Alp Togan KÖMÜRLÜ

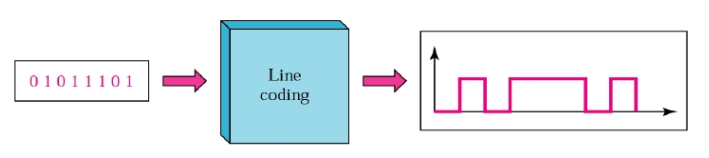
040160085 15.04.2021

Sayısal Haberleşme – ÖDEV 1

**HAT KODLAMA**

Hat kodlama, sayısal veriyi birkaç farklı kodlama tekniği kullanarak sinyale dönüştüren bir sayısal haberleşme alanı disiplinidir. Temel band haberleşme sistemlerinde sıkça kullanılır.

Hat kodlamada, her bit ayrık bir darbe biçiminde kodlanır. Bu oluşturulan sinyal modülasyona uğratılabilir ve bu yüzden sayısal haberleşmenin önemli bir parçasını oluşturur.



1.Hat kodlama şeması

Bit hızı saniyede iletilen bit sayısını belirtir(bps). Sinyal hızı ise saniyede iletilen sinyal sayısını belirtir(baud/s). Hat kodlama yaparken amaç, sinyal oranını düşürüken bit oranını yükseltmektir.

İletilebilecek maksimum veri oranı aşağıda Nyquist ve Shannon formülleri ile sınırlıdır[[1]](#footnote-1):

2 \* B \* log2(n) bit (1)

B \* log2(1+ S/N) (2)

B: Bant genişliği , n: Ayrık sinyal seviye sayısı , S: Sinyal gücü , N: Gürültü gücü

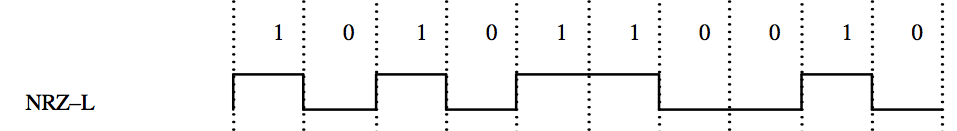
**Hat Kodlama Teknikleri**

1. NRZ(non return to zero) Kodlama

En çok kullanılan metodlardan biri olan NRZ kodlamada her bit belli bir seviyedeki işaret ile ifade edilir. Seçilen bir bit için yüksek veya düşük, diğeri için ise tam tersi olacak şekilde kodlanır. “Clock” değişimi sırasında 0’a gitmediği için bu kodlamaya sıfıra dönüşsüz denir[[2]](#footnote-2). Bu metodun 3 farklı yöntemi NRZL , NRZM ve NRZS’ dir.

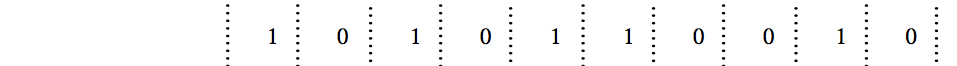
* 1. NRZ-Level (NRZL)

Bu kodlamada standart olarak 1 bit için yüksek, 0 bit için alçak gerilim çıkışta görülür. Aşağıda her kodlama tekniğinde olduğu gibi örnek bir zaman aralığı gösterilmiştir[[3]](#footnote-3).



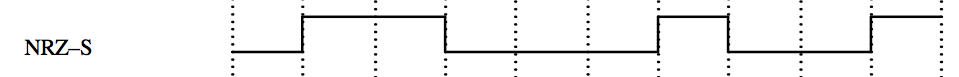
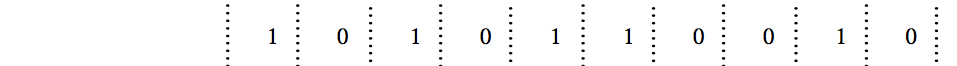
* 1. NRZ-Mark(NRZM)

NRZM kodlamada, bite bağlı olarak çıkıştaki yüksek veya alçak gerilim değiştirilir. 1 bitlik giriş için çıkıştaki seviye önceki durumun çıkışına göre değiştirilir, tam tersi yapılır. Örnek olarak, ilk gerilimi yüksek olan çıkışa 1 bit girişi verildiğinde çıkış alçak gerilim verecektir. 0 bitlik giriş için ise değişim gözlenmeyecektir ve bir önceki andaki seviye korunacaktır.



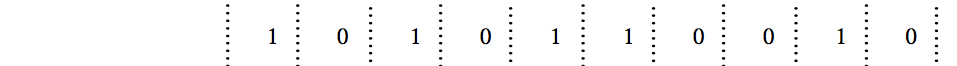
* 1. NRZ-Space(NRZS)

Bir önceki metoda benzer olarak bu kodlamada da alınan bite göre çıkış değiştirlmektedir. Ama NRZS kodlamada 0 biti için çıkış bir önceki seviyeye göre değiştirlmektedir. 1 bitlik giriş için ise seviye aynı kalmaktadır.



1. RZ Kodlama

Bu kodlama tekniğinde her türlü girişte çıkışın yarım periyodunda gerilim 0’a gider ve bu yüzden sıfıra dönüş adını alır. Girişten 1 bit alındığında yarım dalga şeklinde periyodun yarısına kadar yüksek gerilim kalanında ise düşük gerilim uygulanır. 0 bit girişinde ise bütün periyod boyunca düşük gerilim görülür[[4]](#footnote-4).

