



Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Ciudad de México

Proyecto Final

Comunicaciones Digitales

Prof. Dr. Edgar Omar López Caudana

Integrantes:

| | |
|---------------------------|-----------|
| Alfredo Zhu Chen | A01651980 |
| Juan Pablo Valverde López | A01656127 |
| Andrés Islas Bravo | A01339391 |
| Luis Arturo Dan Fong | A01650672 |

Objetivo:

Desarrollar una herramienta de aprendizaje para la materia de Comunicaciones Digitales. Se desarrolla un código de MATLAB que consiste en varias secciones que contienen diversos temas de la materia y cada sección se corre de manera independiente.

Para ejecutar cada sección de código, se debe de seleccionar la sección del código y dar click en “Run Section”. Cada sección está dividida al poner “%% ” en el encabezado.

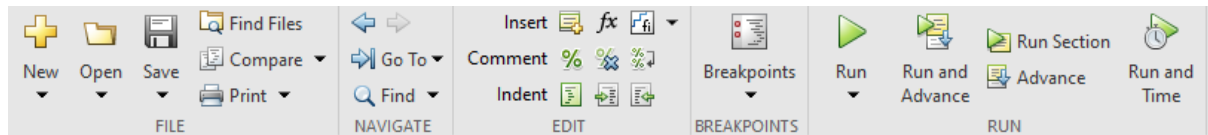


Fig. 1 Muestra de las opciones disponibles para ejecutar el código de MATLAB

Sección 1 del código:

Esta sección es para conocer si una frecuencia de muestreo permite que se recobre la señal con un filtro pasabandas de acuerdo al teorema de de muestreo pasabandas. Los parámetros que se pueden modificar en esta sección son los siguientes:

fu: frecuencia superior

fb: Ancho de banda

fs: frecuencia de muestreo

```
%% Teorema de muestreo pasabanda
clc
clear all
fu=25;%frecuencia superior
fb=10;%Ancho de banda
k=floor(fu/fb);%número entero más grande si exceder fu/fb
%fs=25;%Frecuencia de muestreo
fs=45;%Frecuencia de muestreo
if(fs==(2*fu/k))|
    display("Si se puede recobrar la señal con la frecuencia de muestreo indicada y con un filtro pasabanda")
else
    display("No se puede recobrar la señal con la frecuencia de muestreo indicada y con un filtro pasabanda")
end
```

Fig. 2 Código del teorema de muestreo pasabanda

Al ejecutar el código, se podrá observar que en la ventana de comandos muestra una respuesta si sí es posible o no recuperar la señal con un filtro pasabandas.

```
"No se puede recobrar la señal con la frecuencia de muestreo indicada y con un filtro pasabanda"
```

Fig. 3 Ventana de comandos del resultado del código del teorema de muestreo pasabanda

Sección 2 del código:

Esta sección del código permite calcular el número de niveles, tamaño de paso, la varianza(valor cuadrático medio), la razón señal a ruido y la razón señal a ruido en dB. Estos conceptos se vieron en el tema de cuantificación del primer parcial. Los parámetros que se pueden modificar son:

mp: Amplitud pico de la señal

R: número de bits por muestra

```
%% Cuantificación
clc
clear all
mp=1;%Amplitud pico
R=6;%número de bits por muestra(Resolución)
L=2^R;%número de niveles
delta=2*mp/L%Tamaño de paso
varianza=(mp^2)/(2^(2*R))/3
SNR_dB=1.8+6*R%Razón señal a ruido con carga de 1 ohm
SNR=10^(SNR_dB/10)
```

Fig. 4 Código que determina diversos parámetros de la cuantificación

```
delta =

    0.0312

varianza =

    8.1380e-05

SNR_dB =

    37.8000

SNR =

    6.0256e+03
```

Fig. 5 Ventana de comandos de los resultados del código que determina diversos parámetros de la cuantificación

Sección 3:

La tercera sección del código es útil para graficar cuantificaciones e PCM no lineales como el de la ley de μ y el de la ley A. Ambas gráficas muestran valores de entradas normalizadas. Los parámetros que se pueden modificar son μ y A.

```

%% Cuantificación no lineal con ley de mu y ley de A
clc
clear all
close all
entrada=0:0.01:1;%entrada normalizada
mu=5;
salida_mu=log10(1+mu.*entrada)/log10(1+mu)
plot(entrada,salida_mu)
xlabel("entrada normalizada")
ylabel("salida normalizada")
title("Ley mu")
A=100;
for i=1:length(entrada)
    if(entrada(i)<1/A)
        salida_A(i)=(A.*entrada(i))/(1+log10(A))
    else
        salida_A(i)=(1+log10(A*entrada(i)))/(1+log10(A))
    end
end
figure
plot(entrada,salida_A)
xlabel("entrada normalizada")
ylabel("salida normalizada")
title("Ley A")

```

Fig. 6 Código para graficar la cuantificación por ley de μ y por ley A

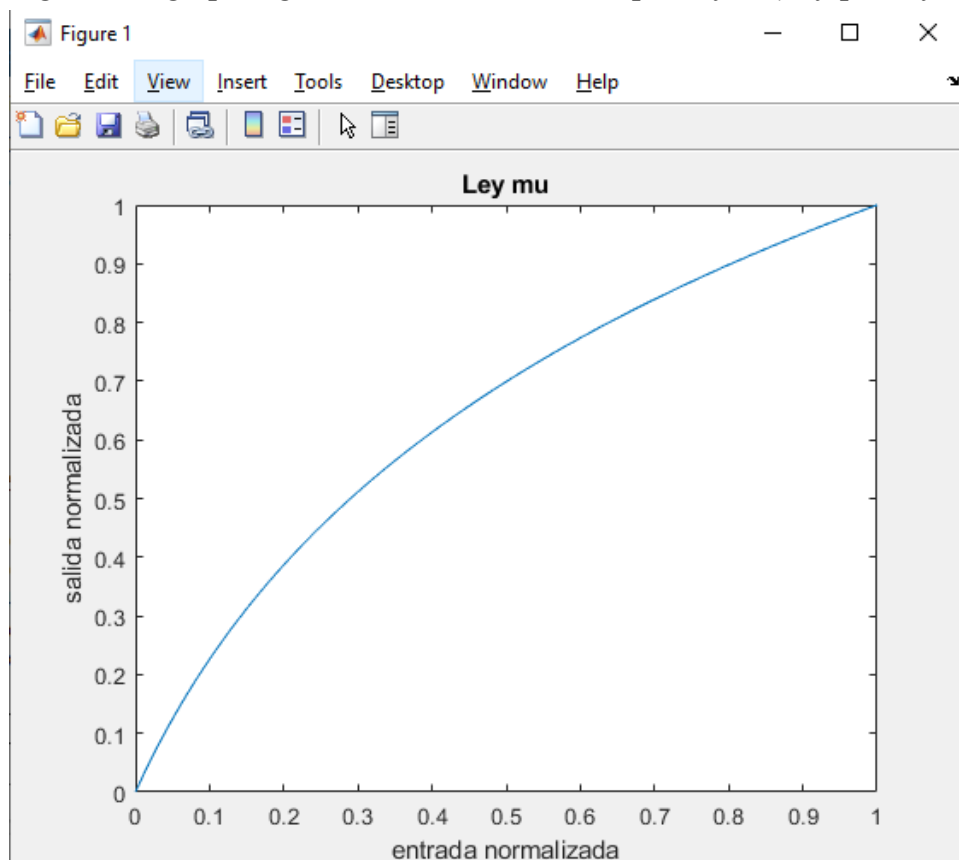


Fig. 7 Cuantificación en PCM por ley de μ

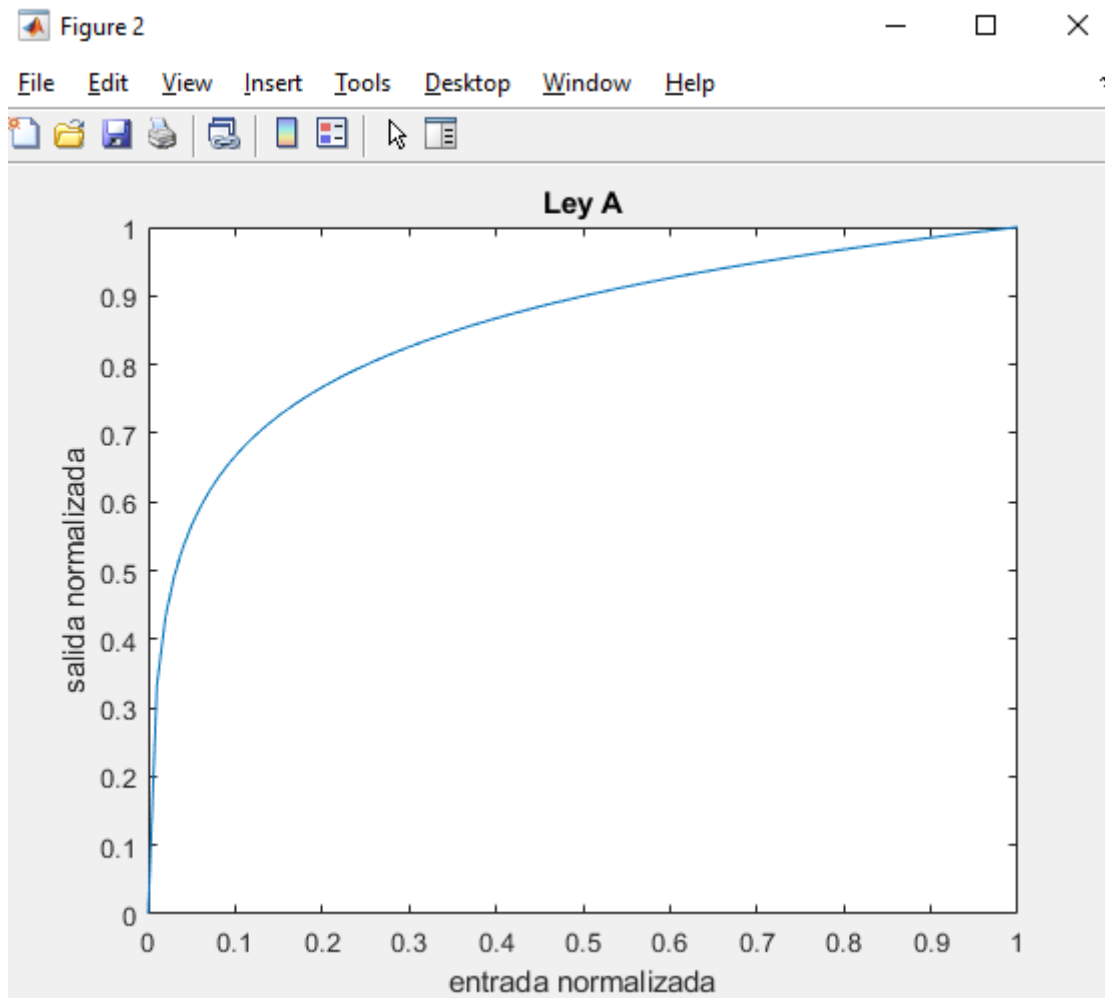


Fig. 8 Cuantificación en PCM por ley A

Sección 4 del código:

Esta sección del código determina la probabilidad de error de bit de diferentes esquemas de modulación. Los parámetros a variar son los siguientes:

A: Amplitud de la señal en V

N0: Ruido en W/Hz, notar que aquí se toma $N0/2$

R: La tasa de transmisión en bps

```
% BER de diferentes esquema de modulación
A=10^-6;%Amplitud de la señal en V
N0=2*10^-20;%N0/2, ruido en W/Hz
R=2.5*10^6;%tasa de transmisión en bps
T=1/R;%periodo
Eb=0.5*A^2/R
Pe_FSK_binaria_coherente=0.5*erfc(sqrt(Eb/(2*N0)))*FSK binaria coherente,ASK
Pe_MSK_coherente=0.5*erfc(sqrt(Eb/(N0)))*MSK coherente,QPSK coherente, FSK binaria coherente,QPSK
Pe_FSK_binaria_no_coherente=0.5*exp(-Eb/(2*N0))*FSK binaria no coherente
Pe_DPSK=0.5*exp(-Eb/N0)*DPSK
```

Fig. 8 Cálculo del BER para diferentes esquemas de modulación

Se calcula y se muestra en la ventana de comandos de MATLAB la probabilidad de error para modulaciones como FSK binaria coherente, MSK, FSK binaria no coherente, DPSK y entre otras.

```

Pe_FSK_binaria_coherente =

    7.8270e-04

Pe_MSK_coherente =

    3.8721e-06

Pe_FSK_binaria_no_coherente =

    0.0034

Pe_DPSK =

    2.2700e-05

```

Fig. 9 Resultados del código que determina los BER de diferentes esquemas de modulación

Sección 5 del código:

Esta sección hace un proceso similar a la sección anterior, la diferencia es que se puede mostrar las gráficas de las probabilidades de error de bit en la misma figura. Esto nos permite analizar y realizar una comparación del funcionamiento de los esquemas.

```

%% Curvas de desempeño de error de diferentes esquemas de modulación
%%Pe vs Eb/N0
clc
clear all
close all
Eb_N0_dB=0:0.1:12.5;%Rango de la razón señal a ruido Eb/N0
Eb_N0=10.^(Eb_N0_dB/10);%Razón señal a ruido en dB
Pe_coherent_BFSK=1/2.*erfc(sqrt(Eb_N0));%FSK binaria coherente
Pe_coherent_BFSK=1/2.*erfc(sqrt(Eb_N0/2));%FSK binaria coherente
Pe_QPSK=1/2.*erfc(sqrt(Eb_N0));%QPSK
Pe_MSK=1/2.*erfc(sqrt(Eb_N0));%MSK
Pe_DPSK=1/2.*exp(-Eb_N0);%DPSK
Pe_noncoherent_BFSK=1/2.*exp(-Eb_N0/2);%FSK binaria no coherente
semilogy(Eb_N0_dB,Pe_noncoherent_BFSK,Eb_N0_dB,Pe_coherent_BFSK,Eb_N0_dB,Pe_DPSK,Eb_N0_dB,Pe_coherent_BFSK,Eb_N0_dB,Pe_QPSK,Eb_N0_dB,Pe_MSK);
grid on
legend("Non coherent binary FSK","Coherent binary FSK","DPSK","","","Coherent binary PSK,QPSK,MSK")
ylabel("Probabilidad de error Pe");
xlabel("E_b/N_0 (dB)")
title("Comparación de diferentes esquemas de modulación Pe vs E_b/N_0 (dB)")
ylim([10^-4 1])%Establecer rango de la figura en y

