El repositorio es: https://github.com/GonzalezL0310/gnu-radio-sys-2.git

Ejercicio 1

Dos señales analógicas x1(t) y x2(t) son como siguen:

$$x_1(t) = \cos(10 \cdot 2\pi t)$$

$$x_2(t) = \cos(50 \cdot 2\pi t)$$

Las dos señales son muestreadas a una tasa de muestreo fs=200 Hertz.

a) Encuentre las secuencias x1[n] y x2[n] de manera teórica.

Cuando una señal $x(t)$se muestrea con período $T_s = \frac{1}{f_s}$, se obtiene:

$$x[n] = x(nT_s)$$

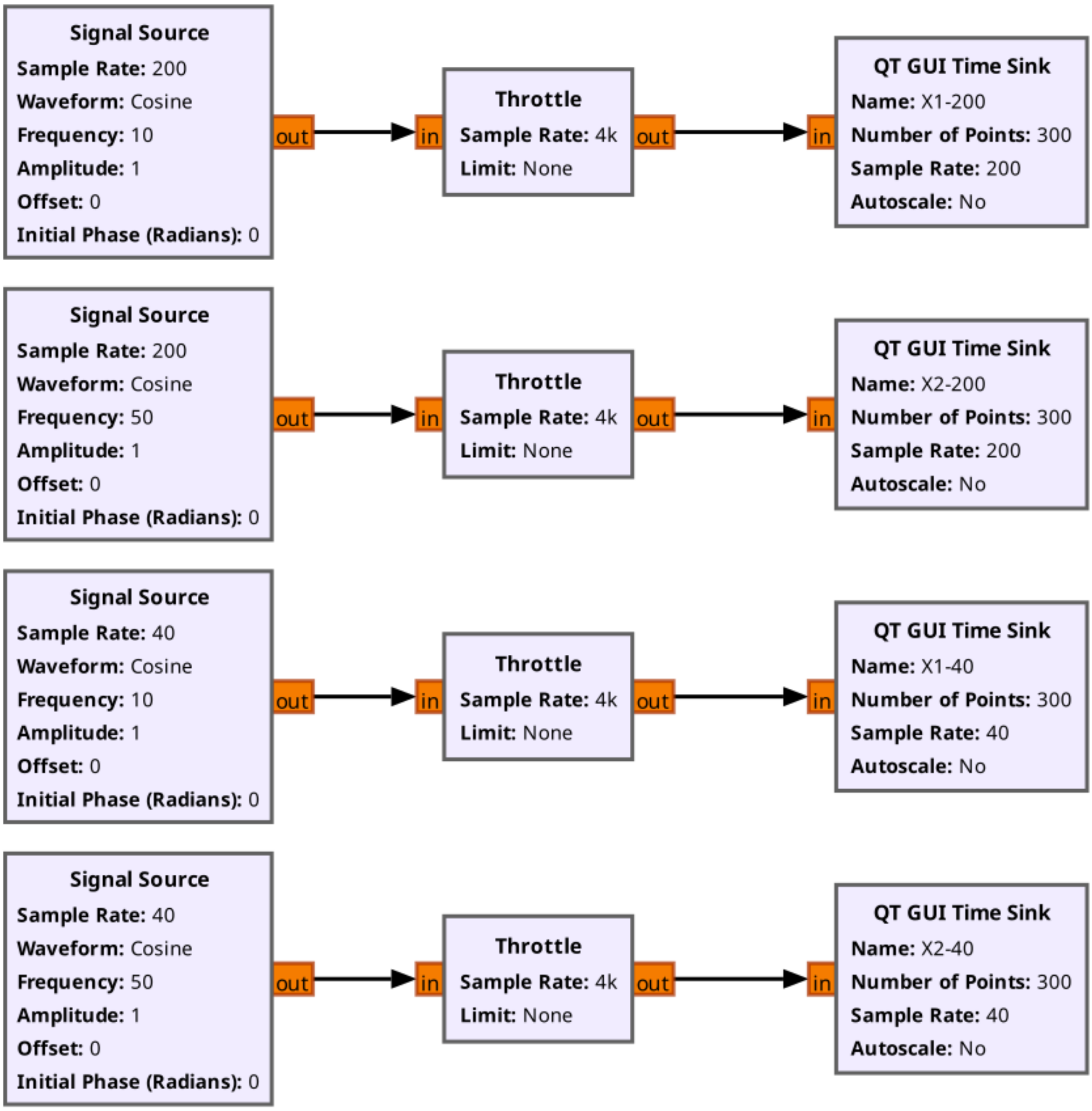
Para $x_1(t)$:

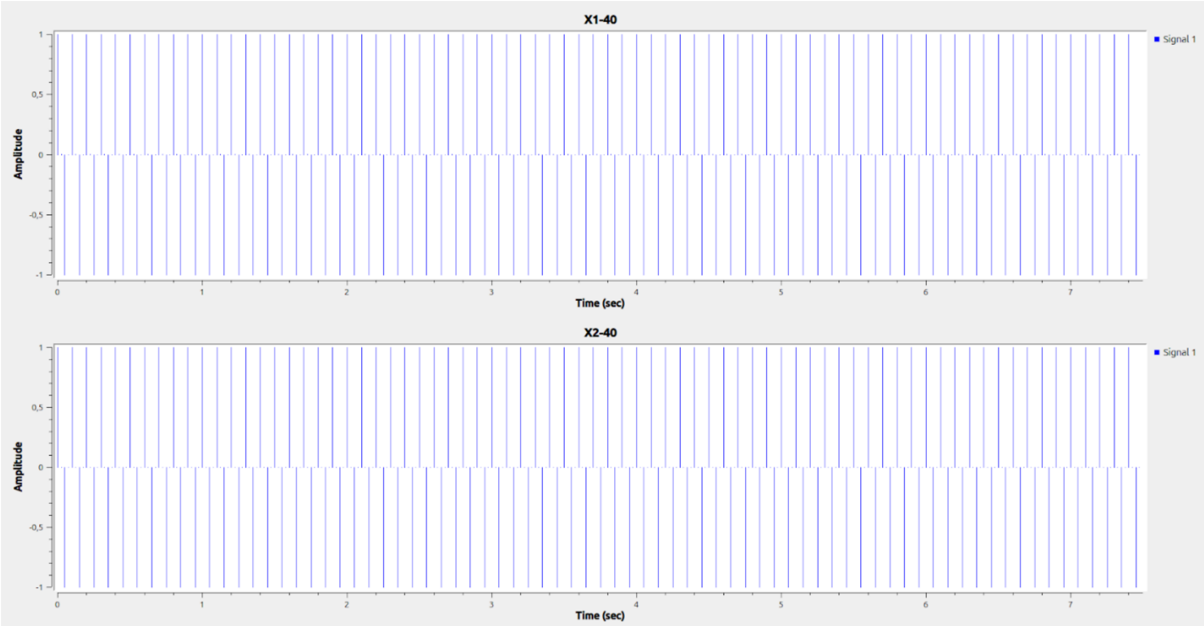
$$x_1[n] = \cos(10 \cdot 2\pi \cdot nT_s)$$

Con $T_s = \frac{1}{200}$:

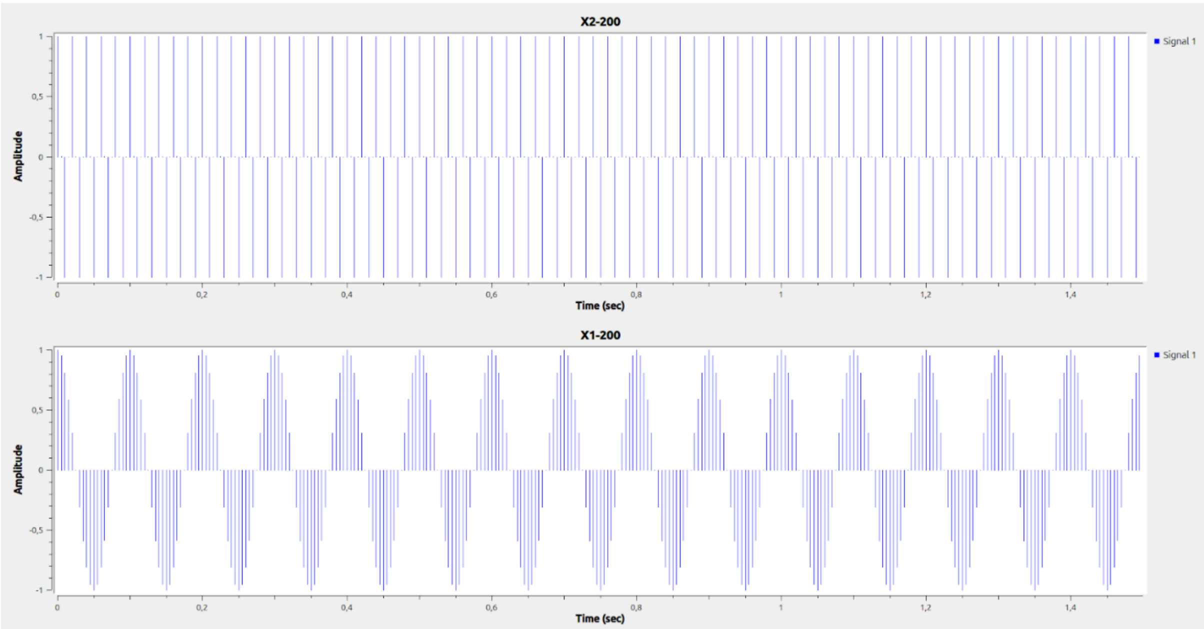
$$x_1[n] = \cos(2\pi \cdot 0.05\, n)$$
$$x_2[n] = \cos(2\pi \cdot 0.25\, n)$$

b) b. Implemente usando el Companion, y visualice las señales. Utilice Stem plot, para ver los puntos muestreados en el bloque QT GUI Time Sink. Recuerde usar el bloque Throttle. Nota: Configure el Number of Points en QT GUI Time Sink en 300 muestras. Para este ejercicio particular, configure el bloque Throttle en Sample Rate=4000.





Ahora cambie la tasa de muestreo fs, a 40 muestras por segundo (40 Hertz). Vuelva a graficar cada señal.



c) Explique el fenómeno de aliasing que se observa en las gráficas del inciso b.

Al muestrear, las componentes de frecuencia continua se “pliegan” dentro del intervalo $[0, f_s/2]$(espectro de bandas base) porque la respuesta en tiempo discreto es periódica en frecuencia con periodo f_s. Si existe energía por encima de $f_s/2$, esa energía aparece como una frecuencia distinta dentro de la banda base: eso es *aliasing*.

d) ¿Qué otras señales producen un alias de la señal x1[n] cuando fs=40 Hertz? Muestre un ejemplo en GNURADIO Companion.

e) General: dos cosenos son idénticos en las muestras si sus frecuencias verifican, en Hz:

$$f = \pm f_1 + k f_s, k \in \mathbb{Z}$$

Aplicado a $f_1 = 10\text{Hz}$ y $f_s = 40\text{Hz}$:

$$f = \pm 10 + 40k, k \in \mathbb{Z}$$

Eso genera la familia: $\{\dots, -70, -30, -10, 10, 30, 50, 70, 90, \dots\}$. Cualquiera de esas frecuencias produce la misma secuencia discreta $x_1[n]$cuando se muestrea a 40 Hz.

En dominio discreto la frecuencia angular $\omega = 2\pi f / f_s$está definida módulo 2π. Y $\cos(\omega n) = \cos((-\omega + 2\pi k)n) \rightarrow$ la condición algebraica que llevó a la fórmula anterior.

Ejercicio 3

Una señal de tiempo continuo x(t) tiene la forma de:

$$x(t) = 5cos(2\pi 5000t)$$

Considere inicialmente una frecuencia de muestreo tal que permita una correcta reconstrucción de la señal. Utilice el bloque QT GUI Range del Companion para modificar la misma.

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?

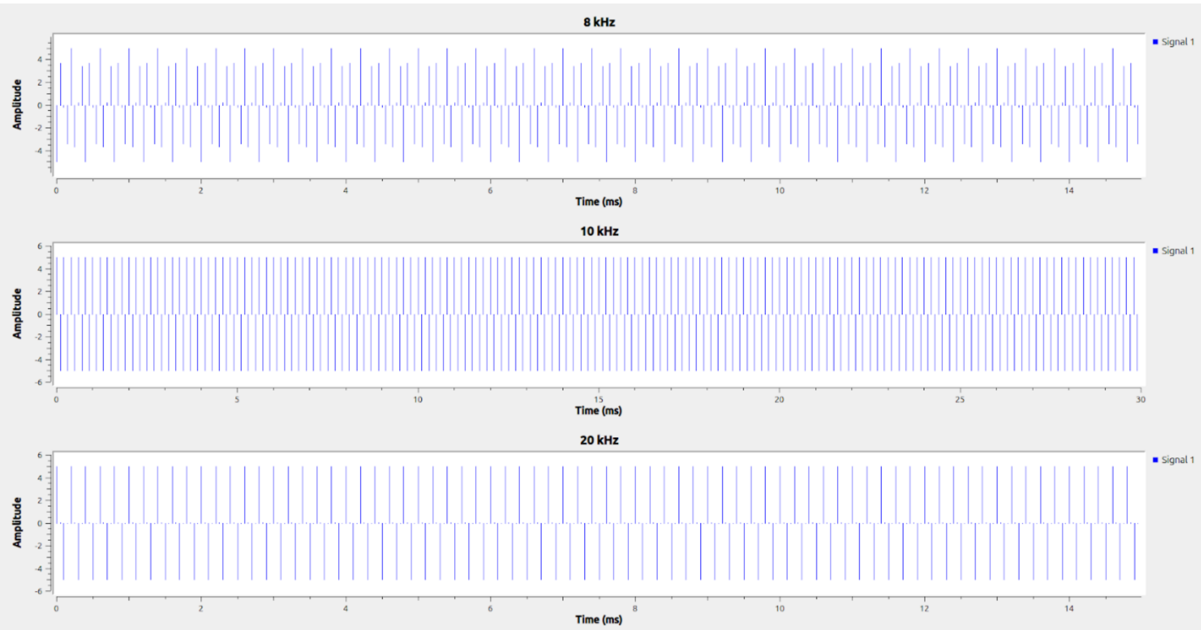
Fs < 10000 (menor que Nyquist) — aliasing: la señal se pliega y la frecuencia observada en las muestras es distinta.

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

Fs = 10000 **(igual a Nyquist)** — la señal queda en la frontera: la frecuencia discreta es $\omega = \pi \rightarrow x[n] = 5\cos(\pi n) = 5(-1)^n$. Se muestrea, pero está en el límite (sensible a jitter y al filtrado).

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

Fs >10000 **(mayor que Nyquist)** — correcta representación; a más f_s mejor resolución temporal, menor exigencia del filtro antialias y menos error por cuantización/aliasing residual.



