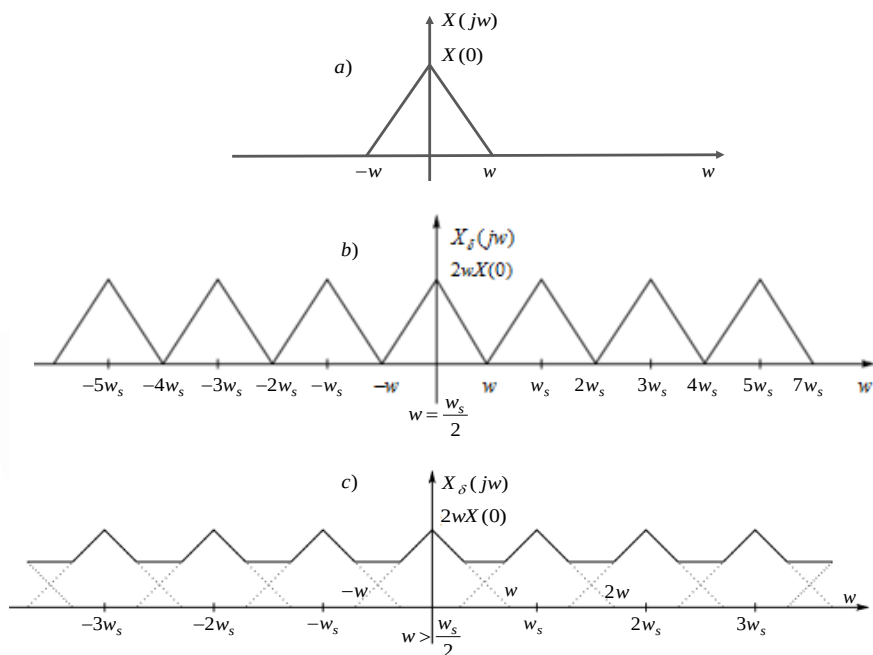
$$\delta_{T_s}(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \Leftrightarrow \frac{1}{T_s} \sum_{n=-\infty}^{+\infty} \delta(j\omega - \frac{n}{T_s})$$

$$X_\delta(j\omega) = X(j\omega) * \frac{1}{T_s} \sum_{n=-\infty}^{+\infty} \delta(j\omega - \frac{n}{T_s}) = \frac{1}{T_s} \sum_{n=-\infty}^{+\infty} X(j\omega) * \delta(j\omega - \frac{n}{T_s})$$

Poslednja relacija pokazuje da se Furijeova transformacija $X_\delta(j\omega)$ sastoji od periodičnog ponavljanja signala $X(j\omega)$ centriranog oko celobrojnog multipla učestanosti odabiranja ω_s . Izborom učestanosti u slučaju pod a) $\omega_s = 2\omega$, omogućeno je da ne dodje do gubitka informacija originalnog signala, dok u slučaju b) $\omega_s = 2\omega$, dolazi do efekta preklapanja (eng. *aliasing effect*). U slučaju efekta preklapanja, spektralni sadržaj signala $x_\delta(t)$, nastao odabiranjem signala $x(t)$ je različit, što za posledicu ima gubljenje informacija originalnog signala. Da bi se ova negativna pojava izbegla, mora biti ispunjen Nyquist-ov

kriterijum, tj. mora biti zadovoljena *teorema o odabiranju*, koja tvrdi da se prilikom odabiranja kontinualnog signala odabiranje mora vršiti učestanošću odabiranja ω_s koja je bar dva puta veća od maksimalne učestanosti u spektru signala koji se odabira, tj.

$$\omega_s \geq 2\omega.$$



Slika 61: Ilustracija odabiranja signala u frekvencijskom domenu

Zadatak 3.4.

U programskom jeziku Python, ponoviti proces odabiranja diskretnog sinusoidalnog signala za različite učestanosti sinusoidalnog signal

$$x(t) = \sin(2\pi kt)$$

držeći konstantnu učestanosti odabiranja i analizirati vizuelno oblike signala za različite osnovne učestanosti sinusoidalnog signala:

Rešenje:

U ovom zadatku fiksiraćemo učestanost odabiranja signala, a menjaćemo osnovnu učestanost sinusoidalne funkcije

$$x(t) = \sin(2\pi kt)$$

k "ciklusa" u jednoj sekundi, odgovara frekvenciji od k Hz
Kako odabrati periodu odabiranja?

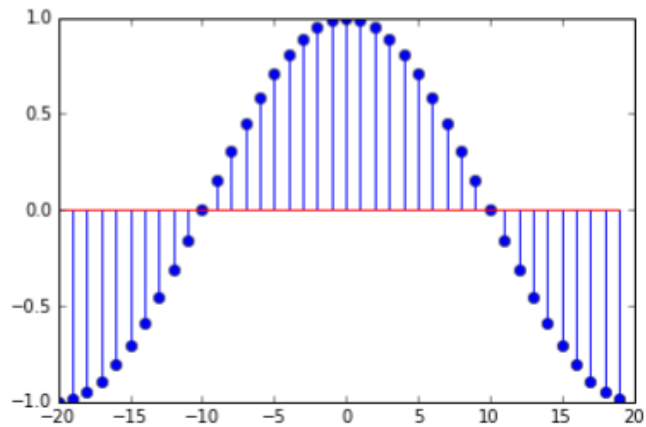
$$T_s < \frac{1}{2f_g} = \frac{1}{(2 \times k) \text{ Hz}}$$

Uzimajući u obzir da je

$\omega_s = 2\pi / N$, gde je N fundamentalni period, dobijamo sledeće rešenje:

```
In [36]: # Brze i spore diskretne sinusoide  
N=40  
n = sp.arange(-20,20,1)  
k=1  
x = sp.cos(2* pi* k / N * n)  
plt.stem(n,x)
```

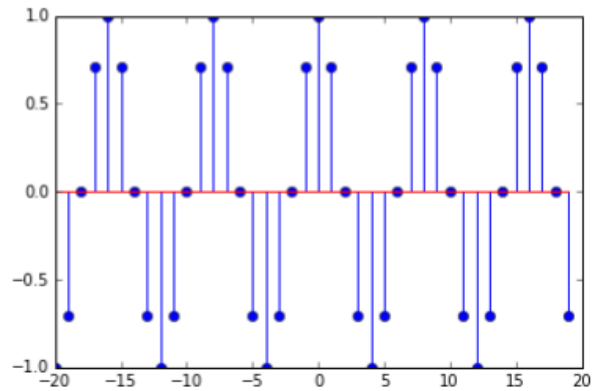
Out[36]: <Container object of 3 artists>



Slika 62: Osnovna učestanost signala je 1Hz, a učestanost odabiranja
 $\omega_s = 2\pi / 40$

```
In [37]: k=5
x = sp.cos(2* pi* k / N * n)
plt.stem(n,x)

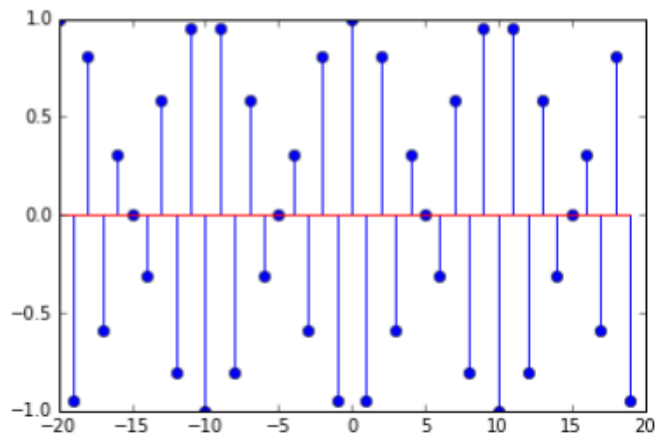
Out[37]: <Container object of 3 artists>
```



Slika 63: Osnovna učestanost signala je 5Hz, a učestanost odabiranja
 $\omega_s = 2\pi / 40$

```
In [39]: k=18
x = sp.cos(2* pi* k / N * n)
plt.stem(n,x)

Out[39]: <Container object of 3 artists>
```



Slika 64: Osnovna učestanost signala je 18Hz, a učestanost odabiranja
 $\omega_s = 2\pi / 40$

Zadatak 3.5.

Neka je dat signal

$$x(t) = \cos(2\pi 2t) + \cos(2\pi 11t)$$

Izračunati Diskretnu Furijeovu Transformaciju za učestanost odabiranja, uzeti da je $F_s = 100\text{Hz}$

- Iscrtati njegov spektar
- Iscrtati samo levu polovinu spektra
- Signalu $x(i)$ dodati jednosmernu komponentu i ponoviti postupak

Rešenje:

a)

```
In [1]: import scipy as sp
        %matplotlib inline
        import matplotlib.pyplot as plt
```

```
In [2]: #Fs - Sampling frequency ; frekvencija odabiranja
        Fs = 100

        #obratiti paznju na 1. Zasto koristimo tacku ?
        t = sp.arange(0,2,1./Fs)
```

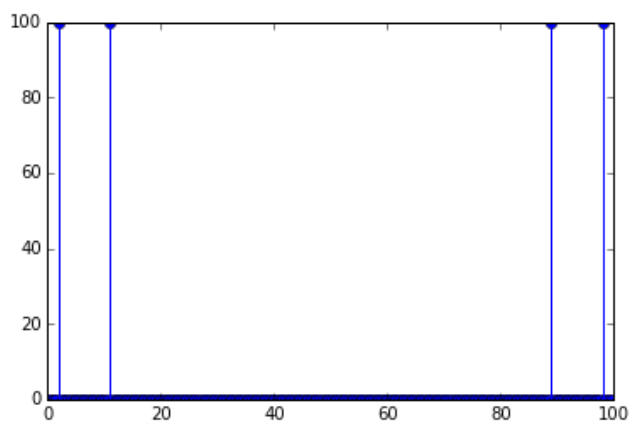
```
In [3]: # Definisanje signala x
        x = sp.cos(2*sp.pi*t*2) + sp.cos(2*sp.pi*11*t)
```

```
In [4]: # Primena FFT - Brza Furijeova transformacija
        X = sp.fft(x)
```

```
In [5]: # Kreiranje frekvencijske ose.
        F = sp.linspace(0,Fs, len(X)+1)
        F = F[:-1]
```

```
In [6]: plt.stem(F, sp.absolute(X))
```

```
Out[6]: <Container object of 3 artists>
```

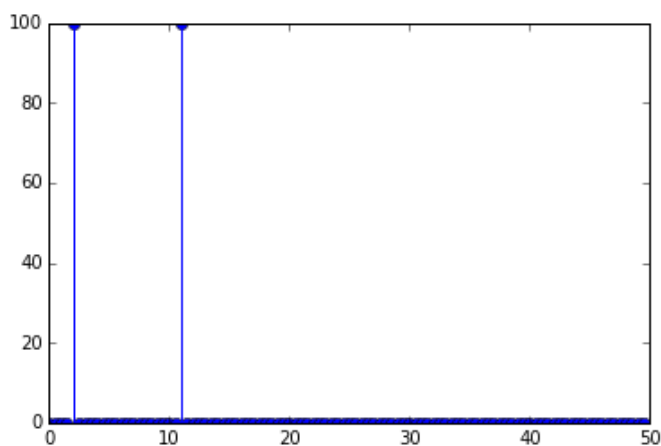


b)

```
In [9]: N = len(X)
```

```
In [10]: # Iscrtavanje samo prvog dela spektra  
plt.stem(F[0:N/2], sp.absolute(X[0:N/2]))
```

```
Out[10]: <Container object of 3 artists>
```



SIGNALI I SISTEMI

c)

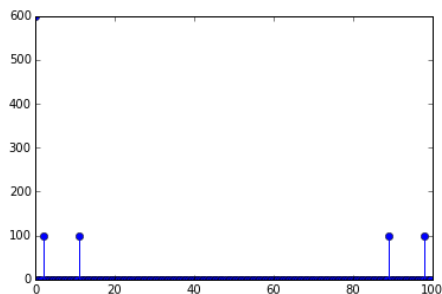
```
In [11]: # Definisanje signala koji ima DC komponentu
```

```
x1 = 3+ sp.cos(2*sp.pi*t*2) + sp.cos(2*sp.pi*11*t)
```

```
In [12]: X1 = sp.fft(x1)
```

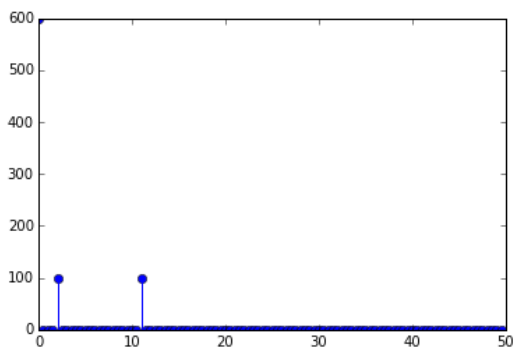
```
In [13]: # Iscrtavanje spektra. Obratite paznju na vrednost za frekvencijsku vrednost 0 Hz.  
plt.stem(F, sp.absolute(X1))
```

```
Out[13]: <Container object of 3 artists>
```



```
In [14]: # Iscrtavanje korisnog dela spektra.  
# Praksa je da se duzina signala uzima uvek kao paran broj,  
# a veoma cesto i kao stepen broja 2.  
# Npr. duzina signala u nasem slucaju je 200.  
# Bolje bi bilo da smo izabrali takav opseg da je duzina signala 256.  
plt.stem(F[0:N/2], sp.absolute(X1[0:N/2]))
```

```
Out[14]: <Container object of 3 artists>
```



Zadatak 3.6.