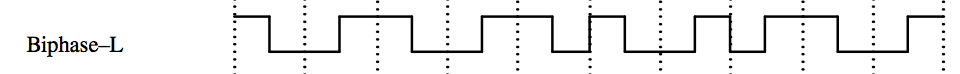
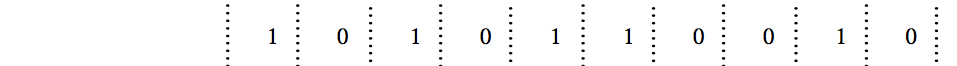


1. İki Fazlı (Biphase) Kodlama

İki fazlı kodlamada her bir bit için çıkışta en az bir değişim gözlenir.

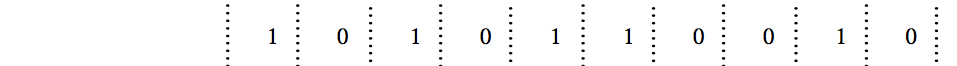
* 1. Biphase-L Kodlama

Manchester kodlama olarak da bilinir. Bu kodlama tipinde 1 bit girişi için negatife geçiş olur yani gerilimde periyodun yarısında yüksekten alçağa doğru bir geçiş kullanılır. 0 bit girişinde ise tam tersi alçaktan yükseğe bir geçiş olur.



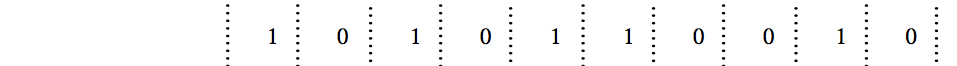
* 1. Biphase-M Kodlama

Bu kodlamada girişten alınan bitin değerinden bağımsız olarak çıkışta önceki seviyeye göre bir değişim olur(Yüksek ise alçak ile alçak ise yüksek gerilim ile başlar). Bitlerdeki değişimler ise ; 1 bit için periyodun yarısında değişim gözlemlenir , 0 bit için ise periyot boyunca sabit kalır.



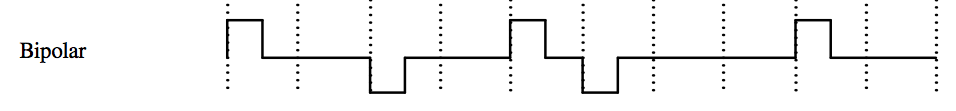
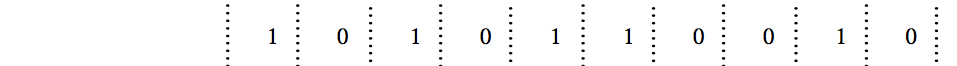
* 1. Biphase-S Kodlama

Burada da bir önceki metod gibi sinyalin başında değişim gözlenir. Ama bitlere göre çıkış değişimi tam tersidir. Bu sefer 0 bit girişi için periyodun yarısında çıkış seviye değiştirir. 1 bit giriş için ise periyod boyunca değişim gözlenmez.



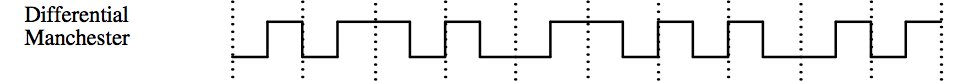
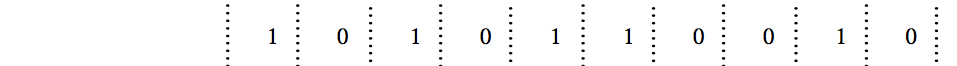
1. İki kutuplu (Bipolar) Kodlama

İki kutuplu kodlamada her bit için gerilim düzeyi 0 dan geçer. 1 bit giriş için yüksek veya düşük sonra ise 0 geriliminden geçer. 0 bit giriş için ise sabit sıfır gerilim kullanılır.



1. Farksal Manchester Kodlama

Bu kodlama tekniğinde periyodun ortasında her zaman değişim yaşanır. 1 bit girişin çıkışında önceki çıkış devam ettirilirken , 0 bit girişi için ise değişim gözlenir.



1. AMI Kodlama

AMI kodlamada 0 bit girişi için çıkış alçaktır. 1 bit girişinde ise çıkış bir pozitif bir negatif gerilim olur[[5]](#footnote-5).

