

Univerzitet u Kragujevcu
Fakultet inženjerskih nauka



Seminarski rad iz predmeta Digitalna obrada signala

Tema:
Jednostavan prenos digitalnog signala uz slabljenje
i BER kriva

Studenti:
Nikola Mitrevski
Ana Pavlović

Predmetni profesor:
Vladimir Milovanović

Kragujevac 2020.

SADRŽAJ:

Uvod	2
Projektni zadatak	3
Realizacija projektnog zadatka	6
Prilog kodovi	6
Zaključak.....	9
Literatura	9

1. UVOD

Tema seminarskog rada je prenos digitalnog signala preko kabla koji po prirodi sadrži šum.

Kada se primi signal koji je prenesen preko kabla, on je zapravo modifikacija originalnog signala koji sadrži prigušenje i šum.

Prigušenje se može popraviti pojačavanjem primljenog signala ali se time pojačava šum.

Odnos između signala i šuma predstavlja se pomoću definicije SNR-a (signal noise ratio) koja upoređuje nivo željenog signala sa nivoom pozadinskog šuma.

SNR je definisan kao odnos snage signala i snage šuma, često izražen u db.

$$\text{SNR} = \frac{P_{\text{signal}}}{P_{\text{noise}}}$$

Pošto je digitalni signal sastavljen od niza bitova, kada se prenese može se utvrditi odnos greške bita (BER) koji je zapravo broj grešaka bitova po jedinici vremena, često se izražava u procentima.

Primer:

- Niz bitova koji prenosimo putem kabla:

0 1 1 0 0 0 1 0 1 1

- Primljen niz bitova:

0 0 1 0 1 0 1 0 0 1

Broj pogrešnih bitova (podvučeno) u ovom slučaju je tri. BER se dobija kao broj pogrešnih bitova podeljen sa ukupnim brojem prenesenih bitova.

$$\text{BER} = 3/10 = 0.3 = 30\%$$

Naš zadatak je prikazati odnos između SNR-a i BER-a koji će biti prikazan BPSK BER krivom.

Povećanjem SNR-a smanjuje se BER i obrnuto.

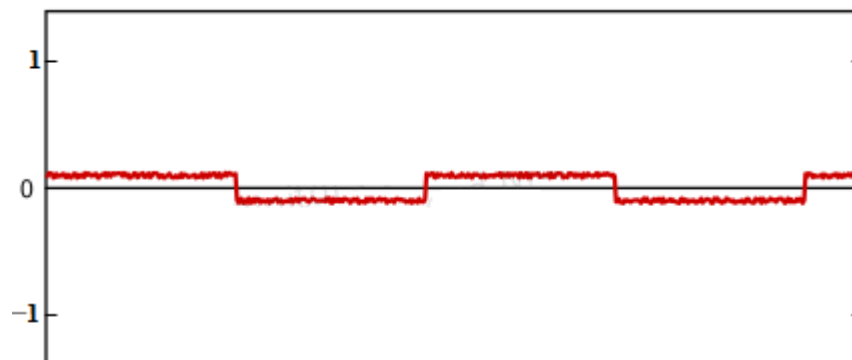
2. PROJEKTNI ZADATAK

Neka je dat signal $x(t)$ koji se prenosi preko kabla od izvorišta do odredišta.



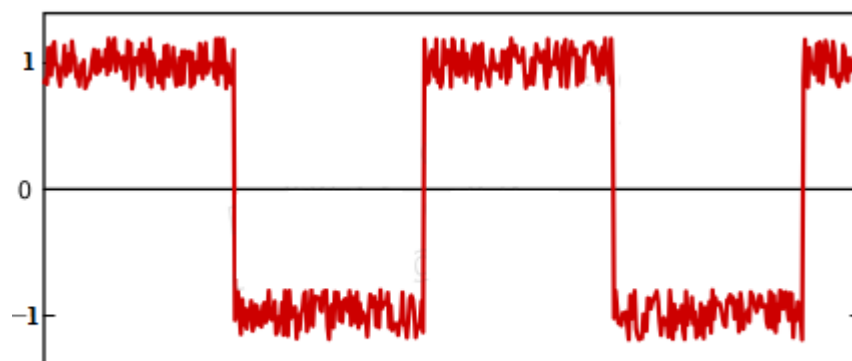
Slika 1. $x(t)$

Primljen signal je modifikacija originalnog signala i on sadrži prigušenje(G) i šum($\sigma(t)$).