Nacrtati vremenski izgled impulsnog odziva kola zadržke nultog reda.

Za periodu odabiranja uzeti da je $0.25s$ (T), a vremenska osa t u rangu od 0 do $2s$.

Zadatak 3.7.

Nacrtati sledeće diskretne signale i izračunati diskretnu Furijeovu transformaciju:

1. Dirakov impuls
2. Hevisajdovu funkciju
3. Puls širine 4, 8 i 12 odbiraka
4. Kosinusnu funkciju za 4 proizvoljne vrednosti frekvencijske učestanosti
5. Za signale generisane pod 4. dodati i "konstantnu" vrednost (DC) komponentu. Uporediti razlike sa signalima amplitudskih spektara dobijenih u 4.

U svim slučajevima uzeti da je dužina signala $N=32$.

Zadatak 3.8. (nastavak):

Za crtanje Diskretne Furijeove transformacije koristiti naredbe:

```
import scipy as sp
sp.fft
sp.absolute-za iscrtavanje apsolutne vrednosti kompleksnog broja
```

Zadatak 3.9.

Neka je dat signal

$$A_1 \sin(2\pi 33t) + A_2 \sin(2\pi 61t)$$

- Generisati odgovarajuću vremensku osu za ovaj signal u skladu sa Teoremom odabiranja. Dužina vremenske ose neka bude 1024 (2048) odbirak, s početkom u 0.
- Nacrtati amplitudski spektar signala, za vrednosti kombinacija A_1 i A_2 iz skupa $\{0.5, 3, 7\}$. Npr. $A_1=3$, a $A_2=7$
- Posmatrati razlike u amplitudskom spektru signala.
- Generisati adekvatnu frekventijsku osu.

Zadatak 3.10. (nastavak)

Za skiciranje frekvencijske amplitude, koristiti sledeće naredbe:

```
import scipy as sp
 $\omega = \text{sp.linspace}(0, 2 * \text{sp.pi}, 1024) * F_s$ 
 $f = \text{sp.linspace}(0, 1, 1024) * F_s$ 
```

Uraditi isti zadatak za primer kada F_s nije odabrana prema Teoremi odabiranja i posmatrajte ponašanje spektra. Da li se "pikovi" pojavljuju na istim frekvencijama?

4. Laplasova transformacija

Laplasova transformacija je metoda rešavanja linearnih diferencijalnih jednačina. Postupak se sastoji od nekoliko koraka. U prvom koraku, diferencijalna jednačina, odnosno funkcija, transformiše se u algebarsku jednačinu. Transformacija neke funkcije može da se izvrši tako što se izvrši promena jedne promenljive u funkciji da bi se pojednostavilo rešenje problema. Drugim rečima, ako postoji problem promenljive x , promenljiva x se zameni drugim izrazom, npr. $x = \sin y$. Tako dobijena jednačina se reši, a u sledećem koraku rešenje se transformiše u traženo rešenje originalne diferencijalne jednačine. Uz pomoć Laplasove transformacije, veličine koje su funkcije vremena t , preslikavaju se u nove veličine, koje su funkcije kompleksne promenljive $s = \sigma + j\omega$. Na taj način, vremenskoj funkciji $x(t)$ se pridružuje odgovarajuća funkcija $X(s)$. Primenom Laplasove transformacije, realna oblast se preslikava u matematički izvedenu Laplasovu oblast, u kojoj određene računске operacije iz realnog područja poprimaju jednostavan oblik, pa zadatak postaje prikladniji za istraživanje i lakše se dolazi do njegovog rešenja. Laplasova transformacija važi samo za kontinualne funkcije, dok za diskretne funkcije važi Z transformacija o kojoj će biti reči u narednom poglavlju.

4.1. Definicija Laplasove teoreme

Neka je data funkcija $x(t)$, definisana za pozitivne vrednosti parametra vremena t . Množenjem funkcije $x(t)$ sa e^{-st} i integraljenjem s ugranicama od nula do beskonačnosti, dobija se relacija za $X(s)$

$$X(s) = \int_0^{+\infty} x(t) e^{-st} dt$$

U literaturi, za Laplasovu transformaciju, koristi izraz

$$\mathcal{L}(x(t)) = X(s)$$

Oblast definisanosti funkcije $X(s)$ je skup vrednosti $s \in C$ za koje izraz $\int_0^{+\infty} e^{-st} x(t) dt$ konvergira.

4.2. Inverzna Laplasova transformacija

Ukoliko se prine Inverzna Laplasova transformacija na signal $X(s)$, dobija se signal $x(t)$. Postoje nekoliko načina da se dobije inverzna Laplasova transformacija

1. Formula za inverziju

Polazeći od činjenice da je Laplasova transformacija u suštini Furijeova transformacija signala $x(t)e^{-\sigma t}$, najlakše se dobija izraz za inverznu Laplas-ovu transformaciju. Dakle, ako je analitička relacija Furijeove transformacije data izrazom:

$$X(s) = X(\sigma + j\omega) = \int_{-\infty}^{+\infty} [x(t)e^{-\sigma t}] e^{-j\omega t} dt$$

Tada se sintetička relacija može napisati na sledeći način:

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

Ako se leva i desna strana poslednje jednakosti pomnoži sa $e^{\sigma t}$, dobija se da je inverzna Laplasova transformacija:

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

Koristeći smenu $s = \sigma + j\omega$, $ds = j d\omega$, dobija se konačni izraz za $x(t)$:

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

Data relacija se naziva inverznom Laplas-ovom transformacijom, u literaturi označena kao $x(t) = \mathcal{L}^{-1}(X(s))$. Ova relacija nam omogućava da se originalan signal u vremenskom domenu može rekonstruisati ukoliko nam je poznata njegova Laplasova transformacija. Rešavanje ovog integrala često može biti složen postupak, tako da se prihvata sledeći manje kompleksan način. Ukoliko signal $X(s)$ postoji, tada se integral za računanje inverzne Laplasove transformacije, računa po pravoj $\sigma = \text{const.}$ koja mora pripadati oblasti konvergencije funkcije $X(s)$ gde $\sigma_1 < \text{Re}(s) < \sigma_2$.

2. Korišćenje tabele za inverznu Laplas-ovu transformaciju

Za često korišćene signale, da ne bismo morali svaki put da koristimo definicione integrale za izračunavanje Laplasove transformacije, na osnovu navedenih osobina, dobija se sledeća tabela.

Signal $x(t)$	Laplasova transformacija $X(s)$	Oblast konvergencije
$\delta(t)$	1	Svako s
$u(t)$	$\frac{1}{s}$	$\text{Re}(s) > 0$
$-u(-t)$	$\frac{1}{s}$	$\text{Re}(s) < 0$
$e^{-at}u(t)$	$\frac{1}{s+a}$	$\text{Re}(s) > -a$
$e^{-at}u(-t)$	$\frac{1}{s+a}$	$\text{Re}(s) < -a$
$te^{-at}u(t)$	$\frac{1}{(s+a)^2}$	$\text{Re}(s) > -a$
$\sin(\omega_0 t)u(t)$	$\frac{\omega_0^2}{s^2 + \omega_0^2}$	$\text{Re}(s) > 0$
$\cos(\omega_0 t)u(t)$	$\frac{\omega_0^2}{s^2 + \omega_0^2}$	$\text{Re}(s) > 0$
$e^{-at} \sin(\omega_0 t)u(t)$	$\frac{\omega_0^2}{(s+a)^2 + \omega_0^2}$	$\text{Re}(s) > -a$
$e^{-at} \cos(\omega_0 t)u(t)$	$\frac{\omega_0^2}{(s+a)^2 + \omega_0^2}$	$\text{Re}(s) > -a$

Tabela 1. Laplasova transformacija često korišćenih signala

Na osnovu Tabele 1, interesantno je primetiti da određeni signali iako su različiti imaju iste Laplasove transformacije, a razlikuju se samo po oblasti konvergencije. Takođe, može se primetiti da za kauzalne signale ili ako postoji neko t_0 takvo da je za $t < t_0$, $x(t)=0$, tada njihova oblast konvergencije ima formu $\text{Re}(s) > \text{const.}$, dok za nekauzalne sisteme, oblast konvergencije ima formu $\text{Re}(s) < \text{const.}$

Da bi se primenio ovaj način za izračunavanje inverzne Laplasove transformacije, potrebno je Laplasovu transformaciju podeliti na sabirke

$$X(s) = X_1(s) + X_2(s) + \dots + X_n(s),$$

Gde su

$$X_1(s), X_2(s) \dots X_n(s) \text{ Laplasove transformacije članova } x_1(t), x_2(t), \dots x_n(t).$$

Iz osobine linearnosti Laplasove transformacije važi

$$x(t) = x_1(t) + x_2(t) + \dots + x_n(t)$$

Primer 1:

Dati su signali $X(s)$, želimo da izračunamo inverznu Laplas-ovu transformaciju:

$$a) X(s) = \frac{1}{s+1}, \text{Re}(s) > -1$$

Na osnovu Tabele 1

$$x(t) = e^{-t}u(t)$$

$$b) X(s) = \frac{1}{s+1}, \text{Re}(s) < -1$$

Na osnovu tabele 1 dobija se da je

$$x(t) = -e^{-t}u(-t)$$

Zadatak 4.1.

Koristeći programski jezik Python, izračunati Laplace-ovu transformaciju sledećih funkcija:

a) $H(s) = \frac{s}{s^2 + 1}$

b) $H(s) = \frac{2}{(s+1)(s+2)}$

Rešenje:

a)

```
In [4]: from sympy import *
In [5]: init_printing()
In [6]: s,t=symbols('s,t')
In [7]: Laplace_function=s/(s**2+1)
In [9]: inverse_laplace_transform(Laplace_function,s,t)
Out[9]: cos(t)θ(t)
```

b)

```
In [10]: from sympy import *
In [11]: init_printing()
In [12]: s,t=symbols('s,t')
In [14]: Laplace_function=2/(s**2+3*s+2)
In [15]: inverse_laplace_transform(Laplace_function,s,t)
Out[15]: 2(et - 1)e-2tθ(t)
```

Zadatak 4.2.

Koristeći programski jezik Python, ako je dat signal $X(s)$, izračunati njegovu inverznu Laplasovu transformaciju:

$$a) X(s) = \frac{2s+4}{s^2+4s+3}, \operatorname{Re}(s) > -1$$

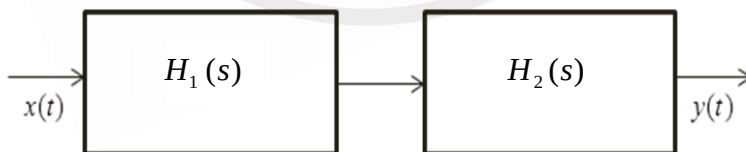
$$b) X(s) = \frac{2s+4}{s^2+4s+3}, \operatorname{Re}(s) < -3$$

$$c) X(s) = \frac{2s+4}{s^2+4s+3}, -1 < \operatorname{Re}(s) < -3$$

4.2.1. Stukturni blok dijagrami sistema

Kada je bilo reči o definiciji signala i sistema, bilo je reči o strukturnim blok dijagramima. Potreba za ovim blok dijagramima nastala je kako su sistemi postajali složeniji. Kao što smo već pomenuli, najjednostavnije transformacije su paralelna veza, redna (kaskadna) veza i veza sa povratnom spregom.

