Facultad de Ingeniería

Electricista
Mecánica
Química
Telecomunicaciones

INGENIERIA
UNRC

UNRC

Universidad Nacional de Río Cuarto

Universidad Nacional de Río Cuarto

Facultad de Ingeniería.

Cátedra:

Sistemas y Señales II

Código: 0021

Guía 1 GNU RADIO

Integrantes de Ingeniería en Telecomunicaciones	
Barroso Maximiliano	40.503.951
Bongiovanni Octavio	44.607.178
Bordón Federico	40.729.999

Fecha de Presentación: 16/10/2025

Calificación: ____________

Firma Docente: ____________

Curso Procesamiento digital de señales con GNU RADIO Companion

GUÍA N° 01

Introducción a GNU RADIO Companion, señales complejas, muestreo y cuantización de señales, interpolación de muestras, diezmado e interpolación.

1)

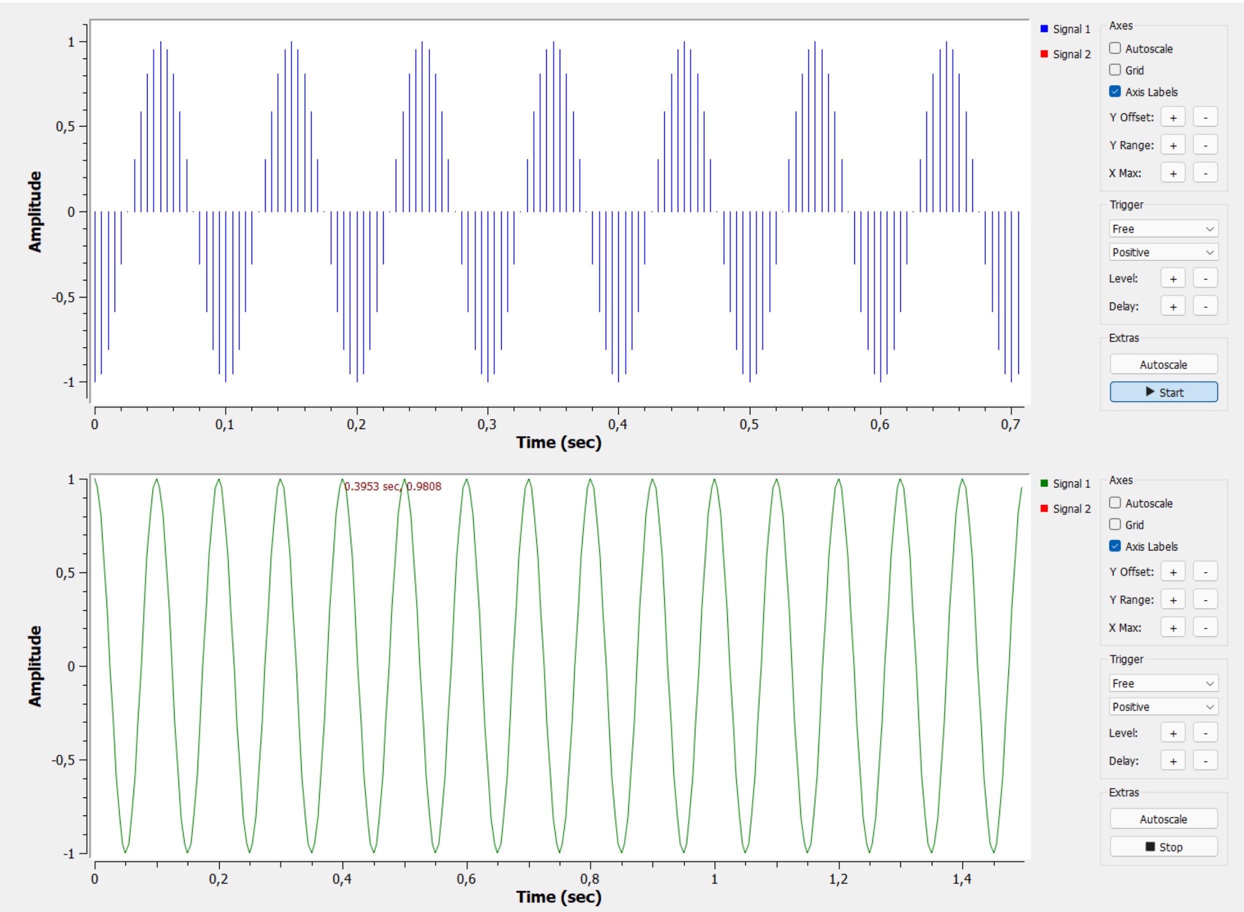
a. $x(t) = A.cos(2\pi.f.t)$

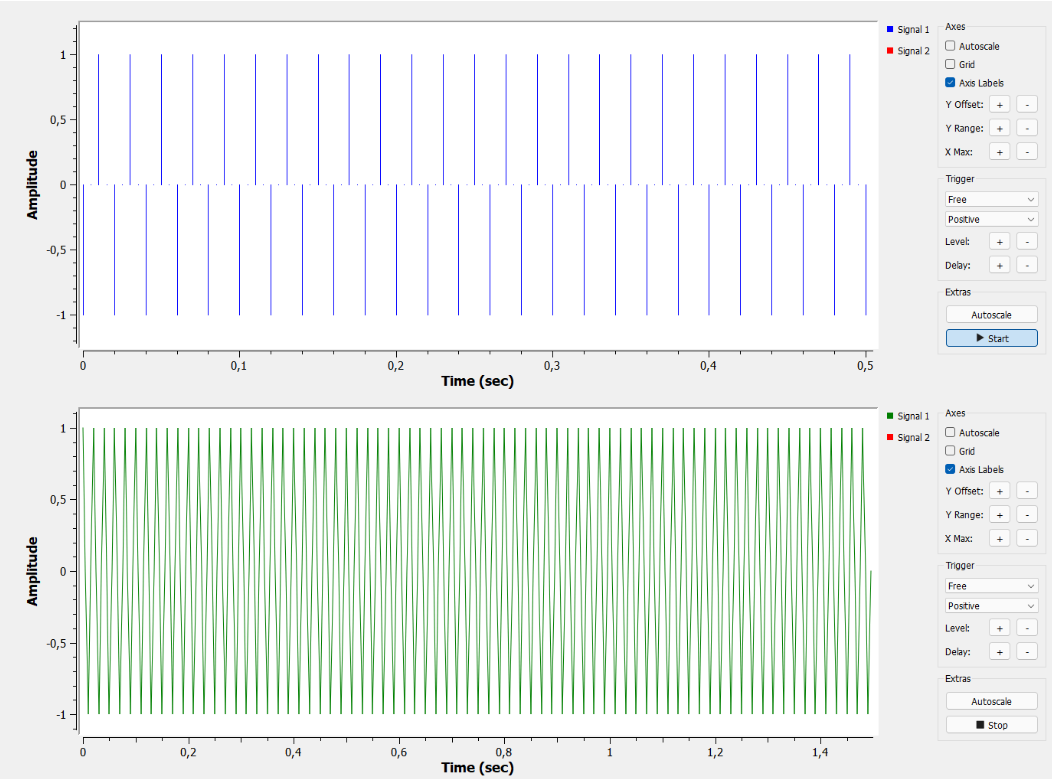
$siendo\ Ts = 1/fs = \frac{1}{200} = 5.10^{-3}s$

$x_1(t) = cos(10.2\pi.t) \rightarrow x[nTs] = cos[10.2\pi.Ts.n] = x_1[n] = cos[\frac{\pi}{10}n]$

$x_2(t) = cos(50.2\pi.t) \rightarrow x[nTs] = cos[50.2\pi.Ts.n] = x_2[n] = cos[\frac{\pi}{2}n]$

b.

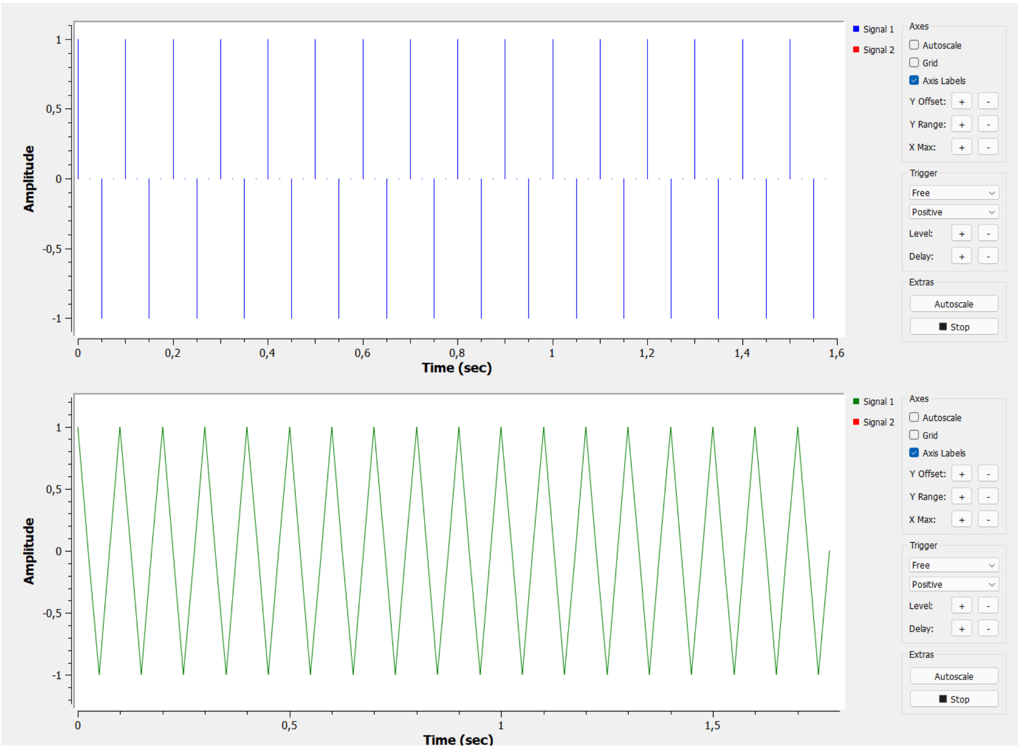




Con nueva fs = 40 Hz por Teorema de Nyquist:

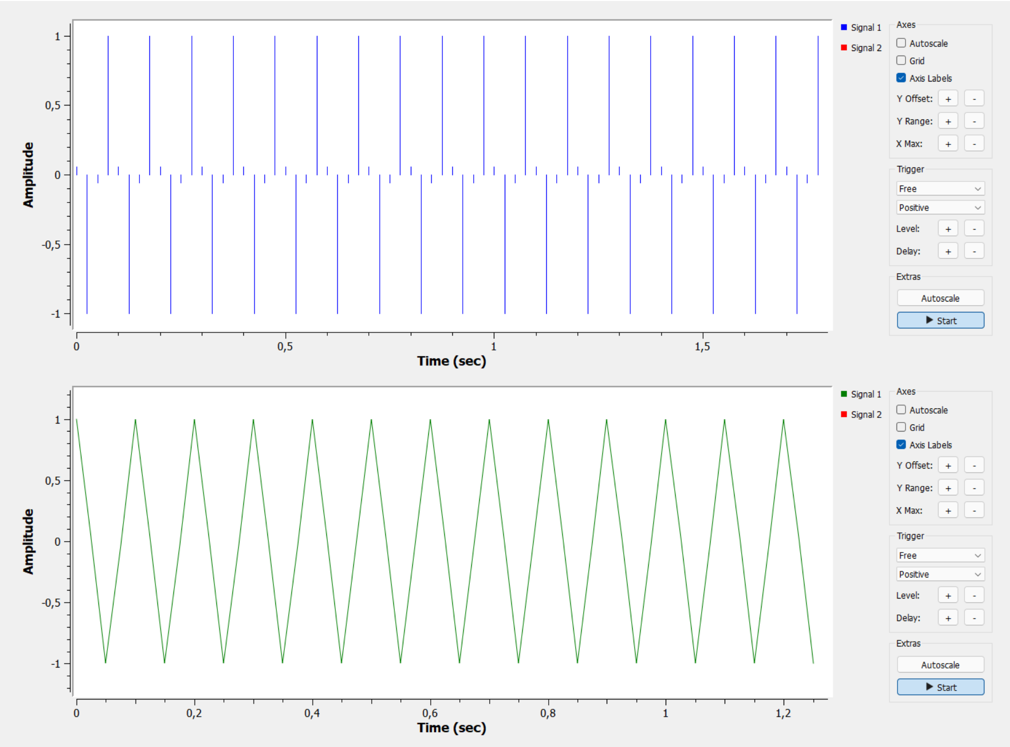
$$fs_1 > 2.fm_1 \rightarrow 40 > 20 \; se\ cumple\ para\ x_1(t)$$

$$x_1(t) = cos(10.2\pi.t) \rightarrow x[nTs] = cos[10.2\pi.Ts.n] = x_1[n] = cos[\frac{\pi}{2}n]$$



$fs_2 > 2. fm_2 \rightarrow 40 > 100\ no\ se\ cumple\ para\ x_2(t)$

$x_2(t) = cos(50.2\pi.t) \rightarrow x[nTs] = cos[50.2\pi.Ts.n] = x_2[n] = cos[\frac{5\pi}{2}n]$



c. Para la nueva fs = 40 Hz observa para la última señal el fenómeno de aliasing, donde no se cumple el teorema del muestreo y no se obtiene la información suficiente para recuperarla, aparece una señal de más baja frecuencia que la fmax. Esto se explicará mejor en la consigna 3).

Otra cuestión es la siguiente, usando la identidad trigonométrica:

$cos(\theta) = cos(\theta - 2\pi k)$ para cualquier entero k, entonces

$cos[\frac{5\pi}{2}n] = cos[\frac{5\pi}{2}n - 2\pi n] = cos[(\frac{5}{2} - 2)\pi n] = cos[\frac{\pi}{2}n]$

Se obtiene como resultado una señal discreta idéntica a la de 50 Hz, que es igual a la de 10 Hz o en otras palabras, no se puede distinguir una de la otra.

d. Para nuestro caso de fo = 10 Hz y fs = 40 Hz:

- **Si k = 1:** fa = 10 + 1 . 40 = 50 Hz. → $caso\ de\ la\ x_2(t)$
- **Si k = 2:** fa = 10 + 2 . 40 = 90 Hz.
- **Si k = 3:** fa = 10 + 3 . 40 = 130 Hz

e. Una señal muestreada no es posible distinguirla de otra si sus frecuencias son un múltiplo entero de la frecuencia de muestreo más la frecuencia original.

La ecuación matemática que describe todas las frecuencias fa que son un alias de una frecuencia original fo es:

$f_a = f_0 + k.f_s;\ con\ k\epsilon Z\ \{...; -2; -1; 1; 2;...\}\ \wedge\ \neq 0$

3)

$x(t) = 5cos(2\pi 5000t)$; **Teorema del Muestreo:** $fs > 2.fm$

a. ¿Qué pasa cuando usamos una fs (fs: frequency sample, o frecuencia de muestreo, o tasa de muestreo) menor a la mínima requerida por el criterio de Nyquist?

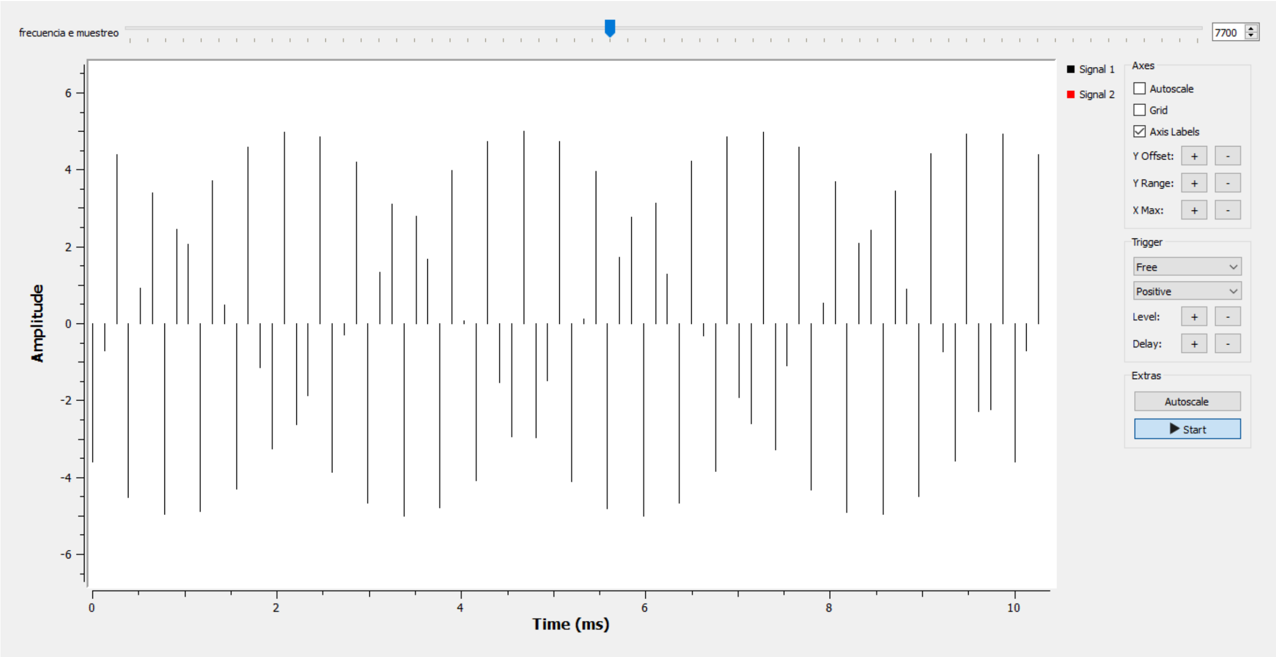
Si se usa una frecuencia de muestreo (fs) menor a la mínima requerida (fs<2fmax), ocurre el fenómeno destructivo e irreversible llamado aliasing o superposición espectral.

Las altas frecuencias de la señal original se "pasan" a bajas frecuencias en la señal muestreada. Esto sucede porque las réplicas del espectro de la señal original, que se generan en el proceso de muestreo, se solapan unas con otras. Una vez que este solapamiento ocurre, es imposible reconstruir la señal original a partir de las muestras. La información está corrupta para siempre.

Ejemplo de aliasing para la $x(t) = 5cos(2\pi 5000t)$

Se muestra con una $fs < 2.fm = 7,7\ kHz\ \ siendo\ Ts = 1/fs = \frac{1}{7700} = 1,3\text{x}10^{-4}s$

$x[nTs] = 5\ cos[2\pi 5000.nTs] = 5\ cos[\frac{13\pi}{12}n]$



Se observa claramente que la señal muestreada es una señal de una frecuencia menor a la original de tiempo continuo, se ha perdido información de esta última y no es posible recuperarla.

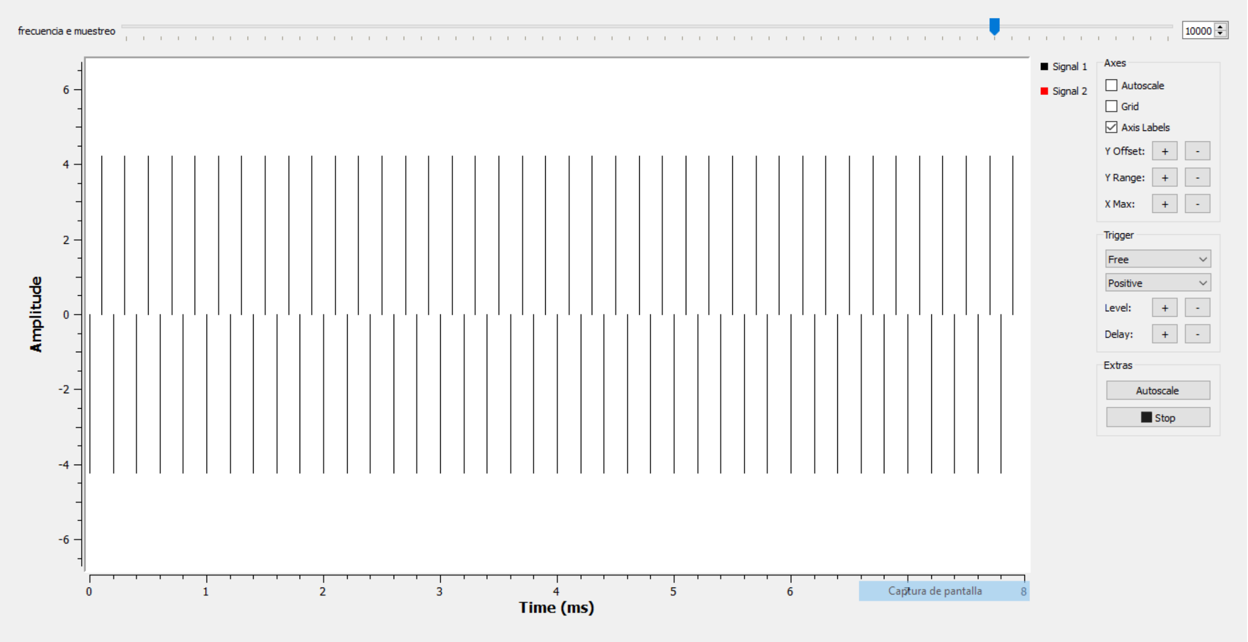
b. ¿Una fs igual a la de Nyquist que produce?

