Datoru tīkli

Jānis Ratnieks (jr09103)

2018. gada 10. oktobrī

1. mājas darbs

1. uzd.

Ieejošais signāls, skatīt attēlus 1. un 2., sastādīts no attiecīgi 20 un 10 harmonikām (atsevišķās harmonikas arī redzamas attēlos). Aprēķinu rezultāti koeficientiem a_n, b_n un c redzami zemāk. Skaitliskās vērtības pārējiem koeficientiem aprēķinātas ar *python numpy* bibliotēkas *fft* funkciju un apskatāmas tabulās. Pielikumā parādīts pilns python kods attēlu un koeficientu ģenerēšanai. Kods nav labākais kodēšanas piemērs, tomēr ātrākais, lai tiktu pie rezultāta, vismaz manā gadījumā.

$$c = \frac{2}{T}\int_0^1 g(t)dt -> 2 \cdot 0.5 = 1$$

$$a_n = \frac{2}{T}\int_0^T g(t)cos(2\pi ntf)dt =$$

(trīs gadījumos $g(t) = 0$, bet pārējos $g(t) = 1$ un izteiksmi var sadalīt divos integrāļos. Pieņemot, ka $f = 1$.)

$$= 2(\int_{\frac{1}{8}}^{\frac{1}{4}} cos(2\pi nt)dt + \int_{\frac{1}{2}}^{\frac{7}{8}} cos(2\pi nt)dt)$$

$$a_1 = 2(\int_{\frac{1}{8}}^{\frac{1}{4}} cos(2\pi t)dt + \int_{\frac{1}{2}}^{\frac{7}{8}} cos(2\pi t)dt) = \frac{1}{\pi}((sin(2\pi t))|_{\frac{1}{8}}^{\frac{1}{4}} + (sin(2\pi t))|_{\frac{1}{2}}^{\frac{7}{8}})$$

$$= \frac{1}{\pi}(sin(\frac{\pi}{2}) - sin(\frac{\pi}{4}) + sin(\frac{7\pi}{4}) - sin(\pi)) = \frac{1}{\pi}(1 - \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2}) = \frac{1}{\pi}(1 - \sqrt{2})$$

$$a_2 = 2(\int_{\frac{1}{8}}^{\frac{1}{4}} cos(4\pi t)dt + \int_{\frac{1}{2}}^{\frac{7}{8}} cos(4\pi t)dt) = \frac{1}{2\pi}(sin(\pi) - sin(\frac{\pi}{2}) + sin(\frac{7\pi}{2}) - sin(2\pi)) =$$

$$= \frac{1}{2\pi}(-1 - 1) = -\frac{1}{\pi}$$

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
-0.132	-0.318	-0.256	0	0.154	0.106	0.0187	0	-0.0147	-0.063

Aprēķinātie a_n:
Aprēķini b_n ir ļoti līdzīgi a_n aprēķiniem, tādēļ parādīts tikai b_1 aprēķins.

$$b_n = \frac{2}{T}\int_0^T g(t)sin(2\pi nft)dt = 2(\int_{\frac{1}{8}}^{\frac{1}{4}} sin(2\pi nft) + \int_{\frac{1}{2}}^{\frac{7}{8}} sin(2\pi nft)dt)$$

(Līdzīgi kā iepriekš $f = 1$ un $T = 1$, bet $g(t) = 1$ apgabalos, kur $V = 1$ un $g(t) = 0$, kur $V = 0$.

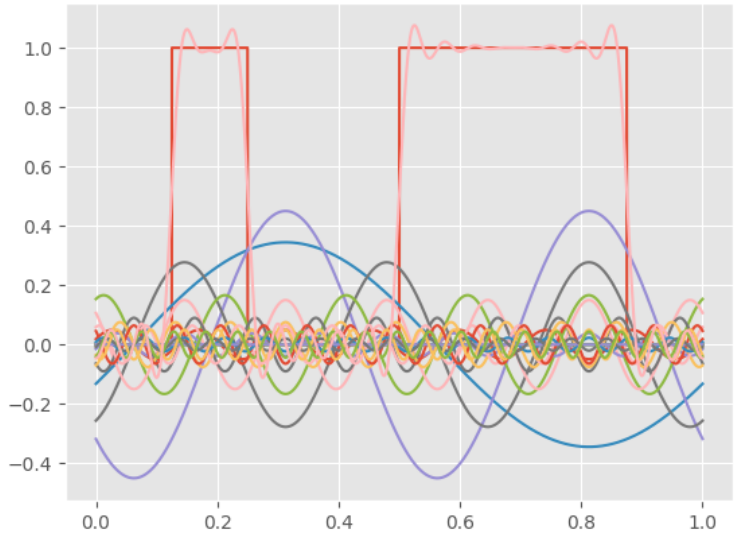
$$b_1 = 2(\int_{\frac{1}{8}}^{\frac{1}{4}} sin(2\pi t) + \int_{\frac{1}{2}}^{\frac{7}{8}} sin(2\pi t)dt) =$$

