Tarea 1:

¿Cada cuánto se procesa en término medio un bloque de muestras cuando se solicita una frecuencia de muestreo de 8000 Hz? ¿Cuál es el tiempo teórico que debería tardar?

-1

- El tiempo teórico que debería tardar es el siguiente: cada bloque que leemos tiene 2048 muestras, si la frecuencia de muestreo son 8000 quiere decir que lee 8000 muestras por cada segundo, esto implica que 2048/8000 = 0.256s , es decir 256.000 µs en promedio.
- El tiempo real:
  ```
  Tiempo bloque: 256020.6 µs (captura: 238091.7 µs; procesado: 0.0 µs; envío: 0.0 µs)
  ```
  el tiempo real promedio en procesar cada bloque fue de 256.020,6 µs, extremadamente similar que el tiempo teórico promedio.

- 2

Para 16khz de sample_rate:

T_teórico = 2048/16000 = 0,128s / 128.000 µs

T_real =

```
Tiempo bloque: 127992.2 µs (captura: 109937.0 µs; procesado: 0.0 µs; envío: 0.0 µs)
```

Tiempo real: 127.992,2 µs

Para 48khz de sample_rate:

T_teórico = 2048 / 48000 = 0,04266667s / 42.666,67 µs

T_real =

```
Tiempo bloque: 42660.96 µs (captura: 24689.05 µs; procesado: 0.0 µs; envío: 0.0 µs)
```

Tiempo real: 42.660,96 µs.

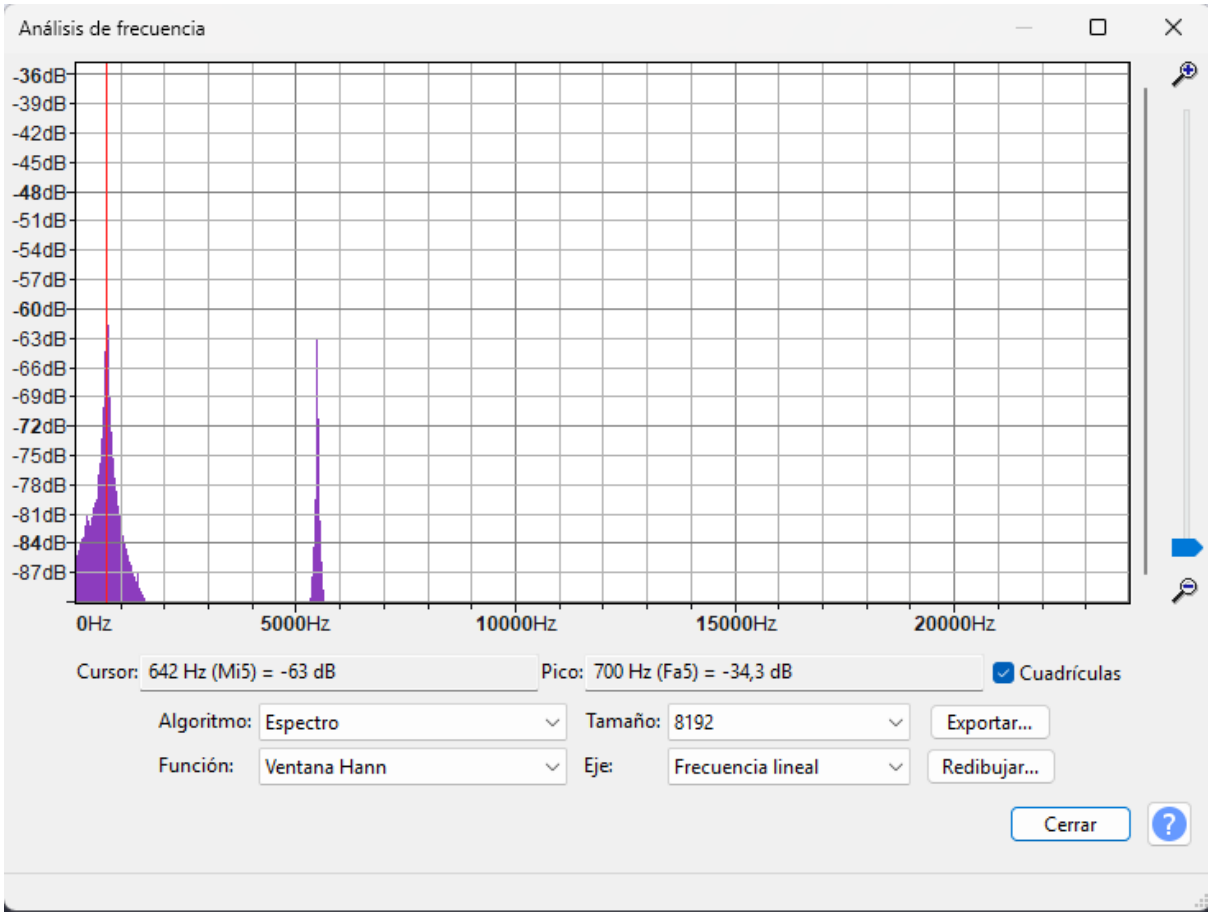
Evidentemente a mayores muestras leídas por segundos, menos será el tiempo de procesado de cada bloque, porque tardará menos en leerse cada muestra de los bloques.