1. Redna veza dva sistema je ostvarena ukoliko je izlaz prvog od njih istovremeno ulaz drugog sistema. Ova veza se ekeivalentno predstavlja jednim sistemom čija je funkcija prenosa jednaka proizvodu pojedinih funkcija prenosa $H(s) = H_1(s) \cdot H_2(s)$. Redna veza sistema je predstavljena na slici 60.

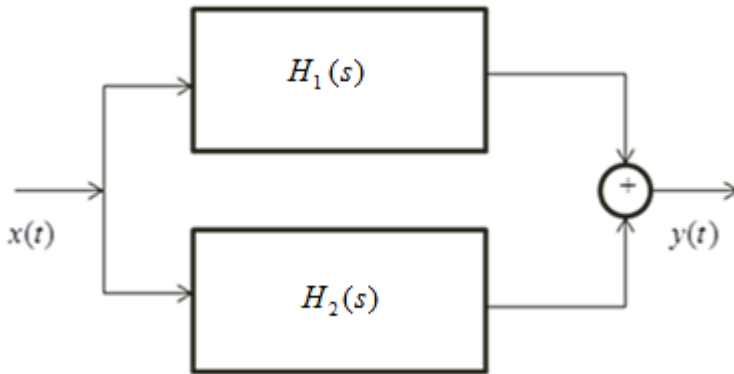


Slika 65: Redna (kaskadna) veza sistema

2. Dva sistema su u paralelnoj vezi ukoliko imaju zajednički ulaz a njihovi izlazi se sabiraju u cilju formiranja zajedničkog odziva

$$H(s) = H_1(s) + H_2(s)$$

Paralelna veza sistema predstavljena je na slici 66.



Slika 66: Paralelna veza sistema

3. U praksi, često se koriste sistemi sa povratnom spregom kakva je prikazana na slici 67. Ova vrsta sistema koristi izmerenu vrednost izlaznog signala i povratnu spregu da bi taj signal bio upoređen sa željenom vrednošću izlaznog signala.

Funkciju prenosa dobijamo na sledeći način:

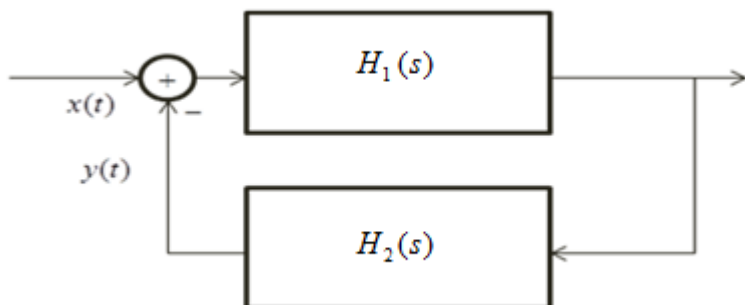
$$Y(s) = H_1 X(s) - H_1(s)H_2(s)Y(s)$$

Uzimajući u obzir da je funkcija prenosa predstavljena kao $H(s) = \frac{Y(s)}{X(s)}$, dobija se da je

ekvivalentna funkcija prenosa sistema sa povratnom spregom data je izrazom

$$H(s) = \frac{H_1(s)}{1 + H_1(s)H_2(s)}$$

Na ovaj način, korišćenjem izraza za ekvivalentne funkcije prenosa, mogu se rešavati mnogi složeni sistemi.



Slika 67: Sistem sa povratnom spregom

Zadatak 4.3.

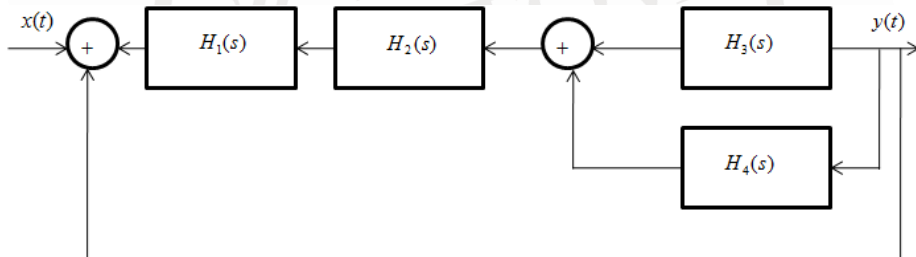
Na slici 68 prikazan je sistem čiji su ulaz i izlaz $x(t)$ i $y(t)$, respektivno. Koristeći programski jezik Python, izračunati ekvivalentni sistem prenosa $H(s)$ ako je:

$$H_1(s) = \frac{1}{(s+1)(s+2)}, \quad H_2(s) = \frac{1}{s}, \quad H_3(s) = \frac{1}{s^2}, \quad H_4(s) = \frac{1}{(s-2)^2}$$

Rešenje:

Ovaj sistem se u dva koraka svodi na sistem sa slike 68, čija je ekvivalentna funkcija prenosa

$$H(s) = \frac{H_1(s)H_2(s)H_3(s)}{1 + H_3(s)H_4(s)}$$



Slika 68: Složeni sistem sa povratnom spregom

```
In [193]: from sympy import *
```

```
In [194]: init_printing()
```

```
In [195]: s,t=symbols('s,t')
```

```
In [196]: H1=1/(s**2+3*s+2)
```

```
In [198]: H2=1/s
```

```
In [199]: H3=1/s**2
```

```
In [200]: H4=1/(s**2-4*s+4)
```

```
In [201]: He=H1*H2*H3/(1+H3*H4)
```

```
In [202]: inverse_laplace_transform(He,s,t)
```

5. z Transformacija

U predhodnom poglavlju, upoznali smo se sa Laplasovom transformacijom koja služi za rešavanje diferencijalnih jednačina u vremenski kontinualnim sistemima, dok u ovom poglavlju upoznaćemo se sa z transformacijom koja služi za pretvaranje diferencnih jednačina u algebarske jednačine. Njihov značaj je u tome što pojednostavljaju analizu vremenski diskretnih sistema i za z transformaciju važe iste osobine koje važe i za Laplasovu.

5.1. z Transformacija

5.1.1. Definicija z transformacije

Ukoliko je dat diskretni signal $x[n]$, tada je njegova z transformacija $X(z)$, (u literaturi poznata kao $Z(x[n]) = X(z)$) data izrazom:

$$X(z) = \sum_{n=-\infty}^{+\infty} x[n]z^{-n},$$

Generalno govoreći, promenljiva z je kompleksan broj, predstavljen oblikom

$$z = re^{j\varpi}$$

5.1.2. Oblast konvergencije z transformacije

Kao i kod Laplasove transformacije, tako i kod z transformacije, skup vrednosti kompleksne promenljive z za koju funkcija $Z(x[n]) = X(z)$ konvergira, naziva se oblast konvergencije.

Oblast konvergencije ima sledeće osobine:

- a) Oblast konvergencije, ne sadrži ni jedan pol.
- b) Ako je $x[n]$ konačan niz, tj. van nekog konačnog intervala važi da je $x[n]=0$ i ako njegova Z transformacija $X(z)$ konvergira za neku vrednost z , tada je oblast konvergencije celokupna z ravan, osim za vrednosti $z = 0$ i $z = \infty$.
- c) Ako je niz $x[n]$ ograničen sa desne strane, tj. $x[n]=0$ za $n < \infty$ i ako z transformacija $X(z)$ konvergira za neku vrednost z , tada oblast konvergencije ima oblik $|z| > r_{\max}$ ili $\infty > |z| > r_{\max}$, gde je $r_{\max}$ najveća amplituda medju polovima transformacije $X(z)$. Dakle, oblast konvergencije je unutrašnjost kružnice $|z| = r_{\max}$ u z ravni sa mogućim izuzetkom $z = \infty$.
- d) Ako je niz $x[n]$ ograničen sa leve strane, tj. $x[n]=0$ za $n > -\infty$ i ako z transformacija $X(z)$ konvergira za neku vrednost z , tada oblast konvergencije ima oblik $|z| < r_{\min}$ ili $0 < |z| < r_{\min}$, gde je $r_{\min}$ najmanja amplituda medju polovima transformacije $X(z)$. Dakle, oblast konvergencije je unutrašnjost kružnice $|z| = r_{\min}$ u z ravni, sa mogućim izuzetkom $z = 0$.
- e) Ako je niz $x[n]$ ograničen sa leve i desne strane i ako z transformacija $X(z)$ konvergira za neku vrednost z , tada oblast konvergencije ima oblik $r_1 < |z| < r_2$, gde su r_1 i r_2 amplitude od dva pola transformacije $X(z)$, a z ravan ne sadrži ni jedan od ovih polova.

Kao i kod Laplasove i kod z -transformacije, da ne bismo svaki put ponovo izračunavali sumu, koristimo tabelu sa transformacionim parovima (Tabela 2).

Signal $x[n]$	Z transformacija $X(z)$	Oblast konvergencije
$\delta[n]$	1	Svako z
$u[n]$	$\frac{1}{1 - z^{-1}}$	$ z > 1$
$a^n u[n]$	$\frac{1}{1 - az^{-1}}$	$ z > a $
$na^n u[n]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z > a $
$-a^n u[-n-1]$	$\frac{1}{1 - az^{-1}}$	$ z < a $
$-na^n u[-n-1]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z < a $
$(\cos \omega_0 n)u[n]$	$\frac{1 - z^{-1} \cos \omega_0}{1 - 2z^{-1} \cos \omega_0 + z^{-2}}$	$ z > 1$
$(\sin \omega_0 n)u[n]$	$\frac{z^{-1} \sin \omega_0}{1 - 2z^{-1} \cos \omega_0 + z^{-2}}$	$ z > 1$
$(a^n \cos \omega_0 n)u[n]$	$\frac{1 - az^{-1} \cos \omega_0}{1 - 2az^{-1} \cos \omega_0 + a^2 z^{-2}}$	$ z > a $
$(a^n \sin \omega_0 n)u[n]$	$\frac{az^{-1} \sin \omega_0}{1 - 2az^{-1} \cos \omega_0 + a^2 z^{-2}}$	$ z > a $

Tabela 2. z transformacija često korišćenih signala

5.2. Inverzna z transformacija

Ukoliko je poznat impulsni odziv kauzalnog sistema z transformacija $X(z)$, tada je sekvenca $x[n] = Z^{-1}(X(z))$, tj. Njegova inverzna z transformacija data izrazom:

$$x[n] = \frac{1}{2\pi j} \oint_C X(z) z^{n-1} dz, \text{ gde je } C \text{ kontura u smeru kazaljke na satu.}$$

5.2.1. Model nula i polova

Kao i u slučaju inverzne Laplasove transformacije, i z transformacija se može predstaviti pomoću modela nula i polova

$$X(z) = k \frac{(z - z_1) \dots (z - z_m)}{(z - p_1) \dots (z - p_n)},$$

Uzimajući u obzir da je $m \leq n$ i da su svi polovi prosti, može se napisati

$$\frac{X(z)}{z} = \frac{c_0}{z} + \frac{c_1}{z - p_1} + \frac{c_2}{z - p_2} + \dots + \frac{c_n}{z - p_n} = \frac{c_0}{z} + \sum_{k=1}^n \frac{c_k}{z - p_k},$$

Gde je

$$c_0 = X(z) \Big|_{z=0} \quad \text{i} \\ c_k = (z - p_k) \frac{X(z)}{z} \Big|_{z=p_k},$$

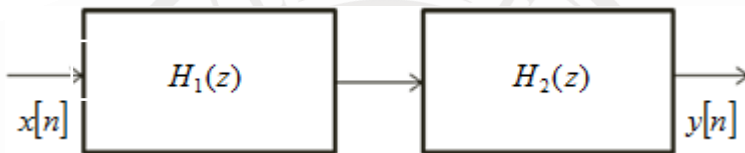
Tada se dobija izraz

$$X(z) = c_0 + c_1 \frac{z}{z - p_1} + c_2 \frac{z}{z - p_2} + \dots + c_n \frac{z}{z - p_n} = c_0 + \sum_{k=1}^n c_k \frac{z}{z - p_k}$$

5.2.2. Stukturni blok dijagrami sistema

Kada je bilo reči o definiciji signala i sistema, bilo je reči o strukturnim blok dijagramima. Potreba za ovim blok dijagramima, nastala je kako su sistemi postajali složeniji. Kao što smo već pomenuli, najjednostavnije transformacije su paralelna veza, redna (kaskadna) veza i veza sa povratnom spregom za koje važe sledeće osobine

1. Redna veza dva sistema je ostvarena ukoliko je izlaz prvog od njih istovremeno ulaz u drugi sistem. Ova veza se ekeivalentno predstavlja jednim sistemom čija je funkcija prenosa jednaka proizvodu pojedinih funkcija prenosa $H(z) = H_1(z) \cdot H_2(z)$. Redna veza sistema je predstavljena na slici 69.