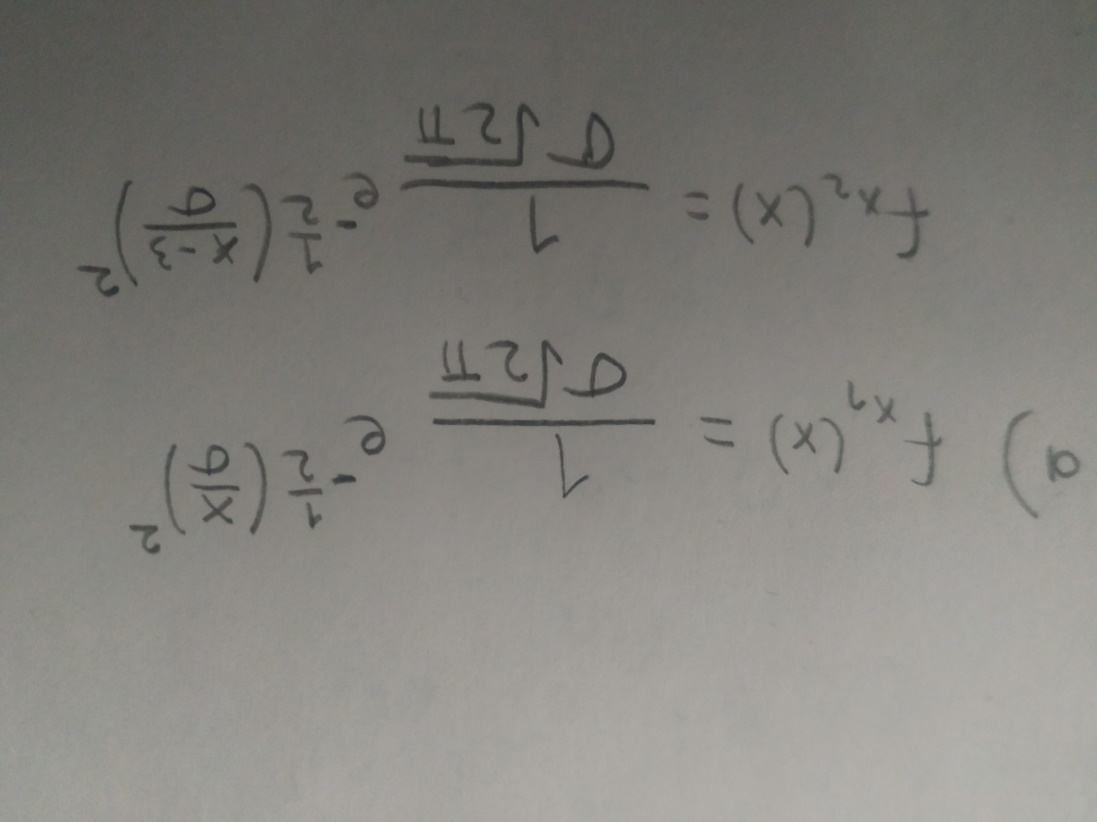


İncelenen hat kodlama tekniklerinin birbirine göre avantajlı ve dezavantajlı özellikleri bulunuyor. NRZ tipi kodlamı uygulaması basit ve fazla band genişliği gerektirmez ama senkronizasyon problemleri ve sabit DC bileşeni vardır. Örnek olarak sürekli gönderilen 1 veya 0 bitler değişime sebep olmayacağından küçük bir kayma bile çıkış sinyalinden anlaşılamıyacak şekilde bozulabilir. RZ kodlama metodunda da aynı sorunlar yaşanmaktadır [[6]](#footnote-6).

Manchester kodlamanın çıkışında 0 geçişi olduğu için zaman bilgisi sinyalin içinde gelir. Bu senkronizasyon için büyük bir avantajdır ama daha fazla bant genişliği gereklidir. Bunun sebebi ise NRZ kodlamaya göre sinyal oranının iki kat daha fazla olmasıdır. Bir diğer avantajı ise DC bileşeninin bulunmamasıdır[[7]](#footnote-7).

1. Soru cevapları:

a)



b)

KOD:

import numpy as np

import matplotlib.pyplot as plt

import math

# In[2]:

# gauss dağılımı pdfini hesaplayan fonksiyon

def gauss(x,ort,sigma):

    f = ( np.power(np.e, (-0.5 \* (x-ort)\*\*2 / sigma ) )  )/ (np.sqrt(2\*np.pi\*sigma) )

    return f

# In[3]:

# hesaplanmak üzere istenen aralıkta oluşturulan x değerleri:

x\_ler = np.linspace(-5,10,100)

# In[10]:

# b şıkkı plotları

plt.title("0 ortalamalı X1 değişkeni gauss pdf'i")

plt.plot(x\_ler, gauss(x\_ler,0 ,1), label = "varyans = 1" )

plt.plot(x\_ler, gauss(x\_ler,0 ,5), label = "varyans = 5")

plt.xlabel("x")

plt.ylabel("fX1(x)")

plt.legend()

plt.show()

# In[12]:

# b şıkkı plotları

plt.title("3 ortalamalı X2 değişkeni gauss pdf'i")

plt.plot(x\_ler, gauss(x\_ler,3 ,1), label = "varyans = 1" )

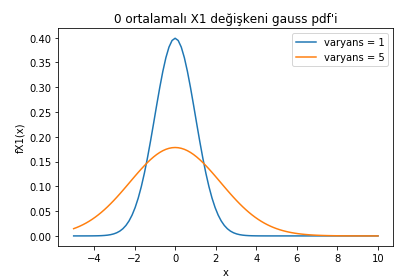
plt.plot(x\_ler, gauss(x\_ler,3 ,5), label = "varyans = 5")

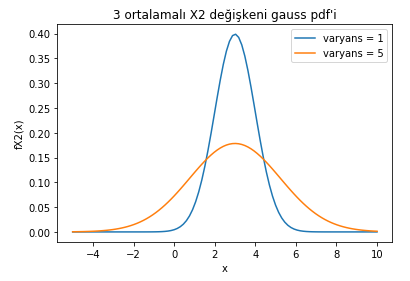
plt.xlabel("x")

plt.ylabel("fX2(x)")

