Proyecto Grupo C

Hecho por Alejandro Comesaña Almuiña y Alberto Manuel Pombar Arias.

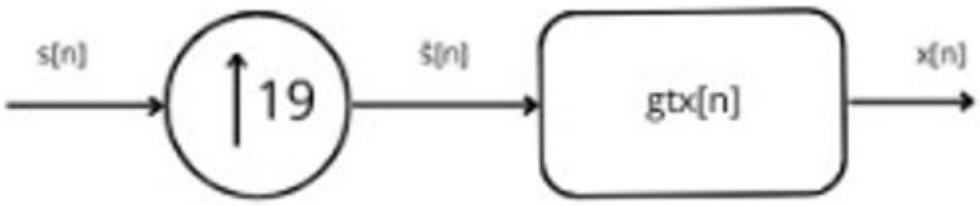
1.Procesado Polifase

1.1 Transmisión, caso específico:

Para la primera parte de nuestro desarrollo polifásico vamos a desarrollar tanto un transmisor monofase como uno polifase. El transmisor monofase será capaz de trabajar a un factor de sobremuestreo de 19/T.

1.1.1.Monofase

El esquema del circuito es el siguiente:



1-Upsampling: introducimo M-1 ceros (es decir, 18) por cada muestra original:

$\hat{s}$[n]=s[m] $\cdot \delta[n-19m]$, entonces: $\hat{s}[n]=\begin{cases} s\left[\frac{n}{19}\right] & \text{si } n \text{ es múltiplo de } 19 \\ 0 & \text{otros casos} \end{cases}$

2-Filtro transmisor: en este paso realizamos la convolución del filtro transmisor con lo obtenido en el paso anterior:

x[n] = $\hat{s}$[n] * g_{tx}[n] = $\sum_{k=-\infty}^{\infty} \hat{s}[k] \cdot g_{tx}$[n-k] , entonces , por el anterior paso sabemos que $\hat{s}[k]$ es distinto de 0 si n es múltiplo de 19, por lo tanto

x[n] = $\sum_{k=19}^{\infty} \hat{s}[k] \cdot g_{tx}$[n-k]; k lo podemos ver como 19*m siendo m un número entero:

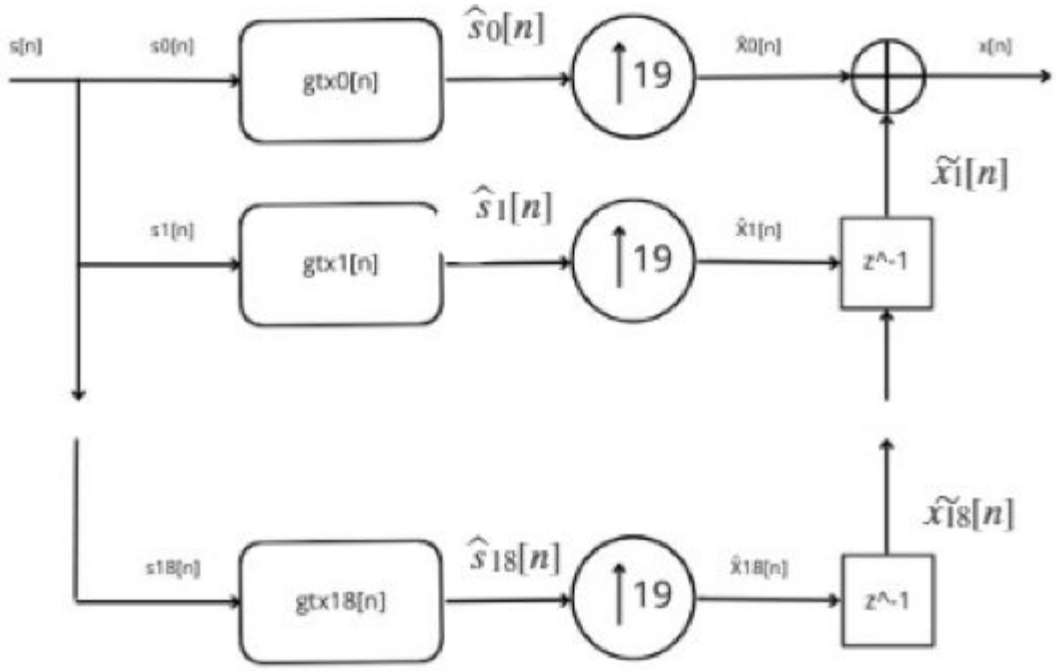
x[n] = $\sum_{m=-\infty}^{\infty} \hat{s}[19m]\ g_{tx}$[n-19m], entonces la salida del transmisor monofase es:

x[n] = $\sum_{m=-\infty}^{\infty} s[m]\ g_{tx}$[n-19m]

1.1.2.Polifase

Ahora vamos a realizar el procesado polifásico, para ello vamos a usar tres procesadores en paralelo con factores de sobremuestreo de 5/T, 7/T y 7/T respectivamente, esto es equivalente a tener 19 ramas de un procesador trabajando cada una a un factor de sobremuestreo de 1/T.

El esquema del procesador es:



1-Tenemos un flujo de datos 's[n]' que ,por ahora, es el mismo en todas las ramas.

2-Convolucionamos el flujo de datos con el filtro transmisor de la rama l-ésima:

$\hat{s}_l$[n] = s[n] * $g_{tx}^{(l)}$[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{tx}^{(l)}$[n-m]

3-Upsampling: introducimos M-1(18) ceros por cada muestra original:

$$\hat{x}_l[n] = \begin{cases} \sum_{m=-\infty}^{\infty} s[m]\, g_{tx}^{(l)}\left[\frac{n}{19} - m\right] & \text{si } n \text{ es múltiplo de } 19 \\ 0 & \text{en otros casos} \end{cases}$$

4-Retardamos cada rama:

$$\widetilde{x_l}[n] = \begin{cases} \sum_{m=-\infty}^{\infty} s[m]\, g_{\text{tx}}^{(l)}\left[\frac{n-l}{19} - m\right] & \text{si } n-l \text{ es múltiplo de } 19 \\ 0 & \text{en otros casos} \end{cases}$$

Siendo l el número de la rama . que está en el intervalor [0,18].

5-Juntamos todas las ramas, teniendo en cuenta que en la rama 'l', habrá valor si n-l es múltiplo de 19, es decir, si n= 19k+l siendo 'k' un número entero, entonces, juntamos todas las ramas y evaluamos para n =19k+l:

x[n]= $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{\text{tx}}^{(l)}[k-m]$, esta es la salida del polifase. De igual forma, evaulamos el monofase en el instante temporal n = 19k+l.

Salida monofase: x[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{\text{tx}}$[n-19m]. La evaluamos como ya hemos mencionado:

x[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{\text{tx}}$[19k+l-19m]

Igualamos ambos filtros transmisores, ya que el resto de la estructura es igual:

$g_{\text{tx}}^{(l)}[k-m]$ = g_{tx}[19k+l-19m] , por lo tanto:

$g_{\text{tx}}^{(l)}[k]$ = g_{tx}[19k+l]. Este es el resultado final que obtenemos al igualar ambos filtros transmisores

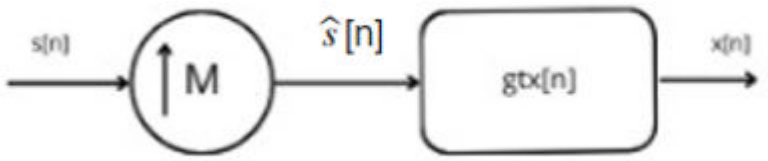
El correcto funcionamiento de este desarrollo se comprobará en matlab , en el archivo 'polifase.mlx', tanto para el monofase como para el polifase.