Cuando la frecuencia de muestreo es exactamente la frecuencia de Nyquist (fs=2fmax), teóricamente, la reconstrucción de la señal original es posible. Sin embargo, esto presenta un problema práctico para reconstruir la señal, se necesitaría un filtro pasabajos ideal. Este filtro tendría que tener una transición instantánea de dejar pasar todas las frecuencias hasta fmax y eliminar por completo todas las frecuencias por encima. El problema: Un filtro ideal es matemáticamente perfecto pero físicamente irrealizable. Por lo tanto, en la práctica, nunca se muestrea exactamente a la frecuencia de Nyquist.

Ejemplo de límite teórico para el muestreo de la $x(t) = 5\, cos(2\pi 5000t)$

Se muestra con una $f = 2.fm = 10\, kHz\ \ siendo\, Ts = 1/fs = \frac{1}{10000} = 1x10^{-4} s$

$x[nTs] = 5\, cos[2\pi 5000.nTs] = 5\, cos[\pi n]$

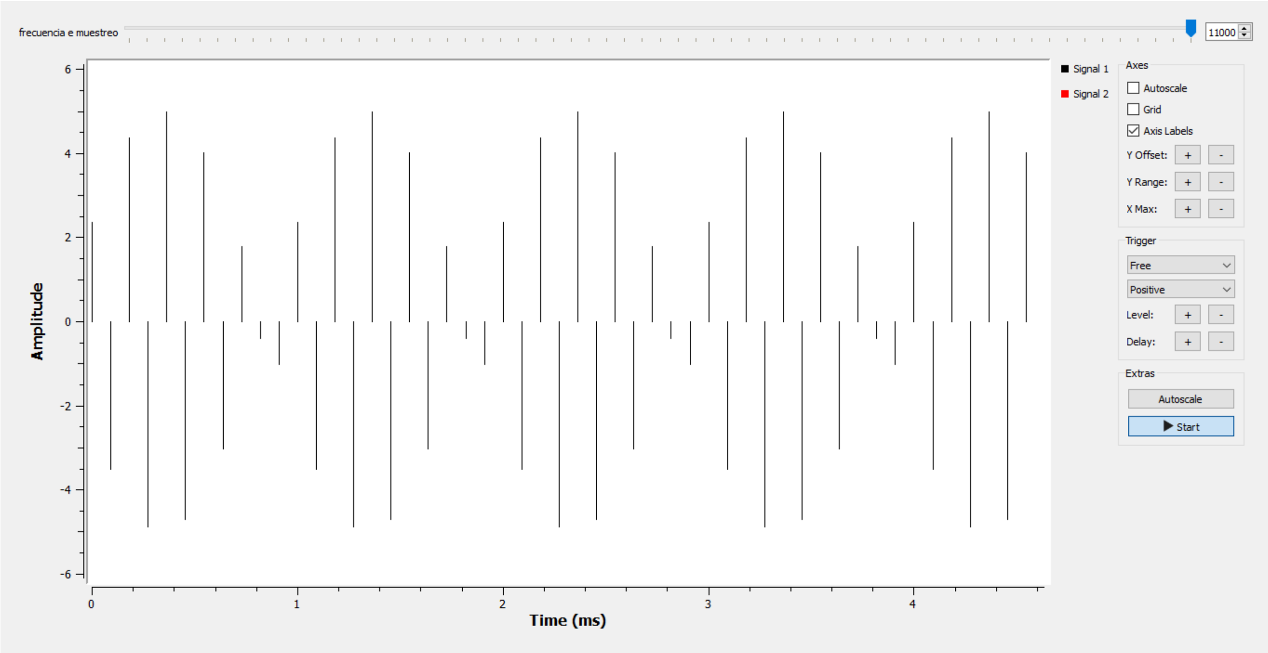


Aquí se muestra una señal que contiene la información suficiente para recuperar la señal original.

Observación, la señal de tiempo continuo tiene una amplitud de 5 mientras que la secuencia discreta tiene una amplitud próxima a 4, se desconoce la causa de esto, probablemente algún bug del propio GNU Radio.

c. ¿y una superior a la de Nyquist?, ¿qué mejora aún más al aumentarla

Esta es la forma más usada en la práctica y habitual de trabajar. Se conoce como sobremuestreo (oversampling). Cuando fs > 2fmax, se crea un espacio vacío en el dominio de la frecuencia entre el espectro de la señal original y su primera réplica. El filtro ya no necesita ser perfecto; su pendiente de atenuación puede caer suavemente dentro de esta banda de guarda, eliminando eficazmente las réplicas no deseadas sin afectar la señal de interés.



Ejemplo de sobremuestreo para la $x(t) = 5cos(2\pi 5000t)$

Se muestra con una $fs > 2.fm = 11\ kHz\ siendo\ Ts = 1/fs = \frac{1}{11000} = 9,1x10^{-5}s$

$x[nTs] = 5\ cos[2\pi 5000.nTs] = 5\ cos[\frac{10\pi}{11}n]$

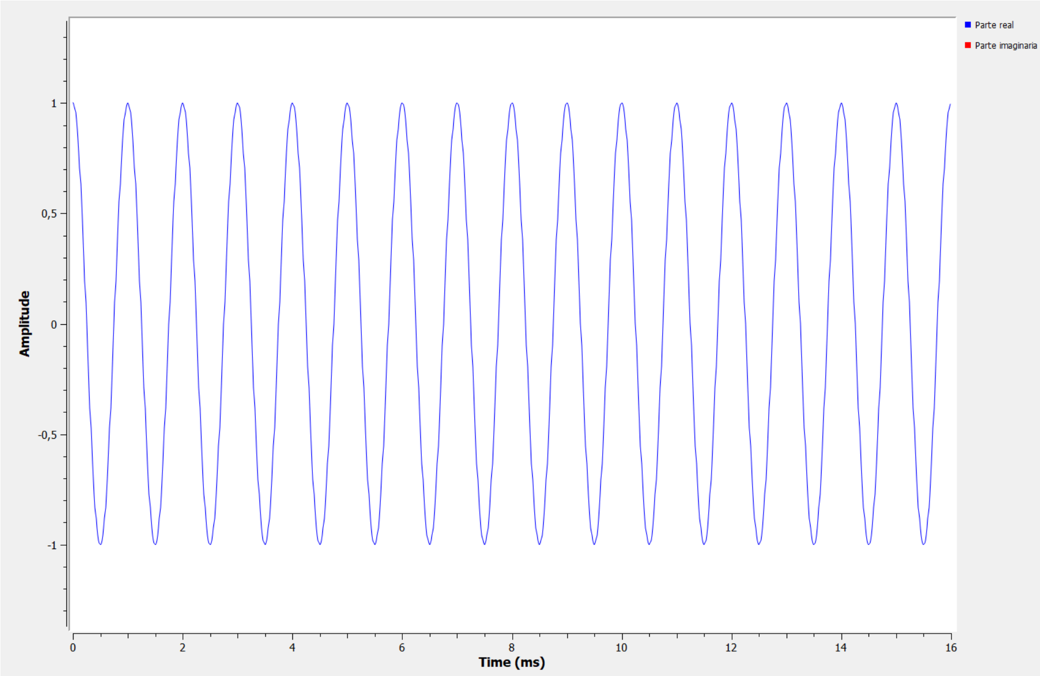
5)

a. Considere la representación fasorial de la señal x(t) ¿A qué velocidad angular se mueve el fasor? rta: ω=2π(1000)=2000π rad/s

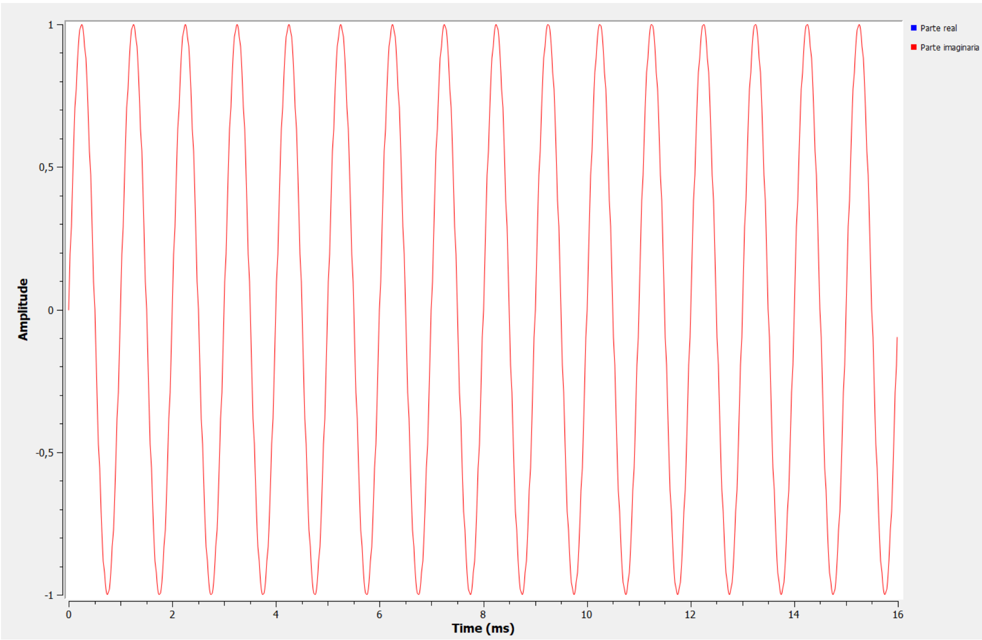
- ¿Cuántas vueltas da el fasor en el tiempo de 1 segundo? rta: 1000 vueltas por segundo

