ITESO
Universidad Jesuita
de Guadalajara

Sistemas de Comunicaciones Digitales

Práctica 1 (Versión 2020)

Raúl Ignacio Baltazar Morán ie710786
Luis Fernando Rodriguez Gutierrez ie705694

Omar Humberto Longoria Gándara

30/03/2020

Raúl Ignacio Baltazar Morán IE710786
Luis Fernando Rodriguez Gutierrez IE705694

Objetivo.

El objetivo de esta práctica es comprobar algunas de las ventajas (y desventajas) que tiene la transmisión digital de la información, en comparación con la transmisión analógica. Hasta ahora hemos estudiado muchos temas en clase relacionados con la transmisión digital de datos: conversión analógica a digital (muestreo, cuantificación), codificación PCM, filtros formadores de pulsos, ancho de banda, errores causados por el efecto del ruido, etcétera. Sin embargo, todavía no tenemos una comprobación de que todo esto, que tiene complejidad considerable, tiene un beneficio real en comparación a las comunicaciones analógicas. Esta práctica nos va a mostrar estos beneficios.

Enunciado.

Primero vamos a simular en Matlab la transmisión analógica de una señal de audio como lo hicimos en una de las tareas. La transmisión se realizará a través de varios canales con ruido AWGN, para diversos valores de N0. Vamos a escuchar el audio después de la transmisión para cada valor de SNR. Después, vamos a transmitir la misma señal, pero en forma digital. La vamos a transmitir por canales con los mismos niveles de ruido N0/2, y vamos a escuchar la señal después de la transmisión. Si la transmisión digital realmente tiene ventajas sobre la analógica, entonces la señal transmitida digitalmente debe escucharse mejor que la analógica, para el mismo valor de SNR. Como puedes ver, ahora la figura de mérito o criterio de desempeño es una medida subjetiva: “cómo se escucha”. Las mediciones de SNR se realizan después del filtro receptor. Vamos a suponer que ambos sistemas (digital y analógico) utilizan el mismo amplificador y por lo tanto transmiten a la misma potencia. Vamos a suponer también que ambos sistemas tienen exactamente el mismo ancho de banda, B = 15 kHz, igual que una estación de FM.

Transmisión analógica.

Analice la Figura 1. La idea es simular la transmisión analógica de una señal de audio almacenada en un archivo WAV/Flac. Vamos a filtrar la señal a 15 kHz, en el canal se le añadirá ruido, y después se filtró de nuevo en el receptor.

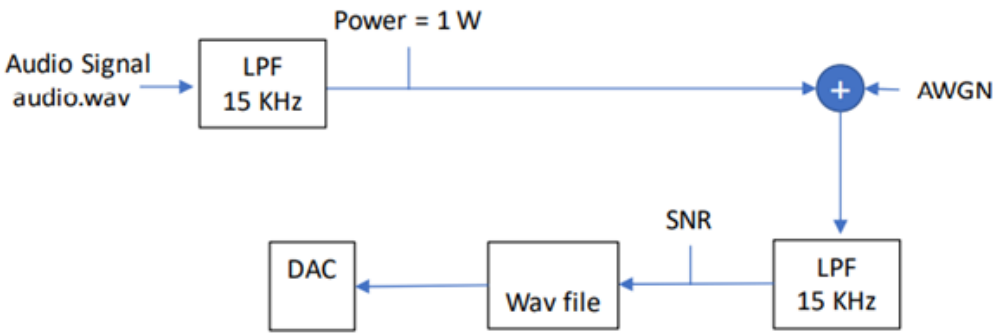


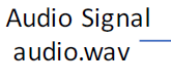
Figura 1. Modelo de un sistema de comunicaciones analógicas

La potencia de la señal transmitida (después de filtrar a 15 kHz) Ps se normaliza a un watt. El ruido blanco presente en el canal no se puede simular, pero si conocemos su PSD (es decir los valores de N0), podemos predecir la potencia que tendrá a la salida del filtro

receptor. Esta potencia es Pn = BN0, donde B es el ancho de banda del filtro. La señal a la salida del filtro receptor tiene una relación señal a ruido igual a Ps/Pn. La señal recibida, igual a audio más ruido, se conecta directamente a un sistema de sonido. Este es el esquema que se usa en transmisión analógica convencional.

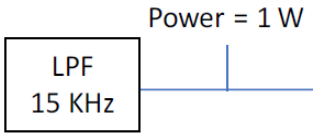
Para poder realizar la transmición analogica seguimos el siguinete proceso

1. **Obtención del archivo WAV**




```
filename= 'Never_Gonna.wav';    %audio to be reproduce

fir_order = 100;    %filter order (15kHz)
Fs_audio = 44100;   %sample frequency of audio
Ts_audio = 1/Fs_audio;  %period of the sample audio
t = 10; %t = 10 seconds
samples = [1,t*Fs_audio]; %vector of samples
[x,Fs] = audioread(filename,samples);   %get wav file
info = audioinfo(filename); %get info of the wav file
info_bps = info.BitsPerSample; %info_bps (info_bits per sample)

%Signal to Mono
xMono = (x(:,1)+ x(:,2))/2; %convert audio stero to audio mono
```

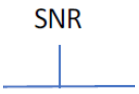
2. Filtrado de la señal y normalización de potencia




```
%% Filtrar la señal
    %LPF Comunication Channel
B = 15e3; %band pass of filter
Fc = B / (Fs_audio/2); %Normalized Freq 0.6803

fc_15KHz = [0 Fc Fc 1]; %Freq of the filter
m = [1 1 0 0];  %Magnitud of filter
order = fir_order; %order of teh filter
LPF = fir2(order, fc_15KHz, m); % LPF (Low Pass Filter)
xa = filter(LPF, 1, xMono); %analog signal filtered
```

```
%% Normalize the Power
pot = sum(xa.*xa)/numel(xa);  % Calculate the power of analog signal
xa = xa/sqrt(pot);  %Unitari normalization of power
P_signal = var(xa); %Check power
```

3. SNR




```
%% SNR
N0 = 1./(B.*10.^(0:0.3:3)); % PSD vector of noise
P_noise = B*N0; % Power of filtered noise
P_noise_dB = 10.*log10(P_noise); % Power of noise in dB
SNR_A = P_signal ./ P_noise; % Signal to Noise ratio
SNR_A_dB = 10*log10(SNR_A); % SNR in dB
```

4. Add up AWGN y obtención del archivo con ruido


```
%% AWGN (White noise all frequency)
% Add up noise to the analog signal, for each value of power density of noise
for i = 1 : numel(N0)
    noise = sqrt(P_noise(i)) * randn(1, numel(xa)); % noise samples for each value noise power
    xa_noised = xa + noise';     % Add up noise to the signal
    xa_noised = xa_noised ./ max(abs(xa_noised)); % Normilized the signal between 1 & -1
    %save name of the new file that contain the audio signal + noise
    filename = strcat('AnalogSignal_', num2str(i), '_', num2str(round(SNR_A_dB(i)), 4), 'dB','.wav');
    audiowrite(filename, xa_noised, Fs); %get new file
end

% Reproduce sound
%soundsc(xMono,Fs_audio);
```

Transmisión digital.

Ver la figura (2). Vamos a transmitir la misma señal de audio, pero ahora de forma digital. El archivo WAV/Flac ya representa una señal muestreada, cuantificada, y en código PCM (con una frecuencia de muestreo y un número de bits por muestra). Lo que resta por hacer es convertir en bits, luego a pulsos, sumarle ruido, recuperar los bits, y volver a grabar el archivo WAV/Flac. Vamos a suponer que el canal presenta los mismos valores de N0 que en la transmisión analógica.

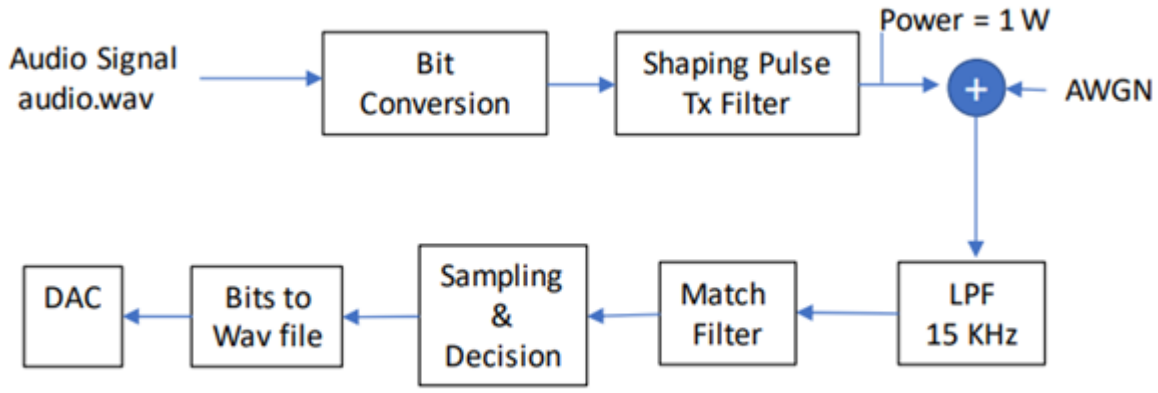
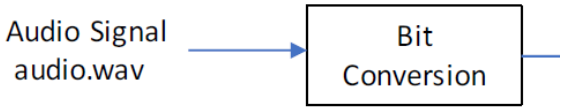