1.2 Transmision, caso general:

Para este punto definimos: L_{tx} que es el número de procesadores en paralelo, cada uno capaz de trabajar a tasa de $M_{\text{tx}}^{(l)}$ con $1 \leq l \leq L_{\text{tx}}$. A la entrada del modulador tenemos $\sum_1^{L_{\text{tx}}} \frac{M_{\text{tx}}^{(l)}}{T}$, entonces , definimos $M = \sum_1^{L_{\text{tx}}} M_{\text{tx}}^{(l)}$ como el factor de sobremuestreo total, por lo tanto, a la entradal del modulador tendremos $\frac{M}{T}$. Usando esta relación podemos escribir las ecuaciones de forma parecida a como lo haciamos en el caso específico. Estos parametros los podemos usar para ambas estructuras , tanto el monofase como el polifase:

1.2.1. Monofase

Como primer paso, vamos a volver a realizar el monofase de transmisión:



1-Upsampling: introducimo M-1 ceros por cada muestra original:

$\hat{s}$[n]=s[m] $\cdot \delta[n - \mathrm{Mm}]$,entonces: $\hat{s}$[n]= $\begin{cases} s\left[\frac{n}{M}\right] & \text{si } n \text{ es múltiplo de } M \\ 0 & \text{otros casos} \end{cases}$

2-Convolución del filtro transmisor con el upsampling:

x[n] = $\hat{s}$[n] * g_{tx}[n] = $\sum_{k=-\infty}^{\infty} \hat{s}[k] \cdot g_{\mathrm{tx}}$[n-k] , entonces , por el anterior paso sabemos que $\hat{s}[k]$ es distinto de 0 si n es múltiplo de M, por lo tanto

x[n] = $\sum_{k=\dot{M}}^{\infty} \hat{s}[k] \cdot g_{\mathrm{tx}}$[n-k]; k lo podemos ver como M*m siendo m un número entero:

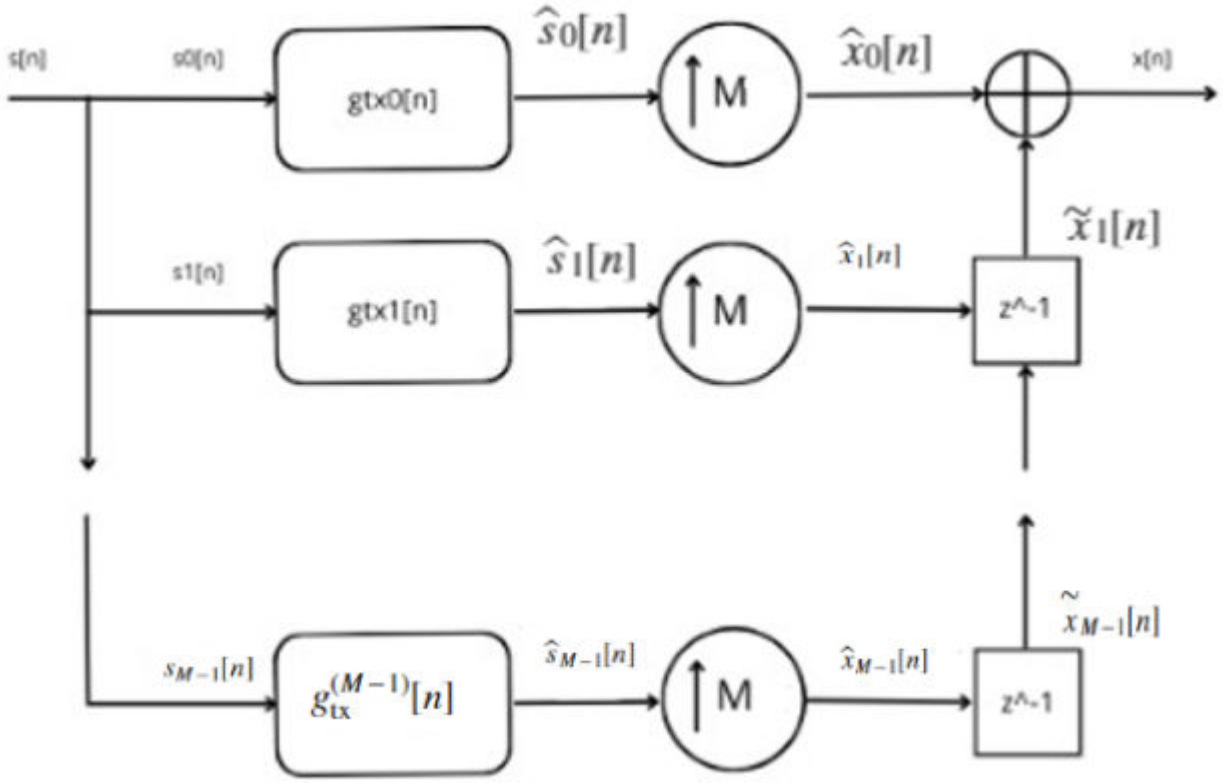
x[n] = $\sum_{m=-\dot{\infty}}^{\infty} \hat{s}[\mathrm{Mm}]$ g_{tx}[n-Mm], entonces la salida del transmisor monofase es:

x[n] = $\sum_{m=-\dot{\infty}}^{\infty} s[m]$ g_{tx}[n-Mm]

1.2.2. Polifase

Ahora en vez de una sola rama, tenemos L_{tx} procesadores trabajando en paralelo, juntanto todas estas ramas tendremos un procesador a tasa M (definido anteriormente) equivalente al monofase anterior:

El esquema del circuito es el siguiente:



1-Tenemos un flujo de datos 's[n]'.

2-Convolucionamos:

$\hat{s}_l$[n] = s[n] * $g_{tx}^{(l)}$[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{tx}^{(l)}$[n-m]

3-Upsampling, introducimos M-1 ceros por cada muestra original:

$$\hat{x}_l[n] = \begin{cases} \sum_{m=-\infty}^{\infty} s[m]\, g_{tx}^{(l)}\left[\frac{n}{M} - m\right] & \text{si } n \text{ es múltiplo de } M \\ 0 & \text{en otros casos} \end{cases}$$

4-Retardamos cada rama:

$$\tilde{x}_l[n] = \begin{cases} \sum_{m=-\infty}^{\infty} s[m]\, g_{tx}^{(l)}\left[\frac{n-l}{M} - m\right] & \text{si } n-l \text{ es múltiplo de } M \\ 0 & \text{en otros casos} \end{cases}$$

Siendo l el número de la rama y siendo este dentro del intervalo [0,M-1]

5-Juntamos todas las ramas, teniendo en cuenta que, para que en una rama haya un valor distinto de cero n-l debe ser múltiplo de M, es decir, n= Mk+l, siendo 'k' un número entero, por lo tanto, juntamos todas las ramas y evaluamos para n=Mk+l:

salida del polifase: x[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{tx}^{(l)}[k-m]$

Ahora, hacemos un proceso similar a el caso específico y evaluamos la salida del monofase para n= Mk+l y tenemos:

salida del monofase: x[n] = $\sum_{m=-\infty}^{\infty} s[m] \cdot g_{\text{tx}}$[Mk+l-Mm]

Volvemos a igualar ambos filtros transmisores:

$g_{\text{tx}}^{(l)}[k-m]$ = g_{tx}[Mk+l-Mm] , por lo tanto:

$g_{\text{tx}}^{(l)}[k]$ = g_{tx}[Mk+l]. Este es el resultado final que obtenemos al igualar ambos filtros transmisores.

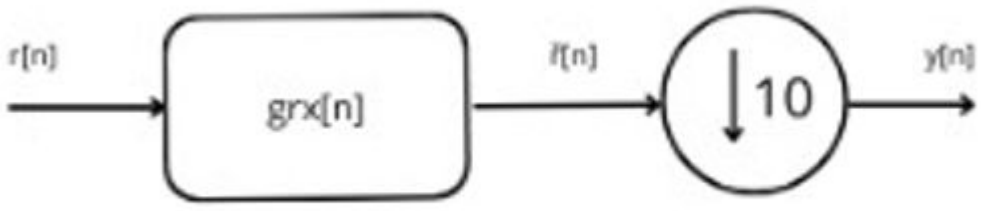
1.3. Recepción, caso específico:

Para el transmisor de recepción tenemos 4 procesadores en paralelo, los dos primeros capaces de trabajar a tasa 8/T y los otros dos a 12/T. A la salida del demodulador tenemos una tasa de 40/T y queremos una tasa de 4/T a la salida del filtro receptor, es decir, queremos que los cuatro procesadores trabajen en DIEZ ramas que hagan un downsampling de factor 10.

De igual forma que hicimos en el transmisor vamos a realizar primero un procesado de una sola fase e igualar su salida al procesado polifásico.

1.3.1. Monofase

El esquema del circuito es el siguiente:



1-El primer paso es la convolución del filtro transmisor con el flujo de datos 'r[n]' de entrada.