Tarea 2:

-1. Captura a 8000 Hz 2 tonos, uno con frecuencia 700 Hz y otro con frecuencia 5500 Hz. Escucha el fichero generado por capture. ¿Se oyen los dos tonos?:

Se escuchan bajo, pero se escuchan perfectamente, de hecho, si vas a Audacity, analizar y trazas el espectro ves como hay enormes picos en los hz de esas señales.

-2. Abre el fichero resultante en Audacity y muestra el espectro (Analyze → Plot Spectrum...). Configura el eje x en modo lineal en el desplegable Axis → Linear frequency. ¿Qué se observa? ¿Por qué ocurre esto?

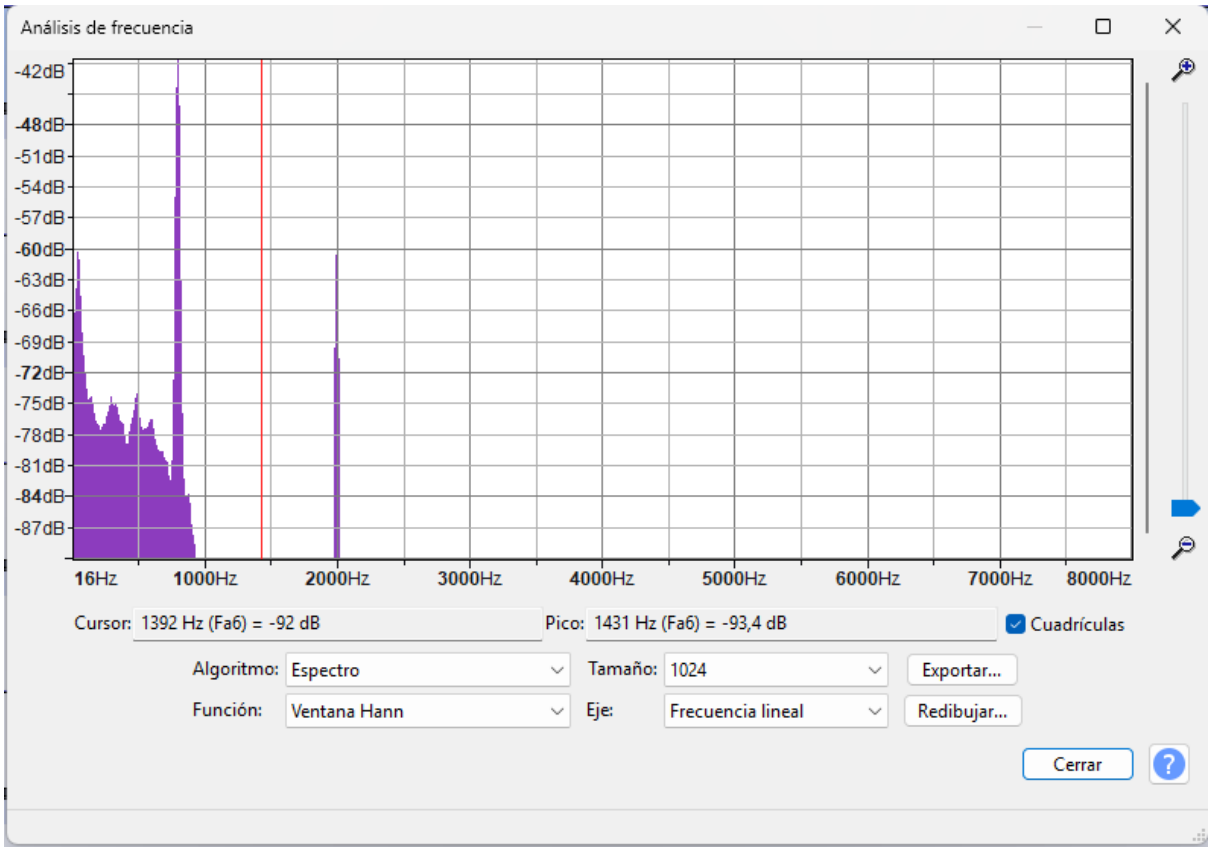


Pues como se dijo antes, se observan dos picos en las frecuencias que creamos desde

https://onlinetonegenerator.com/multiple-tone-generator.html. El gráfico anterior es bidimensional donde el eje de las abscisas corresponde a las frecuencias y el eje de las ordenadas a los dB, según chatgpt, los dB son negativos porque Audacity usa una escala logarítmica relativa donde 0 es el nivel máximo que puede tener un archivo digital.