Slika 2. $x(t)/G + \sigma(t)$

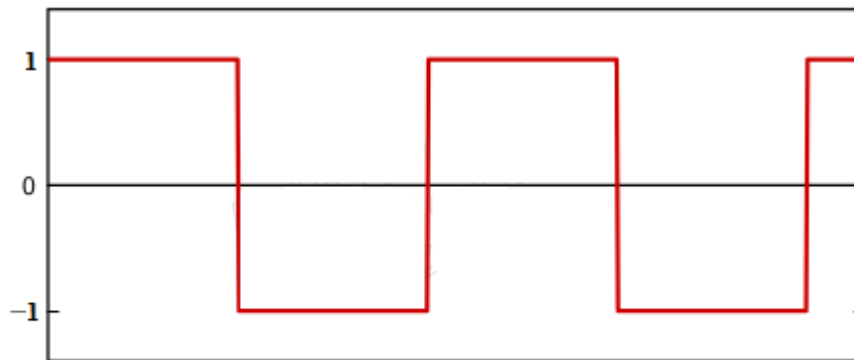
Prigušenje se može popraviti tako što se pojačava signal ali time se pojačava i šum.



$$G[x(t)/G + \sigma(t)] = x(t) + G\sigma(t)$$

Slika 3.

Pojačan šum se može korigovati tako što sve što je iznad 0V je 1V (logička 1), a sve što je ispod 0V je -1V (logička 0).



Slika 4.

$$\hat{x}_1(t) = G\text{sgn}[x(t) + G\sigma(t)]$$

Kako bi se opisale što bolje performanse digitalnog komunikacijskog sistema radi se veći broj iteracija, više puta se prenosi signal preko kabla kako bi se izračunala prosečna greška signala. Metod je poznat kao “Monte Carlo analiza” i uključuje slučajne brojeve.

Za računanje greške u svakoj iteraciji koristi se funkcija “error_calculation”, ona radi sledeće:

- random generiše niz na opsegu [-1, 1], zatim od tog niza generiše digitalni(binarni) signal koji se prenosi, slika 1;
- random generiše šum;
- kada se primi digitalni signal, on izgleda ovako:
primeljen_signal=orginalni_signal+pojačan_šum, slika 3;
- zatim se formira novi niz koji je sastavljen od orginalnog niza i modifikovanog niza;
- na kraju se formira suma u kojoj ulaze elementi novog niza i to je broj

pogrešnih bitova koju funkcija vraća.

Nakon svake iteracije se čuva ukupan broj pogrešnih bitova od kojih kada se završi sa iteracijama formira se prosek pogrešnih bitova i onda se izračunava BER.

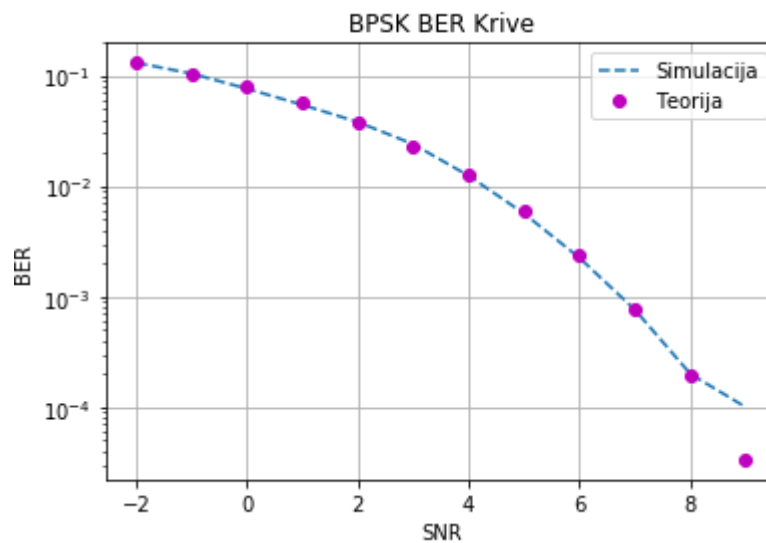
SNR predstavlja niz koji izgleda ovako: [-2 -1 0 1 2 3 4 5 6 7 8 9] i on učestvuje u generisanju random šuma.