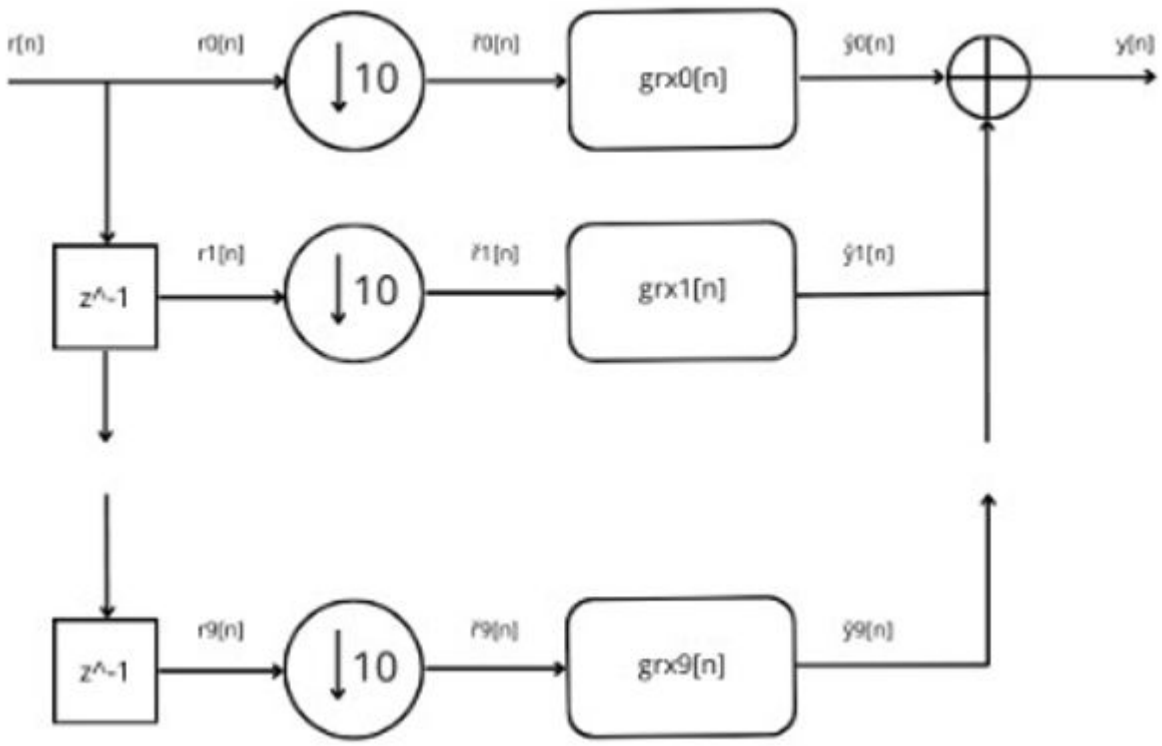
$\hat{r}$[n]= r[n] * g_{rx}[n] = $\sum_{k=-\infty}^{\infty} r[k] \cdot g_{\text{rx}}$[n-k] , este paso es la definición de convolución.

2-Hacemos el downsampling con factor L = 10, es decir, nos quedamos con una de cada L muestras.

y[n] = $\sum_{k=-\infty}^{\infty} r[k] \cdot g_{rx}$[10n-k], esta es la salida del monofase.

1.3.2. Polifase

Ahora vamos a realizar el desarrollo polifásico del receptor, tal y como dijimos más arriba. El esquema del circuito es el siguiente:



1-En la entrada tenemos un flujo de datos r[n].

2-Retardamos cada rama para que entren sincronizados los símbolos en la entrada del downsampling.

r_m[n] = r[n-m], siendo 'm' el número de la rama y siendo 'm' un entero que pertenece al rango [0,9].

3-Aplicamos el downsampling:

$\hat{r}_m$[n] = r[10n-m]

4-Convolucionamos el anterior paso con el filtro receptor:

$\hat{y}_m$[n]= $\hat{r}_m$[n] * $g_{rx}^{(m)}$[n] = $\sum_{p=-\infty}^{\infty} r[10p-m] \cdot g_{rx}^{(m)}$[n-p]

5-Juntamos todas las ramas:

y[n] = $\sum_{m=0}^{9}\sum_{p=-\infty}^{\infty} r[10p-m]\cdot g_{\mathrm{rx}}^{(m)}$[n-p]

En este punto podemos ver que para una señal $q[n,m]$ genérica, $\sum_{l=0}^{M-1}\sum_{k=-\infty}^{\infty} q[n,\mathrm{Mk}+l]$ = $\sum_{m=-\infty}^{\infty} q[n,m]$, ya que, el sumatorio de todas las ramas downsampleadas da lugar a la misma señal de inicio. En nuestro caso particular, evaluamos la salida del esquema monofase para $10p-m$:

salida monofase: y[n] = $\sum_{m=0}^{9}\sum_{p=-\infty}^{\infty} r[10p-m]\cdot g_{\mathrm{rx}}$[10n-(10p-m)]

la salida queda: y[n] = $\sum_{m=0}^{9}\sum_{p=-\infty}^{\infty} r[10p-m]\cdot g_{\mathrm{rx}}$[10(n-p)+m)]

Aquí tenemos r igual para monofase y polifase, solo nos queda comparar ambos filtros:

$g_{\mathrm{rx}}^{(m)}$[n-p] = g_{rx}[10(n-p)+m)] , entonces, los filtros nos quedan

$g_{\mathrm{rx}}^{(m)}$[n] = g_{rx}[10n+m)], esta es la salida igualada de ambos procesadores

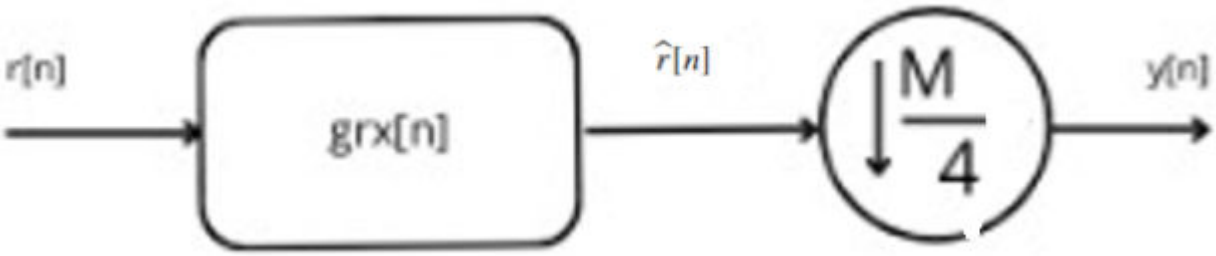
El correcto funcionamiento de este desarrollo se comprobará en matlab , en el archivo 'polifase.mlx', tanto para el monofase como para el polifase.

1.4. Recepción, caso general:

En este caso tenemos L_{rx} procesadores capaces de trabajar con un factor de sobremuestreo $M_{\mathrm{rx}}^{(j)}$ siendo este valor múltiplo entero de 4. A la salida del demodulador tenemos una tasa de $\frac{\sum_{j=1}^{L_{\mathrm{rx}}} M_{\mathrm{rx}}^{(j)}}{T}$, para simplificar los cálculos tomaremos M = $\sum_{j=1}^{L_{\mathrm{rx}}} M_{\mathrm{rx}}^{(j)}$ como el factor de sobremuestreo total, entonces, tendremos una tasa de $\frac{M}{T}$ a la entrada del demodulador y a la salida del receptor queremos una tasa de $\frac{4}{T}$.