$$= \frac{1}{\pi}(\,(cos(\frac{\pi}{2}t) - cos(\frac{\pi}{4}t))\Big|_{\frac{1}{8}}^{\frac{1}{4}} + (cos(\frac{7\pi}{4}t) - cos(\pi t))\Big|_{\frac{1}{2}}^{\frac{7}{8}}) =$$

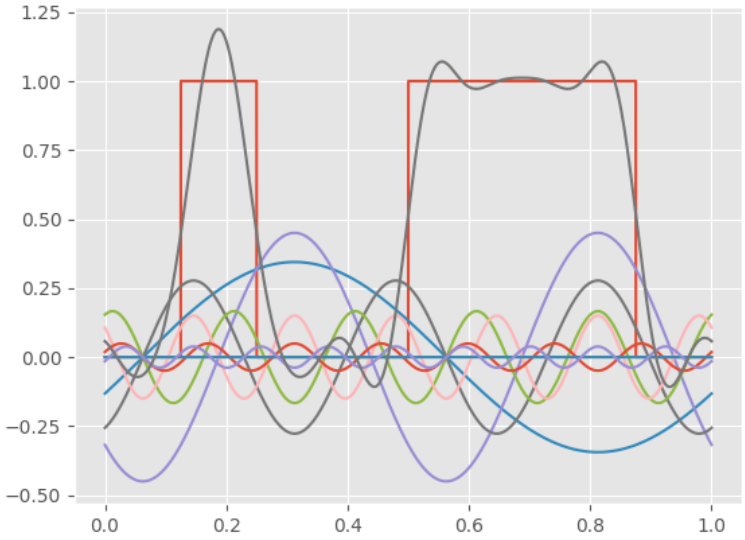
$$= \frac{1}{\pi}(0 - \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} + 1 = \frac{1}{\pi}$$

Apreķinātie b_n redzami tabulā:

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
0.318	-0.318	0.105	0	0.064	-0.106	0.046	0	0.035	-0.064



1. att. Modulējamais signāls, reālais signāls un atsevišķās harmonikas 20 koeficientu pāriem.



2. att. Modulējamais signāls, reālais signāls un atsevišķās harmonikas 20 koeficientu pāriem.

2. uzd.

MDR=max data rate

$$MDR_{Nyquist} = 2Hlog_2V\frac{bit}{sec}MDR_{Nyquist} = 2 \cdot 3e3 \cdot log_24 = 12\frac{kbit}{sec} \quad (1)$$

$$MDR_{Shannon} = Hlog_2(1+S/N)MDR_{Shannon} = 3e3log_2(101) = 3{\cdot}6.66e3 = 20\frac{kbit}{sec} \quad (2)$$

Šajā gadījumā, lai pateiktu, kurš no vienādojumiem norādīs uz datu pārneses apjomu jāsaprot, kurš vienādojums ko aprēķina. Nyquist vienādojums apreķina maksimālo datu pārraides apjomu dotam pārraides līmeņu skaitam, bet Shannon risinājums ir atvasināts no informācijas entropijas teorijas, kuru viņš pats arī izveidoja. Informācijas entropija skatās uz to, cik daudz informācijas var nosūtīt, neatkarīgi no kanālu skaita. Ja Šenona entropija būs maksimāla, kas atbilst pilnīgam troksnim jeb bezgalīgi lielam kanālu saitam, tad $S/N- > 0$, jo $N- > \infty$. Šajā gadījumā ir skaidrs, ka pieņemtais kanālu skaits pirmajam aprēķinam ar Nyquist formulu ir nepietiekams un teorētiski iespējams nosūtīt vairāk informācijas, palielinot kanālu skaitu. Tādēļ jāņem vērā Šenona vienādojums, ja grib zināt maksimāli iespējamo datu nosūtīšanas apjomu.