Ejercicio 5

Una señal compleja puede descomponerse y/o representarse en coordenadas ortogonales (IQ, o bien, parte real y parte imaginaria) o también, en coordenadas polares (módulo o mag, y fase). Ambas representaciones contienen la misma información para describir la señal. Considere una señal compleja de tiempo continuo x(t) de la forma:

$$x(t) = e^{j2\pi 1000t}$$

Utilice una fs=64Khz.

a) Considere la representación fasorial de la señal x(t) ¿A qué velocidad angular se mueve el fasor?, ¿cuántas vueltas da el fasor en el tiempo de 1 segundo?, ¿cuánto tiempo tarda el fasor en dar una vuelta?

Velocidad angular (rad/s):

$$\omega = 2\pi f_0 = 2\pi \cdot 1000 = 2000\pi \text{ rad/s} \approx 6283{,}1853 \text{ rad/s.}$$

Velocidad angular por muestra (rad/muestra):

$$\omega_s = \frac{\omega}{f_s} = \frac{2\pi \cdot 1000}{64000} = \frac{\pi}{32} \text{ rad/muestra} \approx 0{,}09817477 \text{ rad/muestra.}$$

Vueltas por segundo: $f_0 = 1000$vueltas/segundo.
Es decir: **1000 vueltas en 1 segundo**.

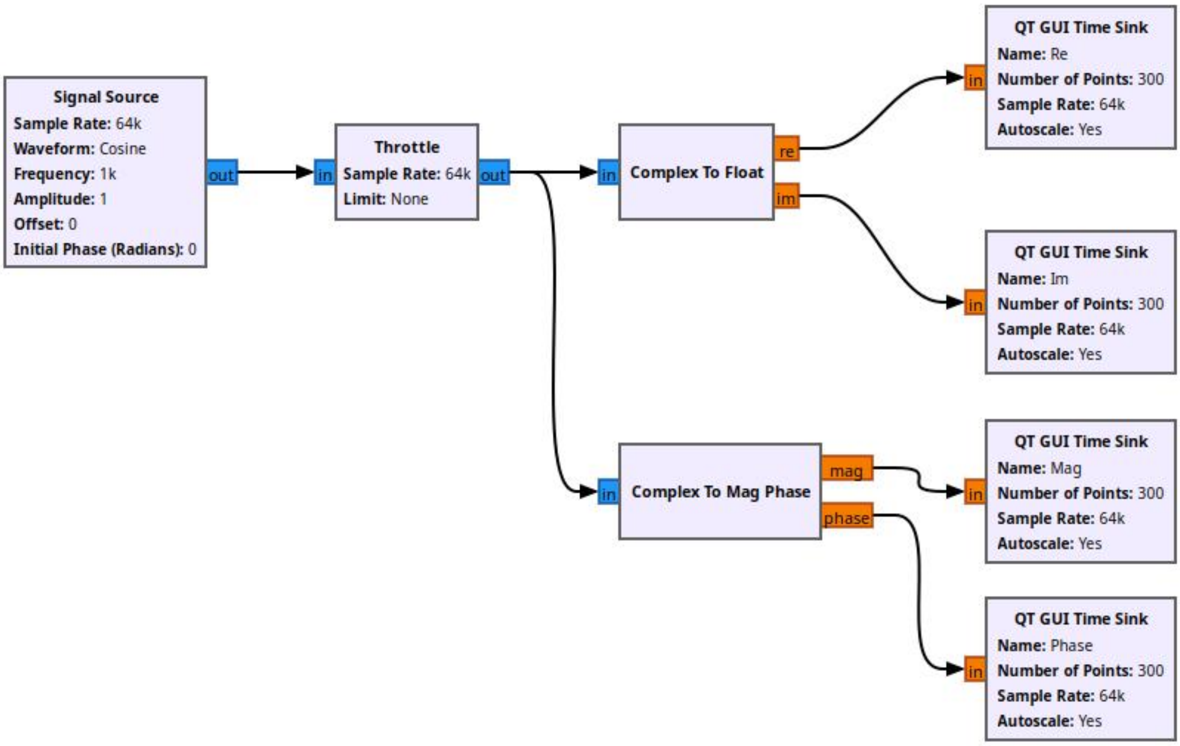
Tiempo por vuelta (periodo):

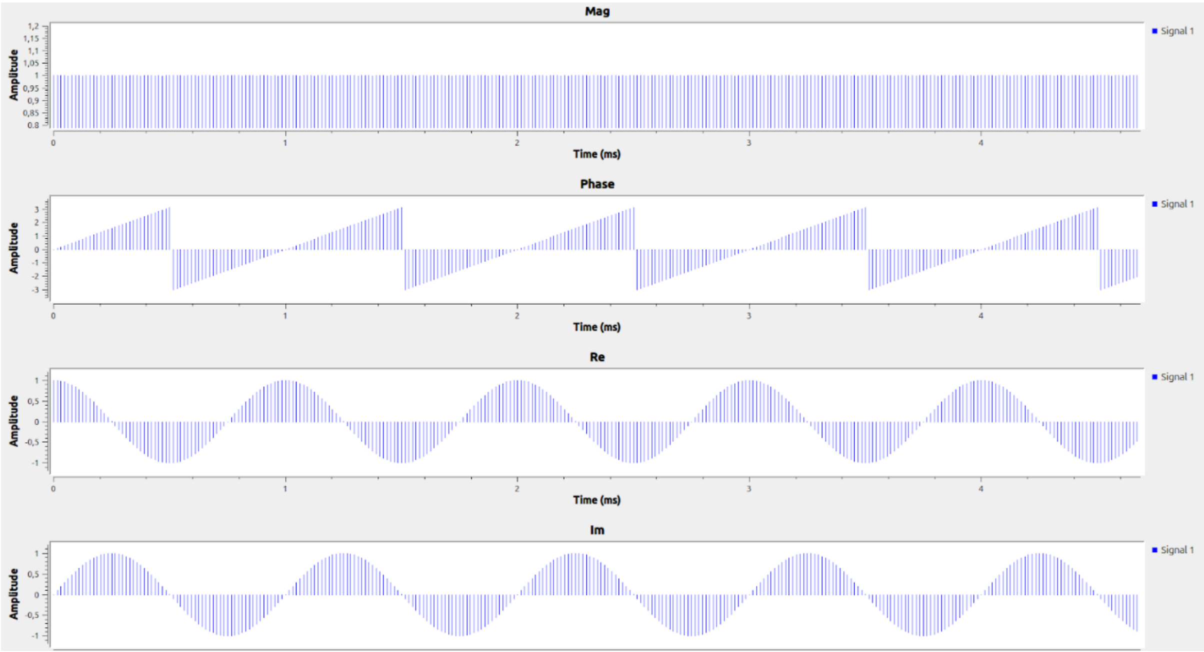
$$T = \frac{1}{f_0} = \frac{1}{1000} = 0{,}001 \text{ s} = 1 \text{ ms}$$

Resumen: el fasor gira a 2000πrad/s, da 1000 vueltas en 1 s y cada vuelta tarda 1 ms.

b) Utilice el bloque Complex to Float y grafique la parte real e imaginaria de la señal, por separado y en el tiempo. ¿Qué señal se ve en cada caso?

c) Utilice el bloque Complex to Mag Phase y grafique ambas salidas. Explique cada gráfico. ¿Cuánto vale la fase a los 10.30ms? Dibuje el fasor en este tiempo de 10.30ms. ¿Cuántas vueltas dió?