```

Fig. 10 Código para graficar los BER de diferentes esquemas de modulación

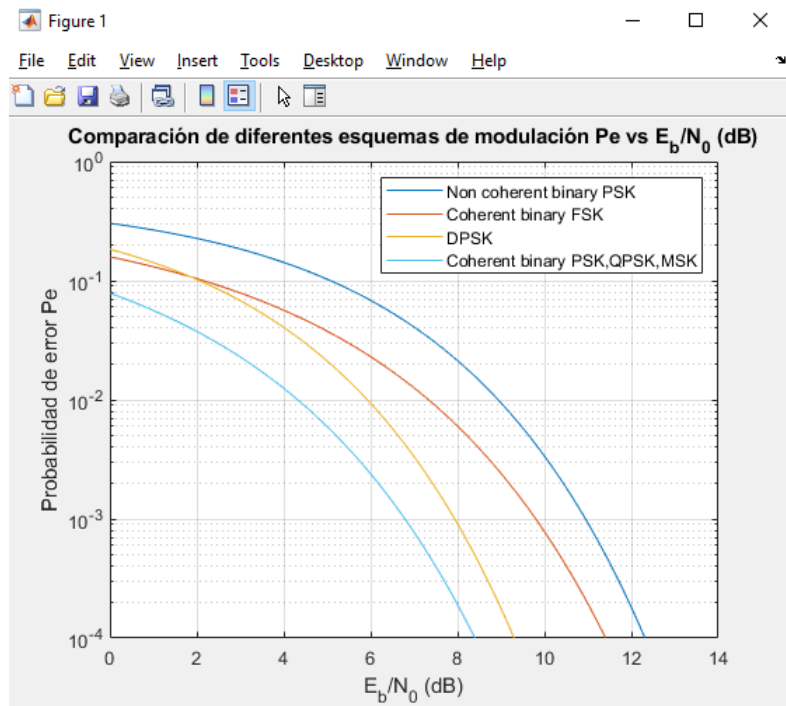


Fig. 11 Comparación de diferentes esquemas de modulación

Sección 6 del código:

La presente sección determina la eficiencia del ancho de banda. Se usa el parámetro la tasa de transmisión en bps y también el tiempo de bit. El tiempo de bit es utilizado para determinar el ancho de banda.

Rb:transmisión en bps

Tb: Tiempo de bit en segundos

```
%% Eficiencia de ancho de banda
clc
close all
clear all
Rb=2.5*10^6;%tasa de transmisión en bps
Tb=5e-6;
B=1/(2*Tb);%Ancho de banda
rho=Rb/B
```

Fig. 12 Código para determinar la eficiencia de ancho de banda

```
rho =

    25.0000
```

Fig. 13 Resultado de la eficiencia de ancho de banda con los valores especificados en Fig. 12

Sección 7 del código:

Uno de los temas importantes en el curso es el espectro disperso, por lo cual se presenta en esta sección el cálculo de la secuencia de pseudo ruido utilizando Flip-flops. Como valores de entrada, se encuentran los siguientes:

m: número de flip flops en el circuito

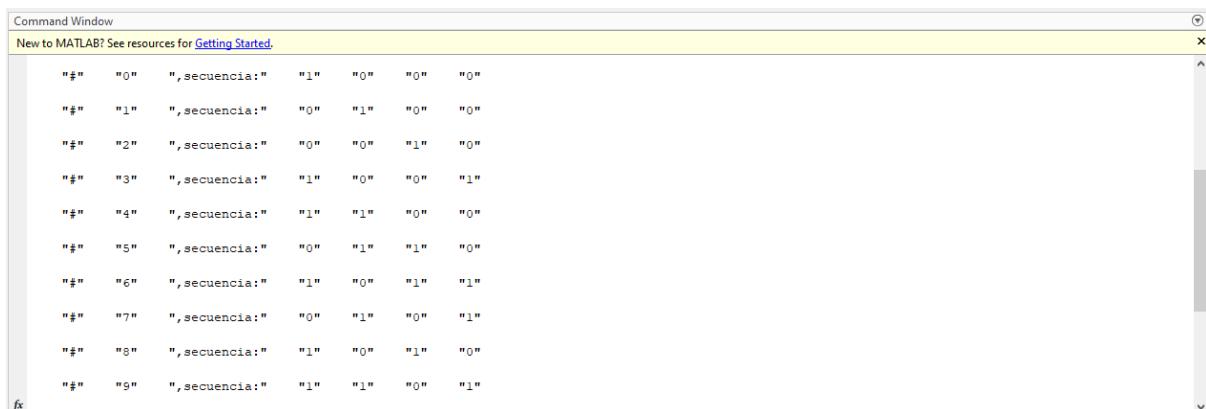
modulo2_adder: es un vector que contiene 1 cuando se encuentra un sumador módulo 2 la salida del flip-flop.