1.4.1. Monofase

De igual forma que en los anteriores puntos vamos a realizar primero el procesado en una sola fase para luego hacerlo en varias fases e igualar la salida de ambas estructuras. El esquema del monofase es:



1-Realizamos la convolución del flujo de datos de entrada r[n] con el filtro transmisor:

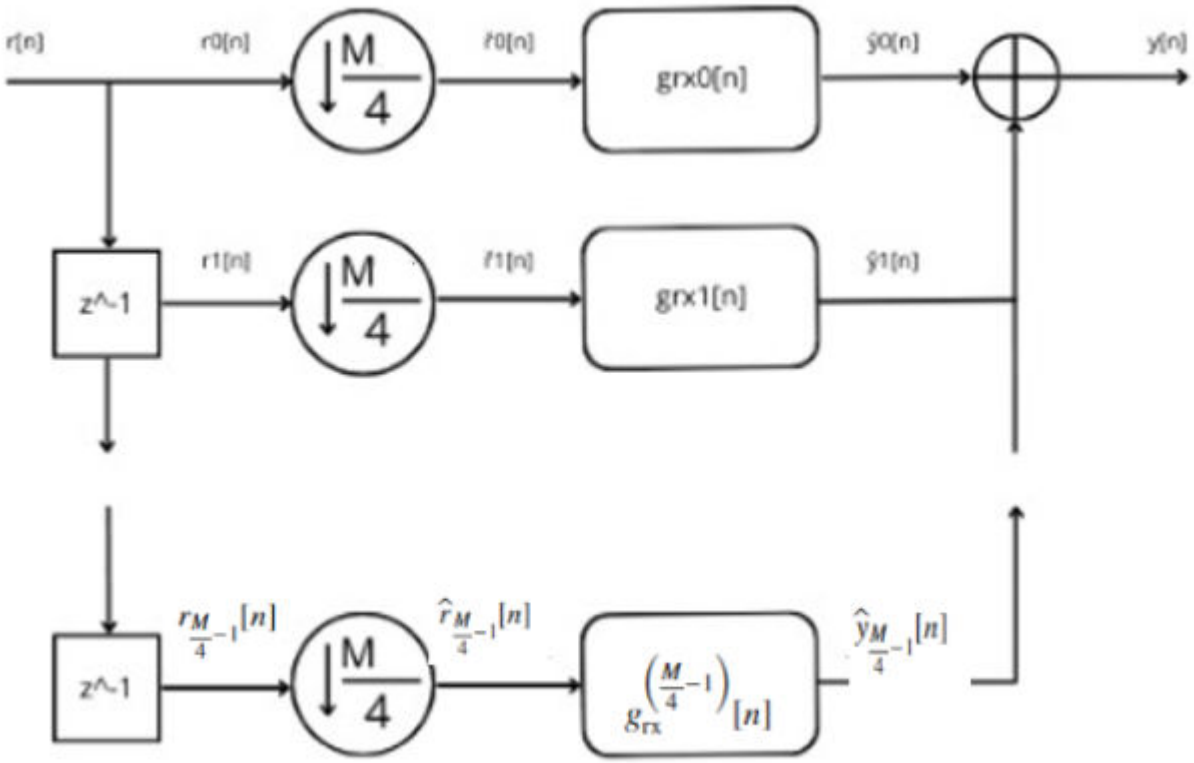
$\hat{r}[n]$= r[n] * $g_{rx}[n]$= $\sum_{k=-\infty}^{\infty} r[k] \cdot g_{rx}$[n-k], este paso es igual que en el caso específico.

2-Realizamos el downsampling con un factor M/4, es decir, nos quedamos con una de cada M/4 muestras, siendo M un entero multiplo de 4.

y[n] = $\sum_{k=-\infty}^{\infty} r[k] \cdot g_{rx}[\frac{Mn}{4}$-k]

Esta es la salida del monofase.

1.4.2.Polifase:



1-En la entrada tenemos un flujo de datos r[n].

2-Retardamos cada rama para que entren sincronizados los símbolos en la entrada del downsampling.

r_m[n] = r[n-m], siendo 'm' el número de la rama y siendo 'm' un entero que pertence al rango [0, $\frac{M}{4}$-1].

3-Aplicamos el downsampling:

$\hat{r}_m$[n] = r[$\frac{M}{4}$n-m]

4-Convolucionamos el anterior paso con el filtro receptor:

$\hat{y}_m$[n]= $\hat{r}_m$[n] * $g_{\mathrm{rx}}^{(m)}$[n] = $\sum_{p=-\infty}^{\infty} r\left[\frac{M}{4}p-m\right]\cdot g_{\mathrm{rx}}^{(m)}$[n-p]

5-Juntamos todas las ramas:

y[n] = $\sum_{m=0}^{M-1}\sum_{p=-\infty}^{\infty} r\left[\frac{M}{4}p-m\right]\cdot g_{\mathrm{rx}}^{(m)}$[n-p]

Aplicamos el mismo razonamiento que en el caso especifico, evaluando la salida del esquema monofase para $\frac{M}{4}p-m$:

salida monofase: y[n] = $\sum_{m=0}^{M-1}\sum_{p=-\infty}^{\infty} r\left[\frac{M}{4}p-m\right]\cdot g_{\mathrm{rx}}$[$\frac{M}{4}$n-($\frac{M}{4}$p-m)]

la salida queda: y [n] = $\sum_{m=0}^{M-1}\sum_{p=-\infty}^{\infty} r\left[\frac{M}{4}p-m\right]\cdot g_{\mathrm{rx}}$[$\frac{M}{4}$(n-p)+m)]

Aquí tenemos r igual para monofase y polifase, solo nos queda comparar ambos filtros:

$g_{\mathrm{rx}}^{(m)}$[n-p] = g_{rx}[$\frac{M}{4}$(n-p)+m)] , entonces, los filtros nos quedan

$g_{\mathrm{rx}}^{(m)}$[n] = g_{rx}[$\frac{M}{4}$n+m)], esta es la salida igualada de ambos procesadores

1.5. Implementación en Matlab

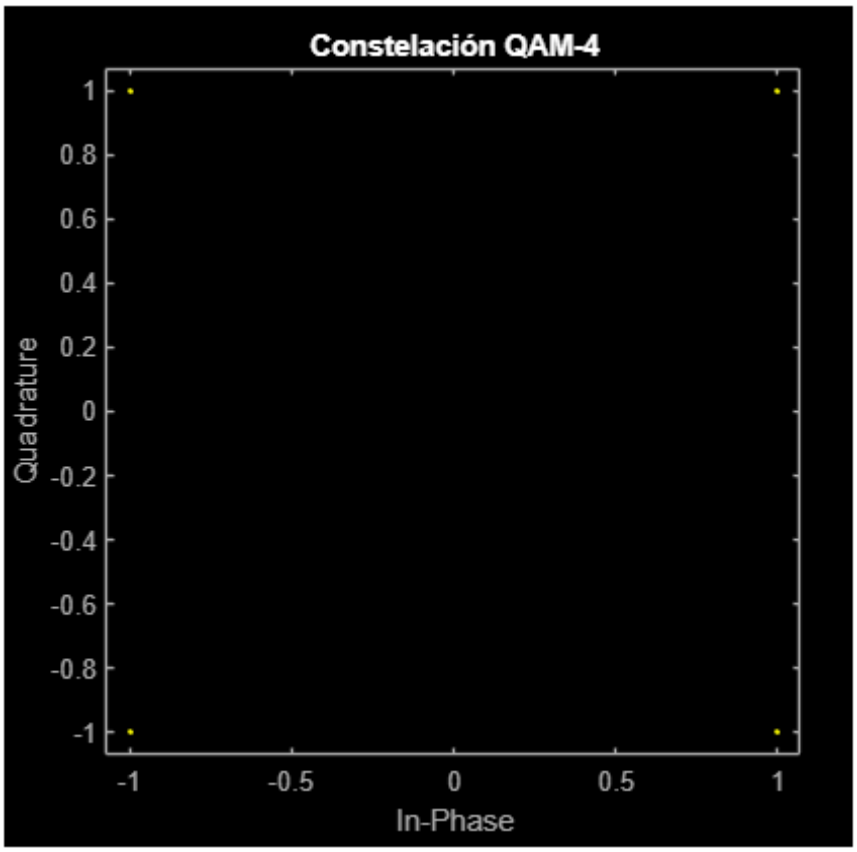
1.5.1.Procesador

Tenemos un canal ideal (sin multitrayecto, ruido ni atenuación). Definimos el filtro transmisor como un SRRC con un α variable de 0 a 1.

```
%El filtro transmisor es SRRC (filtro de raíz de coseno alzado). Parametros
%de este filtro
span = 10; %numero de periodos
alfa = 0.25; %El valor es variable y va desde 0 a 1 que son los valores posibles que tiene alfa
M = 19; %factor de sobremuestreo del procesador
```