Figura 2. Modelo de un sistema de comunicaciones digitales

Para la transmisión digital seguimos los siguientes pasos

1. Archivo WAV, conversión a bits




```
%% Digital Transmition
    %Source: https://www.mathworks.com/matlabcentral/answers/75379-how-to-convert-an-input-sine-wave-into-an-8-bit-digital-signal
    %Source: https://www.mathworks.com/matlabcentral/answers/103315-how-to-convert-digital-data-into-analog-data-using-matlab-code
    % Convert values from xMono to bin

swing = (2^info_bps-1)/2; %offset to be added
xMono2bi = round(xMono*swing+swing);    %add offset to digital audio
    %For Default it is the 'right-msb'
b = de2bi(xMono2bi,info_bps,'left-msb');    %pass decimal to binari
b = b'; %Transponse operation
bits_tx = b(:); %Concatenate the rest of the bits, to make it a single vector
```

2. Generación de pulso

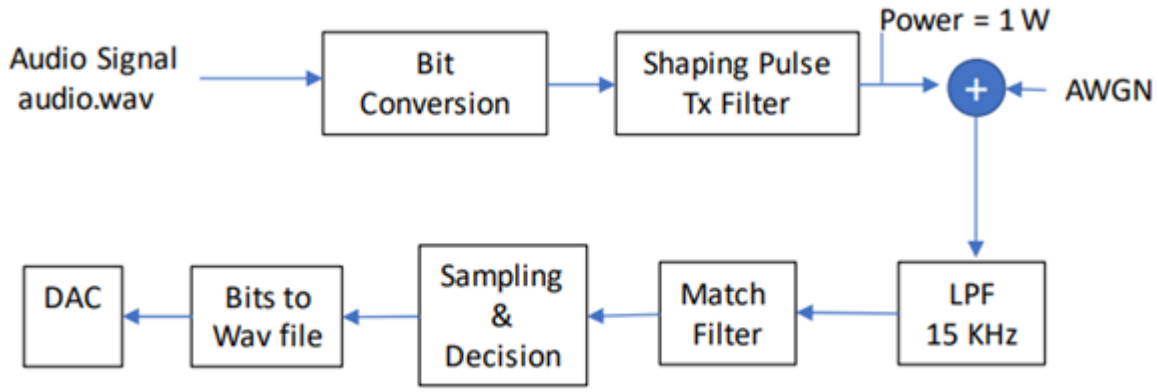
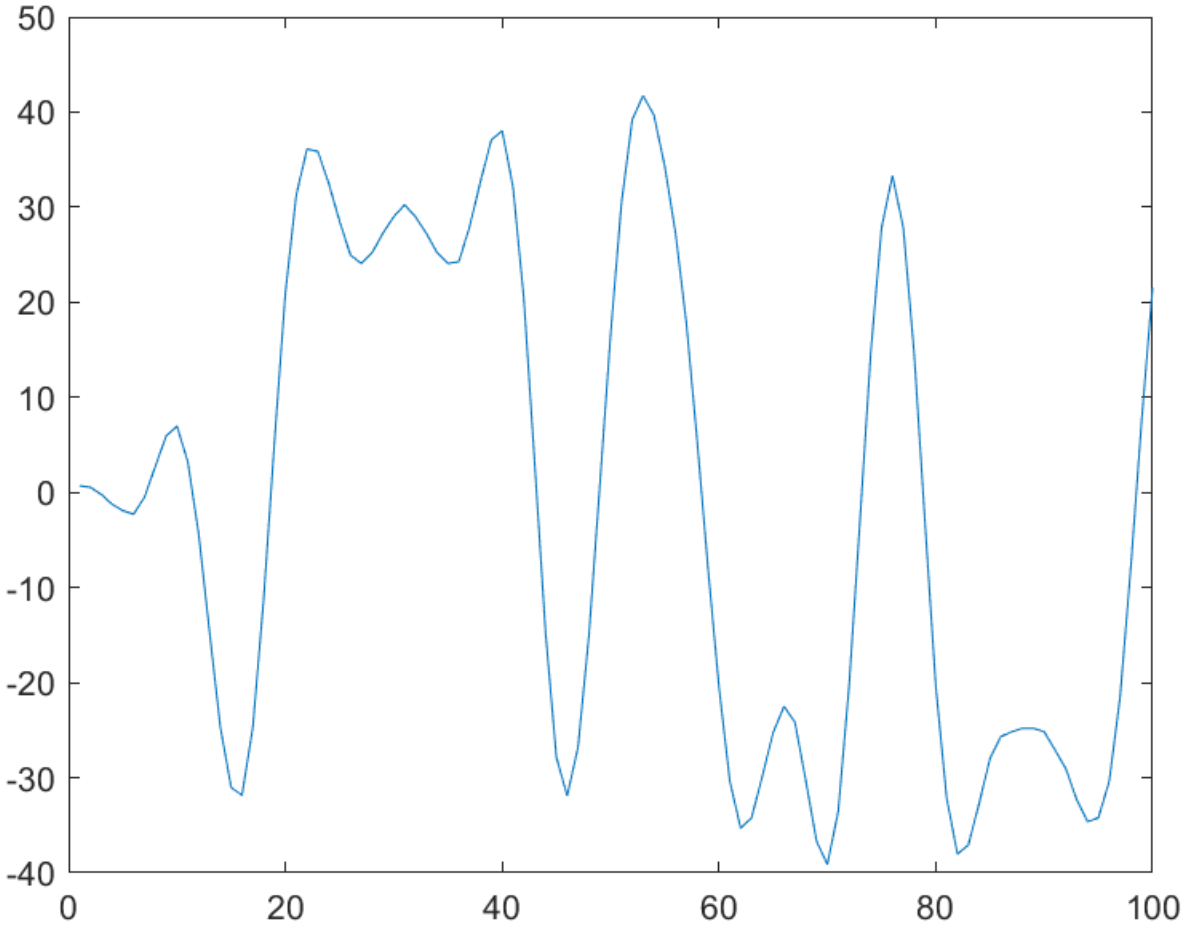


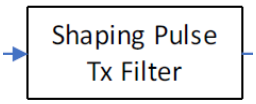
Figura 2. Modelo de un sistema de comunicaciones digitales

```
%% Shapping Pulse
beta = 0.35;
D = 6;
Fs = 96e3;
Rb = (2*B) / (1+beta);
Ts = 1/Fs;
mp = ceil(Fs/Rb); %mp = 5;
Tp = 1/(Fs/mp);
energy = Tp;

[p,t] = rcpulse(beta, D, Tp, Ts, 'srrc', energy); %Genetare
stem(p);
p_energy = Ts*sum(p.*p);    %Calculate the power of the SRRC pulse
p = p/sqrt(p_energy); %Unitari Normalization of SRRC pulse power
```

```
%% Code Line
    %Polar Signal NRZ
PBP = p; % PSPB (Polar Base Pulse)
PS_s = bits_tx; %auxiliar vector that contain bits information
PS_s( PS_s == 0 ) = -1; %convert binary 0 to -1 v
s = zeros(1,numel(PS_s)*mp);    %Vector with the total size
s(1:mp:end) = PS_s;     %Impulse train
PSLC = conv(PBP, s) / mp; %PSLC (Polar Signal Line Code), Pulse train

plot(PSLC(1:mp*20));

    %Normilize pulse
PSLC = PSLC/sqrt( sum(PSLC.*PSLC)/numel(PSLC) ); %Unitari Normalization of L
P_PSLC_d = var(PSLC); %Power of Line code digital
```


50
40
30
20
10
0
-10
-20
-30
-40
0
20
40
60
80
100

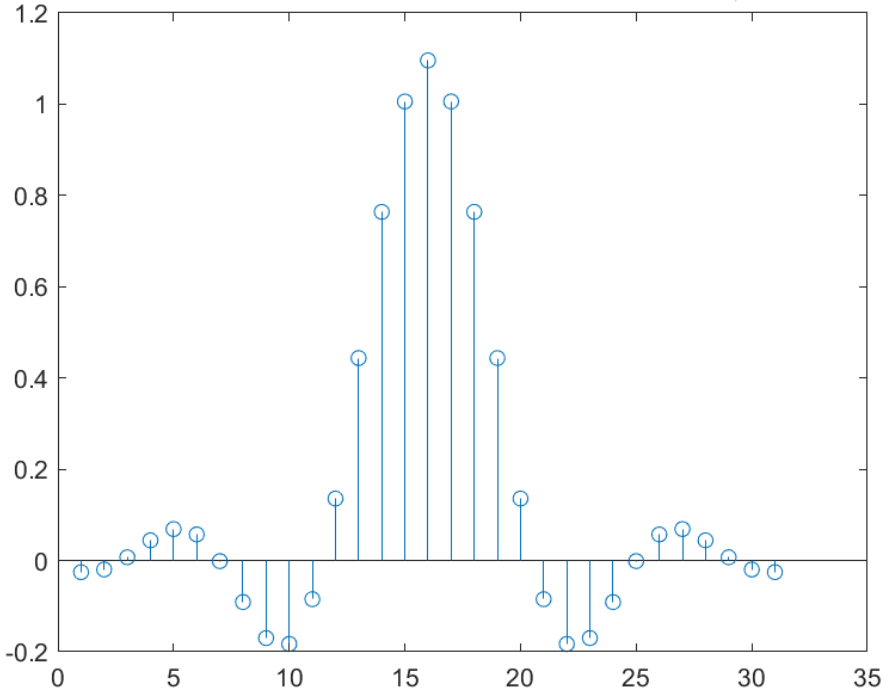
3. Pulso modulador con energía normalizada




```
%% Shapping Pulse
beta = 0.35;
D = 6;
Fs = 96e3;
Rb = (2*B) / (1+beta);
Ts = 1/Fs;
mp = ceil(Fs/Rb); %mp = 5;
Tp = mp*Ts;
energy = Tp;

[p,t] = rcpulse(beta, D, Tp, Ts, 'srrc', energy); %Genetare
p_energy = Ts*sum(p.*p);    %Calculate the power of the SRRC pulse
p = p/sqrt(p energy); %Unitari Normalization of SRRC pulse power
```