Tarea 3

-1Diseña un filtro de L = 10 taps que elimine tonos por encima de 1000 Hz (filtro paso bajo) cuando son muestreados a 16000 Hz. Captura a esta frecuencia una señal compuesta por 3 tonos, uno de 800 Hz, otro de 2000 Hz y otro de 5000 Hz y pásala por el filtro. Muestra el espectro de la señal resultante en Audacity y explica el resultado.



Los taps del filtro son la precisión del mismo, a mayor sean los coeficientes del filtro mayor es la precisión del fitrado, con solo 10 taps el filtrado no es tan preciso y la frecuencia de 2000hz es captada, con respecto a la de 5000hz, dado el bajo volumen de los altavoces el micrófono no captó correctamente la frecuencia, pero si el volumen hubiese sido mayor, esta aparecería también, se mostraría un pequeño pico en 5000hz, tanto el pico de 2000hz como el de 5000hz son atenuados al aumentar el número de taps.

-2. Añade medidas de tiempos: t2 tras el filtrado y t3 tras el envío, y pásalas junto a t0 y t1 también a print_partial(t0, t1, t2, t3). Diseña filtros de tamaños cada vez mayores (L = 15, 20, 25, etc.). ¿Cómo cambian los tiempos? ¿Hasta qué tamaño de filtro le da tiempo a procesar los bloques sin perder muestras?

L=15:

```
Tiempo bloque: 128534.1 µs (captura: 34456.11 µs; procesado: 63195.43 µs; envío: 9284.172 µs)
```

L = 20:

```
Tiempo bloque: 128374.2 µs (captura: 23322.46 µs; procesado: 74013.12 µs; envío: 9016.771 µs)
```

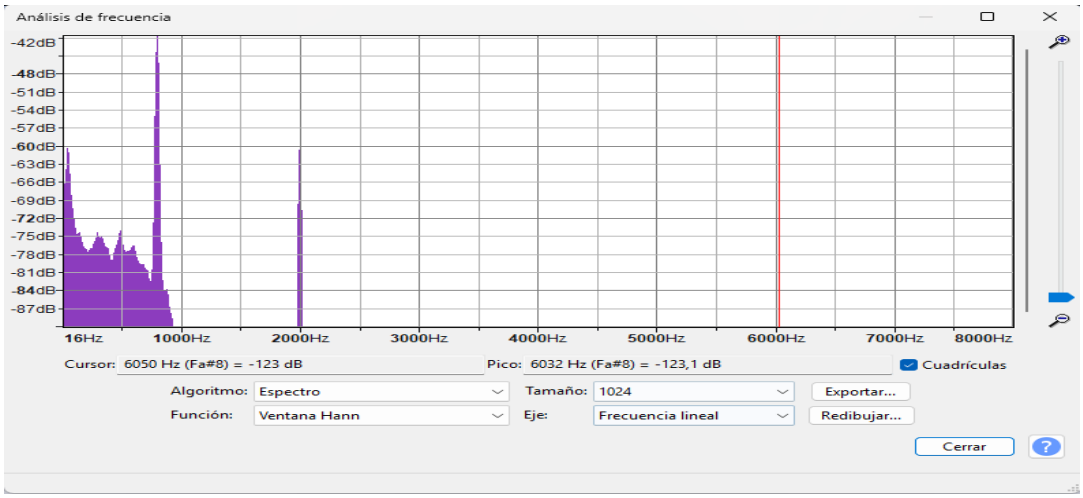
L = 25:

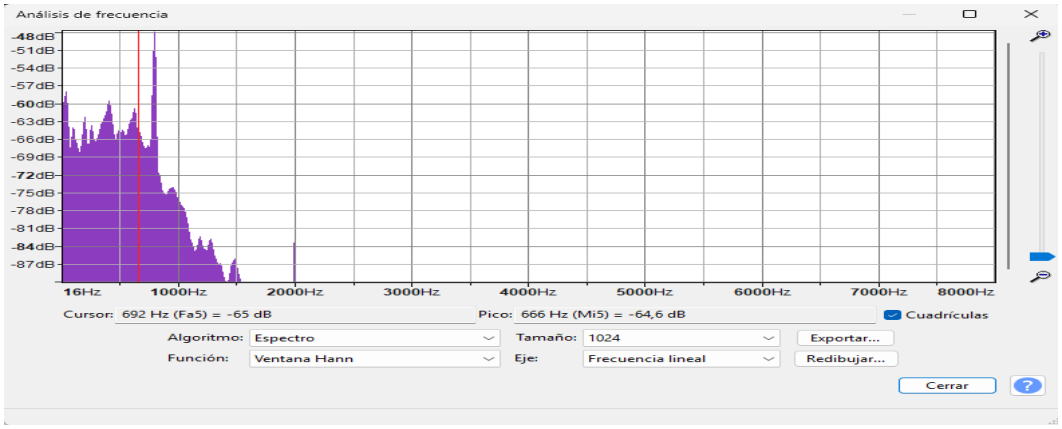
```
Tiempo bloque: 130275.8 µs (captura: 10979.2 µs; procesado: 88310.31 µs; envío: 8678.942 µs)
```

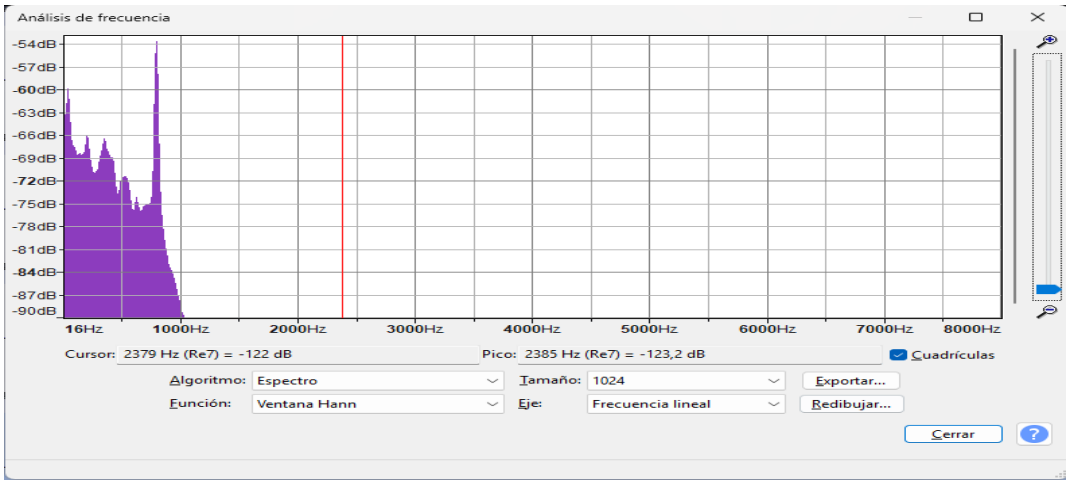
En general podemos observar que el tiempo promedio de procesado de cada bloque no cambia demasiado, sin embargo, lo que sí cambia con una tendencia ascendente directamente proporcional a L es el tiempo de procesado, tiene todo el sentido porque L son los coeficientes de la respuesta al impulso o del filtro. A mayores coeficientes tenga el filtro más tiempo tardará en desarrollarse la convolución resultando en un tiempo de procesado mayor.

--- En16khz el tiempo teórico (aunque ya sabemos que se asemeja mucho al real) que demora en escribir un bloque entero en el buffer interno son 2048/16000 0,128s / 128.000 µs. Eso quiere decir que si nuestro tiempo de bloque (el tiempo de procesado que incluye la lectura/captura en al buffer de procesamiento, el de procesado que incluye el escalado y la convolución y el de envío) es mayor, estaremos perdiendo muestras, por pocas que sean, si es considerablemente mayor en el audio que grabamos incluso escucharemos sonidos entrecortados (para esta tarea se escuchan esos sonidos aunque no se pierdan muestras por este motivo, esto debido a la convolución, el por qué se explicará en la tarea 4. Pero suponiendo que no perdemos muestras por convolución, escucharíamos sonidos entre cortados si el tiempo de procesado de bloque es considerablemente mayor que el de escritura del buffer interno). Como dice el PDF hay que buscar que el tiempo de bloque sea inferior que el de escritura teórico/real, de esta manera simplemente se quedará esperando a que esté listo el nuevo bloque en readinto antes de capturarlo en el otro buffer. Ahora bien, para la implementación desarrollada, tanto en L=15, como L=20, como L=25 se pierden muestras, por pocas que sean, para el momento en que se escribió esto, la terea 4 está finalizada, por lo que probé ejecutar con ese código y la frecuencia de muestreo a 16khz esos taps a ver si se escuchan sonidos entre cortados (grabando un sin de 800hz) y ver si en la práctica la pérdida es grave, en general, en ninguno de los tres se escucharon estos sonidos pero si usas L=50 o 40 la cosa cambia, ahí sí se escuchan. He probado con L=5 y devuelve 127878.1 µs. en promedio, así que en rigor se puede decir que con esta implementación L=5 es el máximo tamaño del filtro para no perder muestras a 16khz

-3Muestra capturas del espectro de la señal filtrada con 10 taps, 20 taps y 30 taps. ¿Qué efecto tiene en la señal de salida aumentar el tamaño del filtro?