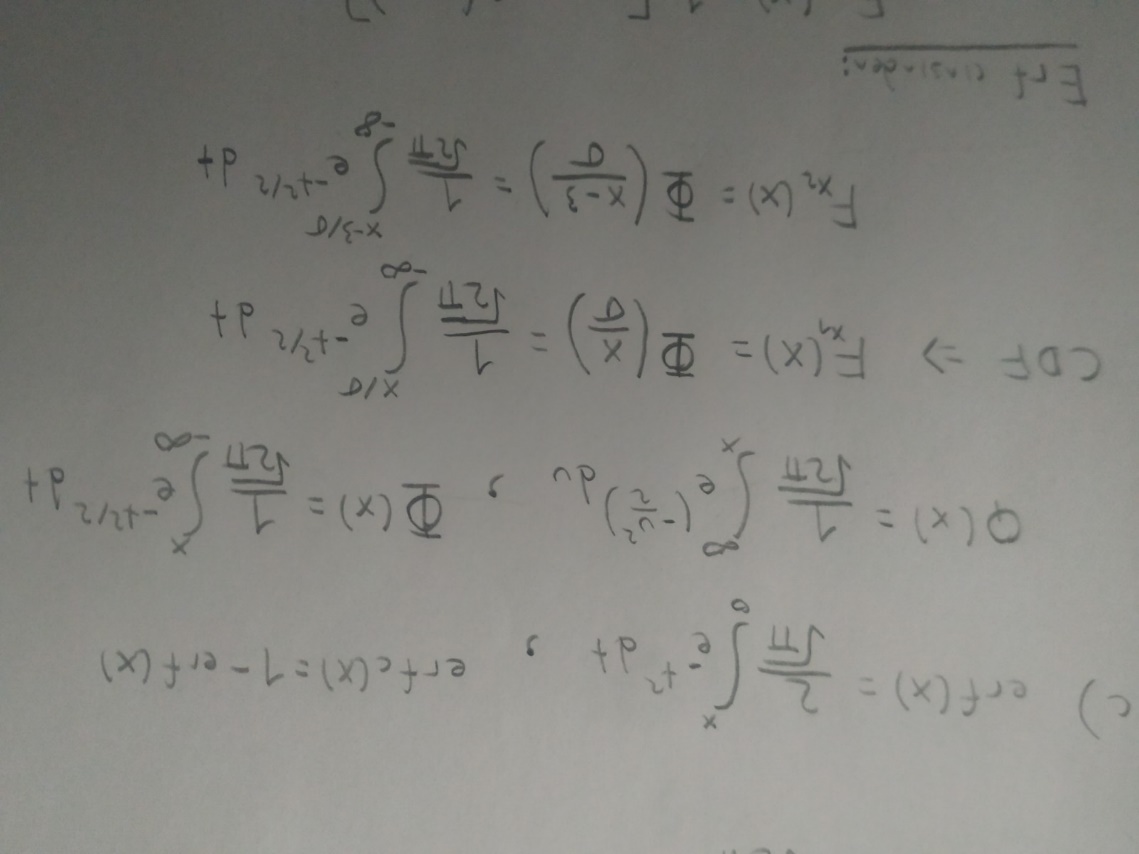
plt.legend()

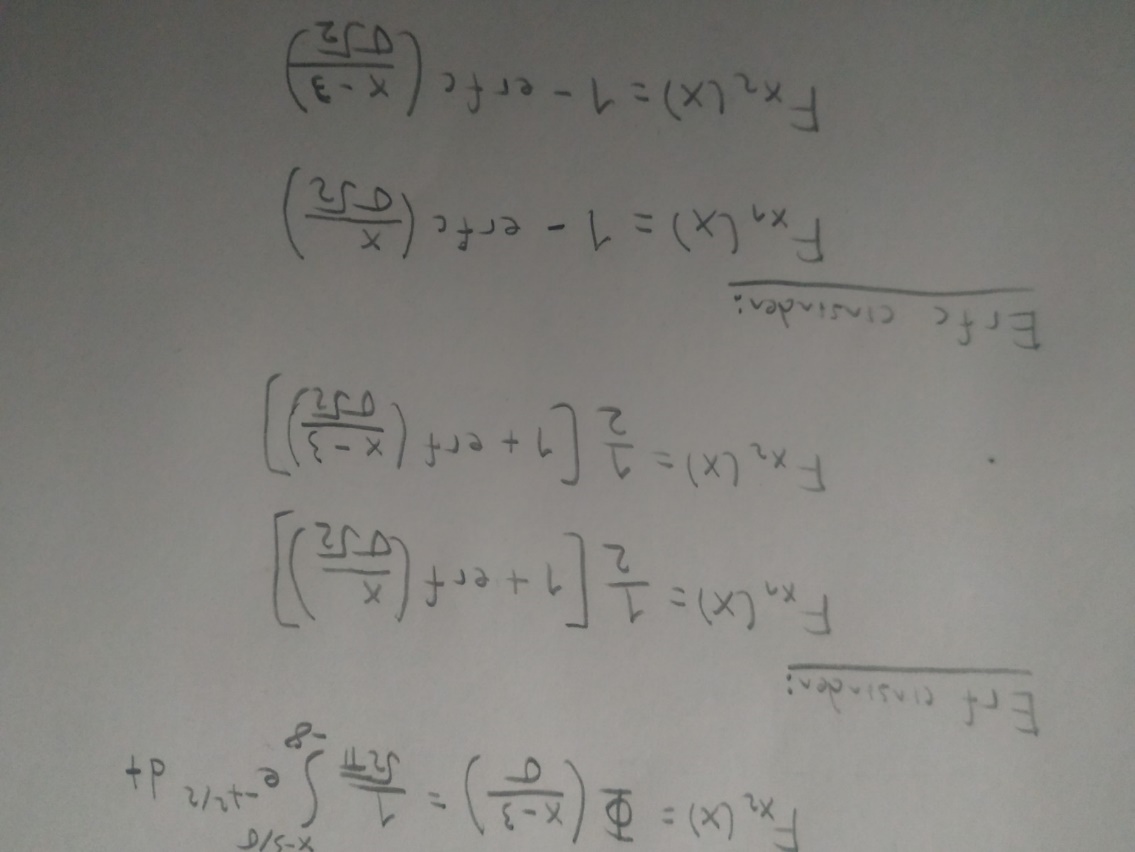
plt.show()