Para la generación del código de línea optamos por utilizar el polar NRZ.

```
%% Code Line
    %Polar Signal NRZ
PBP = p; % PSPB (Polar Base Pulse)
PS_s = bits_tx; %auxiliar vector that contain bits information
PS_s( PS_s == 0 ) = -1; %convert binary 0 to -1 v
s = zeros(1,numel(PS_s)*mp);    %Vector with the total size
s(1:mp:end) = PS_s;     %Impulse train
PSLC = conv(PBP, s) / mp; %PSLC (Polar Signal Line Code), Pulse train

    %Normilize pulse
PSLC = PSLC/sqrt( sum(PSLC.*PSLC)/numel(PSLC) ); %Unitari Normalization of Line code power
P_PSLC_d = var(PSLC); %Power of Line code digital
```



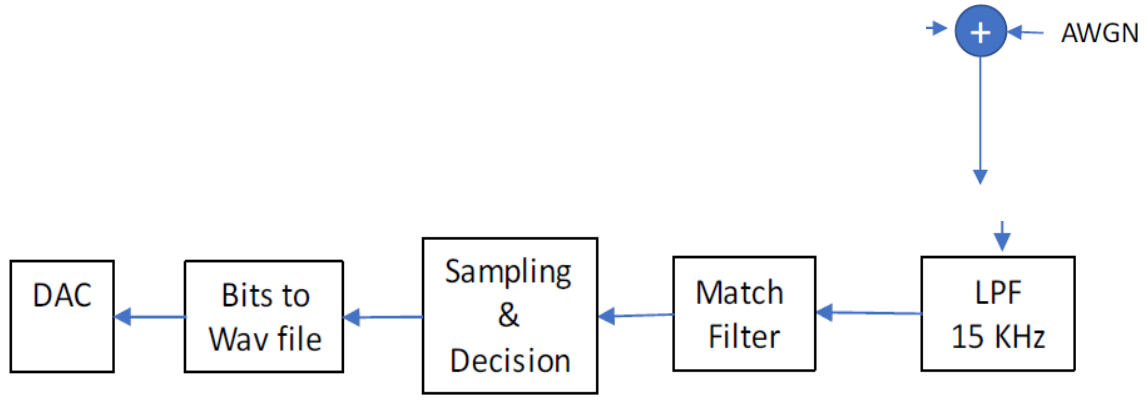
4. Creación de la densidad espectral de ruido y la respuesta del filtro acoplado receptor

```
%% SNR Digital
N0 = 1./(B.*10.^(0:0.3:3)); % PSD vector of noise
P_noise_d = B*N0; % Power of filtered noise
P_noise_dB_d = 10.*log10(P_noise_d); % Power of noise in dB
SNR_D = P_PSLC_d ./ P_noise_d; % Signal to Noise ratio
SNR_D_dB = 10*log10(SNR_D); % SNR in dB
```

```
%% Match Filter
%Match filter
%For the match filter we used the same form of the base pulse (SRRC)
[h_srrc t] = rcpulse(beta,D,Tp,Ts,'srrc',energy); %Response of the match Filter
hsrrc_energy = Ts*sum(h_srrc.*h_srrc); %Calculate the energy of the filter
h_srrc = h_srrc/sqrt(hsrrc_energy); %Normalized power of the filter
```

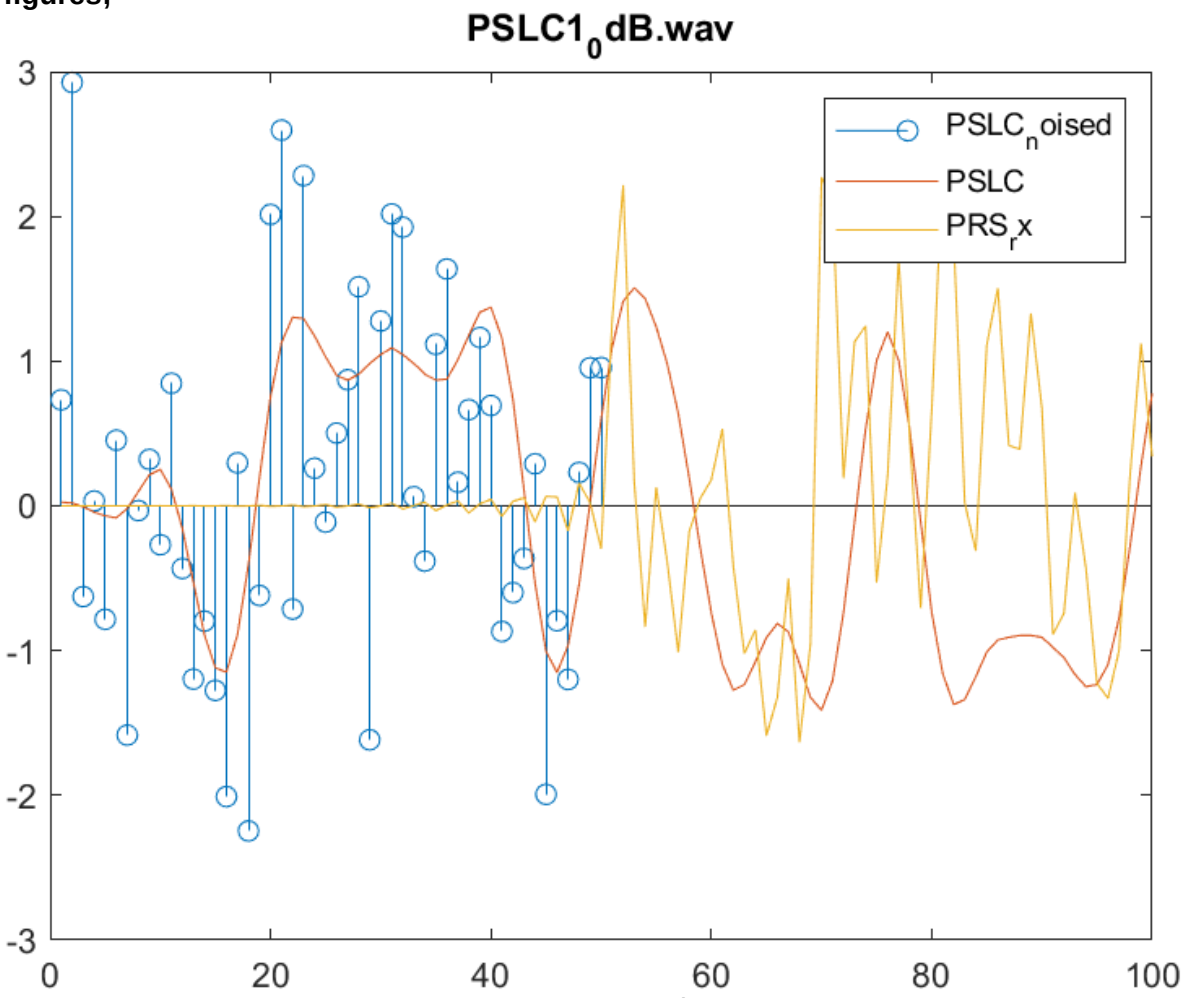
5. Añadir Ruido, filtrado del canal, filtrado del receptor, recuperación y escritura de información y guardado de archivo.

Todos estos procesos se hicieron dentro de un ciclo for, el cual cada que entra generará un vector de ruido con la potencia en dicha iteración, por lo tanto, dentro de ese ciclo for es necesario hacer el filtrado que emule el canal, el filtrado de recepción y la recuperación y escritura de la información obtenida, pues si se hacen fuera, solo estaríamos recuperando el de una sola densidad de ruido, la más baja.