Kada je izračunat BER koji se dobija simulacijom, onda se računa teorijski BER koji se dobija preko formule.

Na kraju se na grafiku prikazuje odnos SNR-a i BER-a i za simulacijski deo i za teorijski deo.

Pr. 1 Neka su dati sledeći podaci:

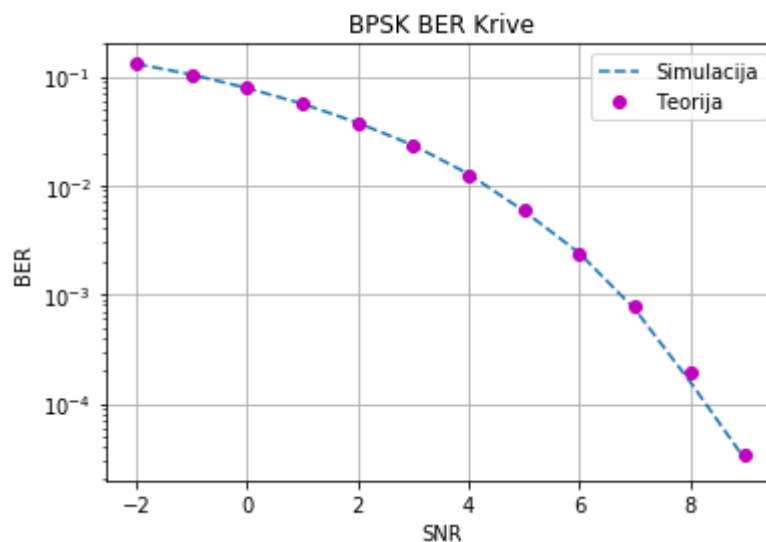
- ukupan broj bitova koji se prenosi: 10 000
- ukupan broj iteracija: 5



Slika 5.

Pr. 2 Neka su dati sledeći podaci:

- ukupan broj bitova koji se prenosi: 10 000
- ukupan broj iteracija: 30



Slika 6.

Ako se uporede slike(slika 5, slika 6), vidi se da veći broj iteracija daje precizniju grešku signala.

3. REALIZACIJA PROJEKTNOG ZADATAKA

Za realizaciju projektnog zadatka koristi se Jupyter notebook koji sadrži Python kod za obradu podataka i prikazivanje figura(npr. grafika).

Za instaliranje Jupyter notebook-a u Ubuntu Linux-u koristiti sledeće komande:

- za ažuriranje sistema:

```
sudo apt-get update
```

- za nadogradnju sistema:

```
sudo apt-get upgrade -y
```

- za instaliranje ipython-a:

```
sudo apt-get install ipython3
```

- za instaliranje Jupyter notebook-a:

```
pip3 install jupyter
```

- za pokretanje Jupyter notebook-a:

```
jupyter notebook --allow-root
```

4. PRILOG KODOVI

```
import numpy as np
from scipy.special import erfc
import matplotlib.pyplot as plt
```

```
#####
```

```
def main():
```

```
    # ukupan broj bitova koji se prenose
```

```
    bit_length = 10000
```

```
    # broj iteracija, promenljiva koja je potrebna za racunanje proseka greske
```

```
    iter_len = 30
```

```
    SNR_db = np.array(np.arange(-2, 10, 1),float)
```

```
    noise = np.zeros(len(SNR_db), float)
```

```
    error = np.zeros((iter_len, len(SNR_db)), float)
```

```
    for iter in range(iter_len):
```

```
        # svaki put se generise druga buka
```

```
        for i in range (len(noise)):
```

```
            noise[i]= 1/np.sqrt(2)*10**(-SNR_db[i]/20)
```

```

# svaki put se racuna ukupan broj pogresnih bitova
error_matrix = np.zeros(len(SNR_db), float)
for i in range(len(noise)):
    error_matrix[i] = error_calculation(bit_length, noise[i])
# svaki put se cuva ukupan broj pogresnih bitova
error[iter]=error_matrix

# prosek greske
BER = error.sum(axis=0)/(iter_len*bit_length)

# teoretska greska
theoryBER = np.zeros(len(SNR_db),float)
for i in range(len(SNR_db)):
    theoryBER[i] = 0.5*erfc(np.sqrt(10**((SNR_db[i]/10))))

plt.semilogy(SNR_db, BER,'--')
plt.semilogy(SNR_db, theoryBER, 'mo')
plt.ylabel('BER')
plt.xlabel('SNR')
plt.title('BPSK BER Krive')
plt.legend(['Simulacija', 'Teorija'], loc='upper right')
plt.grid()

# mogucnost cuvanja grafika
ans = input("Da li zelite da sacuvate grafik(da/ne): ")
if(ans == "da"):
    name_fig = input("Naziv grafika: ")
    plt.savefig("%s.png" %name_fig)
    print("Grafik sacuvan!")
elif(ans == "ne"):
    plt.show()
else:
    print("Pogresan unos!")

#####

def error_calculation(bit_length, noise_amp):
    # kreiranje niza (sadrzi random vrednosti izmedju -1 i 1, duzina bit_length)
    b = np.random.uniform(-1, 1, bit_length)

    # binarni signal generisan iz 'b'
    signal = np.zeros((bit_length), float)
    for i in range(len(b)):
        if b[i] < 0:
            signal[i]=-1
        else:
            signal[i]=1
    # signal se sastoji od niza sa elementima -1 i 1, ukupna duzina bit_length,
    # to je nas digitalni signal koji prenosimo

```



```
# Gausov-a buka
noise = np.random.randn(bit_length)

# primljen_signal=originalni_signal+pojacana_buka
rx_signal = signal + noise_amp*noise

detected_signal = np.zeros((bit_length),float)
for i in range(len(b)):
    if rx_signal[i] < 0:
        detected_signal[i]=-1
    else:
        detected_signal[i]=1
# detected_signal se sastoji od niza sa elementima -1 i 1, ukupna duzina bit_length,
# to je digitalni signal koji je primljen

error_matrix = abs((detected_signal - signal)/2)

# sabiramo sve elemente niza(dobijamo broj pogresnih bitova), jer je to potrebno za BER
error = error_matrix.sum()
return error

#####

main()
```

5. ZAKLJUČAK

Iz priloženog se vidi da nije moguće preneti nijedan signal bez neke njegove modifikacije u vidu dodavanja prigušenja i buke, zbog raznih efekata okruženja u realnosti prilikom prenosa signala.

Prigušenje se može popraviti pojačavanjem primljenog signala ali se time pojačava šum.

Pojačan šum se može korigovati tako što sve što je iznad 0V je 1V (logička 1), a sve što je ispod 0V je -1V (logička 0).

Odnos između signala i šuma predstavlja se pomoću definicije SNR-a.

SNR je definisan kao odnos snage signala i snage šuma, često izražen u db.

Pošto je digitalni signal sastavljen od niza bitova, kada se prenese može se utvrditi odnos greške bita (BER) koji je zapravo broj grešaka bitova po jedinici vremena, često se izražava u procentima.

Naš zadatak je bio prikazati odnos između SNR-a i BER-a s naglaskom na izvršavanje simulacije i teorijskog računanja, što je i prikazano grafikom.

6. LITERATURA

<https://www.youtube.com/watch?v=Yg9AkozItTU>

<https://www.coursera.org/learn/dsp/>

https://en.wikipedia.org/wiki/Bit_error_rate

https://en.wikipedia.org/wiki/Signal-to-noise_ratio