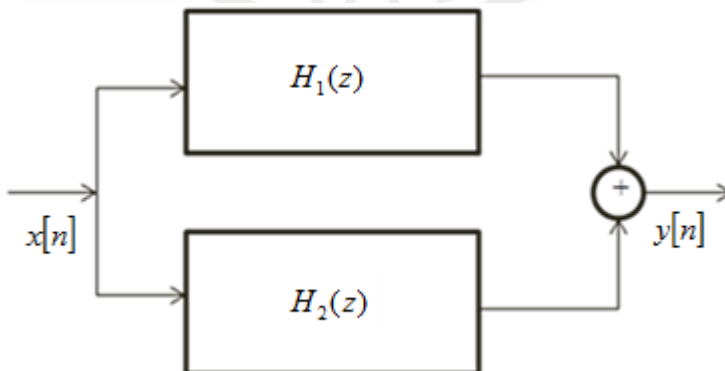


Slika 69: Redna (kaskadna) veza sistema

2. Dva sistema su u paralelnoj vezi ukoliko imaju zajednički ulaz a njihovi izlazi se sabiraju u cilju formiranja zajedničkog odziva

$$H(z) = H_1(z) + H_2(z)$$

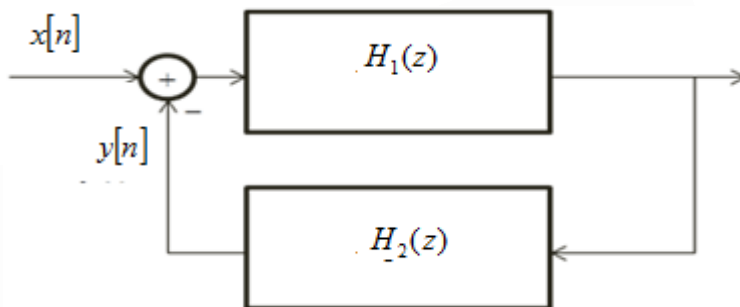
Paralelna veza sistema predstavljena je na slici 70.



Slika 70: Paralelna veza sistema

3. U praksi, često se koriste sistemi sa povratnom spregom kakva je prikazana na slici 71. Ova vrsta sistema, koristi izmerenu vrednost izlaznog signala i povratnu spregu, da bi taj signal bio upoređen sa željenom vrednošću izlaznog signala. Ekvivalentna fuinkcija prenosa sistema sa povratnom spregom data je izrazom

$$H(z) = \frac{H_1(z)}{1 + H_1(z)H_2(z)}$$



Slika 71: Sistem sa povratnom spregom

Na ovaj način, korišćenjem izraza za ekvivalentne funkcije prenosa, mogu se rešavati mnogi složeni sistemi.

Zadatak 5.1.

Funkcija z ravnj uzima brojilac i imenilac prenosne funkcije i crta polove i nule kompleksne z ravnj

$$H(z) = \frac{b_0 + b_1 z^{-1} + \dots + b_N z^{-N}}{a_0 + a_1 z^{-1} + \dots + a_N z^{-N}}$$

Koeficijenti a_i i b_i ($i=1, \dots, N$) će se smeštati u vektore $a = [a_0, a_1, \dots, a_N]$ i

$$b = [b_0, b_1, \dots, b_N]$$

Nacrtati z ravan za sledeću prenosnu funkciju

$$H(z) = \frac{z^{-1} + \frac{1}{2} z^{-2}}{1 + \frac{1}{4} z^{-1} - \frac{3}{4} z^{-2}}$$

```
In [11]: import numpy as np
import matplotlib.pyplot as plt
from matplotlib import patches
from matplotlib.figure import Figure
from matplotlib import rcParams
```

```
In [12]: #crtanje kompleksne ravni za datu funkciju prenosa
```



```

ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position('center')
ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)

# set the ticks
r = 1.5; plt.axis('scaled'); plt.axis([-r, r, -r, r])
ticks = [-1, -.5, .5, 1]; plt.xticks(ticks); plt.yticks(ticks)

if filename is None:
    plt.show()
else:
    plt.savefig(filename)

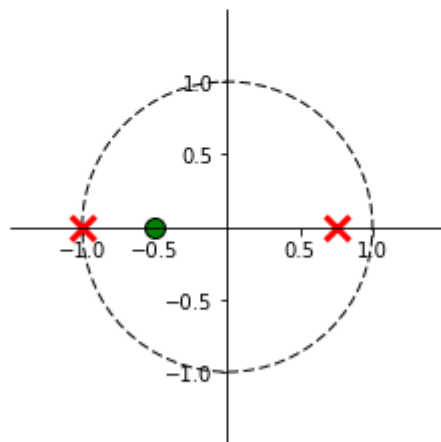
return z, p,

```

```
In [16]: b = np.array([0, 1, 1/2])
```

```
In [17]: a = np.array([1, 1/4., -3/4.])
```

```
In [18]: zplane(b,a)
```



```
Out[18]: (array([-0.5]), array([-1. , 0.75]))
```


6. Filtriranje signala

Funkcija prenosa digitalnog filtra je data u obliku

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + \dots + b_N z^{-N}}{a_0 + a_1 z^{-1} + \dots + a_M z^{-M}}$$

U prošlom poglavlju smo videli kako se uz pomoć softverskog paketa izračunavaju koeficijenti a i b .

Izlaz iz sistema se tada u inverznom obliku može predstaviti pomoću jednakosti

$$y[n] = -\sum_{k=1}^M a_k y[n-k] + \sum_{k=0}^N b_k x[n-k]$$

Kako bismo ilustrovali ideju z transformacije, u ovom poglavlju ćemo opisati jednostavnu i posebno korisnu vrstu filtera pod nazivom Batervortov filter. Filtri ove vrste imaju definisana dva parametra, graničnu frekvenciju i reda filtra. Frekvencijski odziv ovih filtera je monotona funkcija, a oštrina prelaza iz propusnog u nepropusni opseg diktira red filtra. Frekvencijski odziv filtra je dat izrazom

$$|H(j\omega)|^2 = \frac{1}{1 + (j\omega / j\omega_c)^{2N}}$$

Gde ω_c predstavlja graničnu učestanost, a N predstavlja red filtra

Za Butervortovog filter, kontinualan u vremenu, polovi asocirani sa kvadratom vrednosti frekvencijskog odziva su podjednako raspoređeni pod uglom na krugu u s-ravni, poluprečnika jednakom kritične frekvencije. Kada je data granična frekvencija i red filtra, polovi koji karakterišu funkcionisanje sistema se lako dobijaju. Kada su dati polovi, diferencijalna jednačina koja karakteriše filter je jednoznačno određena.

Zadatak 6.1.

Nacrtati karakteristiku Batervortovog filtra 4. reda, koristeći funkciju `signal.butter()`.

scipy.signal.butter

`scipy.signal.butter(N, Wn, btype='low', analog=False, output='ba')`

[\[source\]](#)

Butterworth digital and analog filter design.

Design an Nth order digital or analog Butterworth filter and return the filter coefficients in (B,A) or (Z,P,K) form.

Parameters:

N : int

The order of the filter.

Wn : array_like

A scalar or length-2 sequence giving the critical frequencies. For a Butterworth filter, this is the point at which the gain drops to $1/\sqrt{2}$ that of the passband (the “-3 dB point”). For digital filters, *Wn* is normalized from 0 to 1, where 1 is the Nyquist frequency, π radians/sample. (*Wn* is thus in half-cycles / sample.) For analog filters, *Wn* is an angular frequency (e.g. rad/s).

btype : {'lowpass', 'highpass', 'bandpass', 'bandstop'}, optional

The type of filter. Default is 'lowpass'.

analog : bool, optional

When True, return an analog filter, otherwise a digital filter is returned.

output : {'ba', 'zpk'}, optional

Type of output: numerator/denominator ('ba') or pole-zero ('zpk'). Default is 'ba'.

Returns:

b, a : ndarray, ndarray

Numerator (b) and denominator (a) polynomials of the IIR filter. Only returned if `output='ba'`.

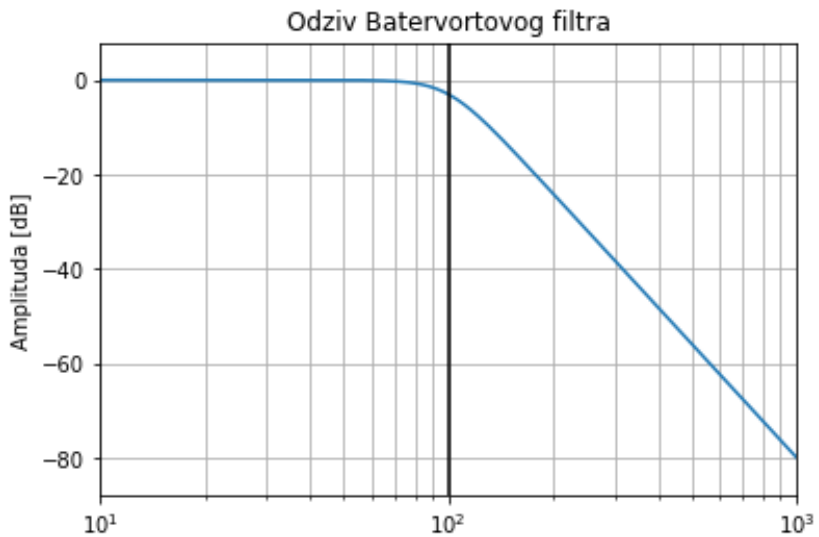
z, p, k : ndarray, ndarray, float

Zeros, poles, and system gain of the IIR filter transfer function. Only returned if `output='zpk'`.

Slika 72: Funkcija `butter()`

```
In [3]: import scipy as sp
import matplotlib.pyplot as plt
from scipy import signal
```

```
In [5]: b, a = signal.butter(4, 100, 'low', analog=True)
w, h = signal.freqs(b, a)
plt.plot(w, 20 * sp.log10(abs(h)))
plt.xscale('log')
plt.title('Odziv Batervortovog filtra')
plt.xlabel('Frekvencija [radians / second]')
plt.ylabel('Amplituda [dB]')
plt.margins(0, 0.1)
plt.grid(which='both', axis='both')
plt.axvline(100, color='k') # cutoff frequency
plt.show()
```



Slika 73: Karakteristika Batervortovog filtra 4. Reda

```
In [1]: import scipy as sp
```

```
In [2]: import numpy as np
```

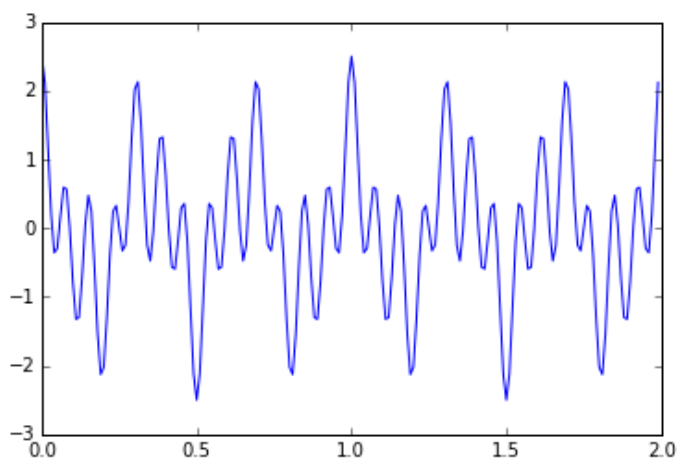
```
In [8]: from scipy import signal
```

```
In [11]: import matplotlib.pyplot as plt
```

```
In [23]: Fs = 100
t = np.arange(0,2,1./Fs )
```

```
In [30]: x = sp.cos(2*np.pi*t*3) + 0.5*sp.cos(2*np.pi*t*7) + sp.cos(2*np.pi*t*13)
```

```
In [31]: plt.plot(t,x)
plt.show()
```