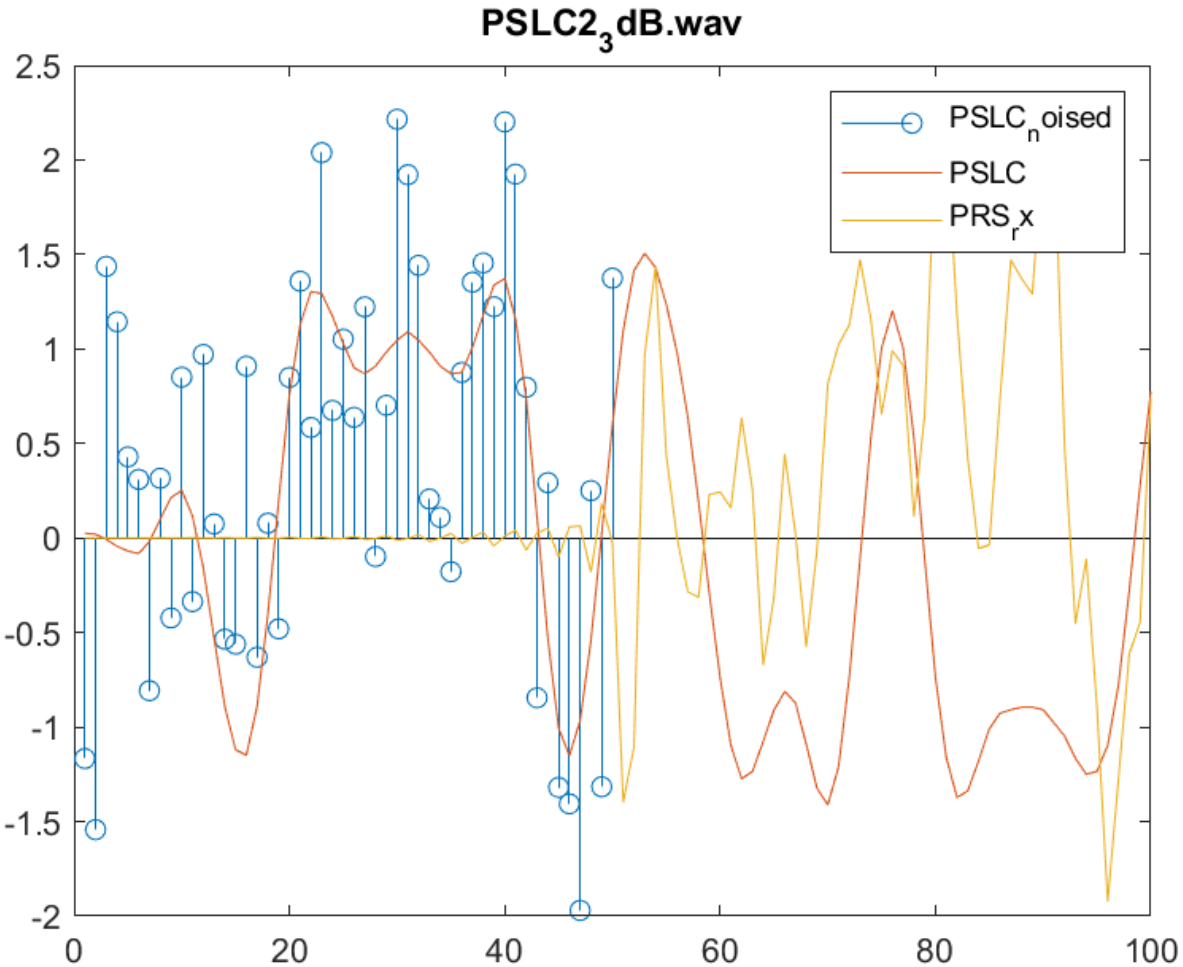


```
%% Add noise to Line Code
% AWGN (White noise all frequency)
%Generate noise vector and add to polar NRZ line code with SRRC as pulse
%base

%We need to add up noise, filtered, and recover the information inside a for loop because each
%iteration represent one noise power, so is diferent eveery time.
for i = 1 : numel(N0)
      %noise samples
    noise = sqrt(P_noise(i)) * randn(1, numel(PSLC)); %created noise vector

    PSLC_noised = PSLC + noise; % Add up noise to the signal

       %Conv function between the filter and the signal with AWGN
    PRS_rx = conv(LPF,PSLC_noised); %PSRS (Polar Received Signal)
    match_filter_recived = conv(h_srrc,PRS_rx); %filtered the recieved line code in the match filter

    %Wake the sampling and desition of the line code recived.
    %We have order/2 for the filter delay.
    %We have mp/2 for the mp size, so we can get the signal in the middle
    %of the signal.
    %Take a sample each mp/2 in time domain to recover the data.
    rx_match_filter = match_filter_recived( ( mp/2 + ((order/2)+(numel(p)/2)) : mp : (end - ((order/2)+(numel(p)/2)))));
%at this point, the information received and passed through the filter that emulates the channel and the match filter (reception)
%is obtained
    audio_rx = sign(rx_match_filter); %Obtenemos valores entre 1 y -1 del arreglo

    bits_rx = (audio_rx+1)/2; %Normalizamos para que los valores sean 1 o 0
        %Por error de acoplamiento de la señal original, eliminamos las
        %primeras 4 muestras y las 3 ultimas.
        %Antes de esto teniamos error de 50% (por estadistica, pues esta
        %bien y mal), despues de arreglar esto, el error fue del 8%
    bits_rx = bits_rx(4:end - 3);
    bits_rx = bits_rx';
        %Checksum de la señal original y la recuperada, obtenemos error del
        %8%
    error = (sum( xor(bits_tx,bits_rx) ) / numel(bits_rx))*100;

    mat = vec2mat(bits_rx,16);      %Convert vector to mat
    audio_bin2dec = bi2de(mat,'left-msb'); %Convert mat to dec
    audio_mat = vec2mat(audio_bin2dec, 1 , Fs_audio); %audio vector to mat
    audio_file = normalize(audio_mat, 'range', [-1 1]); %Normilize audio between 1 & -1
    filename = strcat('DigitalSignalRx_', num2str(i), '_', num2str(round(SNR_A_dB(i)), 4), 'dB','.wav');
    audiowrite(filename, audio_file, Fs_audio); %obtencion del archivo

end
```

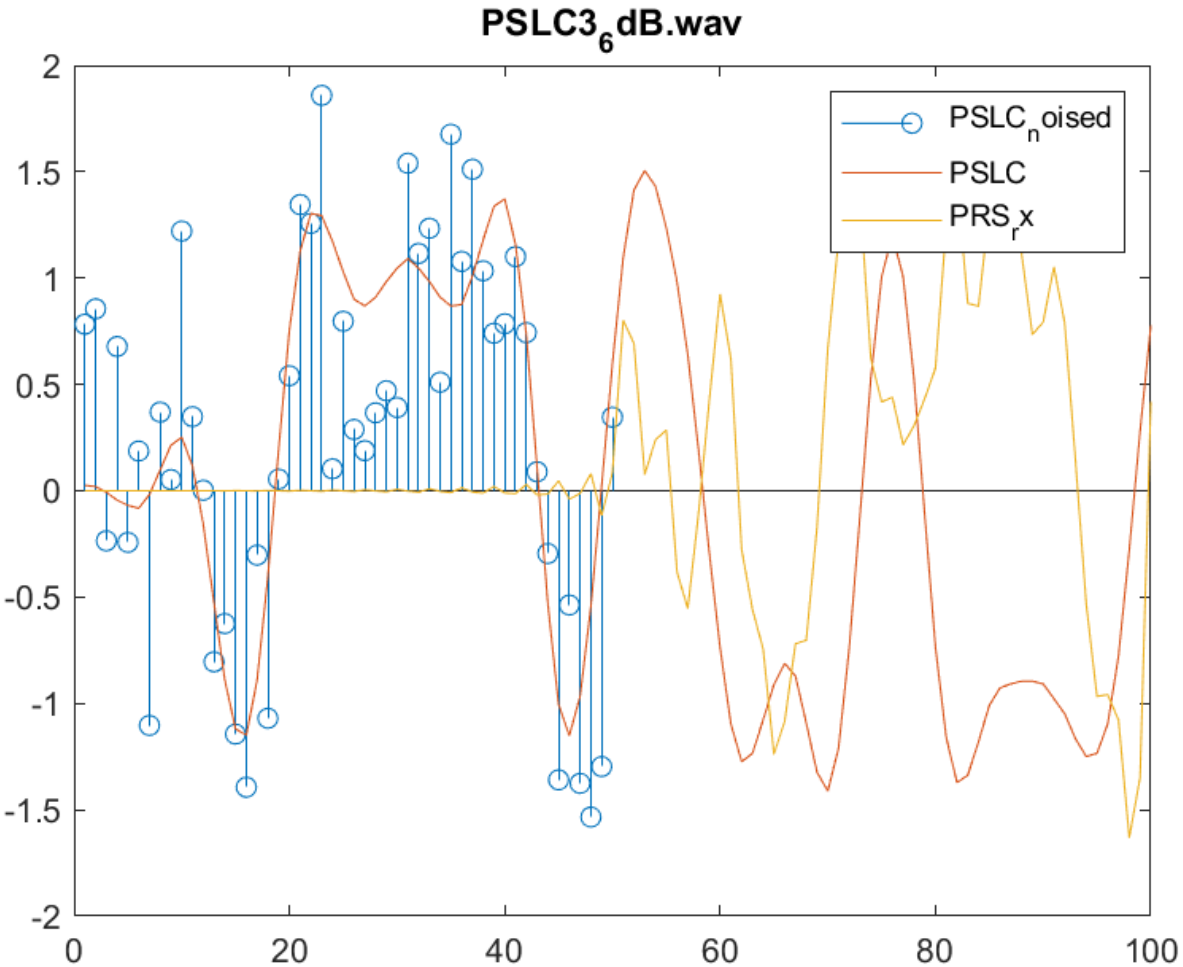
figures;