Fase en $t = 10{,}30$ ms $= 0{,}01030$ s:

$$\phi(0{,}01030) = 2\pi \cdot 1000 \cdot 0{,}01030 = 2\pi \cdot 10{,}3 = 20{,}6\pi \text{ rad.}$$

Reducida módulo 2π (fase principal):

$$20{,}6\pi mod 2\pi = 0{,}6\pi \text{ rad} = 0{,}6\pi \approx 1{,}884955592 \text{ rad.}$$

- **Valor numérico de la fase:** $\phi(10{,}30 \text{ ms}) = 0{,}6\pi$ rad $\approx 1{,}88496$ rad

Fasor en $t = 10{,}30$ ms (coordenadas cartesianas):

$$x(0{,}01030) = e^{j \cdot 0{,}6\pi} = \cos(0{,}6\pi) + j\sin(0{,}6\pi).$$

Valores:

$$\cos(0{,}6\pi) \approx -0{,}30901699, \sin(0{,}6\pi) \approx 0{,}95105652.$$

Entonces el fasor apunta a $(-0{,}3090,\ 0{,}9511)$ en el plano IQ.

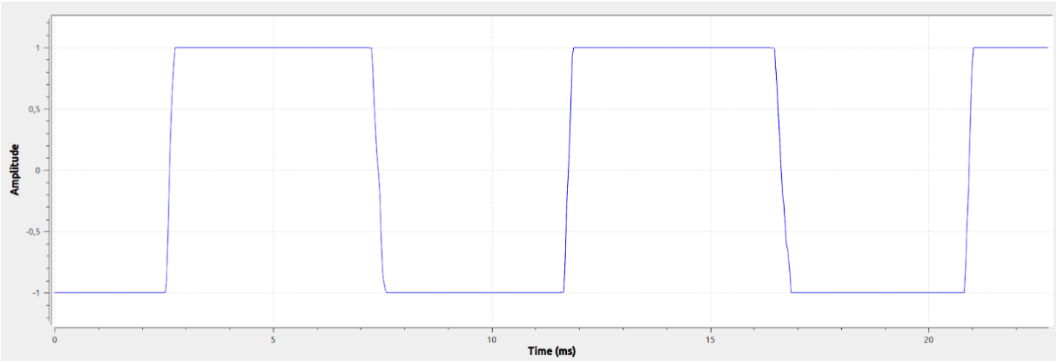
¿Cuántas vueltas dió en 10,30 ms?
Número de vueltas $= f_0 \cdot t = 1000 \cdot 0{,}01030 = 10{,}3$vueltas.
Es decir: **10 vueltas completas + 0.3 de la siguiente**.

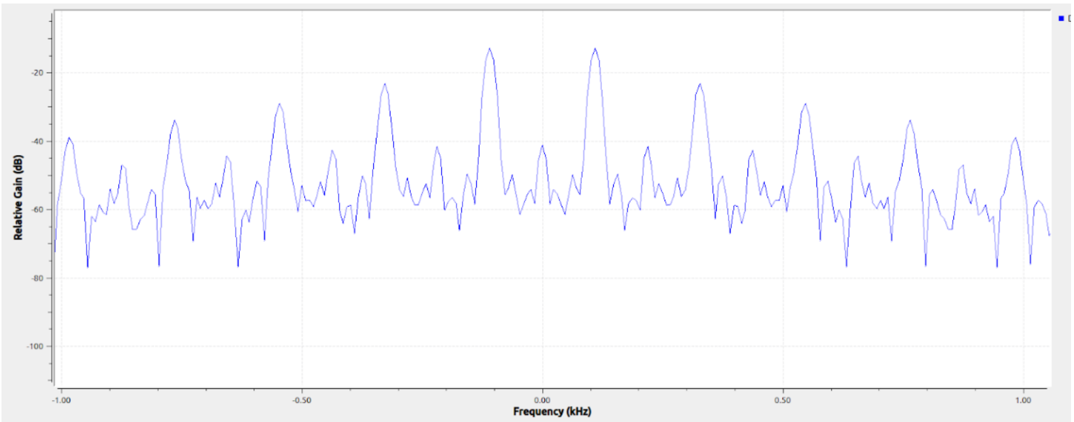
Ejercicio 7

Se ha grabado el tono (el sonido del tono) que produce una cuerda de guitarra tocada al aire, afinada en el estándar A4=440Hz, a una tasa de 32Khz, en el archivo guitar.wav.

a) Usando Companion: ¿Qué forma de onda tiene el tono de la cuerda de guitarra grabada?, ¿qué frecuencia ha identificado?

La frecuencia identificada es de 110 Hz





b) ¿A qué nota musical de la guitarra corresponde este tono? Una nota en una guitarra, se produce cuando tocamos una sola cuerda a la vez.

La nota es un A2 o La 2.

Ejercicio 9

a) Responda:
   a. Esta forma de representar una muestra: ¿Introduce algún error de cuantización?

Sí, el formato **float32 introduce un pequeño error de cuantización**, porque sólo puede representar un número finito de valores dentro del rango continuo real.

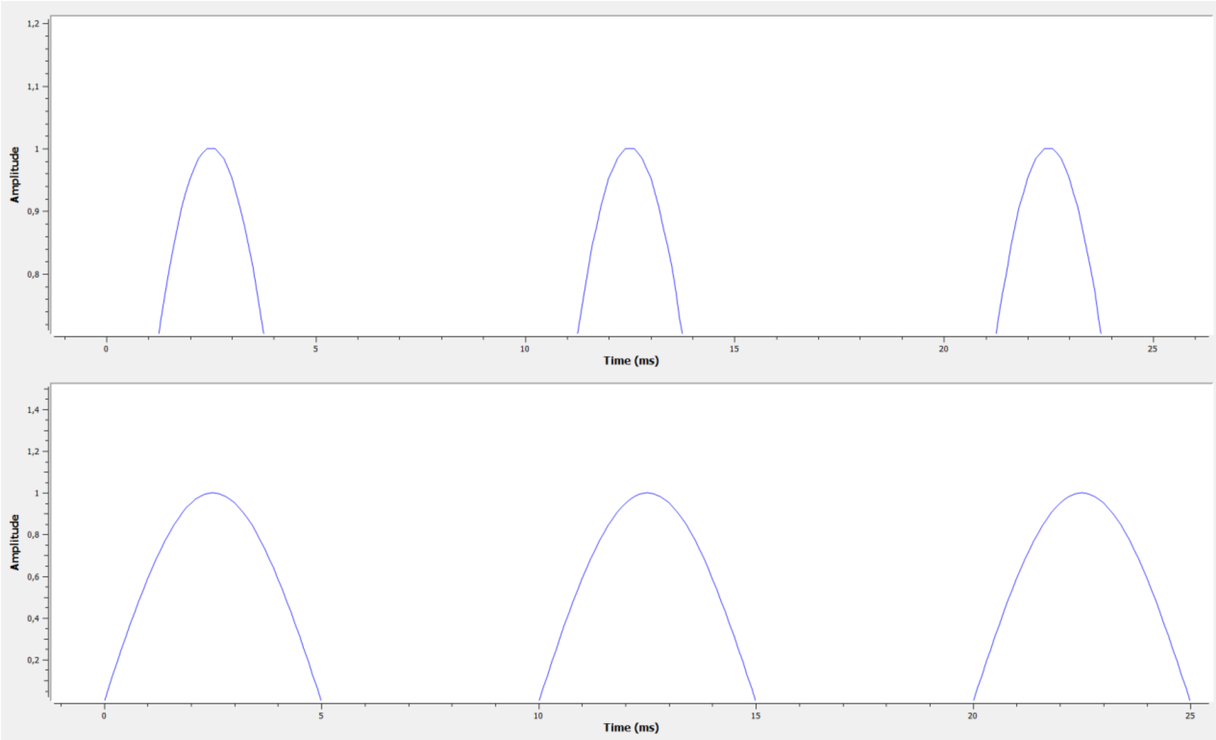
   b. ¿Hay alguna forma de evitar el error de cuantización cuando trabajamos señales digitales?