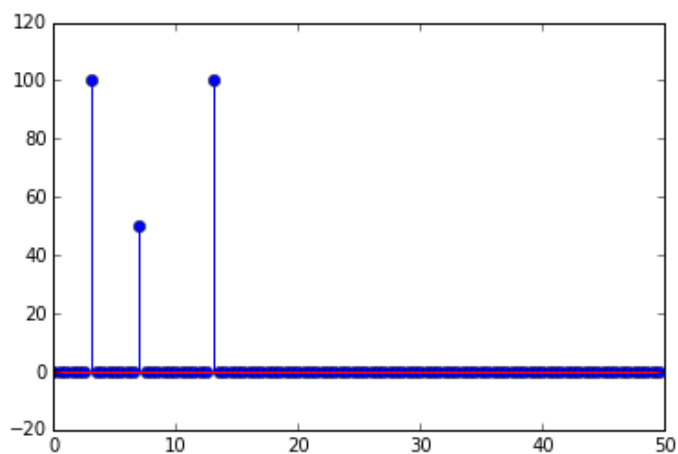
```
In [32]: X = sp.fft(x)
```

```
In [33]: N=len(X)
```

```
In [65]: f=np.linspace(0, Fs/2, len(X)/2+1)  
f=f[:-1]
```

```
In [66]: plt.stem(f, X[0:N/2])
```

```
Out[66]: <Container object of 3 artists>
```



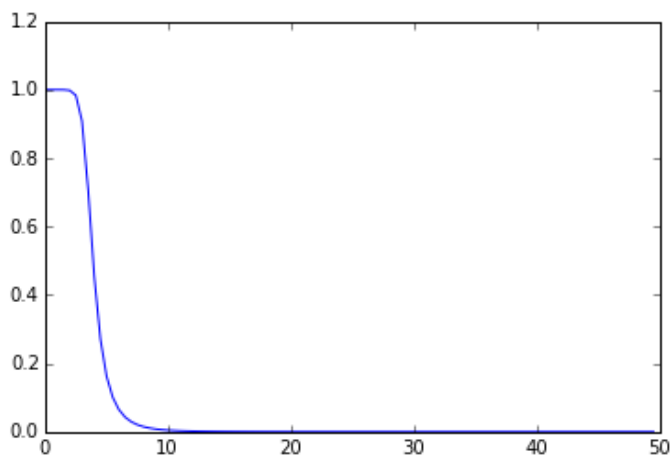
```
In [109]: b, a = signal.butter(3, 0.1)
```

```
In [123]: w, h = signal.freqz(b, a, 100)
```

```
In [124]: w1 = w/sp.pi * Fs/2
```

```
In [125]: plt.plot(w1, abs(h))
```

```
Out[125]: [<matplotlib.lines.Line2D at 0x144963c8>]
```

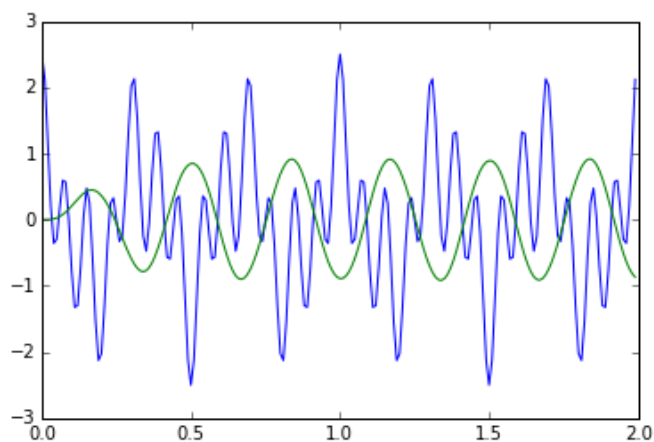


Nacrtana amplitudska karakteristika nam govori koliko će biti pojačanje ili slabljenje signala na datoj frekvenciji. Vidi se da za niske frekvencije, odokativno od 0 do 4 Hz, pojačanje iznosi 1. Takođe, vidimo da u opsegu od 4 Hz do 10 Hz postoji prelazni period. Frekvencije u ovom opsegu će biti oslabljene, ali ne i sasvim potisnute. Vidimo da su frekvencije veće od 20 Hz značajno potisnute. U praksi se koristi amplitudska karakteristika iscrtana u logaritamskoj razmeri.

```
In [126]: y = signal.lfilter(b, a, x)
```

```
In [127]: plt.plot(t, x, t, y)
```

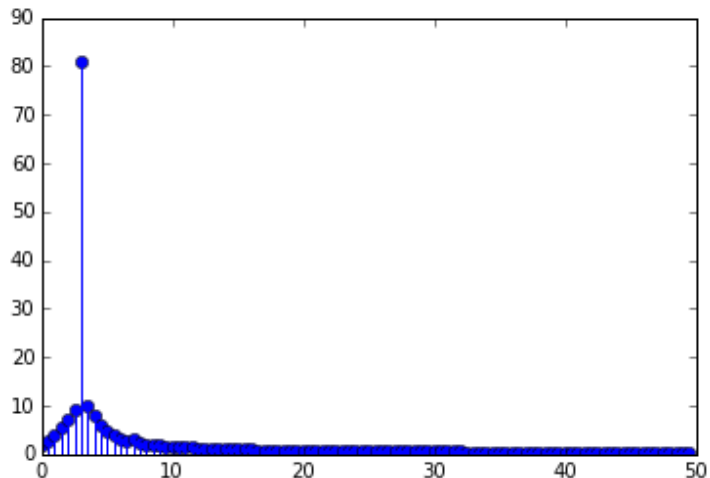
```
Out[127]: [<matplotlib.lines.Line2D at 0x14655f60>,  
<matplotlib.lines.Line2D at 0x146a51d0>]
```



```
In [128]: Y = sp.fft(y)
```

```
In [129]: f=np.linspace(0, Fs/2, len(Y)/2+1)
f=f[:-1]
plt.stem(f, abs(Y[0:N/2]))
```

```
Out[129]: <Container object of 3 artists>
```



U ovom delu vidimo da je propušten samo prvi kosinusni signal na frekvenciji 3Hz. Druga dva signala su relativno izfiltrirana, ali spektar izlaznog signala prikazuje da su ove frekvencijske komponente prisutne. Jedna od strategija da se poboljša amplitudska frekvencijska karakteristika u našem primeru je da se poveća red filtra. On će doprineti da amplitudska karakteristika brže opada, pa je tako prelazni interval manji. Napomena je da u praksi ovaj red filtra ne treba previše povećavati, jer može prouzrokovati nestabilnost filtra

```
In [122]: b, a = signal.butter(5, 0.07)
```

Druga strategija je pomeranje granične frekvencije (engl. "cut-off frequency") na niže vrednosti. U ovom slučaju izabrana normalizovana vrednost od 0.07 odgovara frekvenciji od 3.5Hz. prethodna vrednost (0.1) je odgovarala frekvenciji od 5Hz. U praksi je potrebno razlikovati delove spektra koji čine koristan signal. Sa druge strane, često se desava da se u nekim delovima spektra pored propusnika niskih opsega (low-pass filter), mogu se isprojektovati i

band-pass i high-pass filtri. Kao prvi korak za naš primer, mogu se definisati granične frekvencije, kao aritmetičke sredine 2 frekvencije.

$$\# (3+7) / 2 = 5 \implies 5/50 = 0.1$$

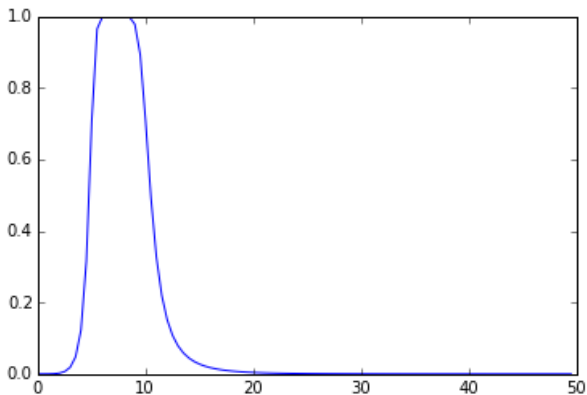
$$\# (7+13) / 2 = 10 \implies 10/50 = 0.2$$

```
In [131]: b1, a1 = signal.butter(4, [0.1, 0.2], btype='bandpass')
```

```
In [132]: w1, h1 = signal.freqz(b1, a1, 100)
```

```
In [133]: w1 = w1/sp.pi * Fs/2
           plt.plot(w1, sp.absolute(h1))
```

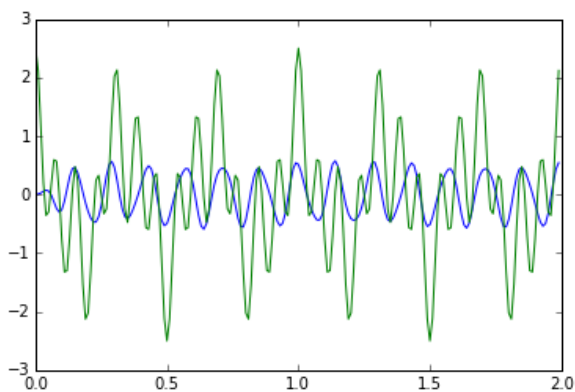
```
Out[133]: [<matplotlib.lines.Line2D at 0x14b9ef28>]
```



```
In [134]: y1 = signal.lfilter(b1, a1, x)
```

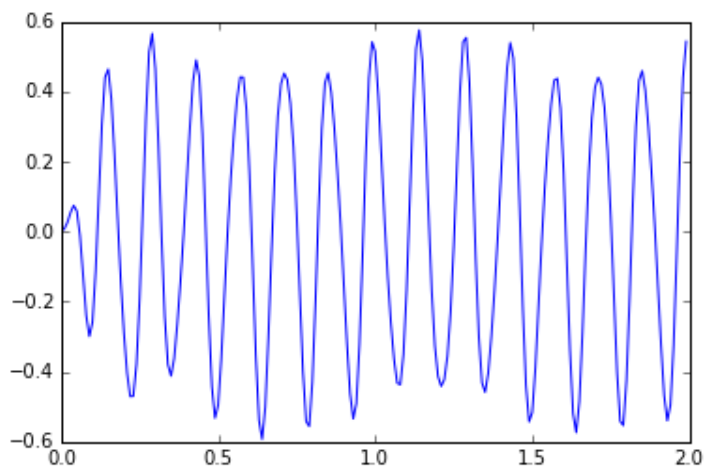
```
In [135]: plt.plot(t, y1, t, x)
```

```
Out[135]: [<matplotlib.lines.Line2D at 0x14f72c88>,  
<matplotlib.lines.Line2D at 0x14f72eb8>]
```



```
In [136]: plt.plot(t, y1)
```

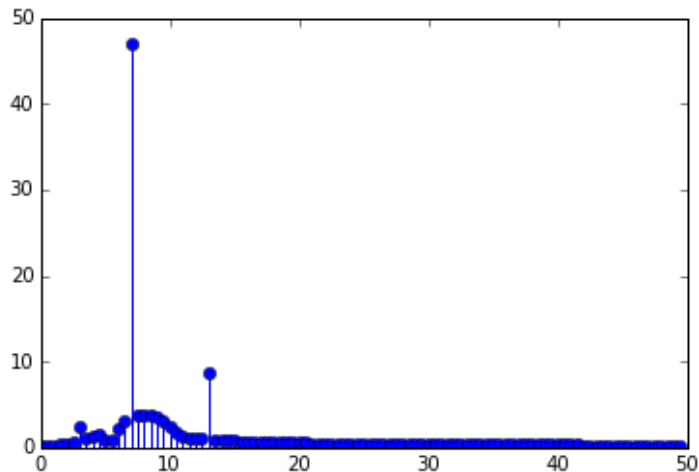
```
Out[136]: [<matplotlib.lines.Line2D at 0x1514d3c8>]
```