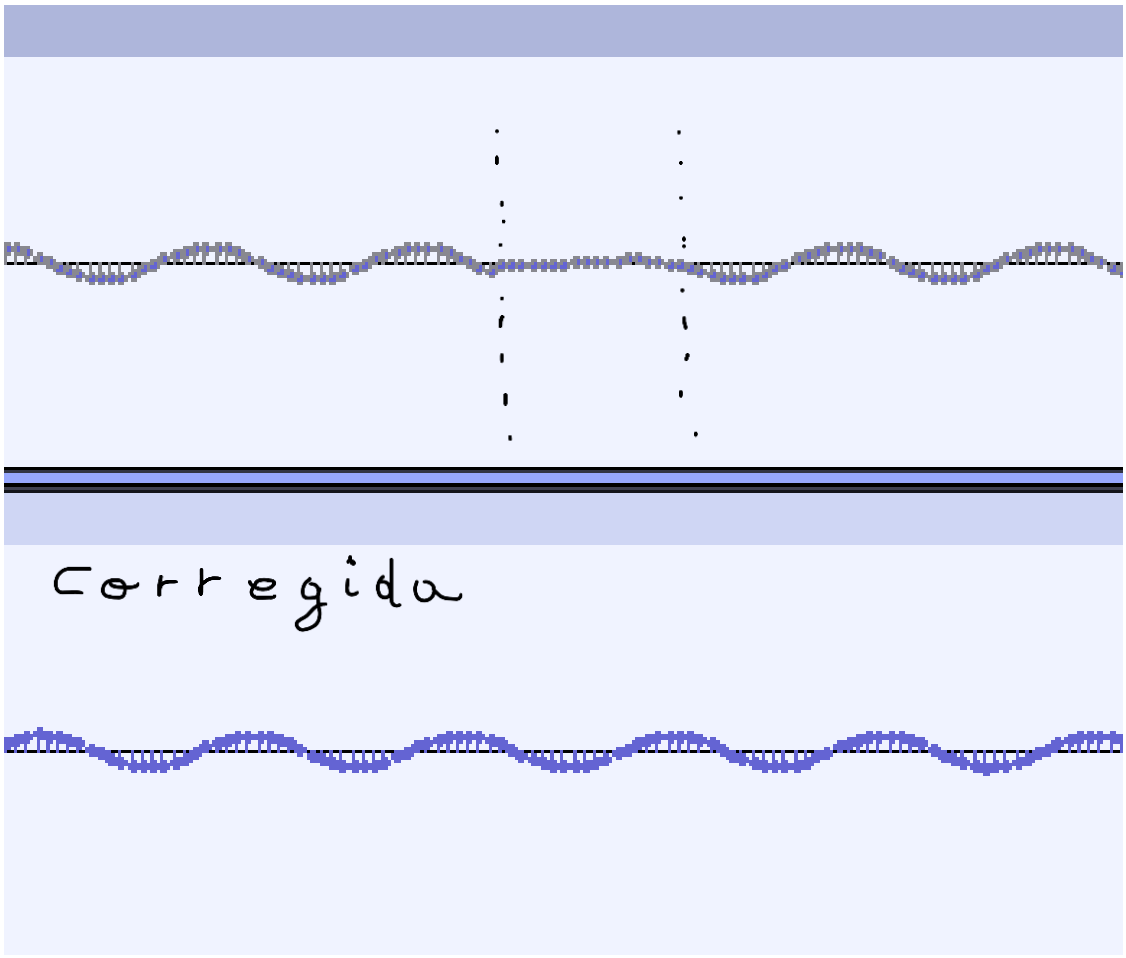
Como podemos ver al aumentar el número de taps o de coeficientes en el filtro podemos hacer que este sea más eficiente a costa de un mayor tiempo de procesado claro.

4. ¿Puedes construir un filtro que deje pasar un tono de 800 Hz y elimine completamente uno de 1200 Hz? ¿Por qué?