-¿cuánto tiempo tarda el fasor en dar una vuelta? rta: 1/1000=1 ms

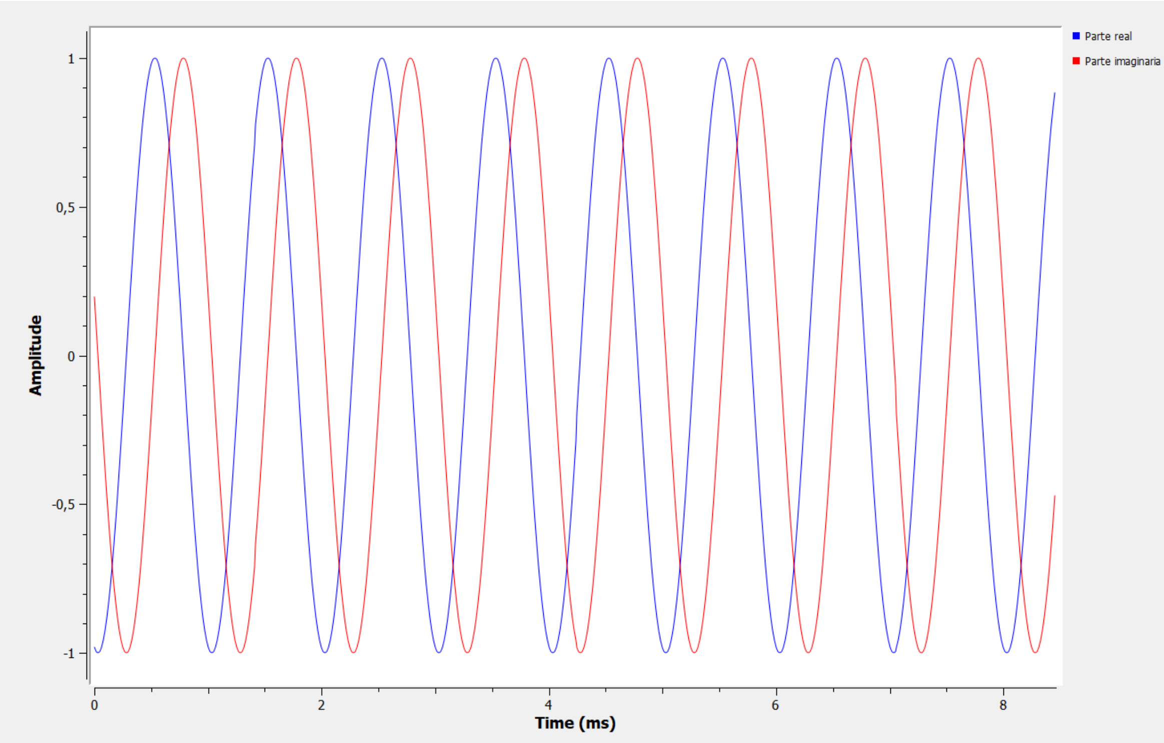
b).



Parte real: cos(2π1000t)

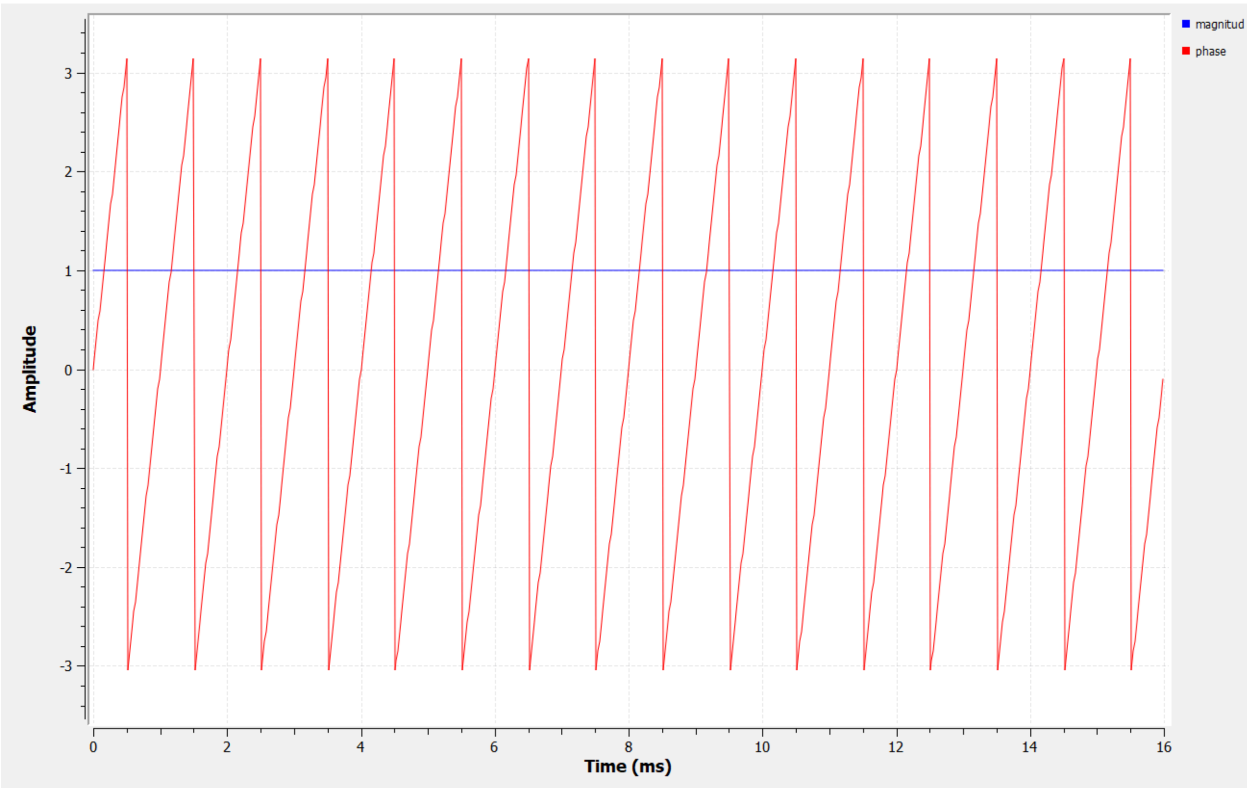


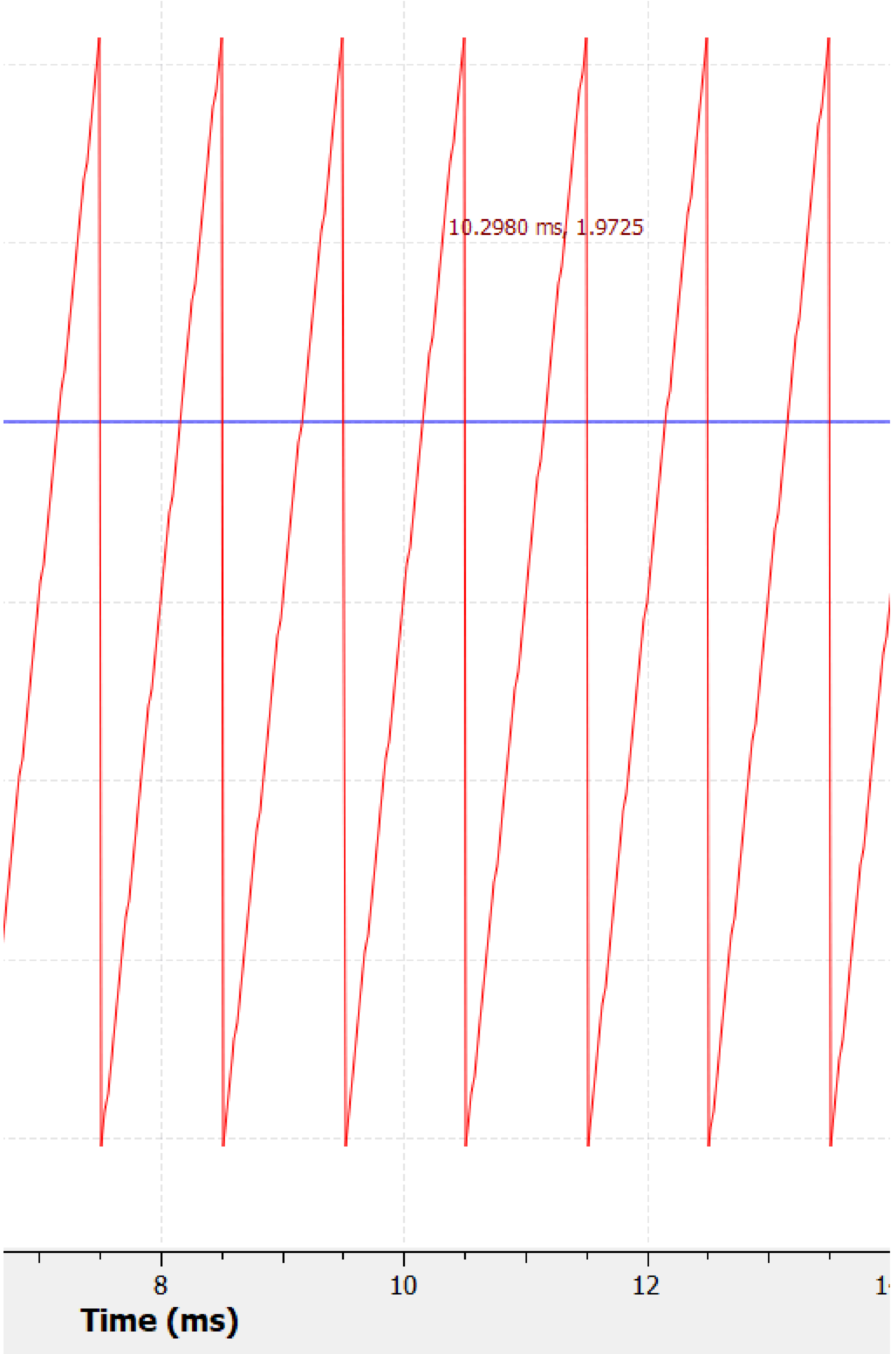
Parte imaginaria: sin(2π1000t)



Vemos que poseen la misma amplitud y están desfasadas 90°.

c. La línea Roja es la fase y la azul es la magnitud, la cual es constante.