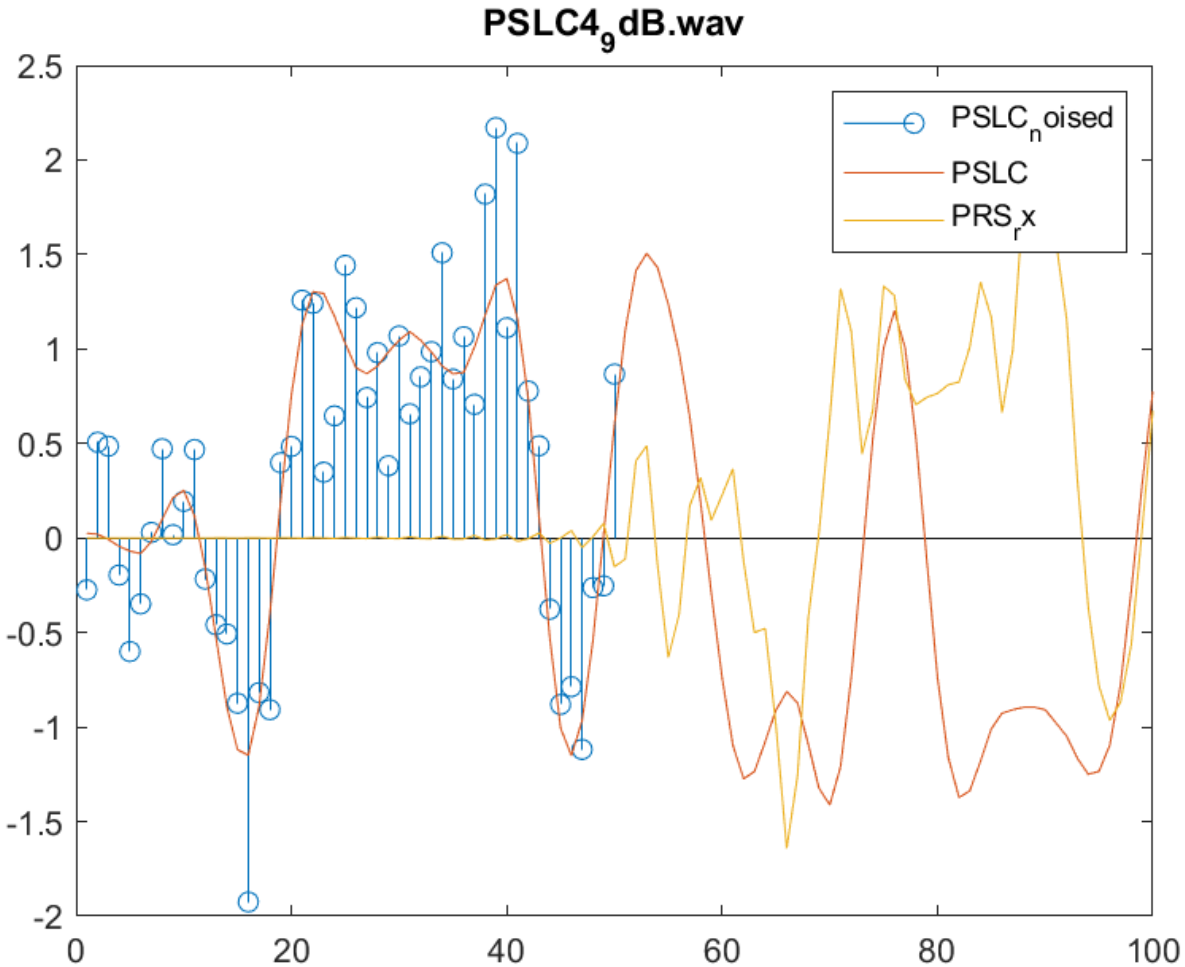
Img1. PSLC_0dB



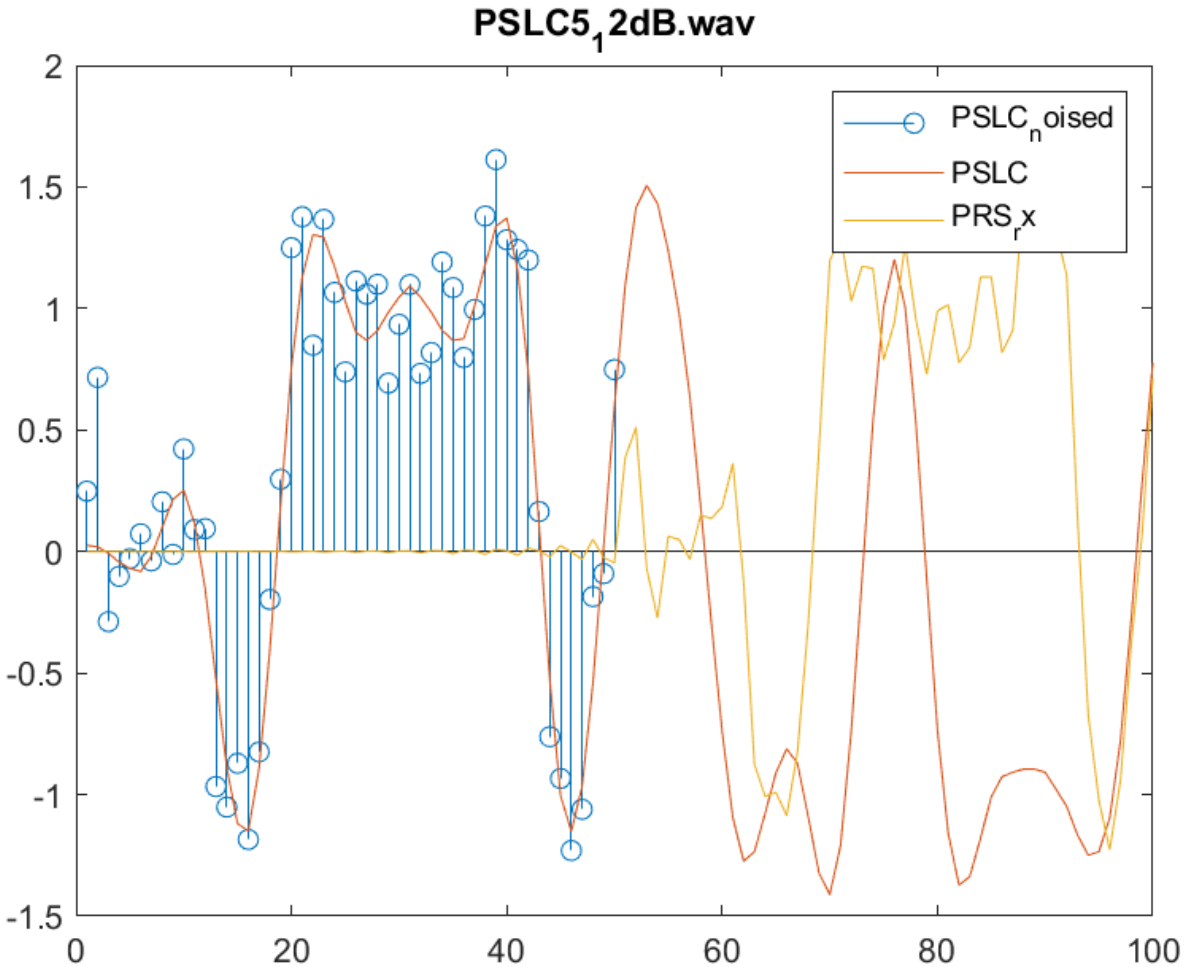
Img2. PSLC_3dB



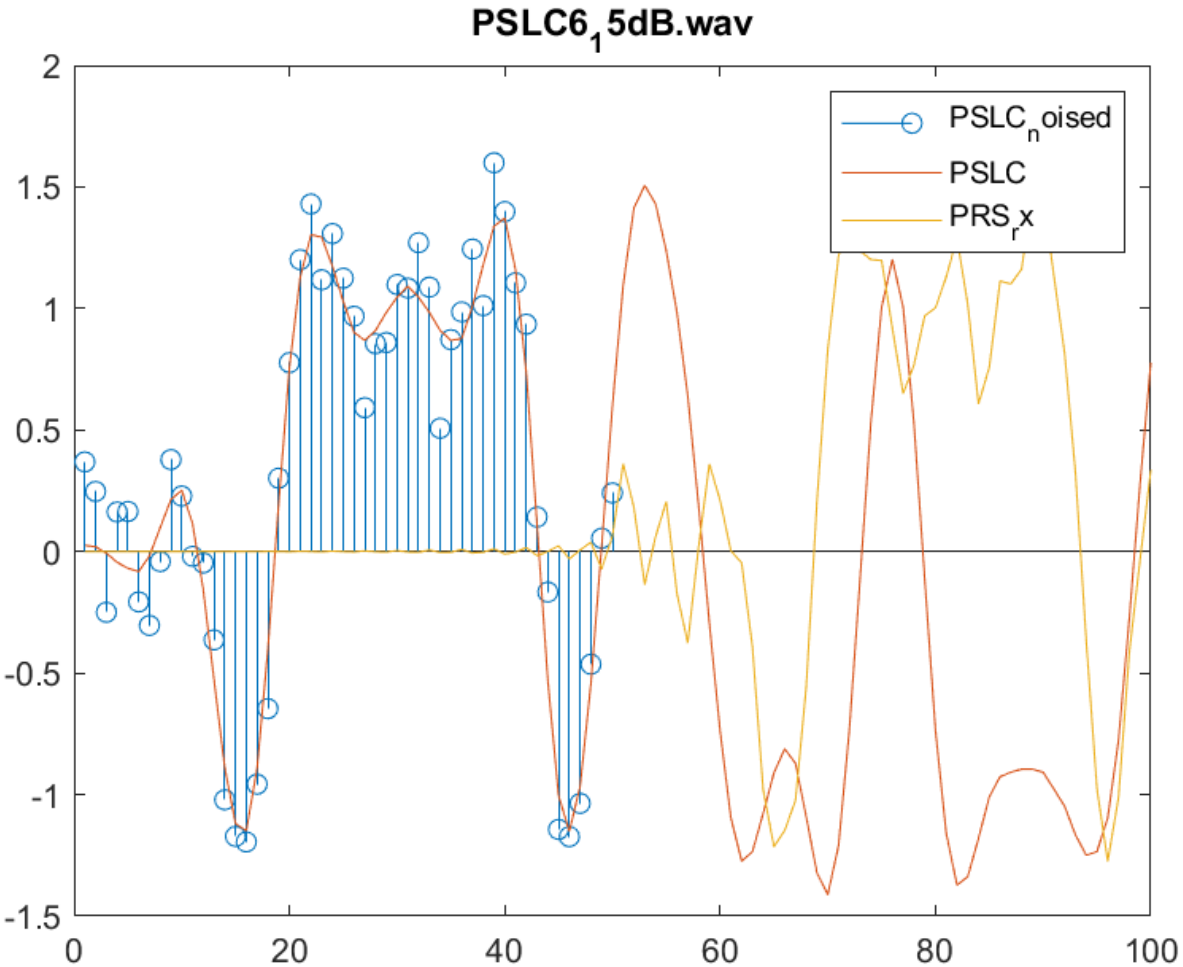
Img3. PSLC_6dB



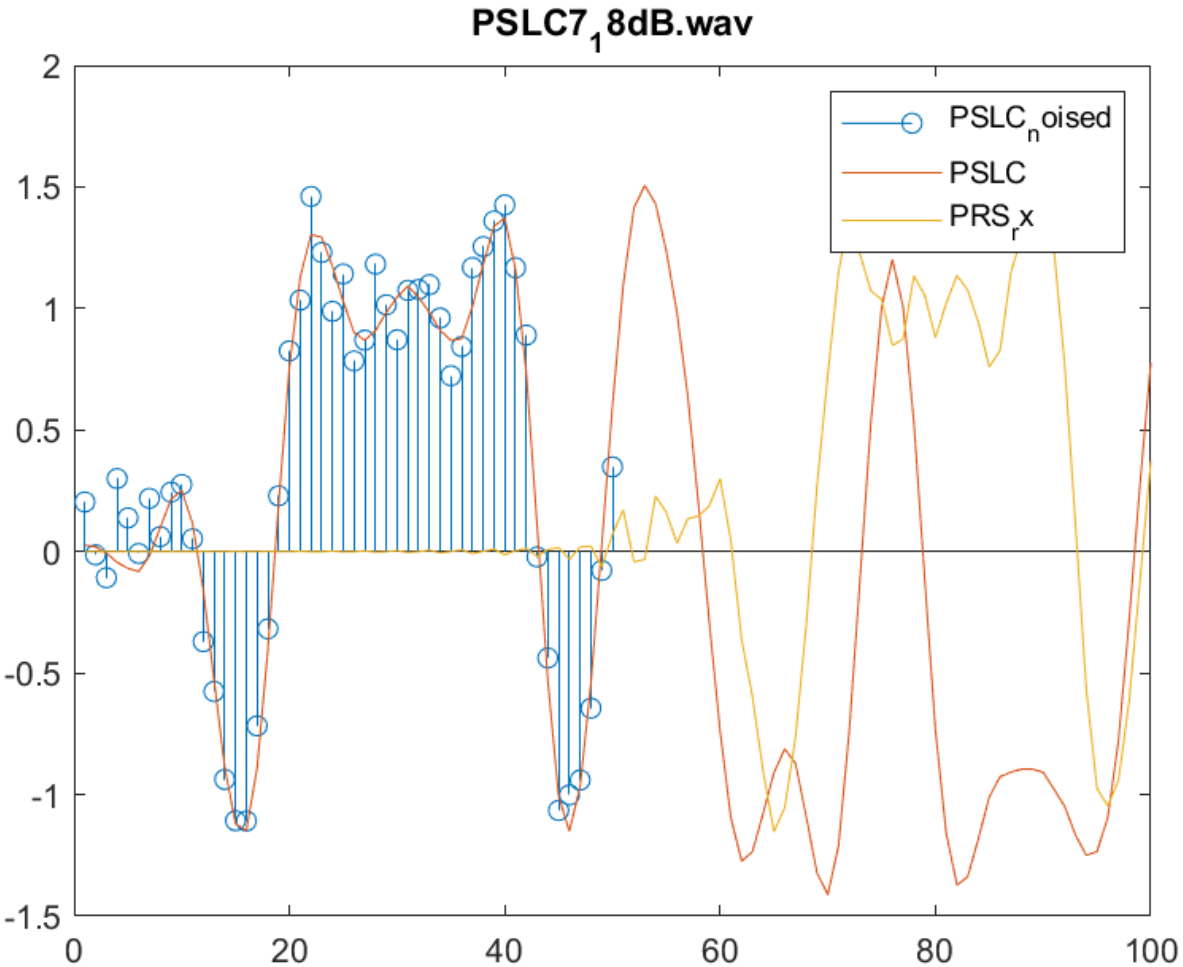
Img4. PSLC_9dB