Tras ir probando, ni el filtro banda ancha ni rechazo banda sirven. Aun aumentando los coeficientes hasta el máximo que permite el código y en consecuencia teniendo pérdidas de muestras (a 63 taps), no es posible eliminar por completo la frecuencia de 1200hz, a lo sumo atenuarla, pero eliminarla, al menos con un filtro FIR que usa convolución no. En teoría con un número infinito de taps podría pero no es físicamente posible. Aunque claro está que con un número elevado de taps podemos hacer que la frecuencia sea inaudible pero eliminada por completo con un número discreto de taps no.

Tarea 4

1. Genera un tono de 800 Hz y pásalo por el filtro paso bajo de 30 taps del apartado anterior, con la convolución sin corregir y corregida. ¿Escuchas alguna diferencia? Muestra una captura de ambas señales de salida en tiempo donde se vea el límite entre dos bloques y explica la diferencia.



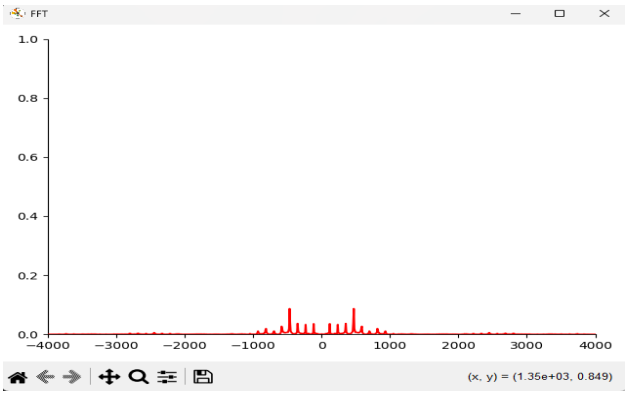
Desde luego se escucha una diferencia, en la primera se escucha un ruido entrecortado mientras que en la segunda se escucha la señal continua. Esto ocurre porque la convolución, por definición matemática de la operación, necesita los L-1 elementos anteriores donde L es el cardinal del conjunto de los coeficientes de la señal, sino la convolución para los valores anteriores los tomará como 0, entonces si taps es 30 por ej, el primer valor de la convolución para el nuevo bloque será 29 ceros + un valor distinto de cero, básicamente tiende a cero, para el siguiente serán 28 ceros, luego 27…. Y así, por eso vemos en la primera imagen “la no corregida” que aunque los valores tienden a cero, a medida que avanza el tiempo la amplitud sube muy muy poco, porque cada vez hay menos ceros hasta que volvemos a trabajar por completo con nuestra señal.

Tarea 5

1. **Captura el sonido ambiente con una frecuencia de muestreo de 8000 Hz y un tamaño de bloque de 1024 muestras. ¿Cómo es el espectro?**

   El espectro sonoro (distribución de las frecuencias) se muestra en una ventana que se abre mostrando la FFT(Fast Fourier Transformation)

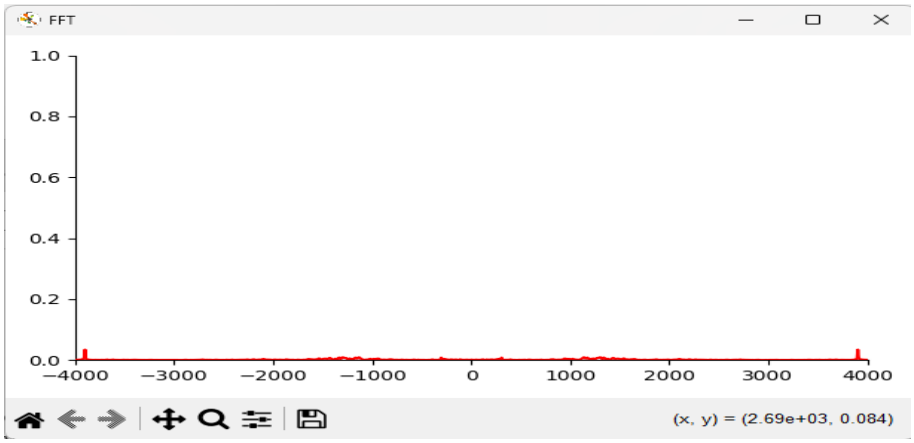
   En tiempo real, es decir, podemos ver las frecuencias que se captan en tiempo real, si reproducimos una frecuencia aguada veremos como hay un pico, lo mismo con una grave, las toma en una escala del 0 – 1 para mostrar la intensidad.