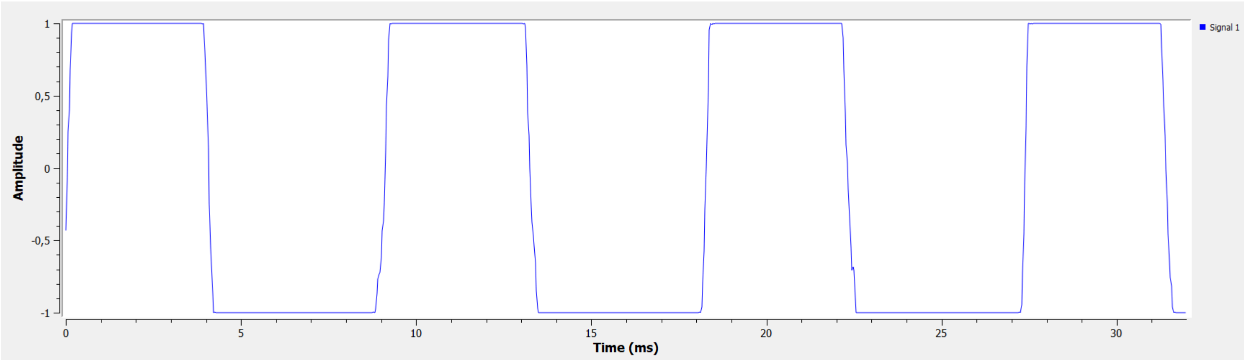




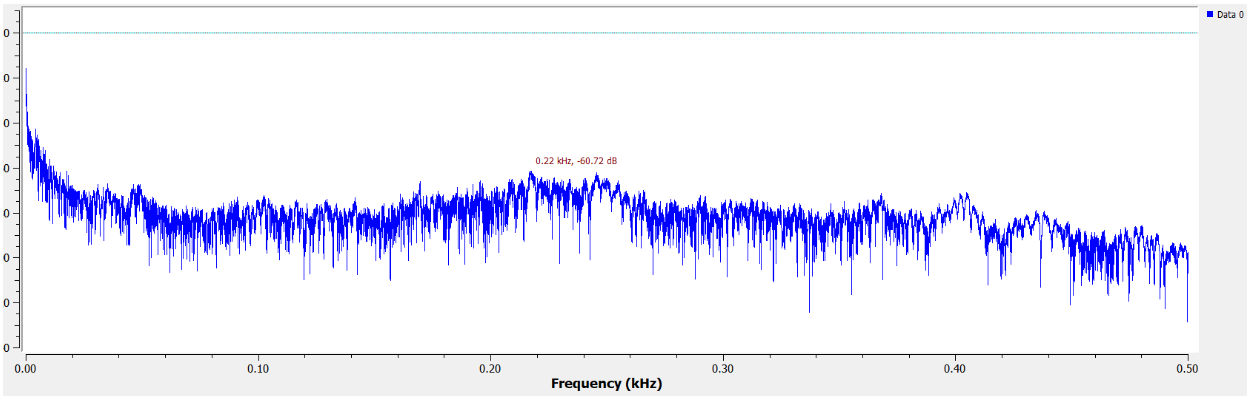
Para f=1000Hz y t≈10.3ms se esperan 10.3 vueltas y una fase de 1.885 radianes. El valor observado en la imagen es cercano, está medido en 10.2980 ms y es de 1.972 radianes. Pero el valor teórico sería de 0.298×2π≈1.872 radianes lo cual indica un error de visualización.

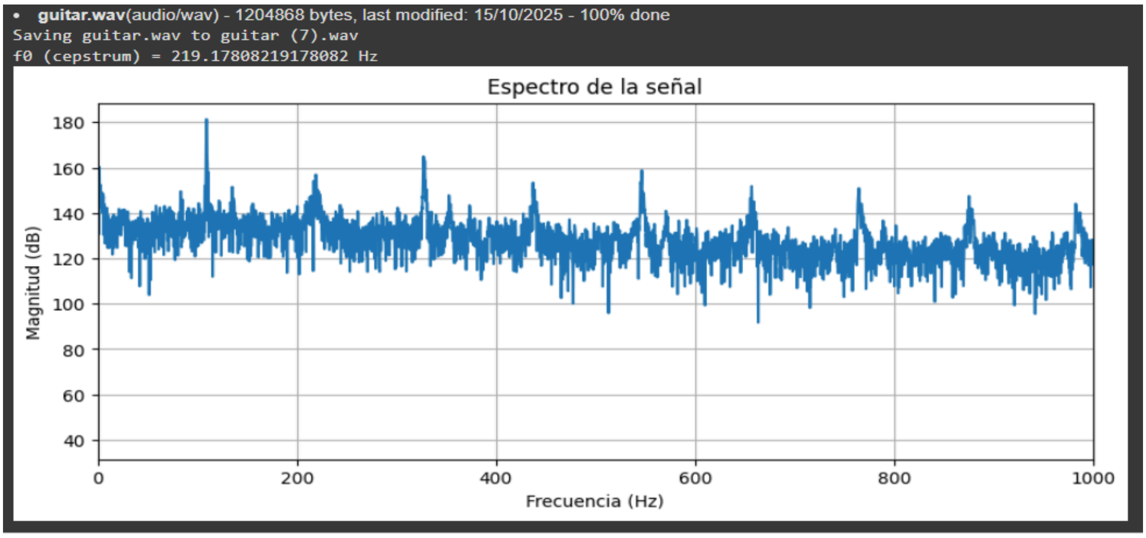
7)

La forma en el tiempo es la siguiente, parece una onda cuadrada.



En frecuencia se observa un pico en 220hz





Con un script en python utilizando cepstrum se obtiene la frecuencia fundamental f0 es de 219.17hz que es la mitad de 440 hz “A” o “La” lo que sería un La3. Pero al graficar el espectro en matplotlib surgen dudas respecto a los picos.

9)

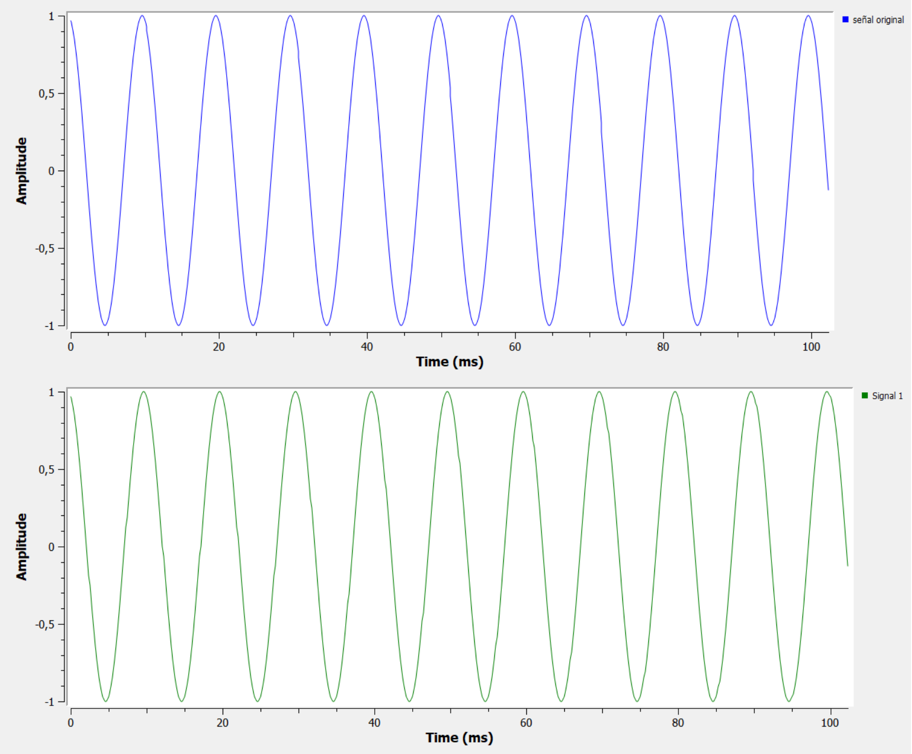
a. i. Sí, absolutamente. El formato de punto flotante de 32 bits (o cualquier formato digital finito) introduce un error de cuantización. La razón fundamental es que se está intentando representar un número potencialmente infinito de valores de una señal analógica continua con un número finito de combinaciones de bits. Esto lleva a que se cometan errores al cuantizar dependiendo si el método es de redondeo o truncado.

Error de Redondeo-Truncado: Cuando el resultado de una medición o un cálculo no coincide exactamente con una de las representaciones posibles, debe ser redondeado al valor más cercano, esta es la fuente directa del error.

ii. No, es teóricamente imposible evitar por completo el error de cuantización cuando se trabaja con señales digitales. El error es una consecuencia inherente de la digitalización. Una señal analógica tiene una amplitud de precisión infinita, mientras que una representación digital tiene una precisión finita, definida por el número de bits, siempre habrá incertidumbre. Entonces, no podemos evitarlo, pero sí podemos reducirlo o minimizarlo hasta un punto en que sea insignificante para nuestra aplicación y es aumentando el número de bits, si el sistema digital lo permite, es la solución principal y más efectiva.