Shifter: Aquí se almacena el estado inicial del sistema en un vector.

Los dos vectores tienen que tener el número de elementos que se indica en m.

```
%% Secuencia de pseudo ruido con Flip Flops
m=4;%Numero de flip flops
modulo2_adder=[0,0,1,1];%posiciones de sumadores modulo 2
shifter=[1,0,0,0]%Estado inicial
fl=1%inicialización de variable
for i=1:2^m
    disp(["#",i-1,"secuencia:",shifter])
    a=fl;
    fl=0;
    for j=1:length(shifter)
        if(modulo2_adder(j)==1)
            fl=xor(fl,shifter(j));%%Modulo XOR
        end
    end
    for j=length(shifter)-1:-1:0
        shifter(j+1)=shifter(j);%%Recorrimiento
    end
    shifter(1)=a;%Recorrer el segundo
    shifter(2)=fl;%Recorrer el primero
end
```

Fig. 14 Sección del código que determina la secuencia de pseudo ruido



The screenshot shows the MATLAB Command Window with the following output:

| Iteration | State (shifter) |
|-----------|-----------------|
| 0 | 1 0 0 0 |
| 1 | 0 1 0 0 |
| 2 | 0 0 1 0 |
| 3 | 1 0 0 1 |
| 4 | 1 1 0 0 |
| 5 | 0 1 1 0 |
| 6 | 1 0 1 1 |
| 7 | 0 1 0 1 |
| 8 | 1 0 1 0 |
| 9 | 1 1 0 1 |

Fig. 15 Ejemplo de los primeros 9 estados del registro, la secuencia de pseudo ruido es el último valor de cada estado.

Sección 8:

```
%% Margen de perturbación
clc
clear all
close all
Tb=4.095e-3;%duración de bit de información
Tc=1e-6;%duración del recorte del pseudo ruido
N=Tb/Tc;%Periodo o también conocido como ganancia de procesamiento
PG=N%Ganancia de procesamiento
m=log2(N+1)%Longitud de registro de corrimiento
SNR=3.1;%
PJ_margenPerturbacion=10*log10(PG)-10*log10(SNR)
```

Fig. 16 Código de Margen de perturbación.

La sección 8 se puede utilizar para calcular la ganancia de procesamiento y también para determinar el margen de perturbación en dB. Los parámetros de entrada son:

Tb: duración de bit de información

Tc: duración del recorte del pseudo ruido

```
PG =
    4095

m =
    12

PJ_margenPerturbacion =
    31.2089
```

Fig. 17 Resultado del margen de perturbacion

Sección 9:

Uno de los temas importantes en el tercer parcial es la entropía, por lo tanto se realizó la siguiente parte del código para eso. Básicamente se calcula la entropía, La longitud de cada onda y también vienen varios refuerzos para saludarlos.

```
%% Entropia
clear all
clc
close all
probabilidades=[1/2 1/8 1/8 1/16 1/16 1/16 1/32 1/32];
L=[1 3 3 4 4 5 6 6]
H=sum(probabilidades.*log2(1./probabilidades))%bits por símbolo
L_bar=sum(probabilidades.*L)
Eficiencia=H/L_bar
```

Fig 18: Código de para la entropía

```
H =  
  
2.3125  
  
L_bar =  
  
2.4375  
  
Eficiencia =  
  
0.9487
```

Fig. 19 Resultados del cálculo de la entropía

Conclusiones:

Alfredo Zhu Chen: El objetivo de este proyecto se cumplió. Pude desarrollar con mi equipo un código de MATLAB que abarca la gran mayoría de los temas vistos en la materia de Comunicaciones Digitales. El código puede servir como una herramienta de estudio y de repaso para la materia. Además, gracias a la sencillez de su uso, sirve como una herramienta para calcular diferentes parámetros importantes. Me agradó mucho poder desarrollar este proyecto con mi equipo. Cada integrante del equipo aportó buenas ideas y cada uno dió su esfuerzo para cumplir con el objetivo. Espero que esta herramienta pueda ser utilizada por otras personas y les sirva como un complemento al aprendizaje de la materia.