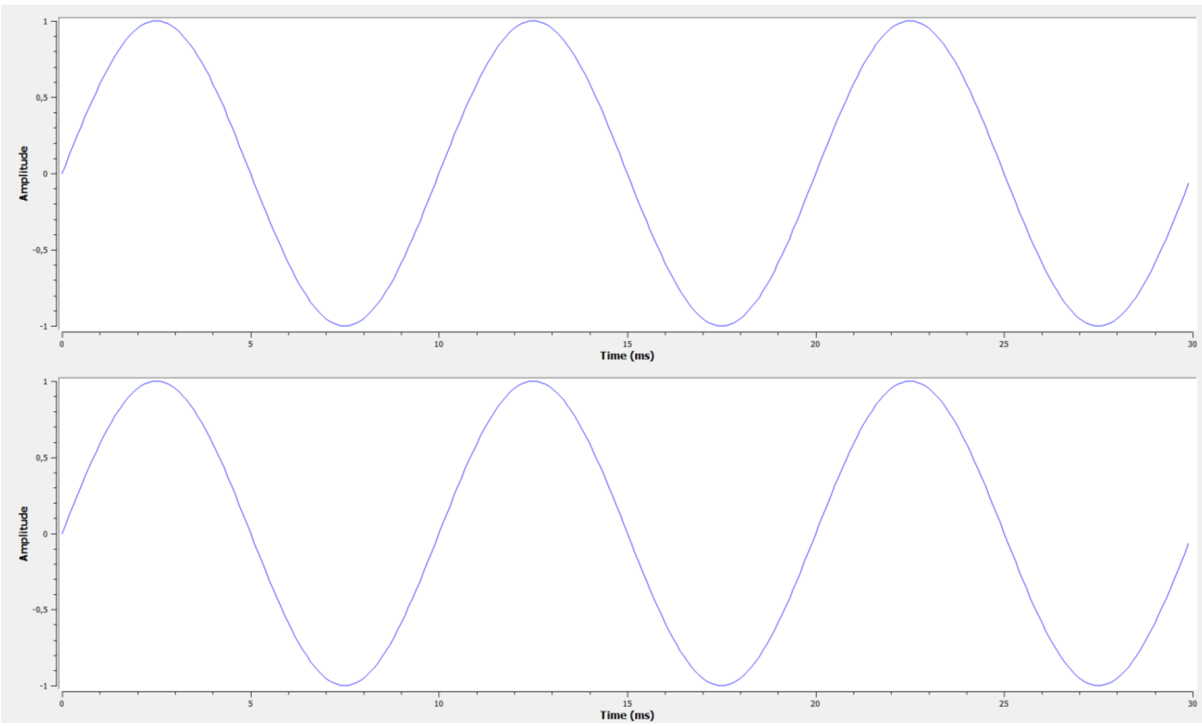
No se puede eliminar completamente el error de cuantización, pero puede **reducirse** usando mayor precisión (por ejemplo, **float64** o **doble precisión**) o manteniendo la señal en el dominio **analógico** antes de cuantizarla.

b) Use el bloque Quantizer, para cuantizar señales. Cree, en Companion, una señal tipo seno, de 100 Hertz de frecuencia, salida Float, y con frecuencia de muestreo de 10 KHz, y amplitud 1. Compare gráficamente la señal original con la misma señal, pero cuantizada a 12 bits. Nota: No olvide usar el bloque Throttle.
   a. ¿Observa diferencias entre ambas señales?

Sí, al comparar la señal original (float) con la señal cuantizada a **12 bits**, se observan pequeñas diferencias, especialmente si se hace zoom en la gráfica.
La **cuantizada** ya no es una curva perfectamente suave: presenta pequeños "saltos" o "escalones" debido a que cada muestra se redondea al nivel más cercano permitido por los 12 bits.

Sin embargo, como se usan **12 bits**, el error de cuantización es muy pequeño, y **a simple vista ambas señales parecen casi idénticas**. La diferencia se vuelve visible solo si se amplía mucho la escala o se resta una señal de la otra.


Amplitude
1,2
1,1
1
0,9
0,8
0
5
10
15
20
25
Time (ms)
Amplitude
1,4
1,2
1
0,8
0,6
0,4
0,2
0
5
10
15
20
25
Time (ms)


Amplitude
1
0,5
0
-0,5
-1
0
5
10
15
20
25
30
Time (ms)
Amplitude
1
0,5
0
-0,5
-1
0
5
10
15
20
25
30
Time (ms)

b. ¿Cuántos niveles de cuantización son 12 bits?

El número de niveles de cuantización se calcula como:

$$N = 2^n$$

donde $n$es el número de bits.
Entonces, para **12 bits**:

$$N = 2^{12} = 4096 \text{ niveles}$$

c) Ahora proponga una cuantización de 4 niveles. ¿Cuántos bits se necesitan para lograr 4 niveles? Compare gráficamente la señal sin cuantizar con la cuantizada, en Companion.

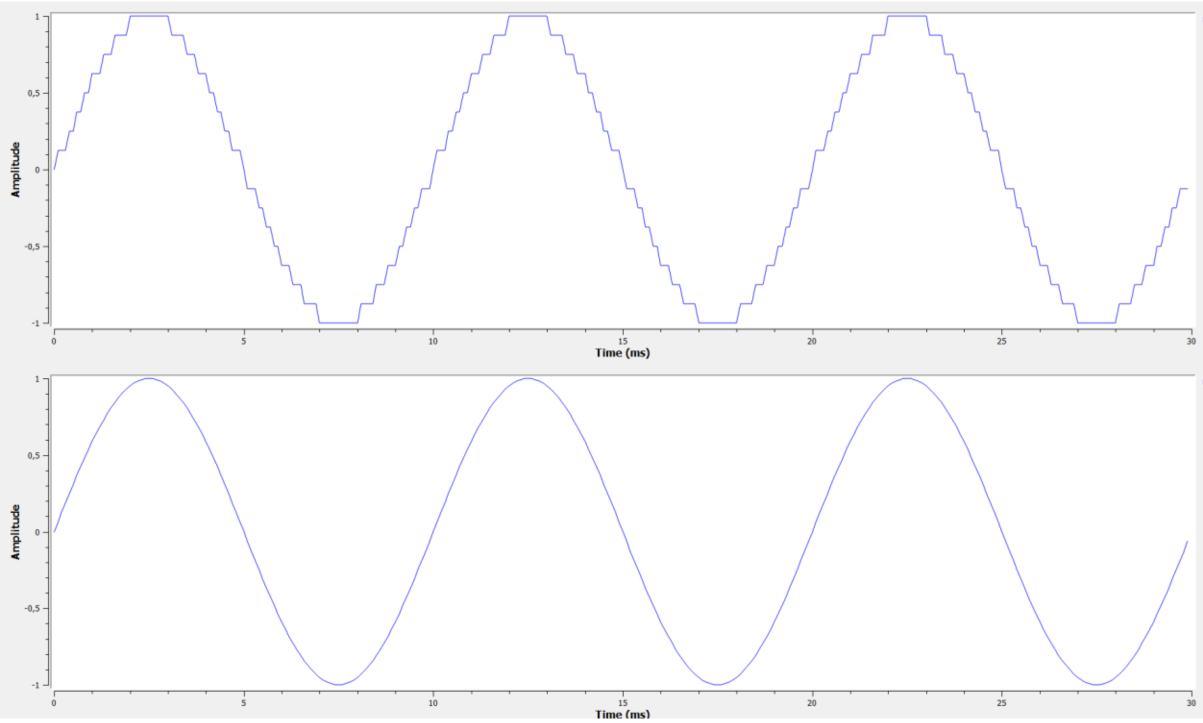
Para determinar cuántos bits se necesitan para representar una cantidad dada de niveles de cuantización, se usa:

$$N = 2^n$$

Queremos **4 niveles**, así que:

$$4 = 2^n \Rightarrow n = \log_2(4) = 2$$

Por lo tanto, **se necesitan 2 bits** para lograr 4 niveles.



d) Calcule de manera teórica la relación señal-ruido de cuantización (SQNR) para señales sinusoidales que usan 4 niveles de cuantización. También calcule el SQNR en dB.
   a. ¿Cuánto aumenta el SQNR por cada bit que se agregue en el cuantizador?, ¿por qué?