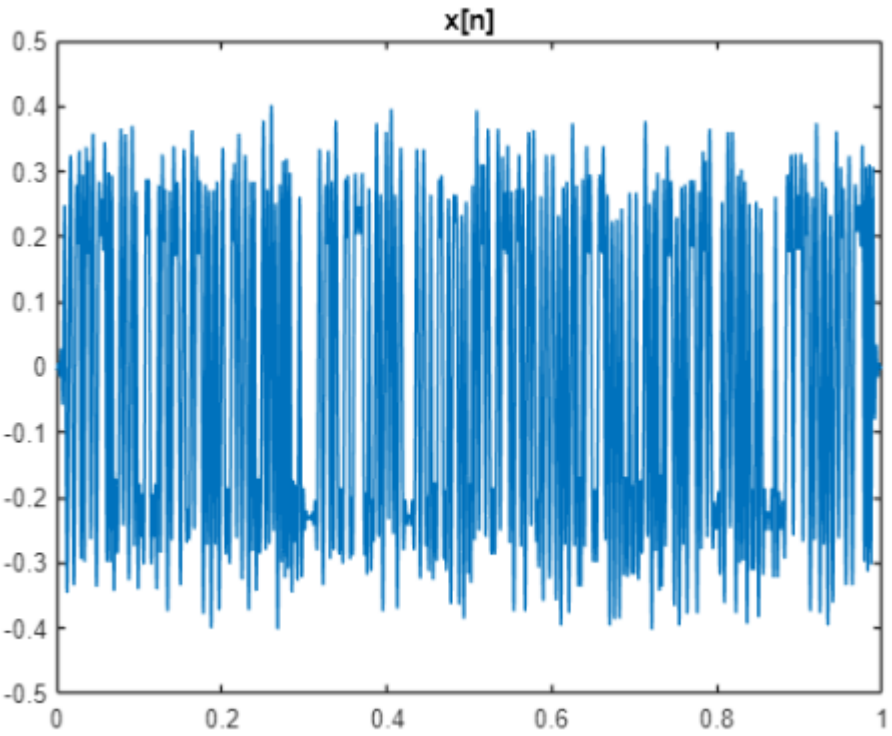
```
srrc = rcosdesign (alfa, span, M, 'sqrt'); %el span es la longitud del filtro
% en simbolos y M las muestras por simbolo
```

Una vez definido, generamos una constelación M-QAM parametrizable y la representamos. En el dibujo podemos observar una 4-QAM:



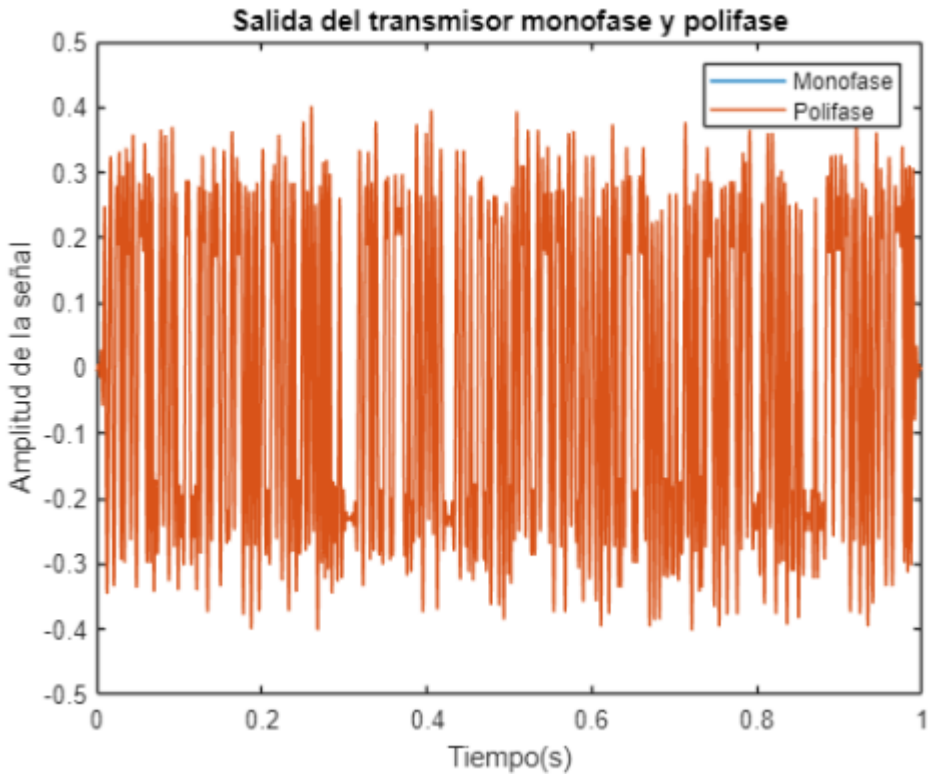
1.5.2. Transmisor monofase

Seguimos el mismo proceso que en el desarrollo teórico, es decir, realizamos el upsampling y después el filtrado dando lugar a la siguiente salida. Así obtenemos:



1.5.3. Transmisor polifase

Al igual que en el apartado anterior, realizamos los mismos pasos que en el desarrollo teórico, y representamos la salida en MATLAB:



Ahora nos disponemos a comparar las salidas de ambos filtros de transmisión. Para ello comparamos las normas de ambos.

```
%Para comparar si ambas salidas son iguales hallamos su norma
normatx = norm(senalfinal - filtrada)
```

```
normatx = 4.4618e-15
```

La norma, básicamente, calcula la diferencia entre cada elemento del vector uno por uno (haciendo el cuadrado de cada vector para tener en cuenta a la vez parte real e imaginaria), al ser tan pequeña la norma, es que prácticamente no hay diferencia entre ambos vectores y podemos concluir que son iguales y, por tanto, que las salidas del monofase y polifase son iguales. Por lo tanto, las salidas teóricas del monofase y polifase para los filtros transmisores son correctas.

1.5.4.Modulación

Primero, tenemos que tener en cuenta que tanto en tx como rx , la mínima frecuencia de portadora es de 1/T, ya que en caso contrario habrá solape entre las muestras. La frecuencia máxima sen tx será fc= M/2T - 1/T , por lo tanto:

```
%modulamos
mod= senalfinal.*exponencialmod;
%resampleamos la señal para adaptarnos a la entrada del demodulador
entrada = 40; %Tasa a la entrada del demodulador
res = resample(mod, entrada, M);
```