Dobijeni signal podseća na signal koji se sastoji samo od kosinusnog signala na 7 Hz. Međutim, moguće je zapaziti da je sada njegova amplituda promenjena i da više nije konstantna 0.5 (pogledati kako je signal $y[n]$ definisan). Da bismo se u ovo uverili možemo ponovo nacrtati spektar izlaznog signala y_1 .

```
In [141]: Y1 = sp.fft(y1)
           plt.stem(w1, abs((Y1[0:N/2])))

Out[141]: <Container object of 3 artists>
```



Vidimo da frekvencija na 13 Hz nije u potpunosti izfiltrirana i da se nalazi u izlaznom signalu. Stoga, možemo pokušati da napravimo filter koji ima uži propusni opseg i nešto strmiju amplitudsku karakteristiku u prelaznom opsegu

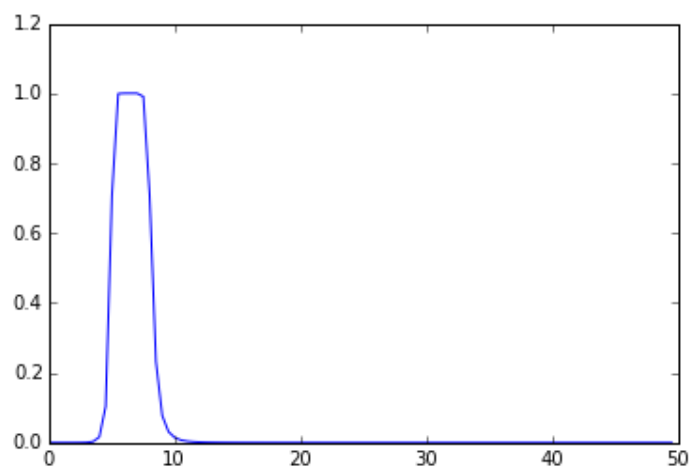
```
In [174]: b2, a2 = signal.butter(6, [0.1, 0.16], btype='bandpass')

In [175]: w2, h2 = signal.freqz(b2, a2, 100)
```

Ovde možemo i da objasnimo vrednost parametra 100. Njime definišemo broj tačaka u kojima želimo da izracunamo vrednosti amplitudske karakteristike.

```
plt.plot(w1, abs(h2))
```

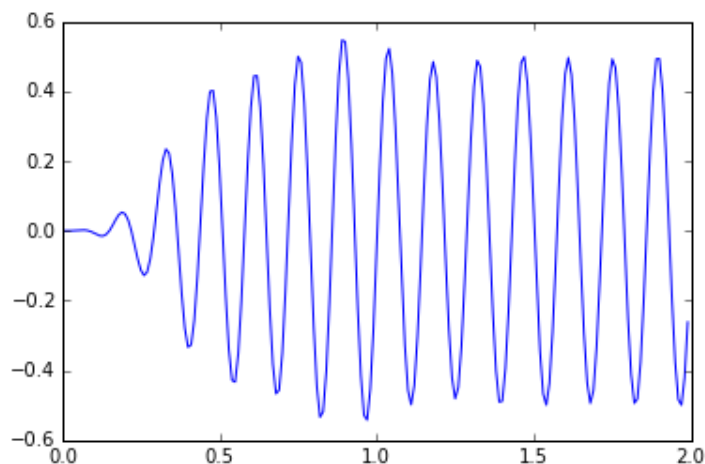
Out[175]: [<matplotlib.lines.Line2D at 0x17993da0>]



```
In [176]: y2 = signal.lfilter(b2, a2, x)
```

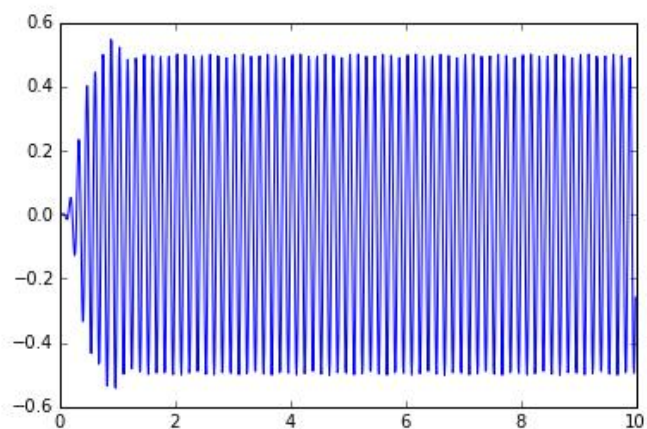
```
In [177]: plt.plot(t, y2)
```

```
Out[177]: [<matplotlib.lines.Line2D at 0x17b9c668>]
```



Da bismo ispitali ponašanje signala posle tzv. Tranzicionog perioda, kalkulacije smo izvršili za relativno duži vremenski interval od 10 sekundi. Sada primećujemo da je amplituda izlaznog signala relativno konstanta u intervalu posle 2 sekunde.

```
Out[188]: [<matplotlib.lines.Line2D at 0x152725c0>]
```



```
In [184]: t1_long
```

```
Out[184]: array([], dtype=float64)
```



Zadatak 6.2.

Generisati signal $x(t) = \sin(2\pi 0.75t) + 0.5\sin(2\pi 1.5t) + n(t)$, gde $n(t)$ predstavlja Beli Gausov Šum i propustiti kroz Butervortov filter. Signal nacrtati pre i nakon propuštanja signala kroz Filter Propusnik Niskih Učestanosti. Koristiti naredbu `lfilter()`.

`scipy.signal.lfilter(b, a, x, axis=-1, zi=None)`

Filter data along one-dimension with an IIR or FIR filter.

Filter a data sequence, x , using a digital filter. This works for many fundamental data types (including Object type). The filter is a direct form II transposed implementation of the standard difference equation (see Notes).

Parameters b : array_like

The numerator coefficient vector in a 1-D sequence.

a : array_like

The denominator coefficient vector in a 1-D sequence. If $a[0]$ is not 1, then both a and b are normalized by $a[0]$.

x : array_like

An N-dimensional input array.

$axis$: int, optional

The axis of the input data array along which to apply the linear filter. The filter is applied to each subarray along this axis. Default is -1.

zi : array_like, optional

Initial conditions for the filter delays. It is a vector (or array of vectors for an N-dimensional input) of length $\max(\text{len}(a), \text{len}(b)) - 1$. If zi is None or is not given then initial rest is assumed. See `lfiltic` for more information.

Slika 74: Naredba `lfilter()`

Rešenje:

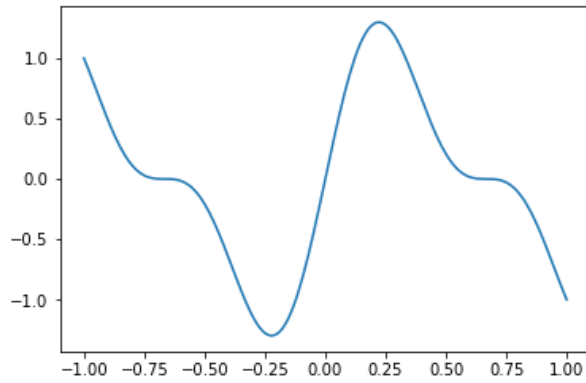
```
In [4]: import numpy as np
        from scipy import signal
```

```
In [5]: import matplotlib.pyplot as plt
```

```
In [8]: t=np.linspace(-1,1,201)
```

```
In [13]: x=np.sin(2*np.pi*0.75*t)+0.5*np.sin(2*np.pi*1.5*t)#signal
```

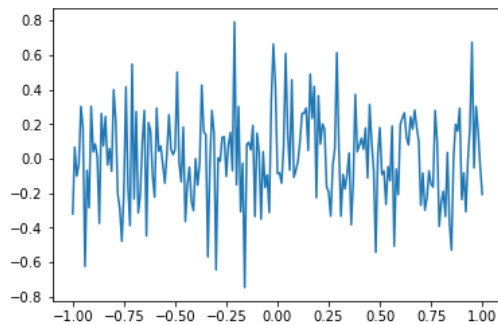
```
In [14]: plt.plot(t,x)
        plt.show()
```



Slika 75: Signal $x(t) = \sin(2\pi 0.75t) + 0.5 \sin(2\pi 1.5t)$

```
In [15]: n=0.25*np.random.randn(len(t))
```

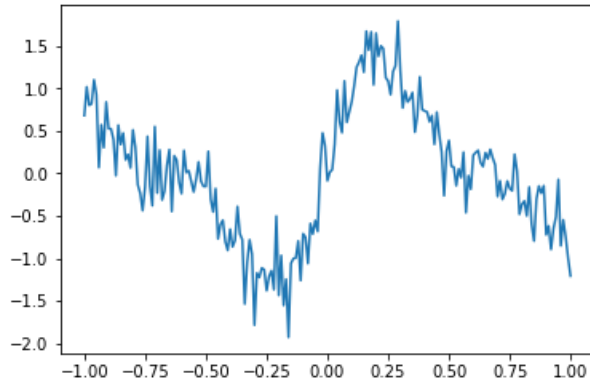
```
In [16]: plt.plot(t,n)
plt.show()
```



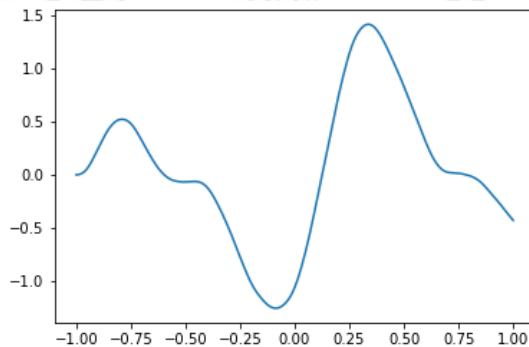
Slika 76: Signal $n(t)$ -Beli Gausov Šum

```
In [17]: xn=x+n
```

```
In [18]: plt.plot(t,xn)
plt.show()
```



Slika 77: Signal $x(t) = \sin(2\pi 0.75t) + 0.5 \sin(2\pi 1.5t) + n(t)$



Slika 78: Signal $x(t) = \sin(2\pi 0.75t) + 0.5 \sin(2\pi 1.5t) + n(t)$
nakon prolaska kroz Batervortov filter

Zadatak 6.3.

Neka je dat signal

$$x(t) = \cos(2\pi 2t) + \cos(2\pi 11t) + \cos(2\pi 22t)$$

Izračunati Diskretnu Furijeovu Transformaciju za učestanost odabiranja, uzeti da je $F_s = 100\text{Hz}$

- Nacrtati signal $x(t)$
- Iscrtati levu polovinu spektra
- Konstruisati filter propusnik niskih učestanosti (Batervortov filter) koji će propustiti signal na 2Hz
- Izfiltrirati signal $x(t)$ datim filtrom i iscrtati dobijeni izlazni signal

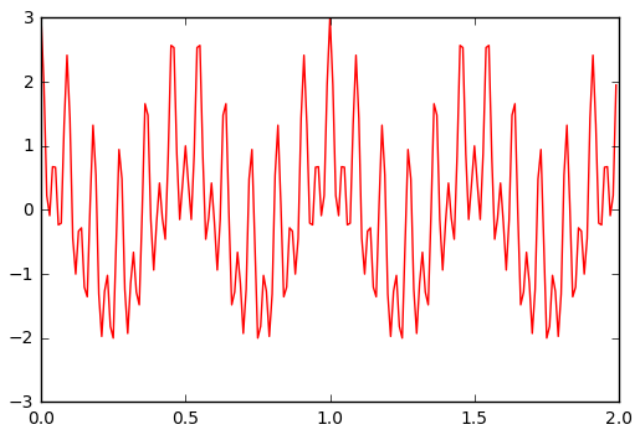
Rešenje:

a)

```
In [1]: import scipy as sp
import matplotlib.pyplot as plt
from scipy import signal
%matplotlib inline
```

```
In [2]: Fs = 100
t = sp.arange(0, 2, 1/Fs)
x = sp.cos(2*sp.pi*t*2) + sp.cos(2*sp.pi*t*11) + sp.cos(2*sp.pi*t*22)
plt.plot(t, x, 'r')
```