Juan Pablo Valverde López: MATLAB es una herramienta de desarrollo matemático capaz de simular tanto gráficamente como por código funciones y relaciones físicas que explican fenómenos de la comunicación digital. Desde formas de transferencia de la información, probabilidad de error (Bit Error Rate). Este proyecto compila las principales temáticas vistas en clase y las resume en un script de MATLAB para que futuras generaciones puedan utilizarlo y mejorarlo. Tanto la materia como este proyecto, es importante para el perfil de telecomunicaciones en donde cada uno se encuentra, el cual es muy apetecido en el mercado laboral. Gracias por tanto y perdón por tan poco.

Andrés Islas Bravo: La realización de este proyecto, más allá de ser una recopilación de aspectos importantes del semestre, espero le sea de utilidad a las futuras generaciones y les facilite diversos cálculos. Finalmente, considero que MATLAB es de las herramientas más importantes en el ámbito ingenieril, entonces el haber hecho este proyecto en esa plataforma le hará útil e universal para su aplicación.

Luis Arturo Dan Fong: Considero que aprender a utilizar una herramienta como lo es MATLAB es fundamental en nuestro desarrollo como ingenieros, ya que esta nos permite desarrollar problemas complejos a través de una interfaz de programación amigable al usuario. Personalmente disfruté mucho poder aprender sobre comunicaciones digitales este semestre y me divertí al poder utilizar las herramientas que se nos presentaron como lo fueron los simuladores y el desarrollo de este código de matlab, lo cual me permitió asentar los conocimientos adquiridos y aplicarlos a un proyecto con el cual me siento satisfecho. Me siento muy contento con nuestro desempeño como equipo y el proyecto en general.

Anexos:

%Proyecto final de Comunicaciones Digitales

%%Alfredo Zhu, Juan Pablo Valverde, Luis Arturo Dan, Andrés Islas

clc

clear all

%% Teorema de muestreo pasabanda

clc

clear all

fu=25;%frecuencia superior

fb=10;%Ancho de banda

k=floor(fu/fb);%número entero más grande si exceder fu/fb

%fs=25;%Frecuencia de muestreo

fs=45;%Frecuencia de muestreo

if(fs==(2*fu/k))

display("Sí se puede recobrar la señal con la frecuencia de muestreo indicada y con un filtro pasabanda")

else

display("No se puede recobrar la señal con la frecuencia de muestreo indicada y con un filtro pasabanda")

end

%% Cuantificación

clc

clear all

mp=1;%Amplitud pico

R=6;%número de bits por muestra(Resolución)

L=2^R;%número de niveles

delta=2*mp/L%Tamaño de paso

varianza=(mp^2)/(2^(2*R))/3

SNR_dB=1.8+6*R%Razón señal a ruido con carga de 1 ohm

SNR=10^(SNR_dB/10)

%% Cuantificación no lineal con ley de mu y ley de A

```

clc
clear all
close all
entrada=0:0.01:1;%entrada normalizada
mu=5;
salida_mu=log10(1+mu.*entrada)/log10(1+mu)
plot(entrada,salida_mu)
xlabel("entrada normalizada")
ylabel("salida normalizada")
title("Ley mu")
A=100;
for i=1:length(entrada)
    if(entrada(i)<1/A)
        salida_A(i)=(A.*entrada(i))/(1+log10(A))
    else
        salida_A(i)=(1+log10(A*entrada(i)))/(1+log10(A))
    end
end
figure
plot(entrada,salida_A)
xlabel("entrada normalizada")
ylabel("salida normalizada")
title("Ley A")
%% BER de diferentes esquema de modulación
A=10^-6;%Amplitud de la señal en V
N0=2*10^-20;%N0/2, ruido en W/Hz
R=2.5*10^6;%tasa de transmisión en bps
T=1/R;%periodo
Eb=0.5*A^2/R
Pe_FSK_binaria_coherente=0.5*erfc(sqrt(Eb/(2*N0)))%FSK binaria coherente,ASK
Pe_MSK_coherente=0.5*erfc(sqrt(Eb/(N0)))%MSK coherente,QPSK coherente, PSK
binaria coherente,QPSK
Pe_FSK_binaria_no_coherente=0.5*exp(-Eb/(2*N0))%FSK binaria no coherente
Pe_DPSK=0.5*exp(-Eb/N0)%DPSK
%% Curvas de desempeño de error de difernetes esquemas de modulación
%%Pe vs Eb/N0
clc
clear all
close all
Eb_N0_dB=0:0.1:12.5;%Rango de la razón señal a ruido Eb/N0