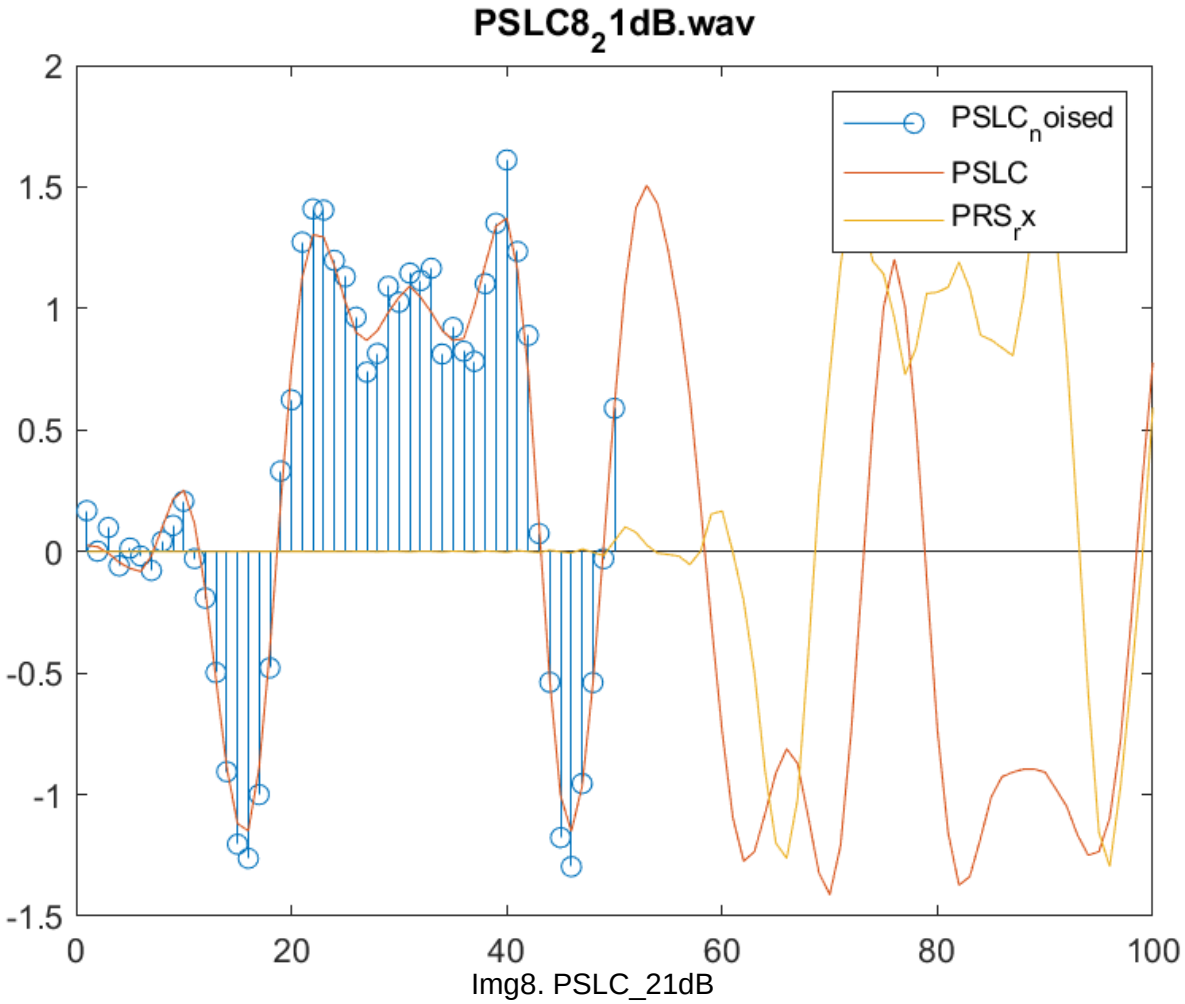
Img5. PSLC_12dB



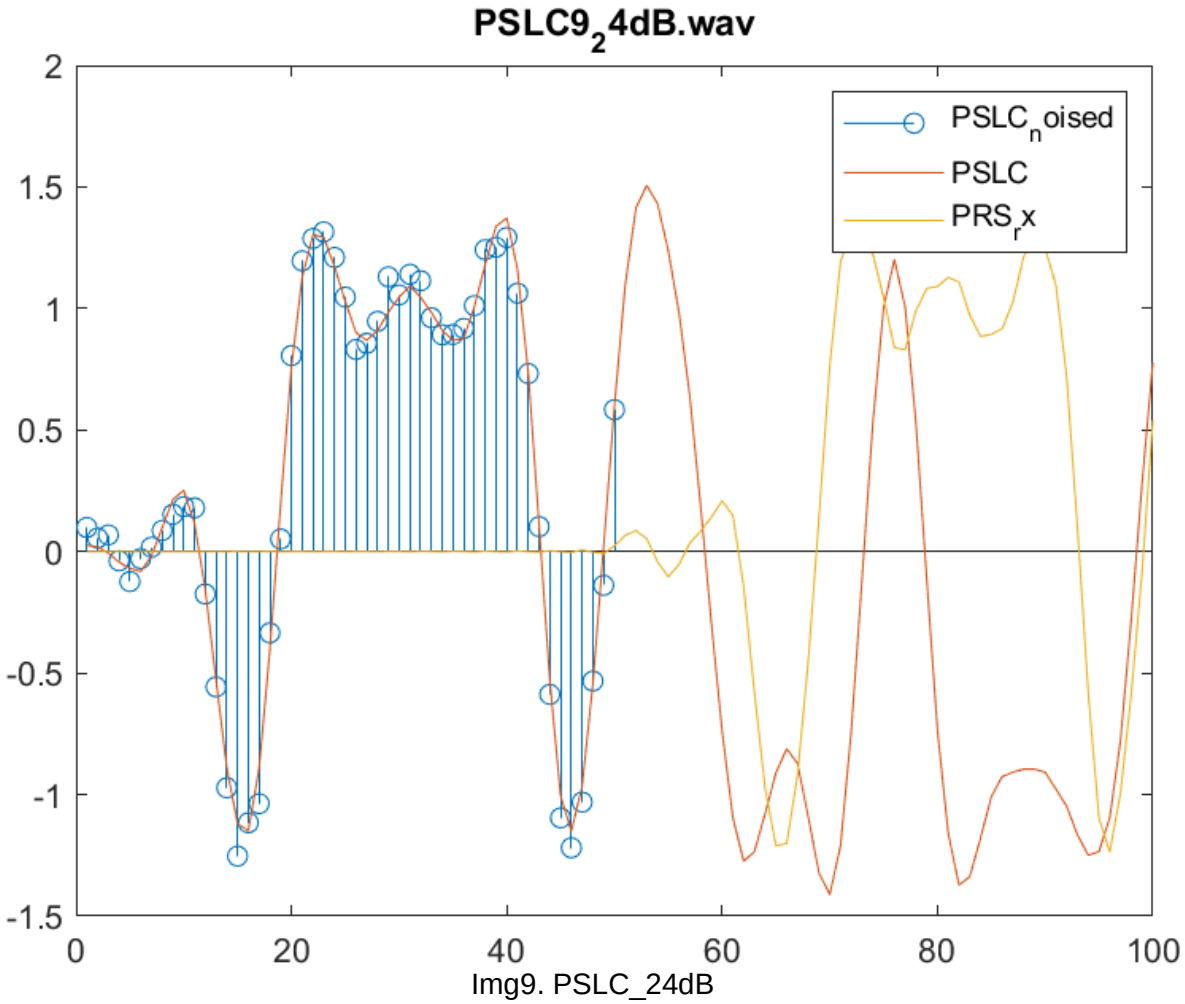
Img6. PSLC_15dB



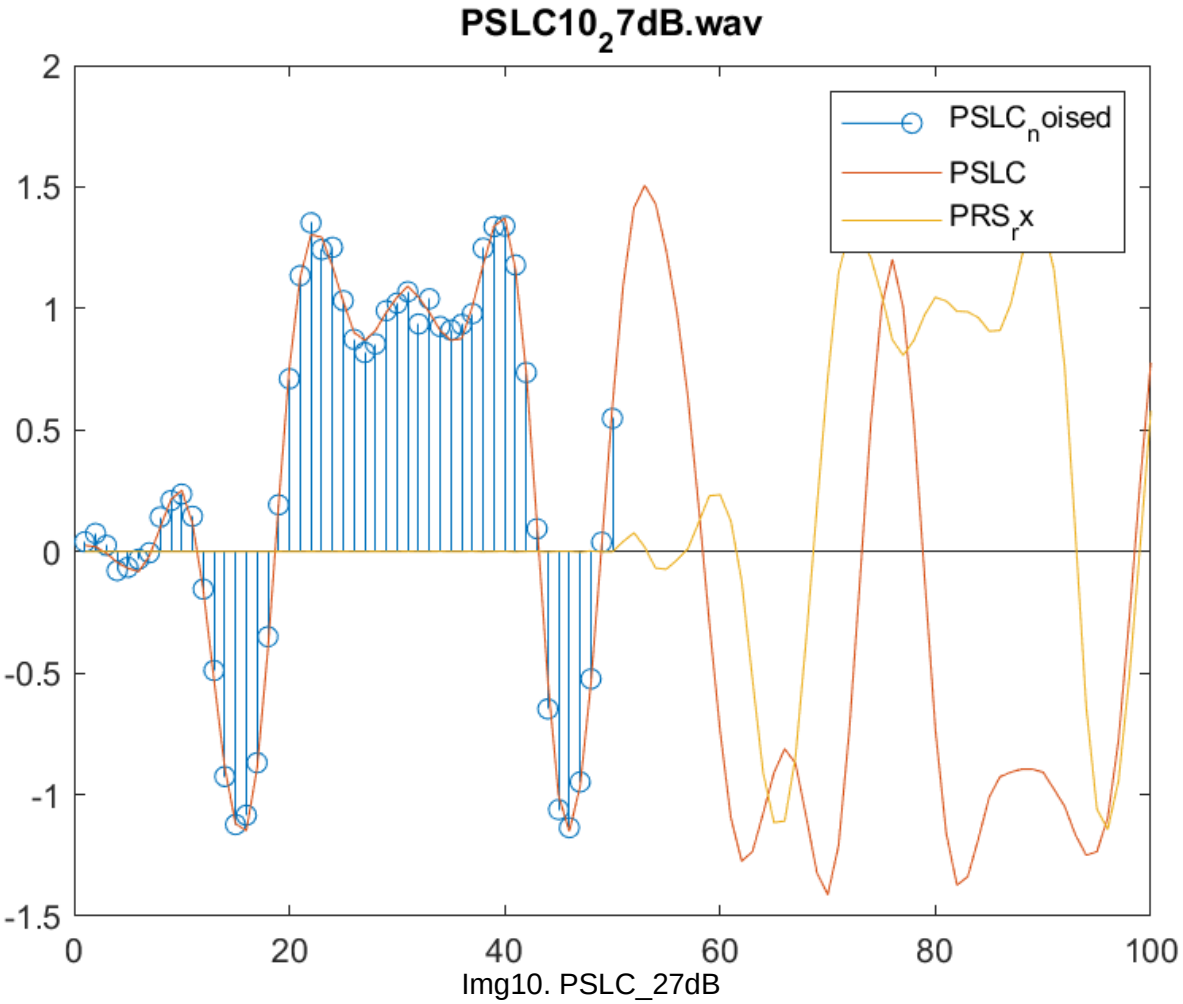
Img7. PSLC_18dB



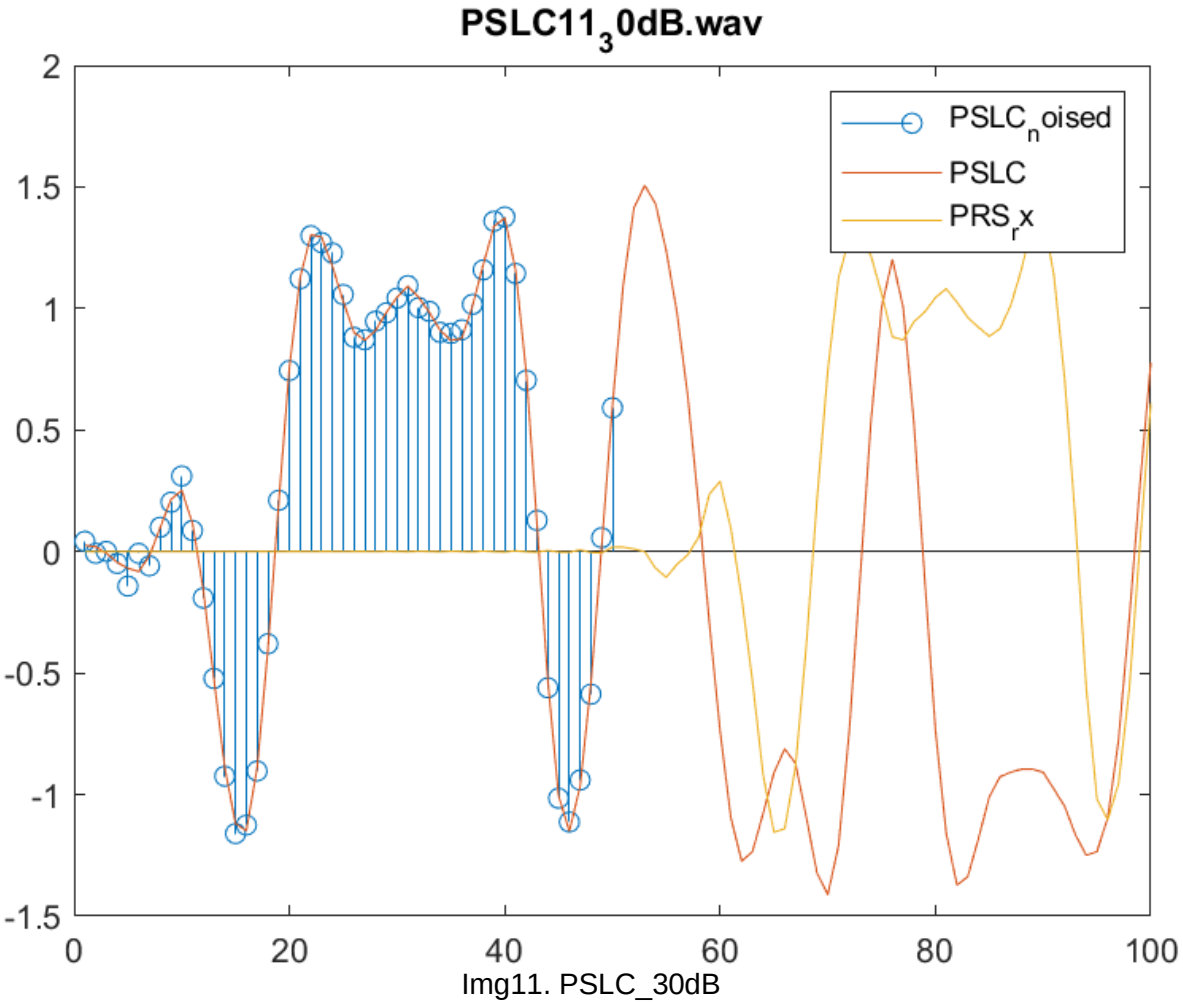
Img8. PSLC_21dB



Img9. PSLC_24dB



Img10. PSLC_27dB



Img11. PSLC_30dB

DIAGRAMA DE OJO DE LA SEÑAL DESPUÉS DEL FILTRO

```
EP = comm.EyeDiagram('SampleRate',Fs*mp,'SamplesPerSymbol',mp);

PRS_rx = PRS_rx';
%Eye Pattern Received Signal
EP(PRS_rx);
```