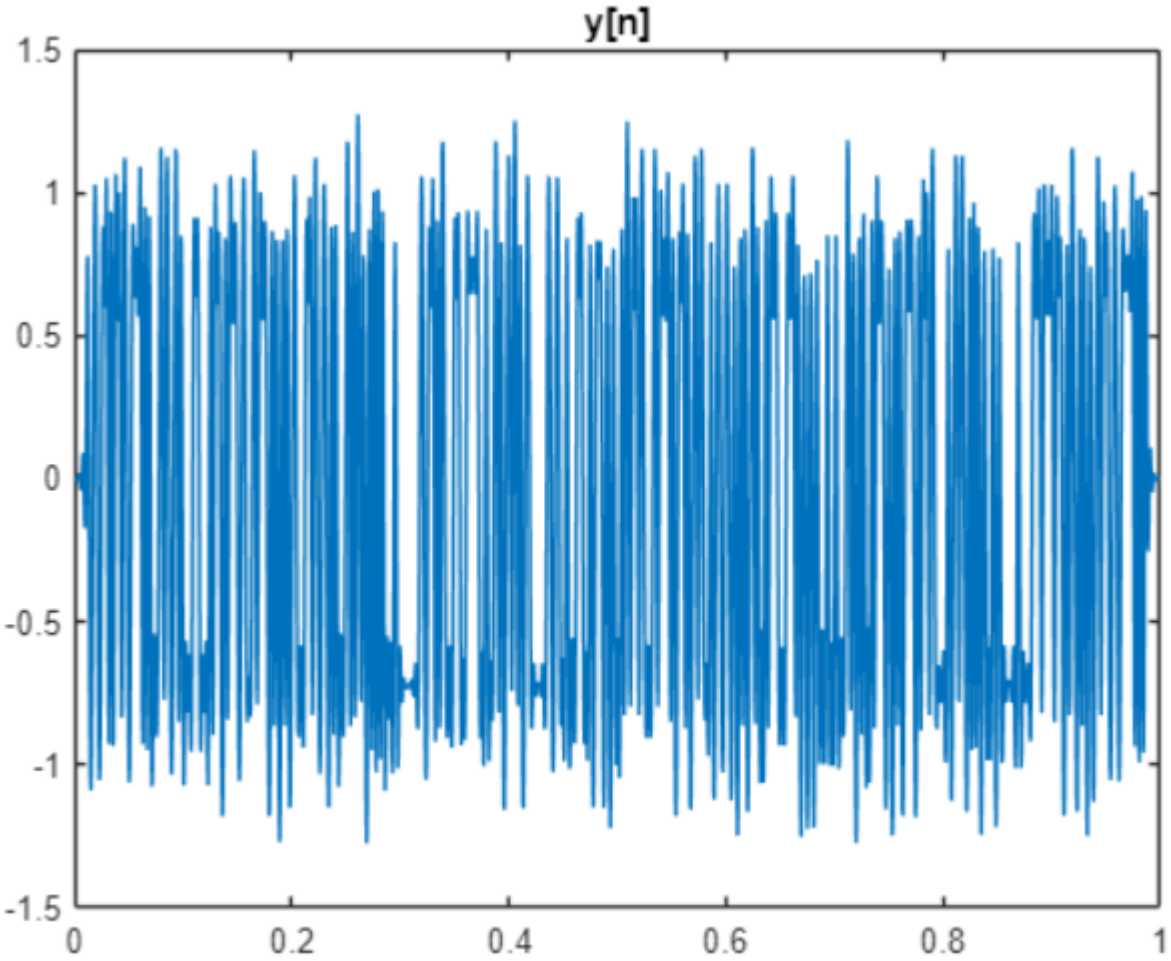
1.5.5.Demodulación

La mínima frecuencia de portadora vuelve a ser la misma que antes, la fc máxima será de : fc =L/4T- 1/2T, por lo tanto:

```
exponencial = exp(-1j * 2 * pi *fcrx *Tdemod);
%Demodulamos
demod = res.*exponencial;
```

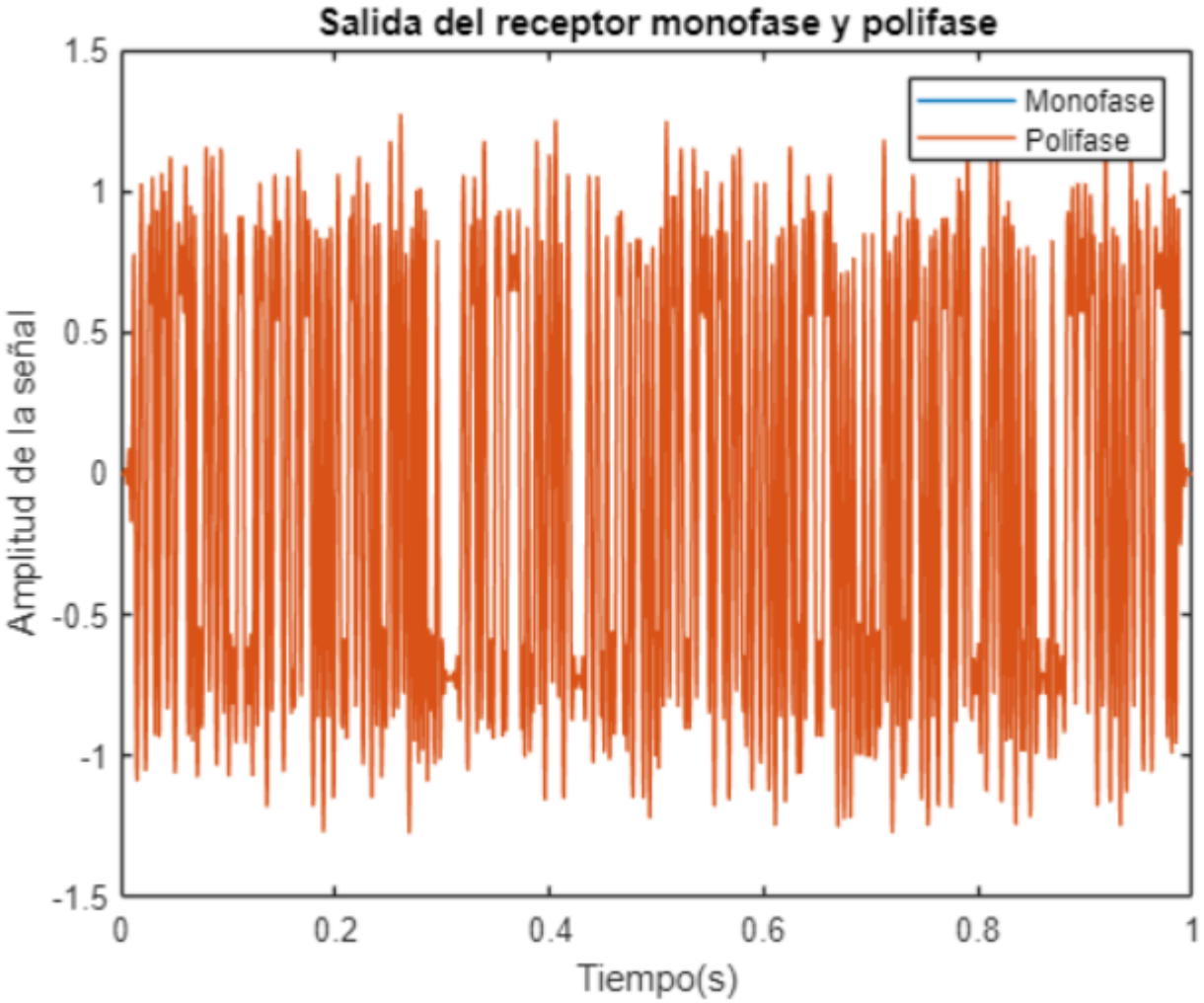
1.5.6. Receptor monofase

Seguimos el mismo proceso que en el desarrollo teórico, es decir, realizamos el filtrado y después el downsamplig dando lugar a la siguiente salida:



1.5.7. Receptor Polifase

Al igual que en el apartado anterior, realizamos los mismos pasos que en el desarrollo teórico, y representamos la salida en MATLAB:




```
%Volvemos a hallar la norma
normarx = norm(salidapolifaserx-monofaserx)
```

```
normarx = 1.6163e-14
```

Al igual que en transmisión, hallamos la norma para ver las diferencias entre la salida del monofase y el polifase y volvemos a concluir que no hay diferencias. Por lo tanto, las salidas teóricas del monofase y polifase para los filtros receptores son correctas.

Por último, hay que fijarse, que aunque las salidas de recepción y transmisión son iguales , tienen distinta amplitud, esto es debido al funcionamiento de la función 'rcosdesing' en matlab, al haber un número de coeficientes de entrada mayor en transmisión su salida está más atenuada por esta función, por tanto, habrá que hacer un ajuste de la ganancia previa a la entrada de recepción, con un valor de : $\sqrt{\frac{10}{19}}$

2.Ajuste de AGC

2.1.SNR teórica

El control automático de ganancia (AGC) es esencial para ajustar el nivel de la señal a la entrada del convertidor analógico-digital (CA/D) dentro de su rango dinámico. Nuestra meta radica en determinar la potencia en la salida del AGC para maximizar la relación señal-ruido (SNR); en otras palabras, buscamos que la proporción entre la potencia de la señal a la entrada del CA/D y la potencia del error de cuantificación sea lo más alta posible. Para calcular esto, asumimos que el CA/D emplea un cuantificador escalar uniforme de L bits (2^L niveles) para cuantificar el intervalo [-1, 1], y que la señal de entrada sigue una distribución laplaciana de media 0.

Fórmula de la relación señal ruido: $\mathrm{SNR} = \frac{\mathrm{Var}(X)}{\mathrm{Var}(E)}$

Fórmula de la función de densidad de la distribución Laplaciana: $f_x(X, b) = \frac{1}{2b} e^{\left(-\frac{|x-\mu|}{b}\right)}$

Que tiene como varianza teorica Var(X) = $2b^2$

Para generalizar el proceso para un valor arbitrario de L, primero definamos algunos conceptos clave.

Dado que hay 2^L en el rango [-1,1] y queremos que estén equiespaciados, la separación entre centroides será $\Delta = \frac{2}{(2^L - 1)}$

El error se define como la diferencia entre la entrada X y el valor resultante de cuantificar esa entrada, denotado como Q(X), donde Q(x) es la función que cuantifica la entrada al centroide más cercano. Entonces, el error se representa como: E = X - Q(X)

A partir de esto y del teorema fundamental de probabilidad:

$f_e(E) = f_x(x - Q(x))$

Ahora, para determinar qué valores de entrada generan un determinado error, distinguimos varios casos:

1-Si el error es 0, el valor de entrada coincide con uno de los centroides. x = c

2.Si el error está en el rango [$-\frac{\Delta}{2}, \frac{\Delta}{2}$] hay 2^L posibles valores de entrada. Cada uno de los centroides puede ser desplazado por ese error. x = -1 + $\Delta i + E$ para todo i en el rango[0, 2^L-1].