```

```

Eb_N0=10.^(Eb_N0_dB/10);%Razón señal a ruido en dB
Pe_coherent_BPSK=1/2.*erfc(sqrt(Eb_N0));%PSK binaria coherente
Pe_coherent_BFSK=1/2.*erfc(sqrt(Eb_N0/2));%FSK binaria coherente
Pe_QPSK=1/2.*erfc(sqrt(Eb_N0));%QPSK
Pe_MSK=1/2.*erfc(sqrt(Eb_N0));%MSK
Pe_DPSK=1/2.*exp(-Eb_N0);%DPSK
Pe_noncoherent_BFSK=1/2.*exp(-Eb_N0/2);%FSK binaria no coherente
semilogy(Eb_N0_dB,Pe_noncoherent_BFSK,Eb_N0_dB,Pe_coherent_BFSK,Eb_N0_dB,Pe_DPSK,Eb_N0_dB,Pe_coherent_BPSK,Eb_N0_dB,Pe_QPSK,Eb_N0_dB,Pe_MSK);
grid on
legend("Non coherent binary PSK","Coherent binary FSK","DPSK","", "", "Coherent binary PSK,QPSK,MSK")
ylabel("Probabilidad de error Pe");
xlabel("E_b/N_0 (dB)")
title("Comparación de diferentes esquemas de modulación Pe vs E_b/N_0 (dB)")
ylim([10^-4 1])%Establecer rango de la figura en y
%% Eficiencia de ancho de banda
clc
close all
clear all
Rb=2.5*10^6;%tasa de transmisión en bps
Tb=5e-6;
B=1/(2*Tb);%Ancho de banda
rho=Rb/B
%% Secuencia de pseudo ruido con Flip Flops
m=4;%Numero de flip flops
modulo2_adder=[0,0,1,1];%posiciones de sumadores modulo 2
shifter=[1,0,0,0]%Estado inicial
f1=1%inicialización de variable
for i=1:2^m
    disp(["#",i-1," ,secuencia:",shifter])
    a=f1;
    f1=0;
    for j=1:length(shifter)
        if(modulo2_adder(j)==1)
            f1=xor(f1,shifter(j));%%Modulo XOR
        end
    end
end
for j=0:length(shifter)-2

```

```

        shifter(length(shifter)-j)=shifter(length(shifter)-j-1);%%Recorrimiento
    end
    shifter(2)=a;%%Recorrer el segundo
    shifter(1)=f1;%%Recorrer el primero
end
%% Margen de perturbación
clc
clear all
close all
Tb=4.095e-3;%duración de bit de información
Tc=1e-6;%duración del recorte del pseudo ruido
N=Tb/Tc;%Periodo o también conocido como ganancia de procesamiento
PG=N%Ganancia de procesamiento
m=log2(N+1)%Longitud de registro de corrimiento
SNR=3.1;%
PJ_margenPerturbacion=10*log10(PG)-10*log10(SNR)
%% Entropia
clear all
clc
close all
probabilidades=[1/2 1/8 1/8 1/16 1/16 1/16 1/32 1/32];
L=[1 3 3 4 4 5 6 6]
H=sum(probabilidades.*log2(1./probabilidades))%bits por símbolo
L_bar=sum(probabilidades.*L)
Eficiencia=H/L_bar

```