Señal sinusoidal: potencia

$$P_s = \frac{A^2}{2}.$$

Ruido de cuantización (modelo uniforme): potencia

$$P_q = \frac{\Delta^2}{12}.$$

SNR (lineal):

$$\text{SQNR} = \frac{P_s}{P_q} = \frac{A^2/2}{\Delta^2/12} = \frac{6A^2}{\Delta^2}.$$

Sustituyendo $\Delta = \frac{2A}{2^b}$:

$$\Delta^2 = A^2 \cdot 2^{2-2b},$$
$$\text{SQNR} = \frac{6A^2}{A^2\ 2^{2-2b}} = 6 \cdot 2^{2b-2} = \frac{3}{2}\ 2^{2b}.$$

Para $b = 2$(4 niveles):

$$\text{SQNR}_{\text{lin}} = \frac{3}{2}\ 2^4 = \frac{3}{2} \cdot 16 = 24.$$

En dB:

$$\text{SQNR}_{\text{dB}} = 10\log_{10}(24) \approx 13.802\ \text{dB}.$$

En forma lineal: SQNR $\propto 2^{2b}$. Añadir **1 bit** multiplica SQNRpor $2^2 = 4$(cuadruplica la potencia de la señal respecto al ruido).

En dB: multiplicar por 4 equivale a sumar $10\log_{10}(4) = 6.0206$ dB

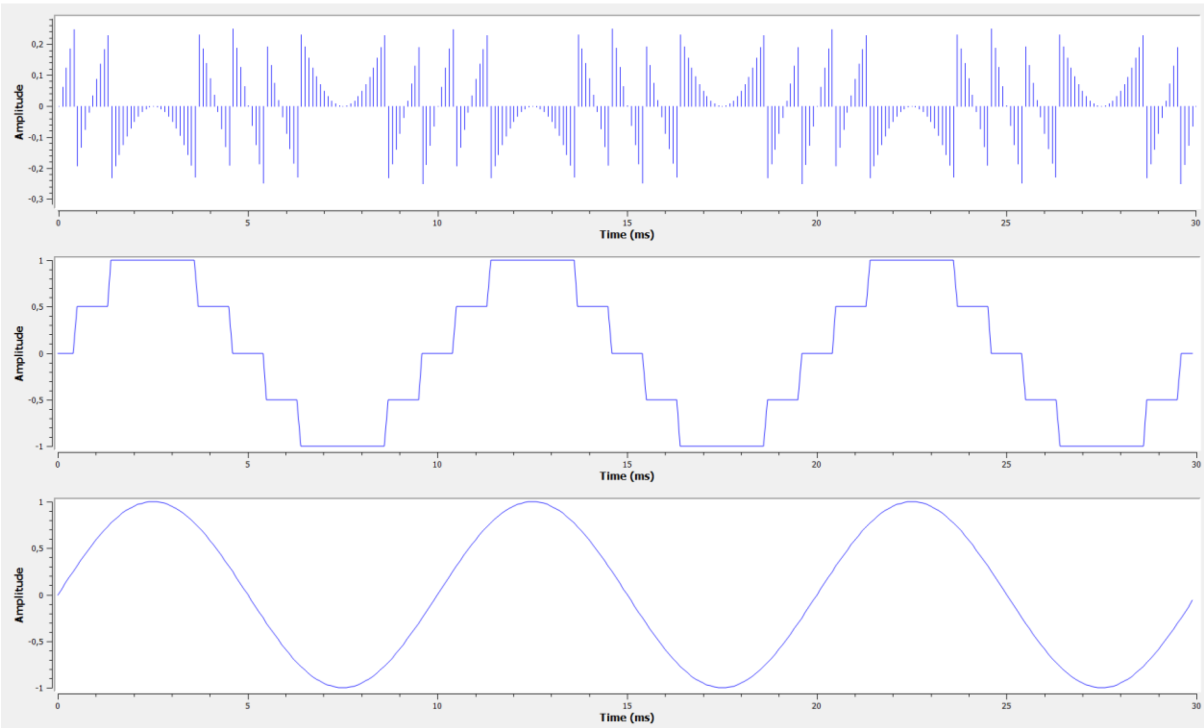
Cada bit adicional **dobla** el número de niveles L→ reduce el paso Δa la mitad → el ruido de cuantización $P_q \propto \Delta^2$se reduce por factor $1/4$. Como SQNR es P_s/P_q, SQNR aumenta por factor 4 (→ +6.02 dB).

e) Vamos a desestimar el error de cuantización introducido por usar variables Float. Considerando una cuantización de 4 niveles, estime el error de cuantificación eq[n] para una señal sinusoidal usando bloques del Companion, y grafique eq[n] usando el bloque QT GUI Time Sink.

Error de cuantización:

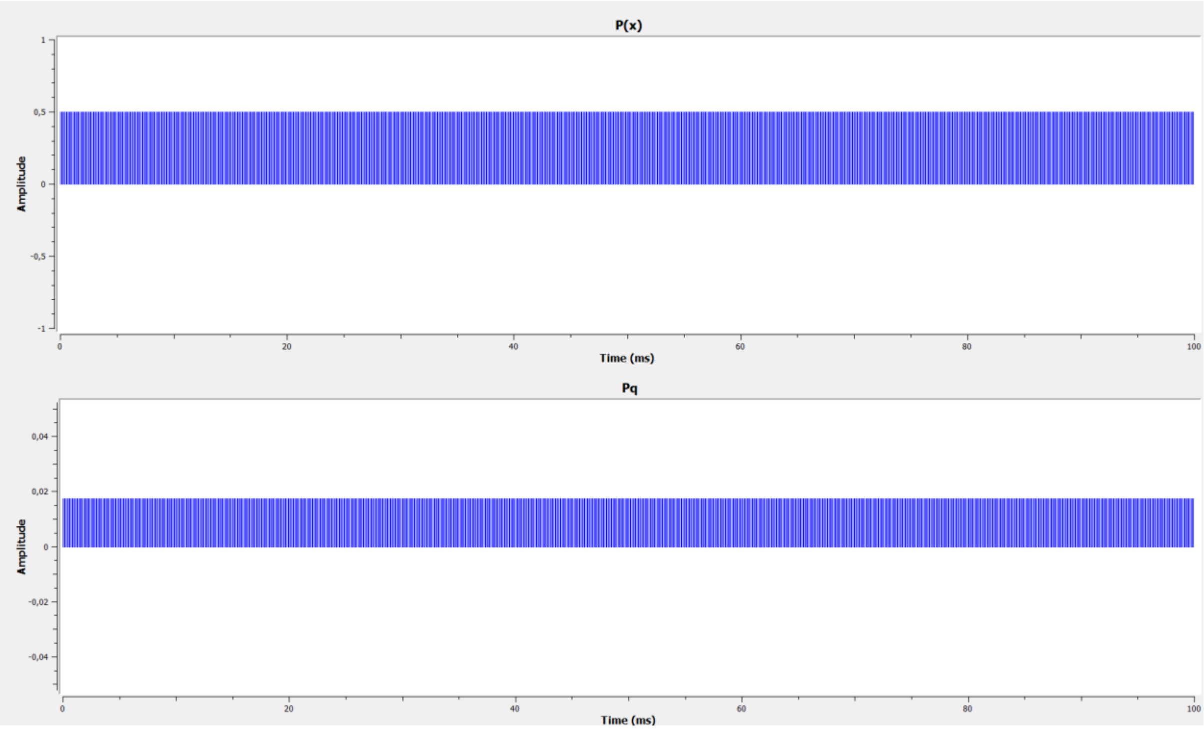
$$e_q[n] = x[n] - x_q[n].$$

- 4 niveles $\Rightarrow b = 2$bits.
- Si la señal usa todo el rango $[-A, A]$con $A = 1$, el paso del cuantizador es
$$\Delta = \frac{2A}{L} = \frac{2}{4} = 0{,}5.$$
- Error máximo (en valor absoluto) $\leq \Delta/2 = 0{,}25$. Por tanto $e_q[n] \in [-0{,}25,\ 0{,}25]$.