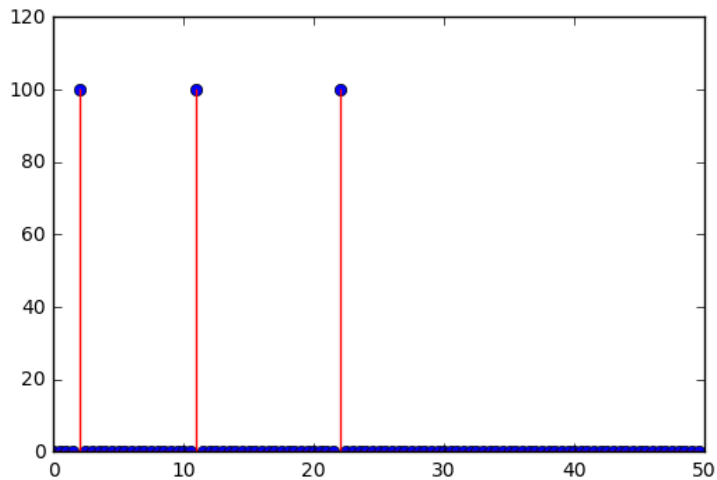
```
Out[2]: [<matplotlib.lines.Line2D at 0x81028d0>]
```



b)

```
In [3]: X = sp.fft(x)
F = sp.linspace(0,Fs,len(x)+1)
F = F[:-1]
N = len(X)
plt.stem(F[0:N/2], sp.absolute(X[0:N/2]), 'r')
```

Out[3]: <Container object of 3 artists>



c)

```
In [15]: from scipy import signal

In [27]: b,a = signal.butter(4, 0.1)

# vrednost f0 odgovara formuli fo = F0 / (Fs/2).
# dakle ukoliko zelimo da postavimo granicu filtra na 5Hz, --> 5Hz / 50 Hz = 0.1

In [28]: w,h = signal.freqz(b,a)

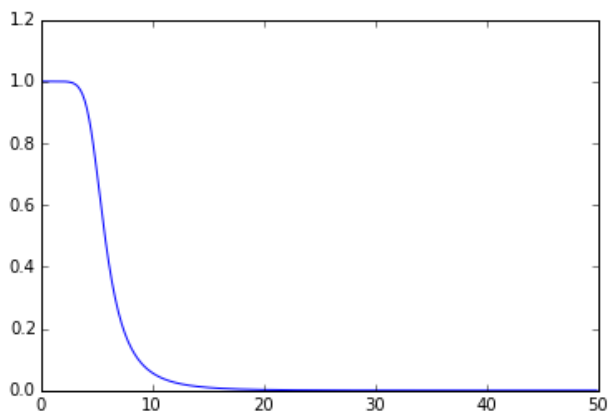
plt.plot(w,sp.absolute(h))

# Vidimo da je amplitudska karakteristika izracunata od nula do sp.pi.
# Mozemo skalirati x-osu tako da bude u sprezi sa frekvencijom odabiranja

f_normalized = w / sp.pi
ff = f_normalized * (Fs/2)
```

```
In [29]: plt.plot(ff, sp.absolute(h))
```

```
Out[29]: [<matplotlib.lines.Line2D at 0x1b7b5400>]
```

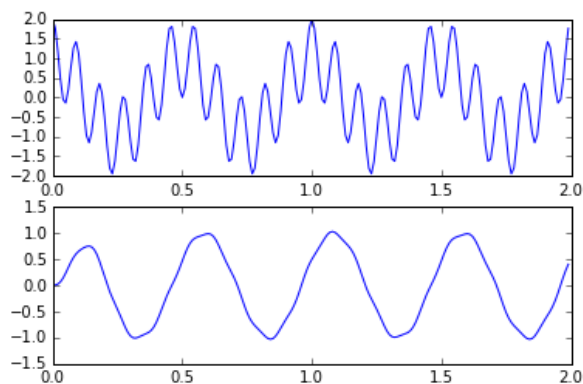


d)

```
In [30]: # Filtriranje signala x.
y = signal.lfilter(b,a,x)
```

```
In [31]: plt.subplot(211)
plt.plot(t,x)
plt.subplot(212)
plt.plot(t,y)
```

```
Out[31]: [<matplotlib.lines.Line2D at 0x1ba39518>]
```



Zvučni signal

U ovom odeljku biće razmatran *zvuk* kao primer signala.

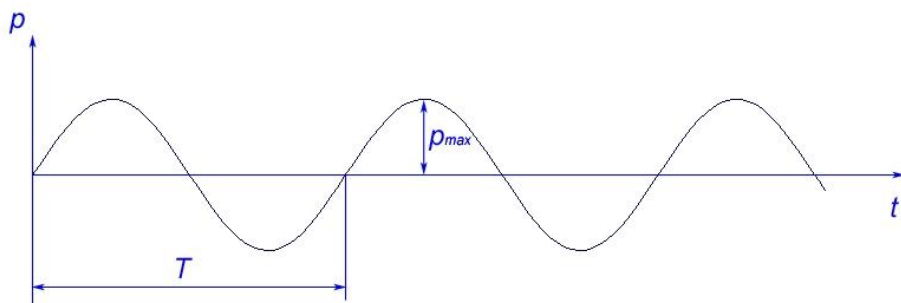
Zvuk je mehanička oscilacija čestica nekog elastičnog medijuma koje se kroz njega najčešće prostiru kao talas i koje (uglavnom) čovek čuje. Čovekov organ sluha može da registruje oscilacije u frekvencijskom opsegu od 16 Hz do 20.000 Hz ako one imaju dovoljni intenzitet.

Zvuk se prostire kroz medijume u sva tri agregatna stanja. Zvuk ne može da se prostire u bezvazдушnom prostoru. Pod zvukom se podrazumevaju i oscilacije izvan napred navedenog frekvencijskog opsega: infrazvuk ispod 16 Hz i ultrazvuk iznad 20.000 Hz.

Zvuk se u vazduhu prostire kao longitudinalni talas brzinom od $c = 344$ m/s na temperaturi od $20\text{ }^{\circ}\text{C}$ pri normalnim atmosferskim uslovima. Oblast fizike koja proučava zvuk zove se *akustika*. Obrada signala zvuka obuhvata *sintezu*, *transformaciju* i *analizu*. Iste metode mogu se primeniti na bilo koji drugi signal (električni, mehaničke vibracije, slike, audio-video fajlovi). Mehaničke oscilacije zvuka se predstavljaju kao periodične:

Parametri zvučnog signala:

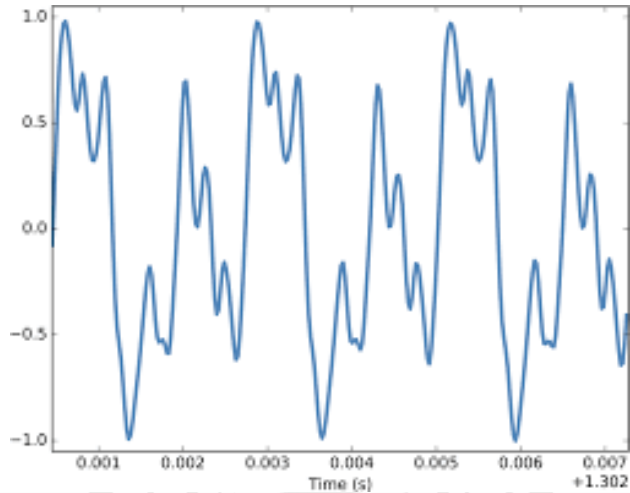
- *amplituda* ($p_{\max}$) je maksimalna vrednost pritiska vazdušnog talasa. Određuje intenzitet (jačinu) zvuka, izražava se u decibelima (dB)
- *frekvencija* (učestanost) je broj oscilacija zvučnog signala u jednoj sekundi i izražava se u hercima (Hz), jedna puna oscilacija signala naziva se perioda (T)



Kvalitet zvučnog signala zavisi od njegove *čistoće*. Čist zvuk, odnosno prost zvuk, ima samo jednu učestanost oscilovanja i on je idealnog kvaliteta. To je tzv. *ton*.

Stepen čistoće se smanjuje dodavanjem signala viših frekvencija (to su umnošci najniže frekvencije koja se naziva *osnovna frekvencija*). Sve ostale frekvencije se nazivaju (viši) *harmonici*). Tako dobijen zvuk naziva se složen zvuk. (primer je akord – istovremeno zvučanje 3 ili više tona na nekom instrumentu – gitara, klavir).

Postoje i šumovi, a to su zvukovi koji se sastoje od velikog broja frekvencija, ali ne postoji, tj. ne može se izdvojiti, osnovna frekvencija, niti postoje viši harmonici. Primer šuma su rad motora, eksplozije, škripanje, grebanje...

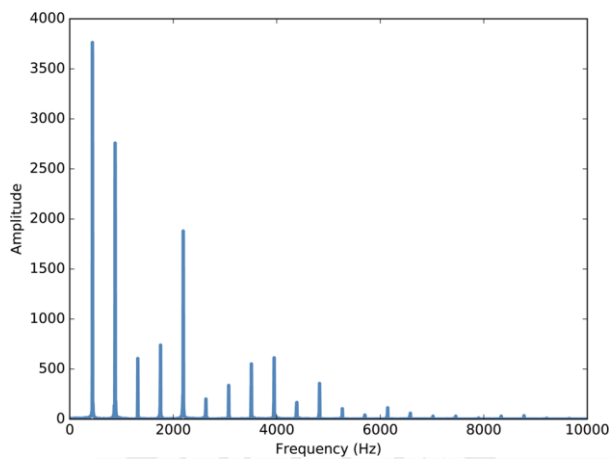


Slika 79: Boccherini's String Quintet No. 5 in E, 3rd movement (Talas predstavlja periodičan signal čiji oblik nije sinusoida)

Spektralna dekompozicija je nastala iz ideje da svaki signal može da se predstavi kao skup sinusoida različitih frekvencija.

Da bi se izvršila spektralna dekompozicija signala, potrebna je primena *Diskretne Furijeove Transformacije*.

Na Slici 75 je prikazan spektar segmenta zvuka violine. Na y osi je predstavljena snaga amplitude svake frekvencijske komponente. Najniža frekvencijska komponenta se naziva osnovna komponenta (440Hz). Ova komponenta je najjače amplitude i zove se dominantna frekvencija.



Slika 80: Spektar segmenta zvuka violine

Zadatak 7.1.

1. Učitavanje, čitanje parametara, ispisivanje audio fajla

Downloadovati zvučni fajl sa stranice <https://www.freesound.org/>, iščitati učestanost odabiranja, odabrati određeni segment datog fajla i ispisati kao novi audio fajl.

Rešenje:

```

In [1]: #Učitavanje audio signala

In [2]: from scipy.io.wavfile import read

In [10]: (fs,x)=read('C:/Users/mmarjanovic/Desktop/predavanja2016-2017/Senzo
<

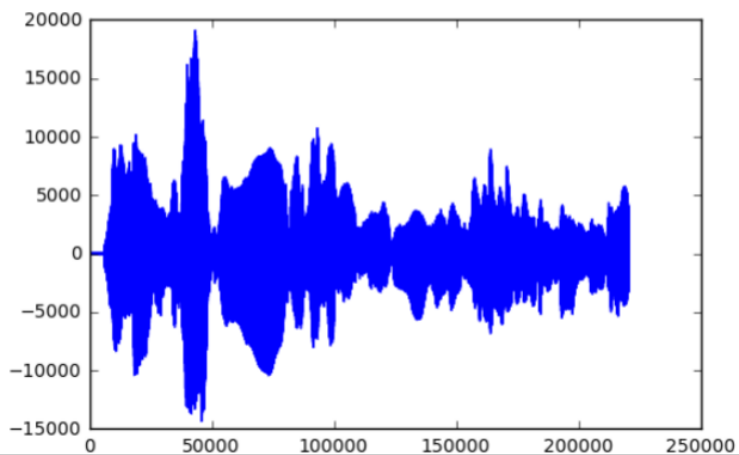
In [11]: fs
Out[11]: 44100

In [12]: x
Out[12]: array([  0,    5,   10, ..., -3288, -3206, -3042], dtype=int16)

In [15]: import matplotlib.pyplot as plt

In [16]: plt.plot(x)
plt.show()

```



Slika 81: Zvučni signal u vremenskom domenu

```
In [22]: y=x[44100:45100]
```

```
In [23]: # računanje maksimalne vrednosti
```

```
In [24]: np.max(y)
```

```
Out[24]: 16615
```

```
In [25]: # želimo da zapisujemo audio fajl
```

```
In [26]: from scipy.io.wavfile import write
```

```
In [27]: write('test.wav',fs,y)
```

```
In [28]: ls #traženje direktorijuma gde se nalazi fajl
```



Zadatak 7.2.