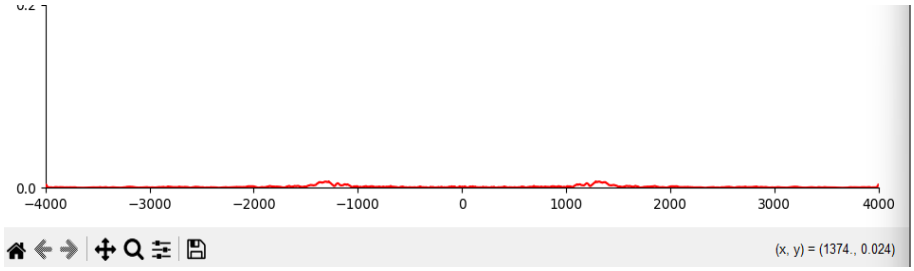
   Un ejemplo de lo que se ve pero por supuesto cambia en tiempo real.

2. **Genera un tono de 3900 Hz. Incrementa gradualmente la frecuencia del tono hasta 4100 Hz. Obtén varias capturas. ¿Qué se observa cuando el tono supera los 4000 Hz? ¿A qué se debe?**

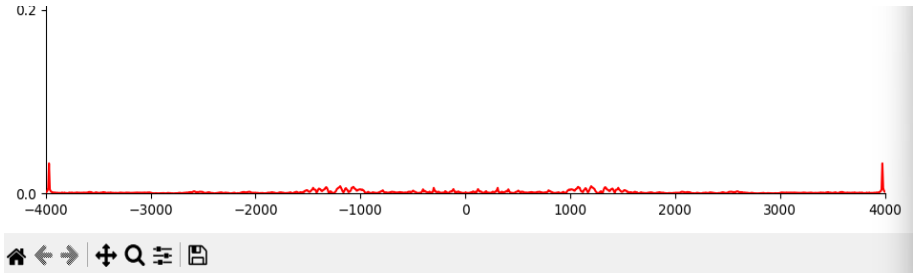
3900 hz:



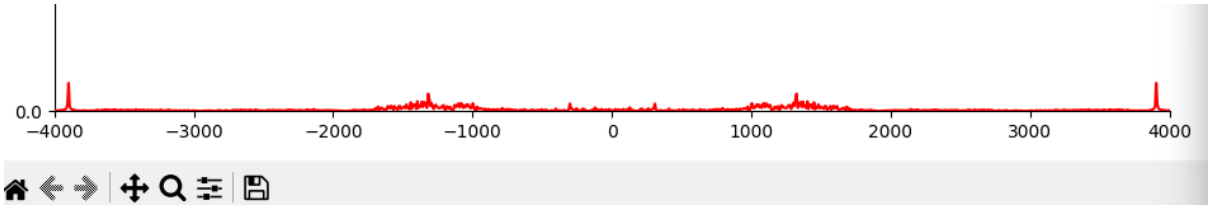
4000hz



4030hz



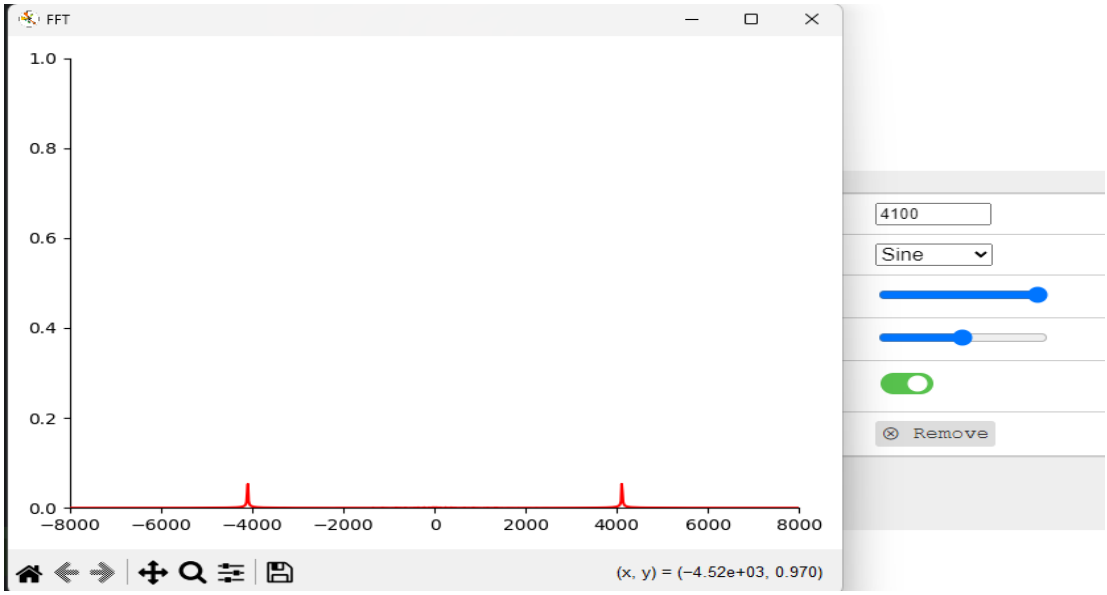
4100hz



En principio se debe a que no podemos recuperar la señal, no la podemos reconstruir digitalmente y mostrarla correctamente. ¿Por qué? Porque no se