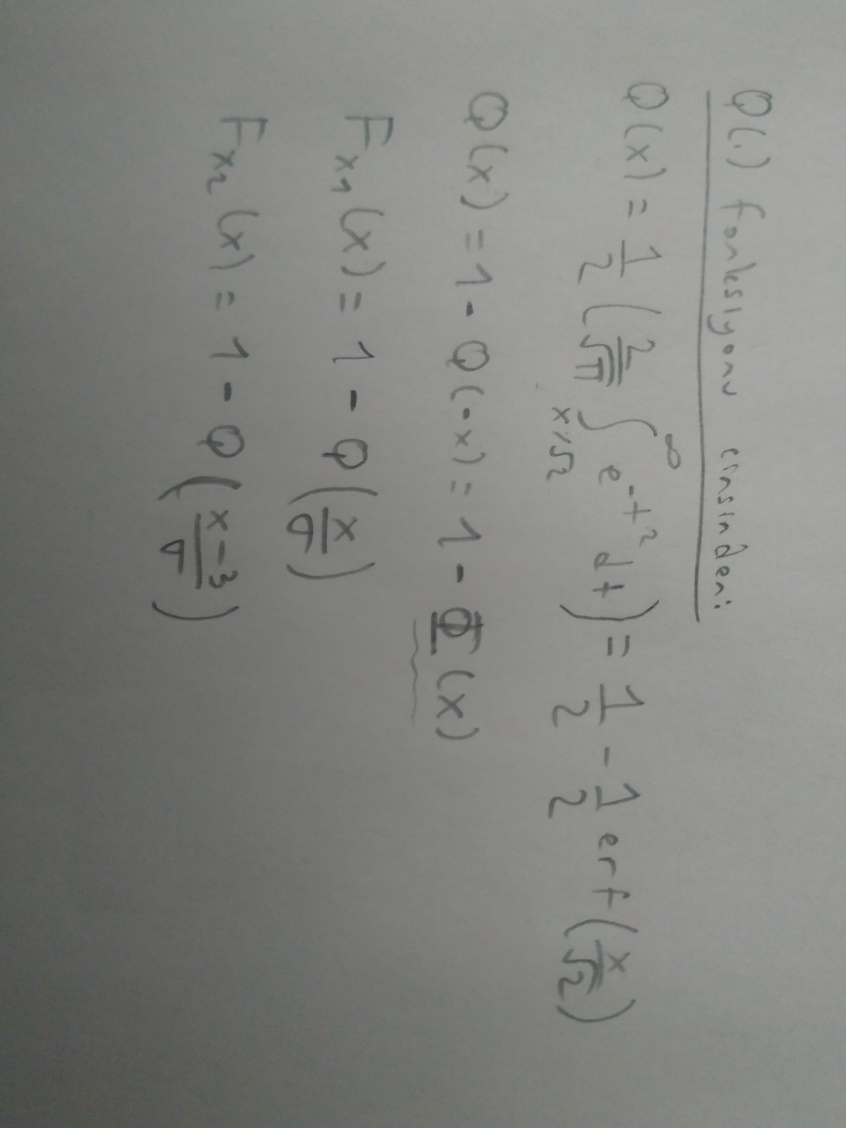




c)







d)

KOD:

# erf ile cdf hesaplayan fonksiyon

def cdf(x,ort,sigma):

    f = 0.5\*(1 + math.erf((x-ort)/(np.sqrt(sigma\*2))) )

    return f

# In[14]:

# fonksiyon sonuçlarının saklanacağı diziler

out\_0\_1 = np.zeros(100)

out\_0\_5 = np.zeros(100)

out\_3\_1 = np.zeros(100)

out\_3\_5 = np.zeros(100)

for i in range(100):

    out\_0\_1[i] = cdf(x\_ler[i],0,1)

    out\_0\_5[i] = cdf(x\_ler[i],0,5)

    out\_3\_1[i] = cdf(x\_ler[i],3,1)

    out\_3\_5[i] = cdf(x\_ler[i],3,5)