3-Si el error es menor que $-\frac{\Delta}{2}$, solo hay un valor posible de entrada: el primer centroide menos el error. x = -1+E.

4-Si el error es mayor que $\frac{\Delta}{2}$, solo hay un valor posible de entrada: el último centroide más el error. x = 1-Δ+E

Esto proporciona una estructura general para determinar los valores de entrada correspondientes a un error dado en un cuantificador escalar uniforme de L bits. Por tanto tendremos como funcion de densidad del error:

$$f_e(E) = \begin{cases} f_x(-1+E) & E < -\frac{\Delta}{2} \\ \sum_{i=0}^{2^L-1} f_x(-1+\Delta i+E) & -\frac{\Delta}{2} \le e \le \frac{\Delta}{2} \end{cases} \qquad f_e(E) = \begin{cases} f_x(1-\Delta+E)) & E > \frac{\Delta}{2} \\ \end{cases}$$

con todo esto podremos empezar a calcular la Var(E), objetivo de este apartado y teniendo en cuenta que $\mu = 0,$ como condición inicial.

$$\mathrm{Var}(E) = \int_{-\infty}^{\infty} E^2 f_e(E)\,\mathrm{dE}$$

es decir, el sumatorio de las integrales de cada una de sus partes

$$\mathrm{Var}(E) = \int_{-\infty}^{-\frac{\Delta}{2}} E^2 f_x(-1+E)\,\mathrm{dE} + \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} E^2 \sum_{i=0}^{2^L-1} f_x(-1+\Delta i+E)\,\mathrm{dE} + \int_{\frac{\Delta}{2}}^{\infty} E^2 f_x(1-\Delta+E)$$

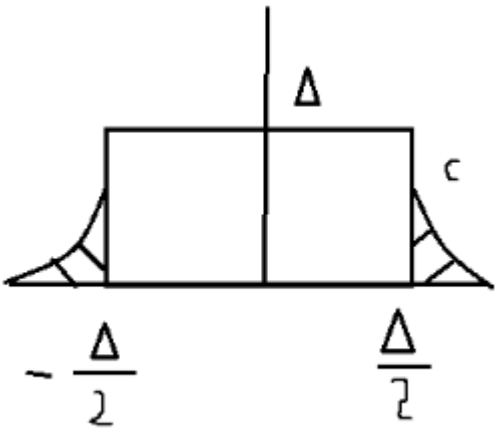
Desarrollamos la primera integral:

$$\int_{-\infty}^{-\frac{\Delta}{2}} E^2 f_x(-1+E)\,\mathrm{dE} = \frac{1}{2}(2b^2 - 2\mathrm{bE} + E^2)e^{\frac{(E-1)}{b}}\Big]_{-\infty}^{-\frac{\Delta}{2}} = \frac{1}{2}(8b^2 + 4b\Delta + \Delta^2)e^{\frac{(-\Delta-2)}{2b}}$$

Y ahora la tercera

$$\int_{\frac{\Delta}{2}}^{\infty} E^2 f_x(1-\Delta+E)\,\mathrm{dE} = \frac{1}{2}(2b^2 + 2\mathrm{bE} + E^2)e^{\frac{(\Delta-E-1)}{b}}\Big]_{\frac{\Delta}{2}}^{\infty} = \frac{1}{2}(8b^2 + 4b\Delta + \Delta^2)e^{\frac{(\Delta-2)}{2b}}$$

Ahora nos centramos en la segunda integral, nos fijamos que su fdp toma un valor constante(c), por lo tanto, la podemos definir como un rectangulo de base Δ y altura c asi que buscaremos calcular su área:



Sabiendo que la fdp de una distribucion siempre debe valer 1, calculamos el area de este rectangulo en funcion de las colas.

$$1 = \int_{-\infty}^{-\frac{\Delta}{2}} f_x(-1+E)\,\mathrm{dE} + c\Delta + \int_{\frac{\Delta}{2}}^{\infty} f_x(1-\Delta+E)\,\mathrm{dE}$$

$$c = \frac{1 - \int_{-\infty}^{-\frac{\Delta}{2}} f_x(-1+E)\mathrm{dE} + \int_{\frac{\Delta}{2}}^{\infty} f_x(1-\Delta+E)\,\mathrm{dE}}{\Delta}$$

$$c = \frac{1 - \left(\frac{1}{2}\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right)\right)}{\Delta}$$

Una vez tenemos el valor de c volvemos a la segunda integral siendo c el valor de la fdp:

$$\int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} E^2 \sum_{i=0}^{2^L-1} f_x(-1+\Delta i+E)\,\mathrm{dE} = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} E^2 * c\,\mathrm{dE} = c * E]_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} = c\frac{\Delta^3}{12} = \frac{\left(1 - \left(\frac{1}{2}\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right)\right)\right)\Delta}{12}$$

Una vez tenemos las 3 integrales solo falta realizar el sumatorio:

$$\mathrm{Var}(E) = \frac{1}{2}(8b^2 + 4b\Delta + \Delta^2)e^{\frac{(-\Delta-2)}{2b}} + \frac{1}{2}(8b^2 + 4b\Delta + \Delta^2)e^{\frac{(\Delta-2)}{2b}} + \frac{\left(1 - \left(\frac{1}{2}\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right)\right)\right)\Delta^2}{12} =$$

$$\frac{1}{2}(8b^2 + 4b\Delta + \Delta^2)\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right) + \frac{\Delta^2 - \left(\frac{\Delta^2}{2}\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right)\right)}{12}$$

Multiplicando por 12 ambos sumandos:

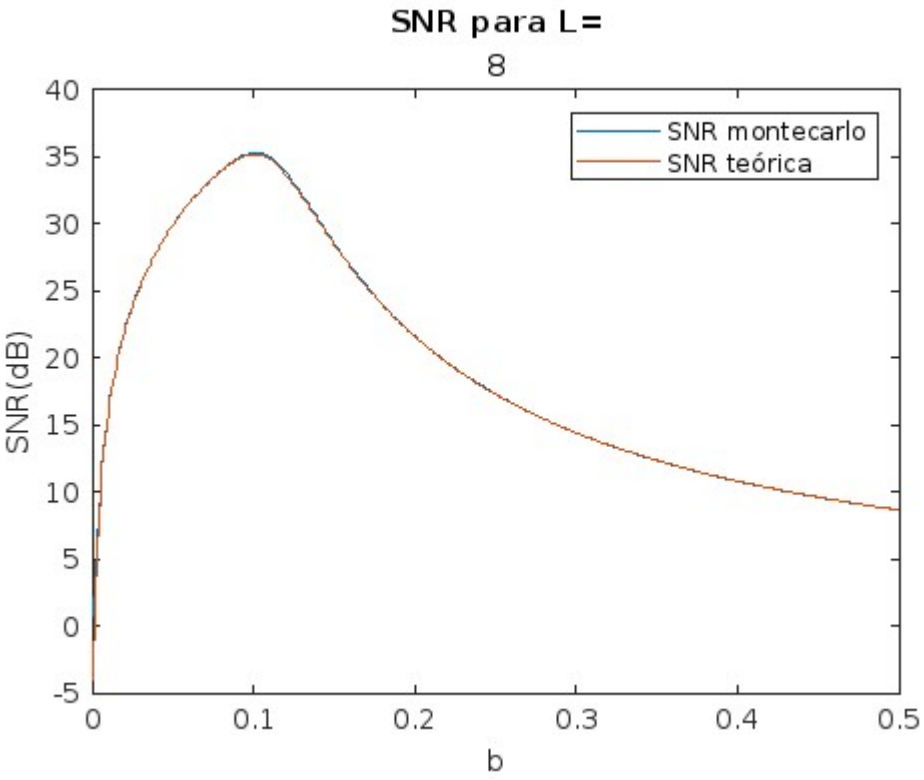
$$\mathrm{Var}(E) = \left(48b^2 + 24b\Delta + \frac{11}{2}\Delta^2\right)\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right) + \Delta^2$$