Koristeći programski jezik Python, učitati određeni audio signal, proveriti da li je mono ili stereo. Zatim, nacrtati mono signal u vremenskom domenu, izračunati i nacrtati njegov spektar:

```
In [40]: import scipy as sp
         from scipy.io.wavfile import read

In [41]: (fs,x)=read('367538__laspaziale__short-circuit.wav')
```

```
In [42]: x.dtype # proveravamo tip audio fajla jer 24 bitni audio fajl nije podržan
         #funkcijom read
Out[42]: dtype('int16')
```

```
In [43]: # ovo znači da vrednosti mogu biti mapirane u rang 2^15 do (2^15)-1

In [44]: x = x / (2.**15) #kako bismo normalizovali vrednosti da budu u rang -1 do 1

In [45]: x.shape #audio fajl ima dva kanala i 2198691 obiraka
Out[45]: (2198691, 2)
```

```
In [46]: 2198691 / fs #računama u sekundama koliko je trajanje fajla
Out[46]: 45.8060625
```

```
In [47]: s1 = x[:,0] #radimo samo sa jednim kanalom (mono, a ne stereo signal)
```

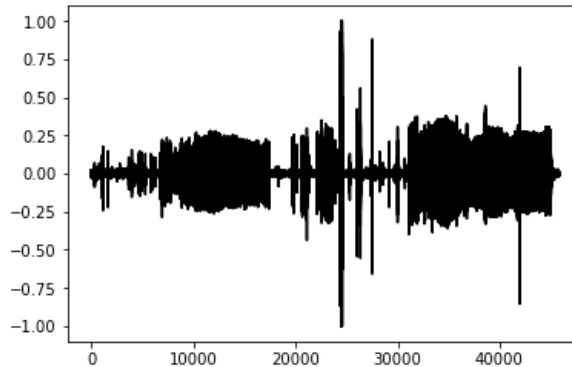
```
In [9]: #crtanje signala u vremenskom domenu

t = sp.arange(0,2198691,1)
t = t / fs
t = t * 1000 #vremenska skala u mili sekundama
```

```
In [10]: plt.plot(t, s1, color='k')
```

```
Out[10]: [<matplotlib.lines.Line2D at 0x1c3375d0be0>]
```

```
In [11]: plt.show()
```



```
In [13]: #crtanje spektra (uzimamo samo prvih 100 odbiraka signala radi bržeg izračunavanja)

X=sp.fft(s1)
X=X[0:100]
```

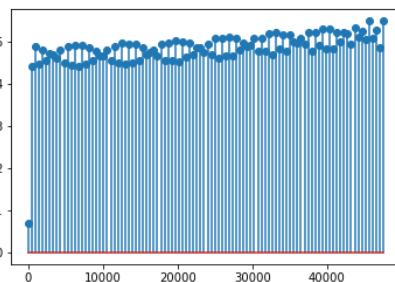
```
In [14]: F=sp.linspace(0,fs,len(X)+1)
```

```
In [15]: F=F[:-1]
```

```
In [16]: plt.stem(F,sp.absolute(X))
```

```
Out[16]: <Container object of 3 artists>
```

```
In [17]: plt.show()
```

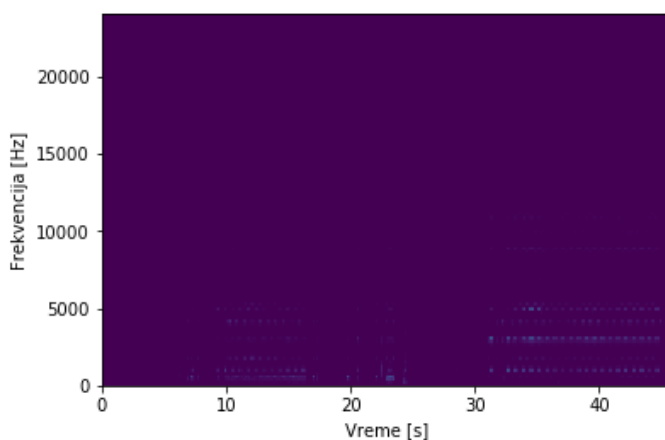


Zadatak 7.3.

Koristeći programski jezik Python, nacrtati spektrogram signala iz drugog zadatka

Rešenje:

```
In [19]: from scipy import signal  
f, t, Sxx = signal.spectrogram(s1, fs)  
plt.pcolormesh(t, f, Sxx)  
plt.ylabel('Frekvencija [Hz]')  
plt.xlabel('Vreme [s]')  
plt.show()
```



Zadatak 7.4.

Koristeći programski jezik Python, učitati dva audio fajla, izračunati njihovu konvoluciju i ispisati u treći audio fajl. Sva tri signala odslušati pažljivo i napisati svoja zapažanja:

```
In [1]: import scipy as sp
        from scipy.io.wavfile import read
        import matplotlib.pyplot as plt

In [2]: (fs,x)=read('367538__laspeziale_short-circuit.wav')
        (fs1,y)=read('367669__bobv2__wind-chimes-01.wav')
C:\Users\mmarjanovic\Anaconda3\lib\site-packages\scipy\io\wavfile.py:267: WavFileWarning: Chunk (non-data) not unders
tood, skipping it.
WavFileWarning)

In [3]: x.dtype # proveravamo tip audio fajla jer 24 bitni audio fajl nije podržan
        #funkcijom read
Out[3]: dtype('int16')
```

```
In [4]: y.dtype
Out[4]: dtype('float32')
```

```
In [5]: # ovo znači da vrednosti mogu biti mapirane u rang 2^15 do (2^15)-1

In [6]: x = x / (2.**15) #kako bismo normalizovali vrednosti da budu u rang -1 do 1

In [7]: y=y/(2.**31)
```

```
In [8]: x.shape #audio fajl ima dva kanala i 2198691 obiraka
Out[8]: (2198691, 2)

In [9]: y.shape
Out[9]: (1051061,)
```

```
In [10]: # iz ovoga zaključujemo da je y signal mono

In [11]: s1 = x[:,0] #radimo samo sa jednim kanalom (mono, a ne stereo signal)kako bi i prvi signal bio mono

In [12]: from scipy import signal

In [13]: # kako bi vreme izračunavanja bilo kraće, uzimamo prvih 1000 odbiraka vakog signala

In [14]: y=y[0:1000]

In [15]: s1=s1[0:1000]

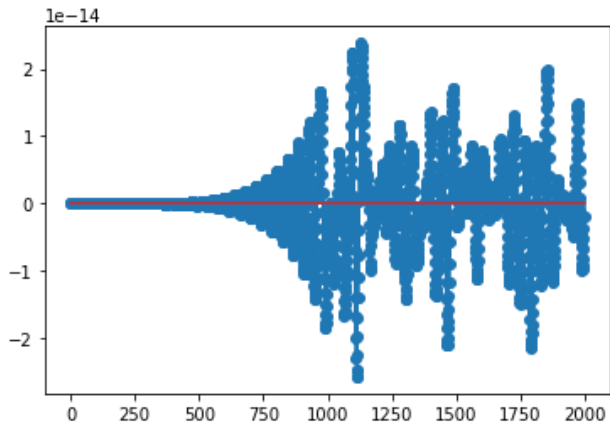
In [16]: convolution=signal.convolve(y,s1)
```

```
In [16]: convolution=signal.convolve(y,s1)
```

```
In [17]: plt.stem(convolution)
```

```
Out[17]: <Container object of 3 artists>
```

```
In [18]: plt.show()
```



```
In [20]: from scipy.io.wavfile import write
```

```
In [23]: write('audioconvolution.wav',fs1,convolution)
```

Literatura

- [1] SIGNALI I SISTEMI. Zbirka rešenih zadataka. Veljko Papić, Predrag Tadić, Aleksandra Marjanović, Akademska misao, ISBN: 978-86-7466-453-7, 2013.
- [2] SIGNALI I SISTEMI. Željko Đurović, Branko Kovačević, Srđan Stanković, Akademska misao, ISBN: 978-86-7466-294-6, 2008.
- [3] THINK DSP. DIGITAL SIGNAL PROCESSING IN PYTHON. Allen B. Downey, O'Reilly Media.
- [4] LEARNING PYTHON. Mark Lutz, David Ascher, Paperback, Second Edition, 1214 pages, Published December 30th 2003 by O'Reilly Media.
- [5] FLUENT PYTHON. CLEAR, CONCISE, AND EFFECTIVE PROGRAMMING, Luciano Ramalho, Published August 20th 2015 by O'Reilly Media (first published January 25th 2015).
- [6] FLUENT PYTHON. CLEAR, CONCISE, AND EFFECTIVE PROGRAMMING, Luciano Ramalho, Published August 20th 2015 by O'Reilly Media (first published January 25th 2015).
- [7] FOUNDATIONS OF PYTHON PROGRAMMING, John Goerzen, Martin Streicher, Published August 17th 2004 by Apress.
- [8] PYTHON PROGRAMMING: A Must Read Introduction to Python Programming (Python Programmig for Beginners, Python Programming in Context, Python Programming Data, Python Programming Language), Robert Richards, Kindle Edition, 52 pages, Published December 11th 2014.
- [9] DIGITAL SIGNAL PROCESSING, Oppenheim, Alan V., and Ronald W. Schafer, Prentice Hall, 1975. ISBN: 9780132146357.
- [10] METHODS OF SIGNALS AND SYSTEM ANALYSIS, Cooper, George, and Clare D. McGillem. Holt, Rinehart and Winston, 1967. ISBN: 9780030637452.
- [11] SIGNALS, SYSTEMS AND COMMUNICATIONS, Lathi, B. P. John Wiley and Sons, 1965.
- [12] SIGNALS AND SYSTEMS, Oppenheim, Alan V., and A. S. Willsky, Prentice Hall, 1982. ISBN: 9780138097318.
- [13] THE FOURIER INTEGRAL ANF ITS APPLICATIONS, Papoulis, A. McGraw-Hill Book Company, 1962. ISBN: 9780070484474.

CIP - Каталогизација у публикацији - Народна библиотека Србије, Београд

621.391(035)(0.034.2)

004.43PYTHON(035)(0.034.2)

МАРЈАНОВИЋ-Јаковљевић, Марина, 1976-

Priručnik iz Signala i sistema [Elektronski izvor] : (sa rešenim primerima u Pythonu) / Marina Marjanović-Jakovljević, Vladimir Matić. - 1. izd. - Beograd : Univerzitet Singidunum, 2017 (Beograd : Univerzitet Singidunum). - 1 elektronski optički disk (CD-ROM) ; 12 cm

Sistemski zahtevi: Nisu navedeni. - Tiraž 10. - Nasl. sa naslovne strane dokumenta. - Sadrži bibliografiju.

ISBN 978-86-7912-664-1

1. Матић, Владимир, 1984- [аутор]

а) Теорија сигнала - Приручници б) Програмски језик «Python» - Приручници

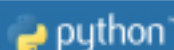
COBISS.SR-ID 247698188

© 2017.

Sva prava zadržana. Nijedan deo ove publikacije ne može biti reprodukovан u bilo kom vidu i putem bilo kog medija, u delovima ili celini bez prethodne pismene saglasnosti izdavača.



Marina Marjanović-Jakovljević
Vladimir Matić



Priručnik iz signala i sistema

(sa rešenim primerima u Pythonu)



Ovaj priručnik je nastao kao rezultat potrebe za odgovarajućim pisanim materijalom iz predmeta Signali i Sistemi koji autori drže na drugoj godini Fakulteta za Elektrotehniku i Računarstvo Univerziteta Singidunum u Beogradu. Pri pisanju je učinjen napor da priručnik bude prihvatljiv za čitaoce koji dobro vladaju znanjem iz matematike. Osnovni cilj autora bio da svi studenti koji slušaju predmet Signali i Sistemi mogu na razumljiv i lak način, uz razmatranje brojnih primera, da savladaju predviđeno gradivo, tj. steknu osnovno znanje iz teorije signala i sistema, kako u vremenskom, tako i u frekvencijskom domenu. Osim razumevanja osnovnih pojmova i uloge transformacija, student treba da se osposobi za korišćenje softvera za analizu signala i sistema.