# In[17]:

# d şıkkı plotları

plt.title("0 ortalamalı X1 değişkeni gauss cdf'i")

plt.plot(x\_ler, out\_0\_1, label = "varyans = 1" )

plt.plot(x\_ler, out\_0\_5, label = "varyans = 5")

plt.xlabel("x")

plt.ylabel("FX1(x)")

plt.legend()

plt.show()

# In[18]:

# d şıkkı plotları

plt.title("3 ortalamalı X2 değişkeni gauss cdf'i")

plt.plot(x\_ler, out\_3\_1, label = "varyans = 1" )

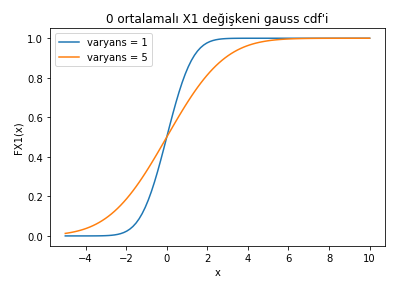
plt.plot(x\_ler, out\_3\_5, label = "varyans = 5")

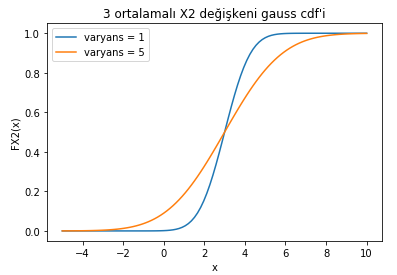
plt.xlabel("x")

plt.ylabel("FX2(x)")

plt.legend()

plt.show()





e) Olasılık hesabı için KOD:

# e şıkkı ihtimal hesapları :

prob = cdf(5,0,1) - cdf(1,0,1)

print("Varyans=1 için, Pr(5>X1>1) = " + str(prob))

prob = cdf(5,0,5) - cdf(1,0,5)

print("Varyans=5 için, Pr(5>X1>1) = " + str(prob))

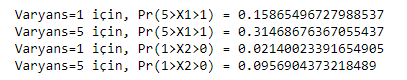
prob = cdf(1,3,1) - cdf(0,3,1)

print("Varyans=1 için, Pr(1>X2>0) = " + str(prob))

prob = cdf(1,3,5) - cdf(0,3,5)

print("Varyans=5 için, Pr(1>X2>0) = " + str(prob))

KOD çıktısı:



1. Soru

a)

KOD:

import numpy as np

from matplotlib import pyplot as plt

from scipy.fft import fft, fftfreq

# sinyal üretildi

S\_R = 1000

t = np.linspace(0, 1, S\_R, endpoint=False)

x\_t = np.cos((2 \* np.pi) \* t \* 1)

# işaret plotu

plt.title("x(t) işareti")

plt.plot(t, x\_t)

plt.xlabel("t[sn]")

plt.ylabel("x(t)")

plt.legend()

plt.show()

# darbe katarı üretildi

fs=10

ts= 1/fs

tao= 0.05

s\_t = np.zeros(S\_R)

for i in range(S\_R):

    if(i == S\_R):

        break

    if(i%100==0):

        for j in range(50):

            s\_t[i+j] = 1

#darbe plotu

plt.title("s(t) darbe katarı")

plt.plot(t,s\_t)

plt.xlabel("t[sn]")

plt.ylabel("s(t)")

plt.legend()

plt.show()

# xs(t) işaret plotu

xs\_t = s\_t\*x\_t

plt.title("x\_s(t) işareti")

plt.plot(t,xs\_t)

plt.xlabel("t[sn]")

plt.ylabel("x\_s(t)")

plt.legend()

plt.show()

# fourier dönüşümü yapılıyor

N = S\_R

yf = fft(x\_t) / S\_R

xf = fftfreq(N, 1 / S\_R)

plt.title("x(f) işareti genlik spektrumu")

plt.plot(xf, np.abs(yf))

plt.xlim([-50,50])

plt.xlabel("f[Hz]")

plt.ylabel("x(f)")

plt.legend()

plt.show()

yfs = fft(xs\_t) / S\_R

plt.title("x\_s(f) işareti genlik spektrumu")

plt.plot(xf, np.abs(yfs))

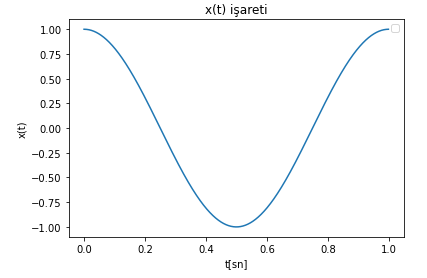
plt.xlim([-50,50])

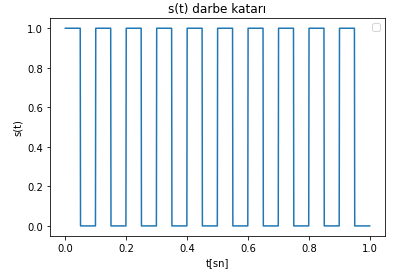
plt.xlabel("f[Hz]")