3. uzd.

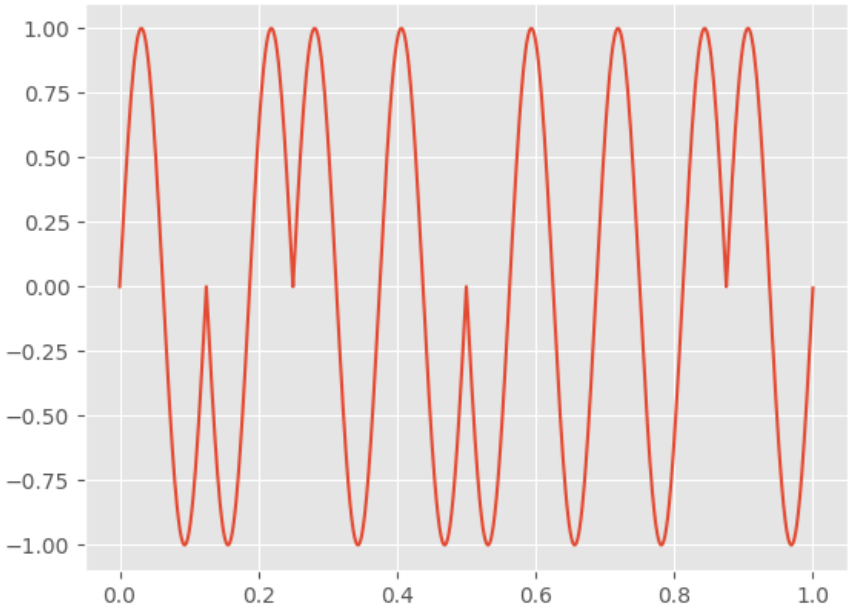
CDMA signāls redzmas tabulā 1. un apzīmēts ar S burtu, tā ir superpozīcija no visiem pārējiem signāliem. Tā kā atslēgas A, B, C un D ir savsarpēji ortogonālas, $A \cdot B = 0$, tad viennozīmīgi iespējams atkodēt signālu. To dara reizinot S ar katru no atslēgām un izdalot ar signāla garumu, reizinājumi atzīmēti tabulā 1. ar apzīmējumu $\Pi_X Y$. Pēc tam izdala ar kopējo signāla garumu jeb šajā gadījumā 8 un iegūst rezultātu. Atkarībā no notācijas -1 var apzīmēt vai nu 1 vai 0 un 1 attiecīgi 0 vai 1. Ja summa ir 0, tad dotais raidītājs attiecīgajā brīdī neraida, kā tas ir A gadījumā.

S	-1	3	-3	1	-1	-1	-1	-1	
A	-1	-1	-1	1	1	-1	1	1	
Π_{AS}	1	-3	3	1	-1	1	-1	-1	=>0
B	-1	-1	1	-1	1	1	1	-1	
Π_{BS}	1	-3	-3	-1	-1	-1	-1	1	=>-1
C	-1	1	-1	1	1	1	-1	-1	
Π_{CS}	1	3	3	1	-1	-1	1	1	=>1
D	-1	1	-1	-1	-1	-1	1	-1	
Π_{DS}	1	3	3	-1	1	1	-1	1	=>1