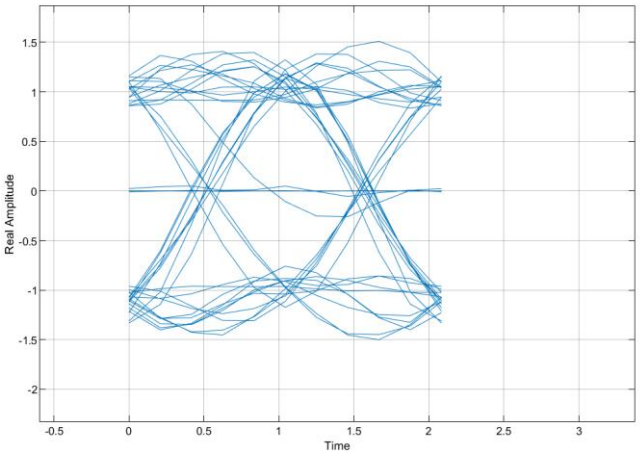
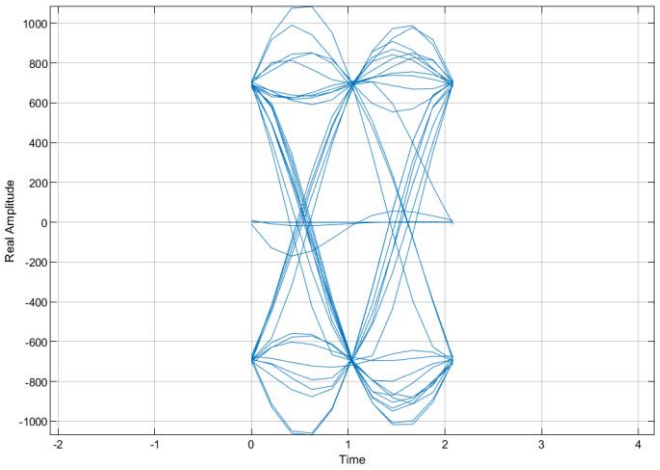


DIAGRAMA DE OJO DE LA SEÑAL DESPUÉS DEL MATCH FILTER RECEIVED

```
EP = comm.EyeDiagram('SampleRate',Fs*mp,'SamplesPerSymbol',mp);

match_filter_recived = match_filter_recived';

%Eye Pattern Received Signal
EP(match_filter_recived);
```



Preguntas:

1. Haga una tabla con cuatro columnas, con los valores de N0, el SNR en la comunicación analógica, el SNR en la digital, y finalmente qué sistema funciona mejor para ese SNR: el digital o el analógico.

N0	SNR Analog	SNR Digital	Best Option
6.667e-05	9.37270021368027e-06	0.999999728883154	Digital
3.3412e-05	3.00000937270021	1.99526177401965	Digital
1.6746e-05	3.98108029727018	3.98107062619937	Digital
8.3928e-06	7.94329949000829	7.94328019368516	Digital
4.2064e-06	15.8489661289251	15.8489276276987	Digital
2.1082e-06	31.6228448482624	31.6227680282163	Digital
1.0566-06	63.0958706178457	63.0957173417028	Digital
5.2955e-07	125.892812873940	125.892507047828	Analog
2.6540e-07	251.189185252801	251.188575049485	Digital
1.3302e-07	501.188315262650	501.187097746970	Digital
6.6667e-08	1000.00215814631	999.999728883154	Digital

En lo que concierne a lo discutido en clase, fue que hasta los 6-9 dB el analogico es mejor, sin embargo, en el resto es en lo que funciona mejor el digital.
Sin embargo por algún error en la toma de decisiones para la recuperación de la señal, o así de los mismos filtros, tenemos el margen de error, sin embargo, si analizamos; realmente la transmisión análoga solamente es mejor con un SNR bajo, en el momento en que subimos, la transmisión digital nota mejor calidad. Dado que esta tiene una mejor cualidad de resistencia al ruido.

2. Si quisiéramos lograr una comunicación en tiempo real (es decir, que el audio pueda ser reproducido sin necesidad de poner buffers, como en telefonía), ¿qué ancho de banda se requiere? Si no cambiamos la potencia transmitida, ¿cuál sería el SNR? Encuentre una fórmula y además calcúlalo para cada valor de N0 utilizado en esta práctica.

$$Rb = N\_bits * Fs = 16*44100 = 705600$$
$$beta = 0.35 \text{ (usada en la práctica)}$$
$$B = ((1 + beta) * Rb) / 2 = 476.28 \text{ k Hz}$$

Como el ancho de banda cambió, la potencia del ruido también, debido a que la potencia del ruido está determinada por la siguiente ecuación

$$\text{Noise Power} = \text{Ancho de banda} * \text{Densidad espectral de ruido}$$

Por lo tanto el SNR cambiará pues sabemos que está determinado por la siguiente ecuación:

SNR = Signal Power /Noise Power

Sustituyendo el Noise Power en la ecuación anterior del SNR

SNR =Signal Power/ (Ancho de banda * Densidad espectral de ruido)

Siguiendo la ecuación encontrada del SNR , y con los valores de la densidad espectral de ruido utilizados en la práctica, calculamos los valores del SNR para esta nueva forma de transmisión:

N0	SNR
6.66666666666667e-05	0.0314940705745513
3.34124822418182e-05	0.0628389321623726
1.67459095433972e-05	0.125380153256468
8.39283607862778e-06	0.250166294837653
4.20638229653462e-06	0.499147380564963
2.10818510677892e-06	0.995929958056700
1.05659546164074e-06	1.98714151365907
5.29552156482854e-07	3.96486857671416
2.65404780368998e-07	7.91095285492206
1.33017487664592e-07	15.7844261069215
6.66666666666667e-08	31.4940705745513

3. En la pregunta anterior, ¿qué potencia debe transmitirse para lograr un SNR de 15dB? Encuentre una fórmula y además calcúlalo para cada valor de N0 utilizado en esta práctica.

Sabemos que el SNR está determinado por:

SNR =Signal Power/Noise Power

Para poder obtener el resultado en dB es necesario aplicar la siguiente ecuación:

SNRdB= 10*log10 (SNR)

Sustituyendo la ecuación del SNR, en la ecuación anterior obtenemos:

SNRdB = 10*log10 (Signal Power/ Noise Power)

De la ecuación anterior conocemos todos los datos excepto la potencia del Ruido, por lo que despejamos para esta variable, resultando la siguiente ecuación:

Noise Power = 1/ 10(SNRdb/10)

Utilizando la ecuación anterior, sustituimos para encontrar el valor de la potencia de ruido transmitido para obtener un SNR de 15dB.

Noise Power = 1/ 10(15/10)= 1/ 31.62=0.03162

N0	SNR dB	Noise power
6.66666666666667e-05	0	1
3.34124822418182e-05	3	0.501
1.67459095433972e-05	6	0.251
8.39283607862778e-06	9	0.125
4.20638229653462e-06	12	0.063
2.10818510677892e-06	15	0.031
1.05659546164074e-06	18	0.015
5.29552156482854e-07	21	0.00794
2.65404780368998e-07	24	0.00398
1.33017487664592e-07	27	0.00195
6.66666666666667e-08	30	0.001

4. Mencione tres ventajas y tres desventajas de las comunicaciones digitales sobre las analógicas.

1. Ventajas

a. Se puede implementar un sistema de detección de errores.
b. Es posible multiplexar señales en las comunicaciones digitales.
c. Puede variar la calidad del audio, dependiendo la cantidad de bits que se quieran transmitir.

2. Desventajas

. Requiere algun metodo de sincronizacion, algunos métodos de comunicaciones digitales facilitan esto, pero no todos.
a. Estas requieren un mayor ancho de banda.
b. Pérdida de información, debido al muestreo, pues este procedimiento no es reversible.

Análisis de Resultados

En lo que logramos analizar en cuanto a estos resultados. Fue las ventajas y desventajas tanto de la transmision de informacion de manera analogica y digital. De manera en que los resultados que obtuvimos fue que en cuanto a distancias cortas a medias. Lo que se analizo es que el mejor método para enviar información es de manera digital. Ya que empleando correctamente el método de transmisión, esta puede ser inmune al ruido y perturbaciones hasta cierto grado.

Conclusiones

Luis Fernando Rodriguez Gtz;

En cuanto al desarrollo de esta práctica, principalmente lo que pude ver y de misma manera efectuar de cierta manera un repaso. Ya que la mayoría de esta práctica fue una recopilación de múltiples tareas realizadas. De manera que la práctica fue en sí un repaso desde mi perspectiva. Aunque la parte de mayor problema que fue que enfrentamos (desde mi punto de vista), fue la recuperación de los bits (en la parte digital). Ya que teníamos un letargo ocasionado en la lectura de la misma. Ya que esto ocasiona que la señal recuperada no fuera correcta.

Raúl Ignacio Baltazar Morán

Como conclusión esta práctica, en lo personal me ayudó para comprender la importancia del uso del filtro receptor que en nuestro caso el filtro acoplado, permitiendo la menor pérdida de información y una mayor eficiencia. En cuanto a la práctica creo que fue de mucha ayuda, pues es la recopilación de todos los temas visto hasta el momento, durante la realización de la misma nos enfrentamos a 2 principales problemas, el primero fue la generación de archivos en la parte digital, pues sólo estábamos obteniendo un solo archivo, este problema se pudo solucionar poniendo los procesos de sumatoria de ruido, filtrado, recuperación y escritura de los archivos dentro de un ciclo for, el cual nos indica la potencia de ruido en dicha iteración, y el otro fue la decisión en donde se estaban tomando las muestras, pues no obtenemos las muestras correctamente debido a los retardos que hay que tomar en cuenta.