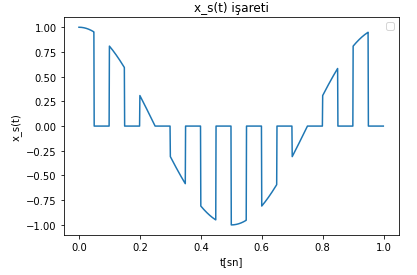
plt.ylabel("x\_s(f)")

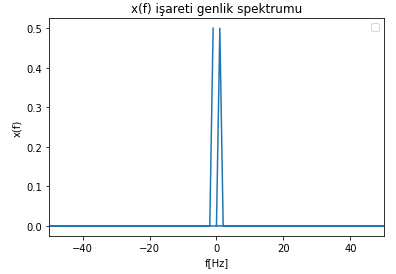
plt.legend()

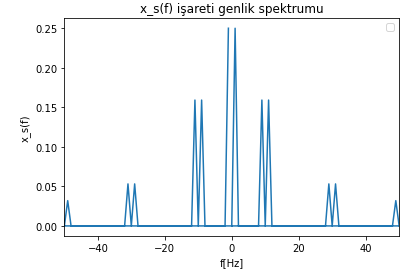
plt.show()











b)

KOD:

# b şıkkı için a(t) üretildi

a\_t = np.zeros(1000)

for i in range(1000):

    if(t[i%100] < 0.05):

        a\_t[i] = 50\*t[i%100] - 1.25

    else:

        a\_t[i] = 3.75 - (50\*t[i%100])

plt.title("a(t) işareti" )

plt.plot(t,a\_t)

plt.xlabel("t[sn]")

plt.ylabel("a(t)")

plt.legend()

plt.show()

# pam işareti gerçekleniyor

x\_pam = np.zeros(1000)

for i in range(1000):

    if(i%100==0):

        tut = x\_t[i]

    x\_pam[i] = tut

plt.title(" pam işareti x\_pam(t)")

plt.plot(t,x\_pam)

plt.xlabel("t[sn]")

plt.ylabel("x\_pam(t)")

plt.legend()

plt.show()

f\_t = x\_pam + a\_t

plt.title(" a(t) + x\_pam(t) işareti toplamı")

plt.plot(t,f\_t)

plt.xlabel("t[sn]")

plt.ylabel("x\_pam(t)+a(t)")

plt.legend()

plt.show()

# pdm işareti hesaplanıyor

x\_pdm = np.zeros(1000)

for i in range(1000):

    if(f\_t[i]<0):

        x\_pdm[i] = 0

    else:

        x\_pdm[i] = 1

plt.title(" x\_pdm(t) PDM işareti")

plt.plot(t,x\_pdm)

plt.xlabel("t[sn]")

plt.ylabel("x\_pdm(t)")

plt.legend()

plt.show()

yf\_pdm = fft(x\_pdm) / S\_R

plt.title(" PDM işareti genlik spektrumu")

plt.plot(xf, np.abs(yf\_pdm))

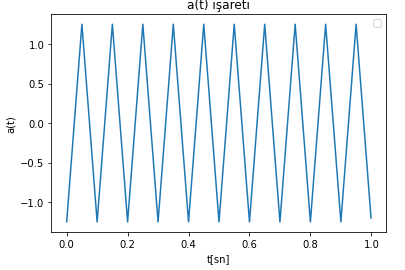
plt.xlim([-50,50])

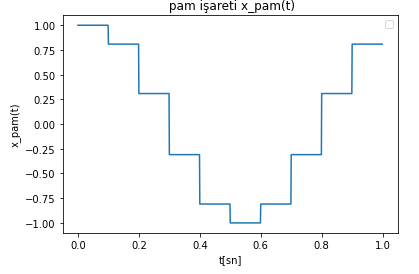
plt.xlabel("f[Hz]")

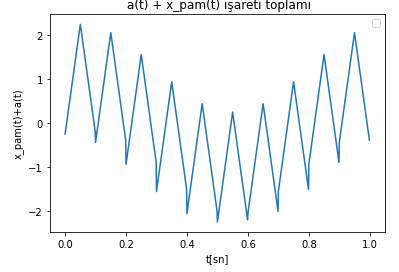
plt.ylabel("x\_pdm(f)")

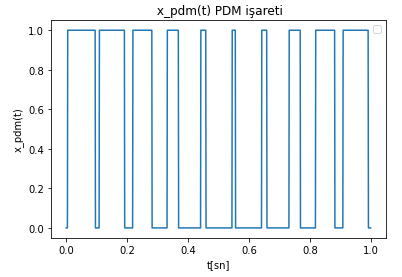
plt.legend()

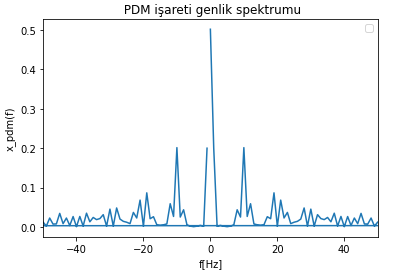
plt.show()











c)

KOD:

# c şıkkı için PPM işareti oluşturuldu

x\_ppm = np.zeros(1000)

for i in range(999):

    if(x\_pdm[i]==1 and x\_pdm[i+1]==0 ):