Más Bits = Más Niveles = Menor Error

b. Señal original y señal cuantizada en 12 niveles, no se observa diferencias:

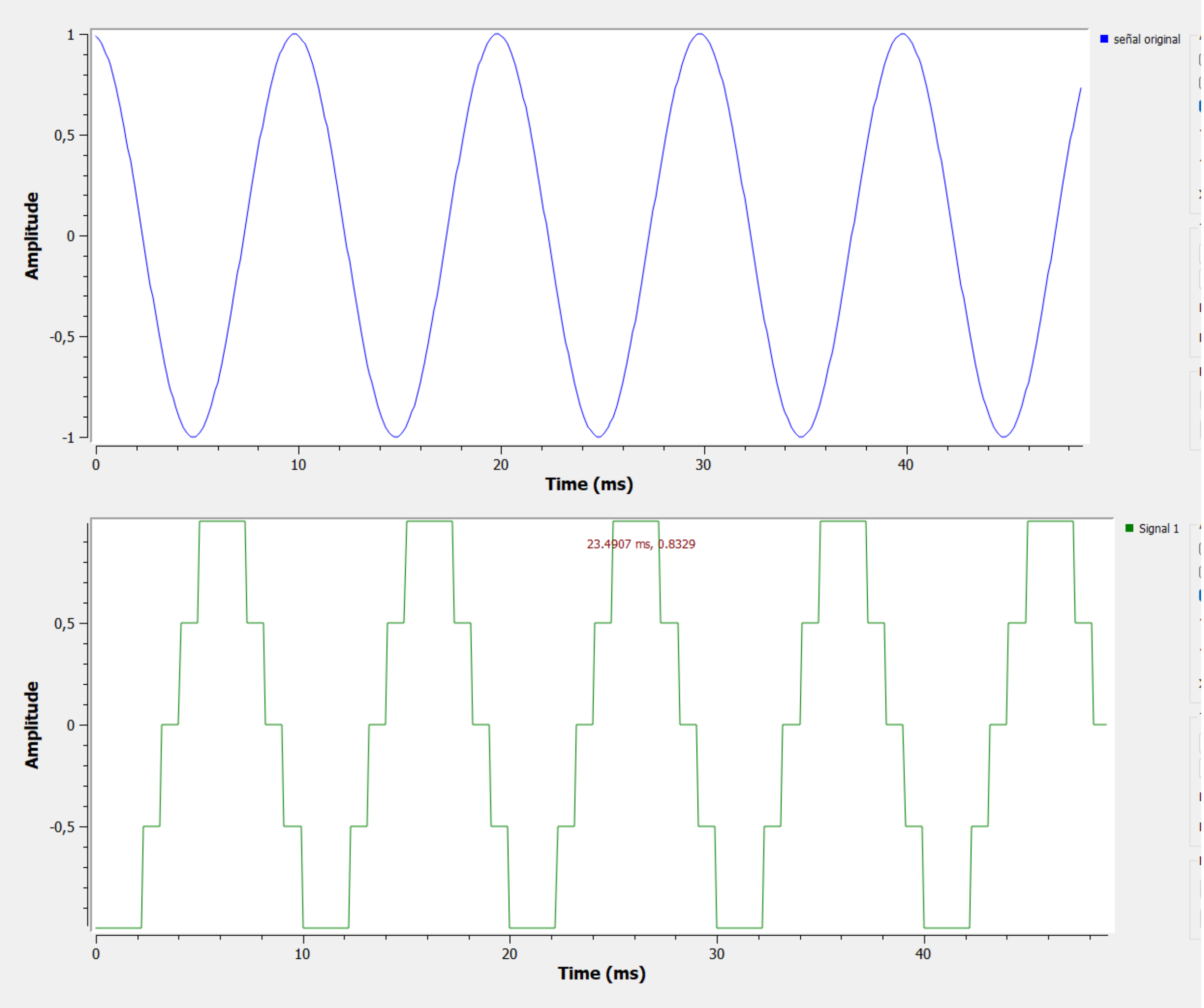


$L = 2^b \rightarrow b = log_2(12) = 3.58$

Por lo cual se necesitan al menos 4 bits para 12 niveles.

c. Para tener 4 niveles: $L = 2^b \rightarrow b = log_2(4) = 2$

Entonces se necesitan 2 bits para 4 niveles.



d.

Calculamos la cantidad de bits para representar 4 niveles: $L \Rightarrow 2^b = 4 \Rightarrow b = 2$

De SyS 1 sabemos que la potencia de una señal sinusoidal $x(t) = A.cos(\omega t + \phi)$ es su valor cuadrático medio: $P_s = \frac{A^2}{2}$

Donde A es la amplitud máxima de la sinusoide.

El error de cuantización se modela como un ruido con una distribución uniforme en el intervalo [- Δ/2, Δ/2], donde Δ (delta) es el tamaño/resolución de un escalón de cuantización. La potencia de este ruido/error (su varianza) es: $P_Q = \frac{\Delta^2}{12}$

El tamaño del escalón (Δ) se calcula dividiendo el rango total de la señal (2A) por el número de niveles: $L\ o\ 2^b \Rightarrow \Delta = \frac{2A}{L} = \frac{2A}{2^b}$

Sustituyendo Δ en la fórmula de la potencia del ruido:

$$P_Q = \frac{1}{12}.(\frac{2A}{2^b})^2 = \frac{4.A^2}{12.(2^b)^2} = \frac{A^2}{3.(2^{2b})}$$

Ahora podemos calcular el SNRQ:

Dividimos la potencia de la señal entre la potencia del ruido. Y vemos que no depende de la amplitud.

$$SNRQ = \frac{P_S}{P_Q} = \frac{A^2/2}{A^2/(3.2^{2b})} = \frac{3.2^{2b}}{2} = 1,5\ .\ 2^{2b} = 24$$

Nos queda para SNRQ en decibelios (dB):

$$SNRQ(dB) = 10.log_{10}(SNRQ) = 10.log_{10}(24) \approx 13.8\, dB$$

i. Por cada bit que se agrega al cuantizador, la Relación Señal a Ruido de Cuantización (SNRQ) aumenta en aproximadamente 6 decibelios (dB), vamos a ver el porqué:

Agregar un bit duplica los niveles, es decir, cada bit que añadimos duplica el número de niveles de cuantización disponibles. Con b bits, tienes 2^b niveles. Con b+1 bits, tienes 2^b+1 = 2 . 2^b niveles. Al tener el doble de niveles en el mismo rango de voltaje, la distancia de un escalón a otro de cuantización Δ se reduce a la mitad.

Al reducir el escalón se reduce el error, este error de cuantización, en su mayor parte, ocurre dentro de los límites de un escalón. Se puede modelar como un error que varía entre -Δ/2 y Δ/2.

Al reducir el tamaño del escalón Δ a la mitad, el rango del error también se reduce a la mitad. Entonces reducir el error reduce la potencia del ruido, la potencia del ruido de cuantización P_q es proporcional al cuadrado del tamaño del escalón Δ^2.

Si reducimos Δ a la mitad $\Delta/2$, la nueva potencia del ruido es proporcional a:

$(\Delta/2)^2 = \Delta^2/4$ ve que la potencia del ruido se reduce en un factor de 4.

Ahora para la conversión a decibelios (dB) la relación en decibelios; si la potencia de la señal se mantiene constante y la potencia del ruido se reduce por 4, el aumento en la SNRQ es:

$$10.\,log_{10}\left(\frac{Px}{P_{Q\,NUEVO}}\right) - 10.\,log_{10}\left(\frac{Px}{P_{Q\,ANTES}}\right) = log_{10}\left(\frac{P_{ANTES}}{P_{Q\,NUEVO}}\right) = 10.\,log_{10}(4) =\approx 6,02$$

Por eso, cada bit adicional, que reduce la potencia del ruido de cuantización en un factor de 4, mejora la SNRQ en aproximadamente 6 dB.

La fórmula general para la SNRQ de una onda senoidal de máxima amplitud en un cuantizador de N bits lo demuestra claramente:

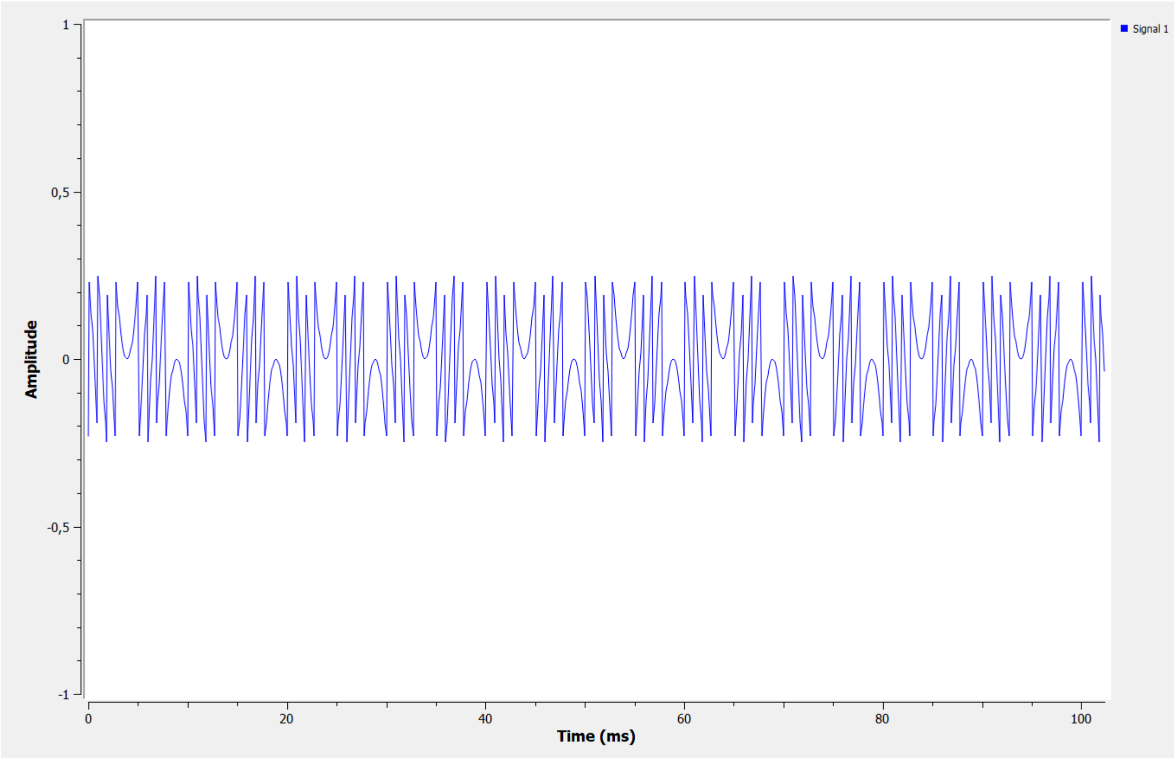
$$SNRQ(dB) \approx 6,02\,.\,b + 1,76$$

Podemos usar la fórmula aproximada para corroborar para el caso de SNRQ = 24:

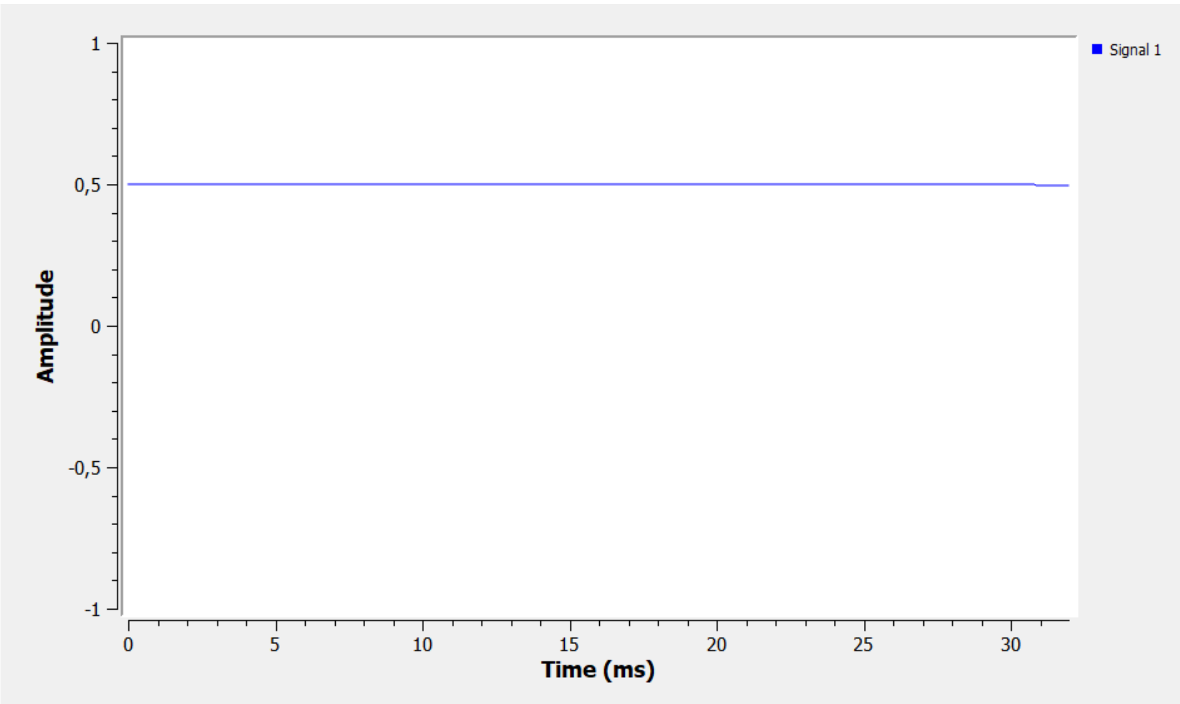
$$SNRQ(dB) \approx 6,02\,.\,2 + 1,76 = 12.04 + 1.76 = 13.8\,dB$$

El resultado coincide perfectamente.

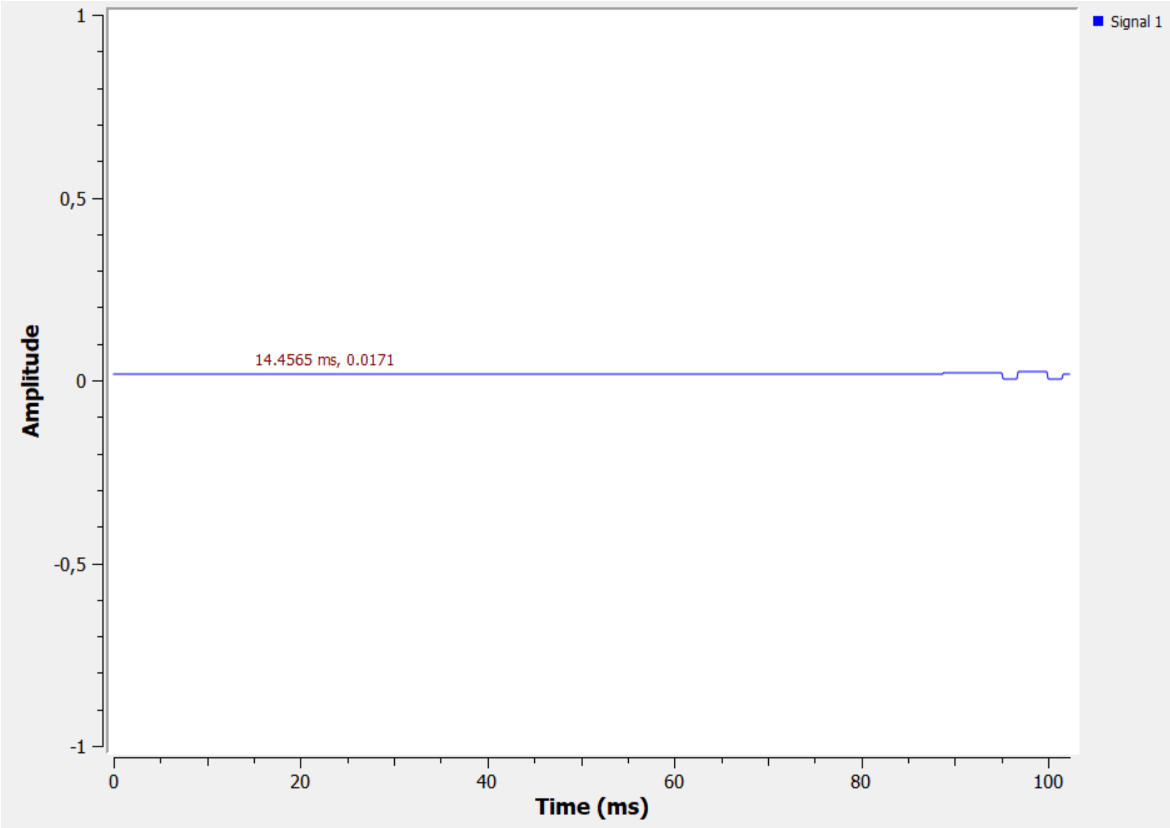
e. Al Restar la señal cuantizada con 2 bits, a la señal “sin cuantizar” se puede estimar el error de cuantización eq[n] :



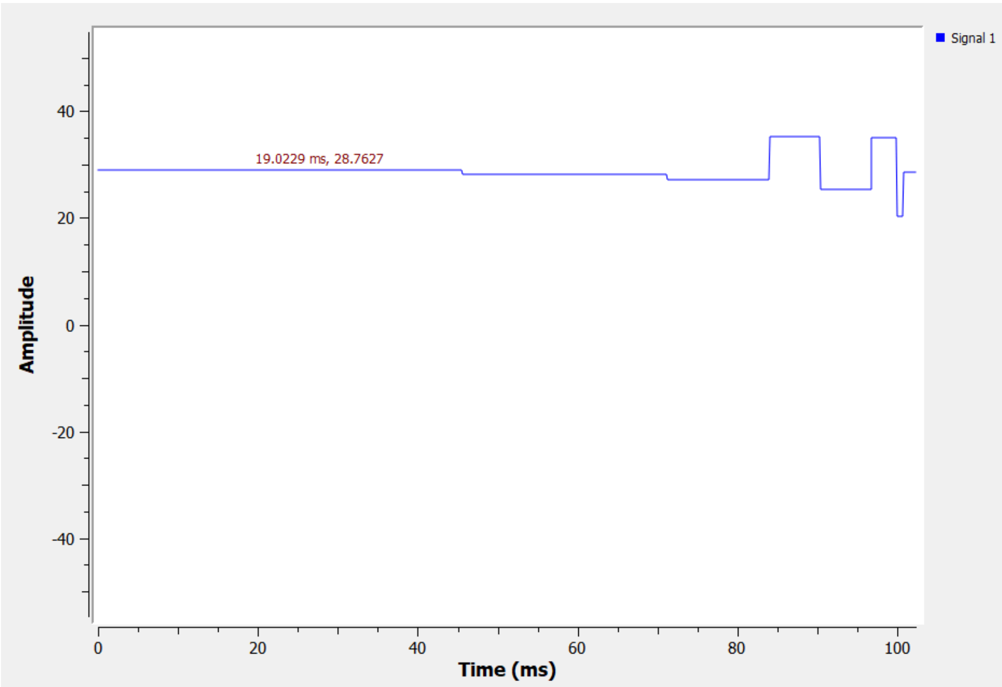
f. $P_x = \frac{A^2}{2} = 0,5$



$$P_q = \frac{A^2/3}{2^{2b}}$$

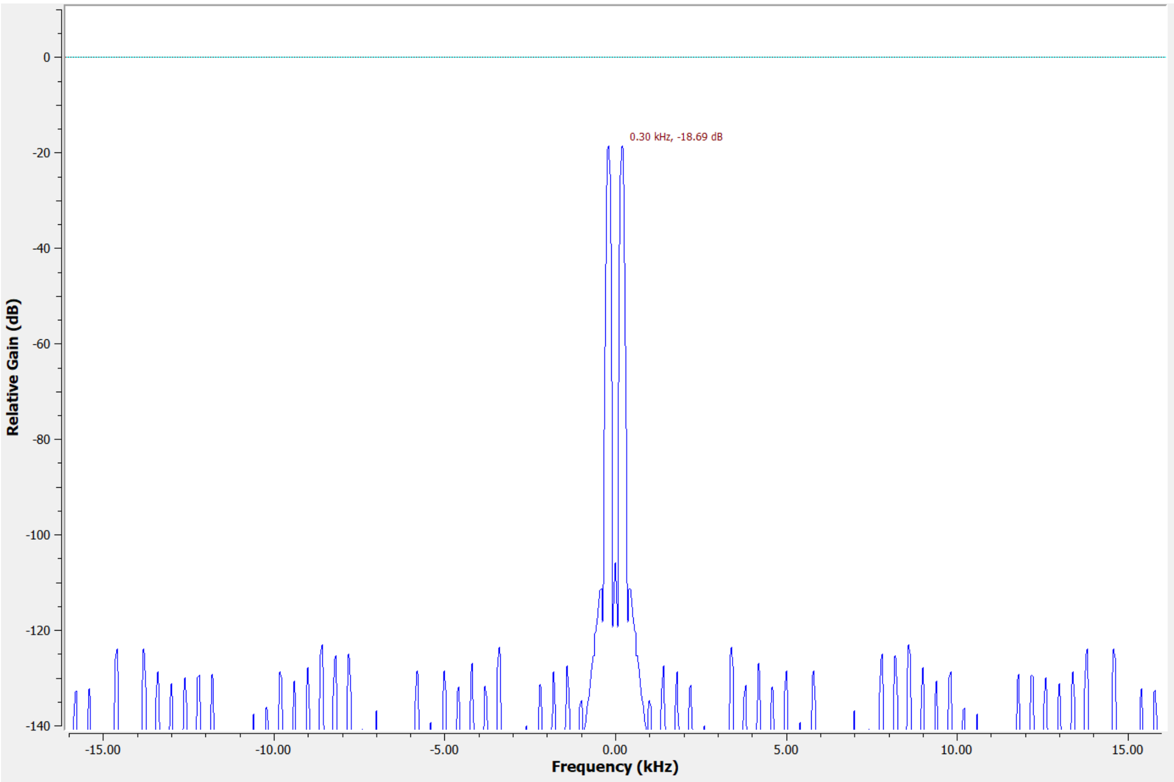


$SNRQ = \frac{Px}{P_q} = 24 \rightarrow$ el obtenido es de 28.7 esto puede deberse a usar variables float.