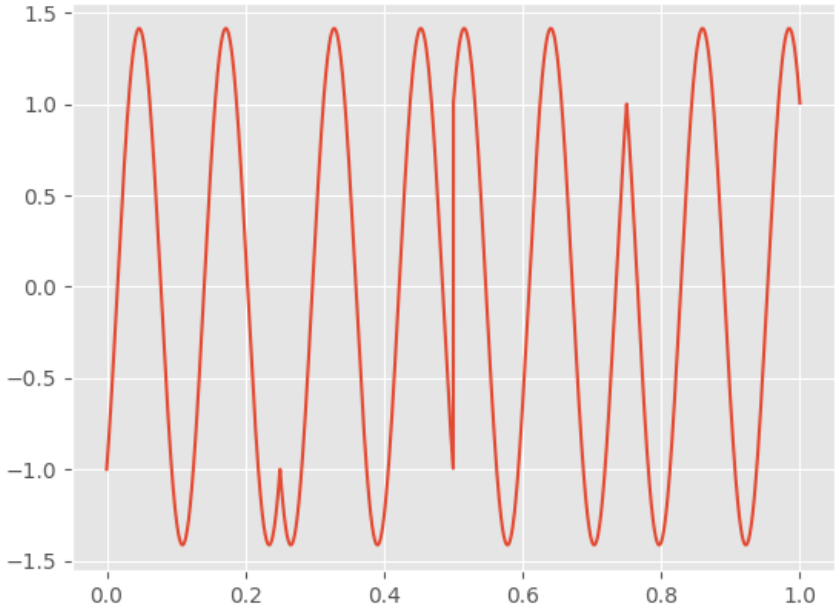
1. tabula. CDMA signālu atšifrēšana.

4. uzd.

Kā redzmas attēlos 3. un 4., tad pirmajā fāzes maiņa notiek pie katras 1 un 0 maiņas, bet otrais gadījums sadalīts 4 daļās, jo spēj pārraidīt pārus ”00”, ”01”, ”10” un ”11”.



3. att. Binary Phase Shift Keying



4. att. Quadrature Phase Shift Keying

Pielikums

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.pyplot import style

style.use('dark_background')

############ A #################
# 01010100
telpa=np.arange(0,1,0.0001)
xs=np.array([])
for i in range(len(telpa)):
    if i<1250:
        xs=np.append(xs,0)
    elif i<2500:
        xs=np.append(xs,1)
    elif i<5000:
        xs=np.append(xs,0)
    elif i<8750:
```

```
        xs=np.append(xs,1)
    else:
        xs=np.append(xs,0)
Is=np.array([])
Qs=np.array([])
for i in range(len(telpa)):
    if i<2500:
        Is=np.append(Is,-1)
        Qs=np.append(Qs,1)
    elif i<5000:
        Is=np.append(Is,-1)
        Qs=np.append(Qs,-1)
    elif i<7500:
        Is=np.append(Is,1)
        Qs=np.append(Qs,1)
    else:
        Is=np.append(Is,1)
        Qs=np.append(Qs,-1)


plt.plot(telpa, xs)
#Do a fourier transform

a=np.fft.fft(xs)

an=np.array(np.real(a[1:10]/5000))
bn=np.array(np.imag(a[1:10]/5000))
summa=np.zeros(10000)
for i in range(len(an)):
    #plt.plot(telpa, an[i]*np.cos(2*np.pi*(i+1)*telpa)+bn[i]*np.sin(2*np.pi*(i+1)*te
    summa=summa+an[i]*np.cos(2*np.pi*(i+1)*telpa)-bn[i]*np.sin(2*np.pi*(i+1)*telpa)

plt.plot(telpa,summa+0.5)
plt.show()

for i in range(len(an)):
    print(','.join(str(a[i]) for i in range(len(an))))


print(an)
print(bn)


plt.plot(np.sqrt(np.real(a[0:10])**2+np.imag(a[0:10])**2),linewidth=0, marker='*')
```

```
plt.show()

######### B ############

'''
A. (Obligātā daļa uz 7) Patstāvīgi atkārtot gramatā 2.1.1 nodaļā ilustrēto Furje tra

B. Četru līmeņu digitāls signāls tiek raidīts caur 3 KHz kanalu, kura trokšņu līmeni

C. CDMA uztvērējs uztvēris sekojošu "čipu" virknīti: (-1 +3 -3 +1 -1 -1 -1 -1). Izma

D. (Neobligata dala uz 10) Grafiski uzkonstruējiet signāla formu (dažus periodus), k


'''

bpsk=np.sin(2*np.pi*8*telpa+xs*np.pi)
plt.plot(telpa, bpsk)
plt.show()


I=Is*np.cos(2*np.pi*8*telpa)
Q=Qs*np.sin(2*np.pi*8*telpa)

plt.plot(telpa, (I+Q))
plt.show()
```