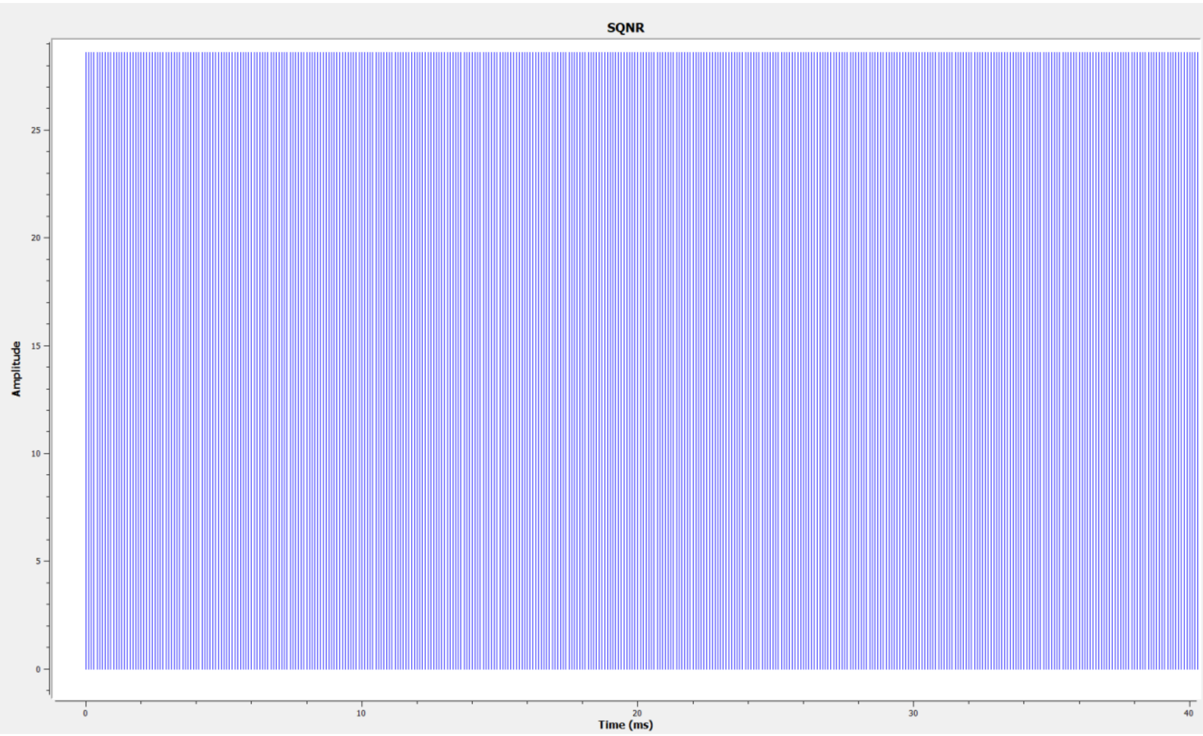


f) Utilizando el bloque Python Block.

   a. Grafique Px[n] y Pq[n] utilizando el bloque QT GUI Time Sink.



   b. Compute y grafique la relación SQNR en Companion, usando el bloque Potencia media.



c. Compare la relación SQNR que obtuvo teóricamente, con la relación SQNR que obtuvo en Companion. ¿A qué puede deberse las diferencias en los valores?

El resultado obtenido fue:

$$\mathrm{SQNR}_{\mathrm{lineal}} \approx 28.5$$

y el teórico era **24**. Las diferencias entre ambos valores pueden darse debido a:

Señal no llena todo el rango del cuantizador → reduce la potencia del error y aumenta SQNR.

Cálculo sobre un número finito de muestras → el promedio puede dar un valor ligeramente menor de error.

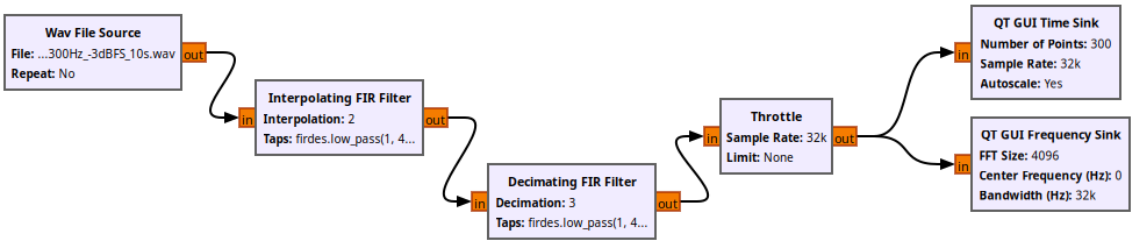
Error de cuantización no perfectamente uniforme con pocos niveles → el modelo teórico subestima la SQNR real.

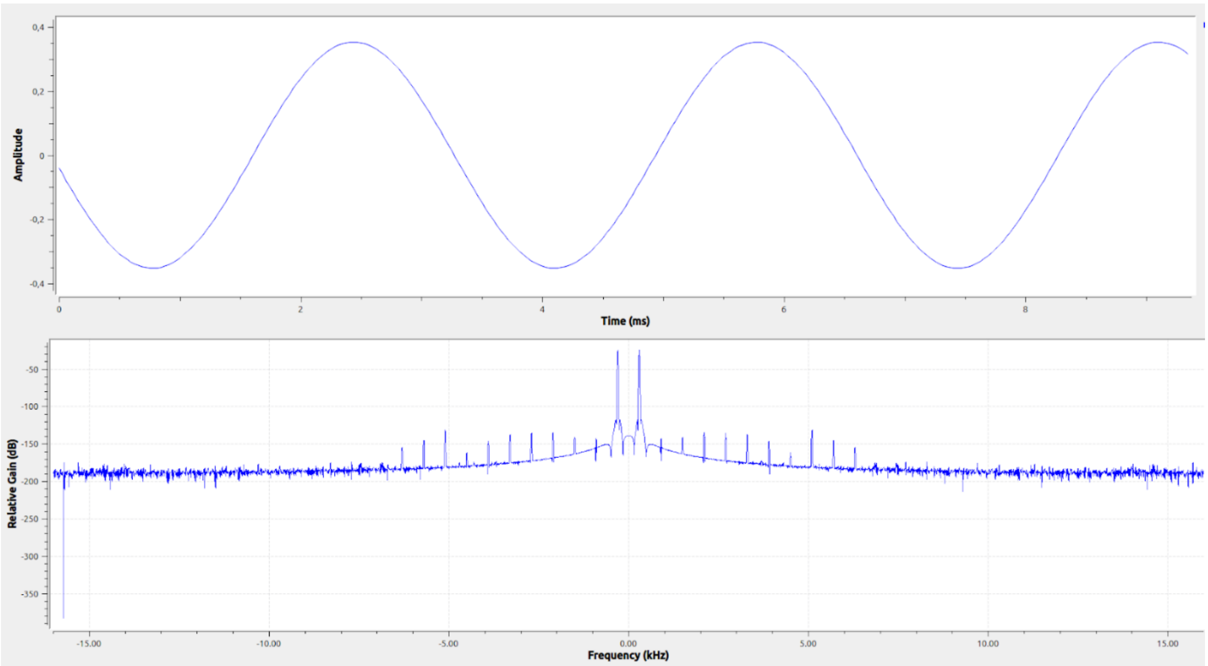
Representación float → mínima variación debido a precisión numérica.