        # 3ms genlikte oluşturuldu:

        x\_ppm[i]=1

        x\_ppm[i+1]=1

        x\_ppm[i+2]=1

plt.title("PPM işareti")

plt.plot(t,x\_ppm)

plt.xlabel("t[sn]")

plt.ylabel("x\_ppm(t)")

plt.legend()

plt.show()

yf\_ppm = fft(x\_ppm)/S\_R

plt.title("PPM işareti genlik spektrumu")

plt.plot(xf, np.abs(yf\_ppm))

plt.xlim([-50,50])

plt.xlabel("f[Hz]")

plt.ylabel("x\_ppm(f)")

plt.legend()

plt.show()

int(kat)

x\_ppm2 = np.zeros(1000)

say = 0

for i in range(900):

    if(x\_pdm[i] == 1):

        say = say +1

    if( i % 100 == 0):

        kat = int(i/100)

        # 3ms lik tk darbesi

        x\_ppm2[i+say+50] = 1

        x\_ppm2[i+(say+1)+50] = 1

        x\_ppm2[i+(say+2)+50] = 1

        say = 0

plt.title("PPM işareti")

plt.plot(t,x\_ppm2)

plt.xlabel("t[sn]")

plt.ylabel("x\_ppm(t)")

plt.legend()

plt.show()

yf\_ppm = fft(x\_ppm2)/S\_R

plt.title("PPM işareti genlik spektrumu")

plt.plot(xf, np.abs(yf\_ppm))

plt.xlim([-50,50])

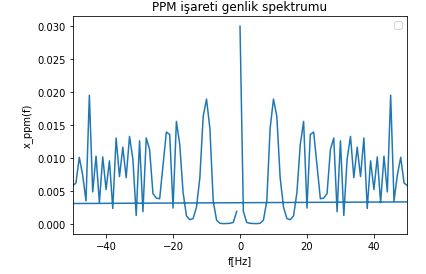
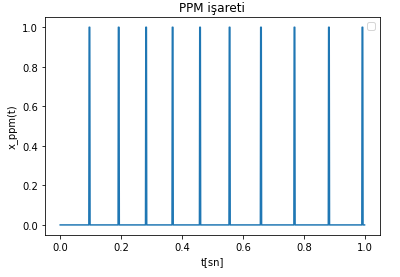
plt.xlabel("f[Hz]")

plt.ylabel("x\_ppm(f)")

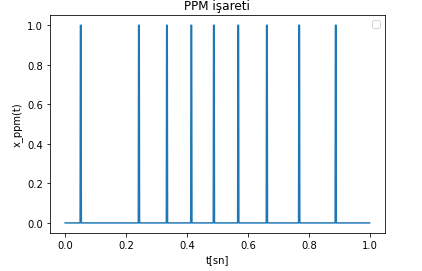
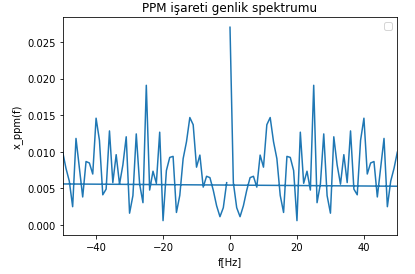
plt.legend()

plt.show()

Derste anlatılan metod ile gerçeklenmiş PPM işareti ve genlik yanıtı:



Soruda anlatılan metod ile gerçeklenmiş PPM işareti ve genlik yanıtı:

Ödevde hazırlanın sistemin avantajı daha düşük maksimum frekans genliğinde çalışması.

NOT: Kodu çalıştırmak için numpy, math, matplotlib ve scipy.fft kütüphanelerinin kurulu olması gereklidir. Kod, kurulum olmadan “Google Colab” üzerinden de çalıştırılabilir. Kodu sisteme istediğiniz üzere .py uzantılı tek dosya halinde yüklüyorum.

1. <https://www.mi.fu-berlin.de/inf/groups/ag-tech/teaching/2010-11_WS/L_19531_Telematics/slides03.pdf> ,Sayfa 4. [↑](#footnote-ref-1)
2. <https://www.elektronikhaberlesme.org/wp-content/uploads/2017/04/5.B%C3%B6l%C3%BCm-Hat-Kodlar%C4%B1-ve-Say%C4%B1sal-Haberle%C5%9Fme-Teknikleri.pdf> , Sayfa 3. [↑](#footnote-ref-2)
3. https://en.wikipedia.org/wiki/Line\_code [↑](#footnote-ref-3)
4. Gupta, Implementation and Analysis of Different Line Coding Schemes using Verilog , 5/ 2, (2016) , Sayfa 396. [↑](#footnote-ref-4)
5. https://co3moz.gitbooks.io/veri-iletisimi/content/HAFTA5.html [↑](#footnote-ref-5)
6. <https://www.elektronikhaberlesme.org/wp-content/uploads/2017/04/5.B%C3%B6l%C3%BCm-Hat-Kodlar%C4%B1-ve-Say%C4%B1sal-Haberle%C5%9Fme-Teknikleri.pdf> ,Sayfa 3. [↑](#footnote-ref-6)
7. <https://web.sonoma.edu/esee/courses/ee442/archives/sp2019/lectures/lecture11_line_coding.pdf> , Sayfa 12 [↑](#footnote-ref-7)