Para acabar volvemos a la SNR

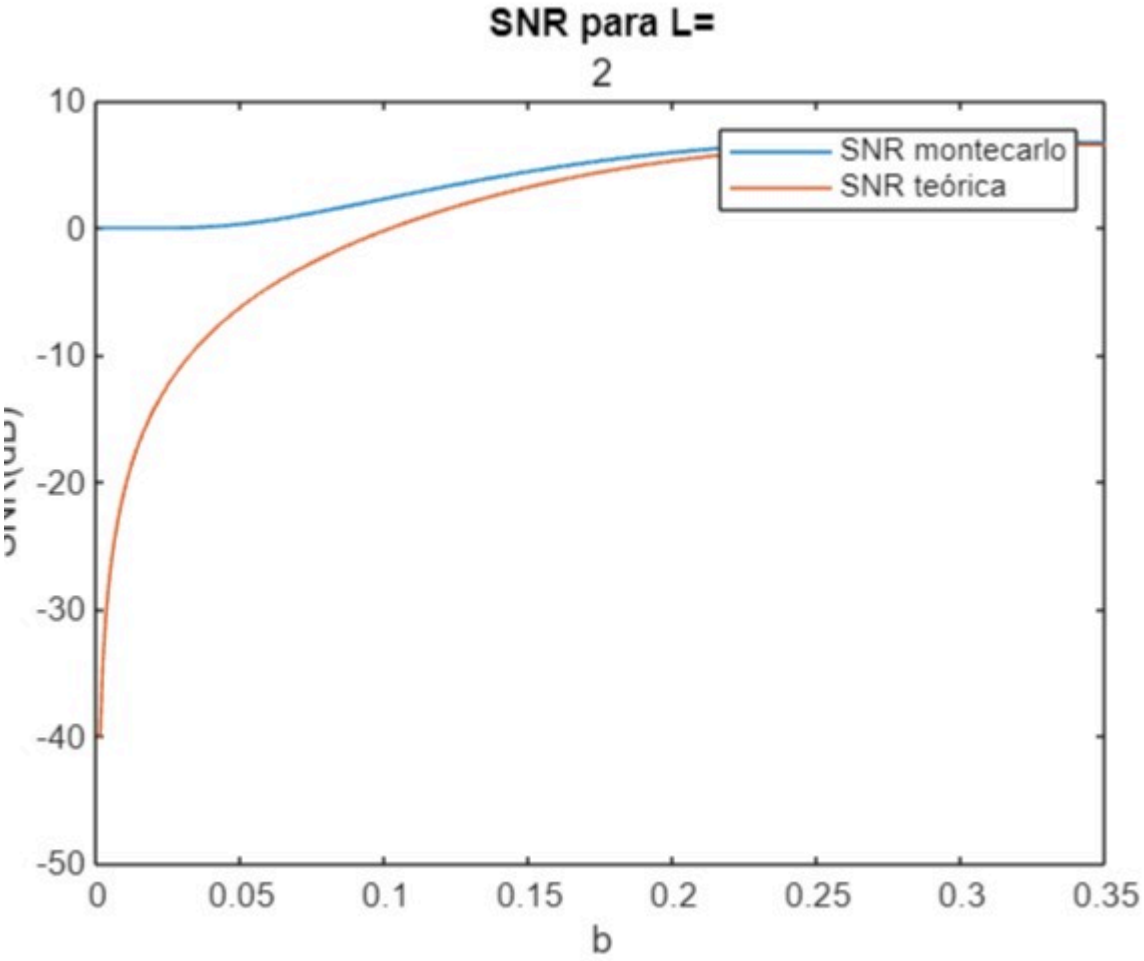
$$\mathrm{SNR} = \frac{\mathrm{Var}(X)}{\mathrm{Var}(E)} = \frac{2b^2}{\left(48b^2 + 24b\Delta + \frac{11}{2}\Delta^2\right)\left(e^{\frac{(-\Delta-2)}{2b}} + e^{\frac{(\Delta-2)}{2b}}\right) + \Delta^2}$$

2.2. Implementación de Montecarlo

En este apartado, interpretaremos los resultados obtenidos de la SNR teórica mencionada anteriormente, comparándolos con una simulación de Monte Carlo de una señal Laplaciana estándar. Los resultados de esta simulación están documentados en el script `montecarlo.mlx` , que se adjunta a esta memoria. Para valores intermedios de L, la SNR teórica y la SNR obtenida mediante Monte Carlo son prácticamente idénticas, con un valor óptimo de b cercano a 0.1.

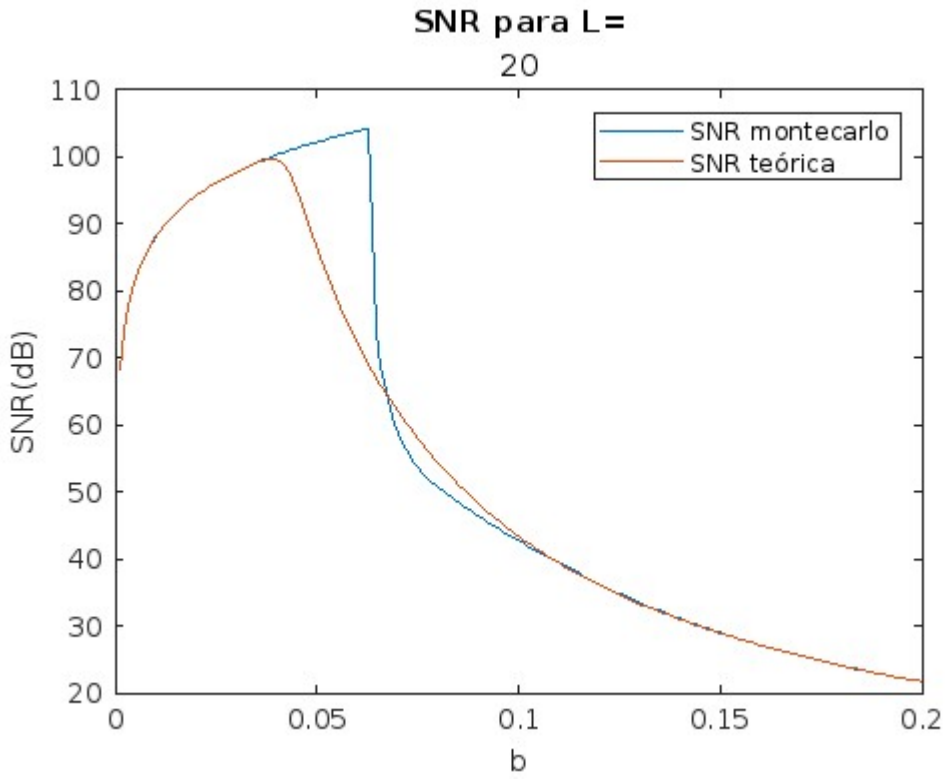


Para valores bajos de L, por ejemplo L=2, tenemos una realidad distinta. En esta ocasión, la SNR teórica toma valores negativos para valores bajos de b mientras que la de Monte Carlo toma valores cercanos a 0.



Dado que esta variación ocurre al inicio de la gráfica, podemos suponer que la diferencia se encuentra en el intervalo de valores que tomamos de la segunda integral, $\left[-\frac{\Delta}{2},\frac{\Delta}{2}\right]$. Si volvemos a la ecuación teórica de esta parte del resultado, podemos observar que este valor está representado por un término $\frac{\Delta^2}{12}$ que multiplica una exponencial en función de b. Al hacer una aproximación rectangular, asumimos que el área de esa integral sería constante, lo cual no es del todo cierto. Esto deja la variación en función de $\frac{\Delta^2}{12}$ para valores bajos de b, lo que hace que, para valores bajos de L, esta variación tienda a 0, a diferencia de los valores obtenidos por Monte Carlo, que tienden a 1.

Para valores altos de L, por ejemplo L=20, notamos una variación para valores intermedios de b y una subida de valores en la SNR.



Esta variación se debe a que, para valores cercanos al borde del intervalo [−1,1], algunos valores caen dentro y otros fuera del rango. Con un L tan grande (es decir, con un Δ muy pequeño), los valores fuera del rango tendrán una variación mucho mayor que los incluidos en este intervalo, lo que resulta en un valor de SNR significativamente diferente del interpretado teóricamente y en valores distintos cada vez que se ejecute la simulación.

Por otro lado, también observamos que el valor óptimo de la varianza (b) decrece significativamente al aumentar L. Esto se puede explicar porque el valor de b que maximiza la SNR se encuentra en el intervalo $\left[-\frac{\Delta}{2},\frac{\Delta}{2}\right]$, dependiendo del valor del factor $\frac{\Delta^2}{12}$, el cual disminuye a medida que aumenta L.

Todas estas gráficas son generadas en el archivo 'Montecarlo.mlx'.