Ejercicio 11:

Cree un tono tipo seno de 300Hz y frecuencia de muestreo de 48KHz, usando el creador de tonos online disponible en el Anexo.

a) Reproduzca la señal en el Companion.
b) Utilice una combinación de bloques de GNURADIO Companion para diezmado e interpolación tal que pueda llevar la señal a una tasa de muestreo de 32Khz. Verifique gráficamente en Companion.





c) ¿Qué bloque debe ir primero (interpolación o diezmado) y por qué?

Cuando reducís la tasa de muestreo, primero debés **aumentar la resolución temporal** para que el filtro posterior tenga suficiente información para eliminar aliasing antes de la reducción.

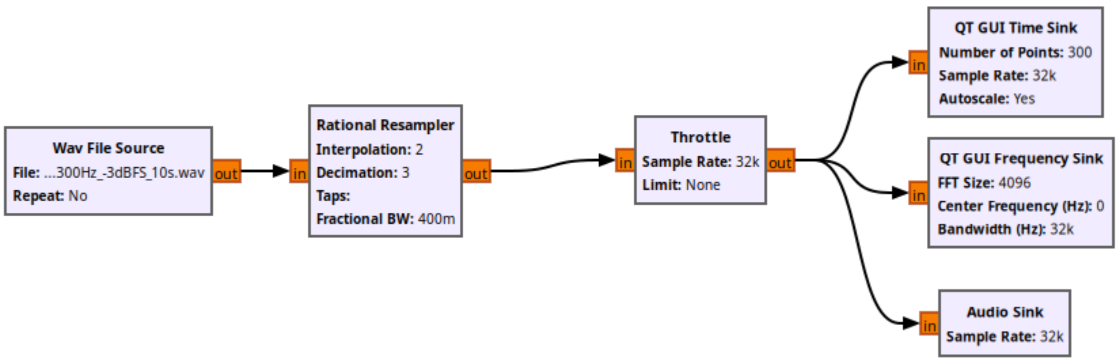
En otras palabras:

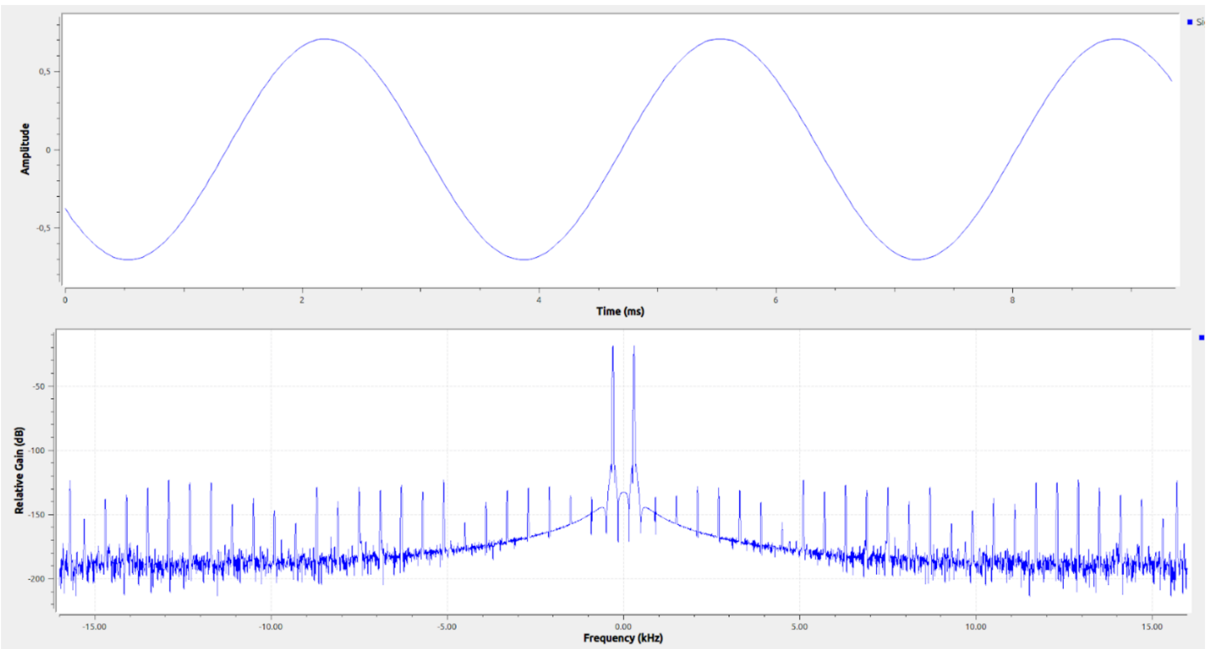
- Si diezmás primero, perderías muestras (información) antes de filtrar → aliasing.
- Si interpolás primero, aumentás la frecuencia de muestreo y el filtrado del decimador se realiza correctamente → señal limpia.

Orden correcto:

$$\text{Interpolación} \rightarrow \text{Diezmado}$$

d) Reemplace la implementación por un bloque Rational Resampler.





Ejercicio 12

Se necesitan obtener muestras de una RTL-SDR, a una frecuencia de muestreo de 1.2Mhz, para procesar cierta información. Considerando los dos canales I-Q:

a) Calcule en bitstream, o tasa de bits por segundo.

Cada muestra compleja = I (8 bits) + Q (8 bits) = **16 bits por muestra compleja**.
Con $f_s = 1{,}2 \cdot 10^6$muestras por segundo (I/Q pairs):

$$\text{bitrate} = 16 \text{ bits/muestra} \times 1{,}2 \times 10^6 \text{ muestras/s} = 19{,}200{,}000 \text{ bits/s} = 19{,}2 \text{ Mbps.}$$

b) ¿Cuántos niveles de cuantización tienen las muestras que se toman?

Si cada componente I y Q es cuantizada con **8 bits**, cada componente tiene $2^8 = 256$niveles discretos.

- Niveles por I = 256
- Niveles por Q = 256

Si preguntas "¿cuántos niveles tiene la muestra compleja en total?", se suele decir que hay 256×256combinaciones posibles por par (I,Q), pero lo correcto es expresar **256 niveles por componente**.

c) ¿Cuál es la frecuencia de muestreo máxima que soporta la SDR (consejo: investigue en la web las características de las RTL-SDR)?

Teórico / límite del hardware (RTL2832U): frecuencia máxima reportada **3.2 MS/s** (mega-muestras/s). Sin embargo, a esa tasa muchos dongles son inestables y pueden perder muestras.

Tasa estable/práctica recomendada: la mayoría de los usuarios y datasheets apuntan a **2.56 MS/s** como el ancho estable sin pérdidas; en muchos PC/USB2.0 la práctica estable ronda **~2.4–2.56 MS/s**. Por encima de eso puede funcionar en algunos equipos (USB3.0, dongles V4, buena CPU/USB), pero no es garantía.