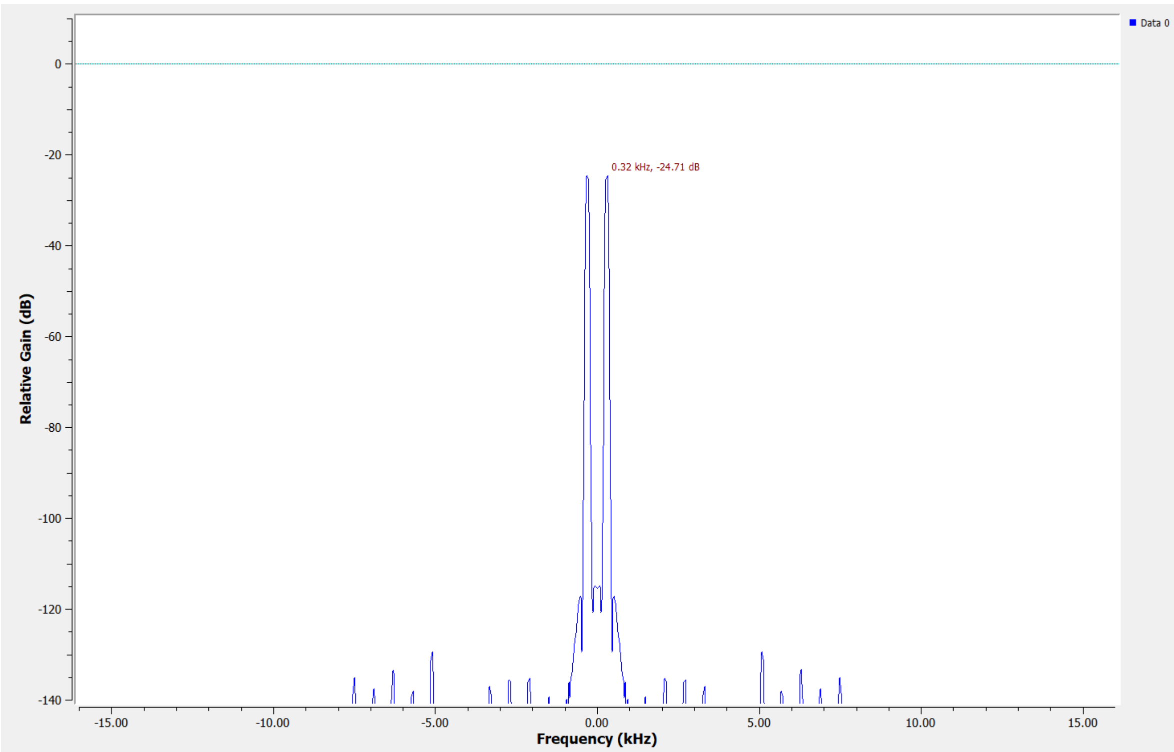


11)

a. tono reproducido en compañion

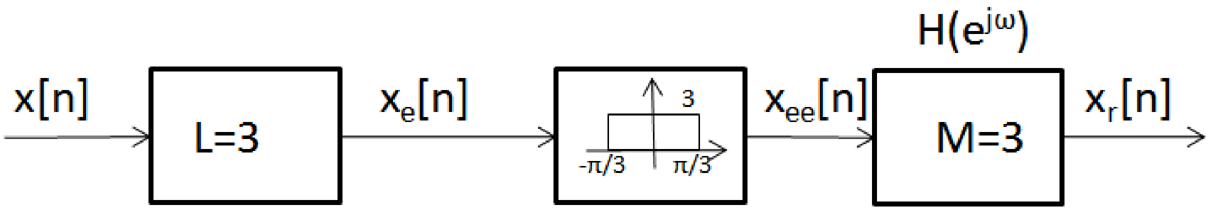


b. Diezmado e Interpolado

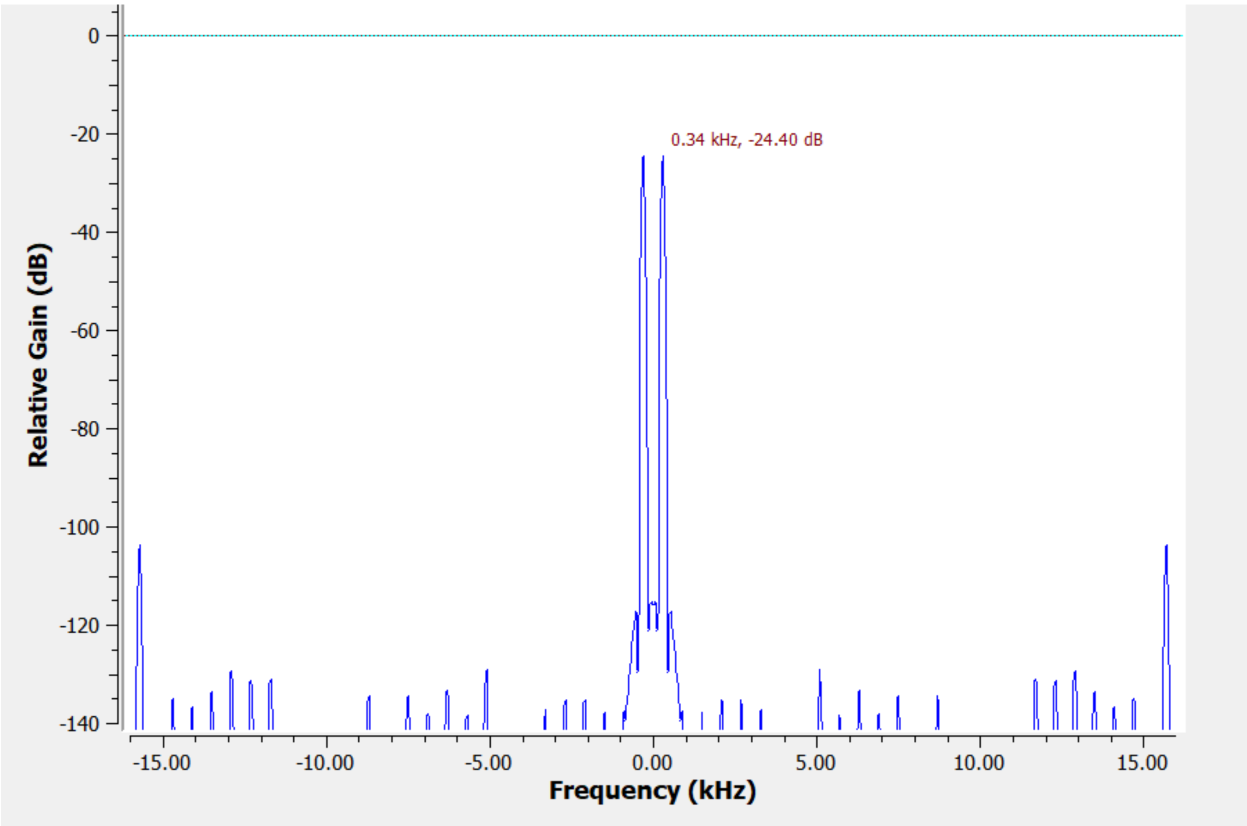


Usando fir interpolation y fir decimation se obtuvo la señal de la imagen, el tono está en 0.32khz.

c. Primero debe ir la interpolación y luego el diezmado para reducir el riesgo de provocar aliasing o pérdida de información. Con L agregamos ceros en el tiempo discreto agregando información en frecuencia alejando las réplicas de los límites de π con el filtro H eliminamos las réplicas y con el M diezmamos para obtener la señal original con su ω_0.



d.



Se observa el tono en 3.4khz

Al reproducir el audio en audio sink luego de usar el rational resampler suena más agudo (aunque la diferencia en frecuencia es mínima y no debería ser notorio) pero de manera continua, mientras que cuando lo reproduje usando los bloques de interpolación y diezmado lo escucho con cortes entre medio.

12)

a. bitstream = frecuencia de muestreo x n° de canales x bits por muestra

fs = 1.2Mhz ; canales= 2 (I y Q) ; bits por muestra = 8 (por los conversores ADC de los sdr)

bitstream= 1.2M muestras/segundos *2 canales* 8 bits /muestra

bitstream= 19.2mbps

b. En el inciso anterior vimos que los sdr estándar usan 8 bits por muestra. Por ende los niveles son L=2^8= 256 niveles

c.

Consultando:
https://www.rtl-sdr.com/wp-content/uploads/2018/02/RTL-SDR-Blog-V3-Datasheet.pdf

Para la frecuencia de muestreo máxima teórica y estable para la mayoría de los dongles RTL-SDR basados en el chip RTL2832U, la datasheet indica que el bandwidth estable es hasta 2.4 MHz. Ya que fs debe ser menor o igual al ancho de banda.