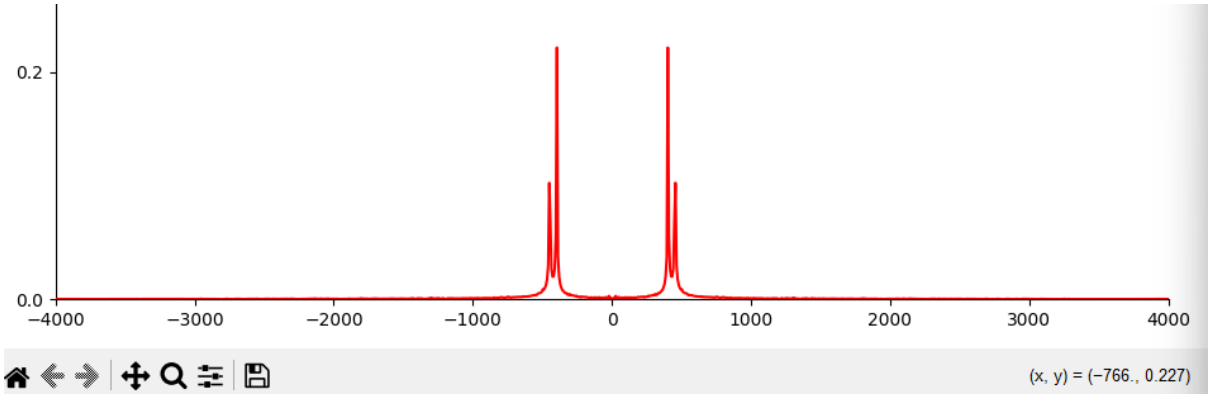
satisface el teorema de Nyquist, fs >= 2f. O sea, si queremos muestrear una señal continua de 4000hz, entonces nuestra frecuencia de muestreo debe ser más del doble, es decir, fs > 8000, y estamos muestreando justo con una frecuencia de 8000, entonces a partir, no podemos reconstruir correctamente la señal y se produce el efecto del aliasing, que es básicamente el nombre de lo anterior descrito, el aliasing es el efecto por el que no se puede distinguir una señal continua al pasarla a digital. Este efecto hace que la señal se refleja hacia abajo, como se ve la señal está retrocediendo en frecuencias en lugar de aumentar, este es el producto de que no se represente correctamente.

Pero como vemos aquí, si muestreamos a 16000hz no hay ningún problema al representar valores por encima de 4000hz.

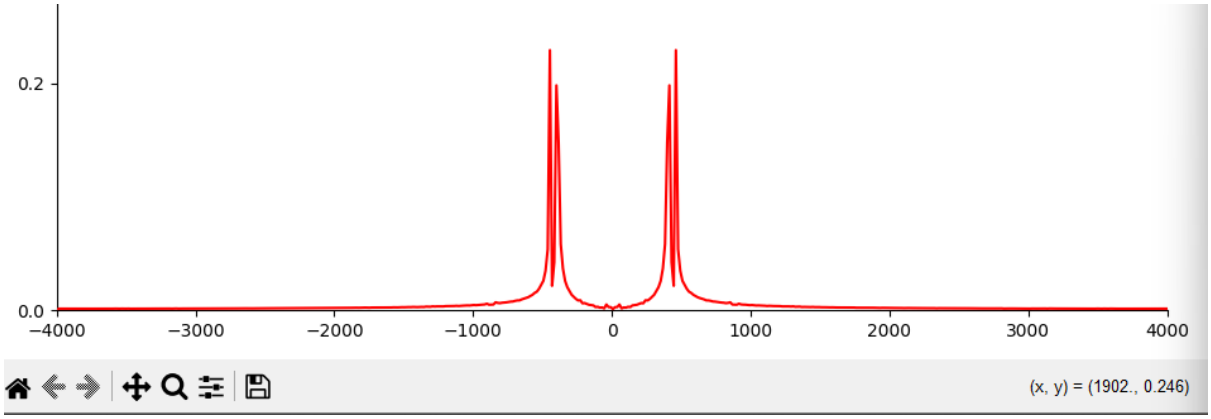


3. **Genera dos tonos, uno de 400 Hz y uno de 450 Hz. Muestra una captura del espectro de la señal mientras suenan los dos tonos a la vez utilizando diferentes tamaños de bloque: 1024, 512, 256 y 128 muestras. ¿Como afecta la reducción del tamaño de bloque al espectro de la señal? ¿A qué se debe?**

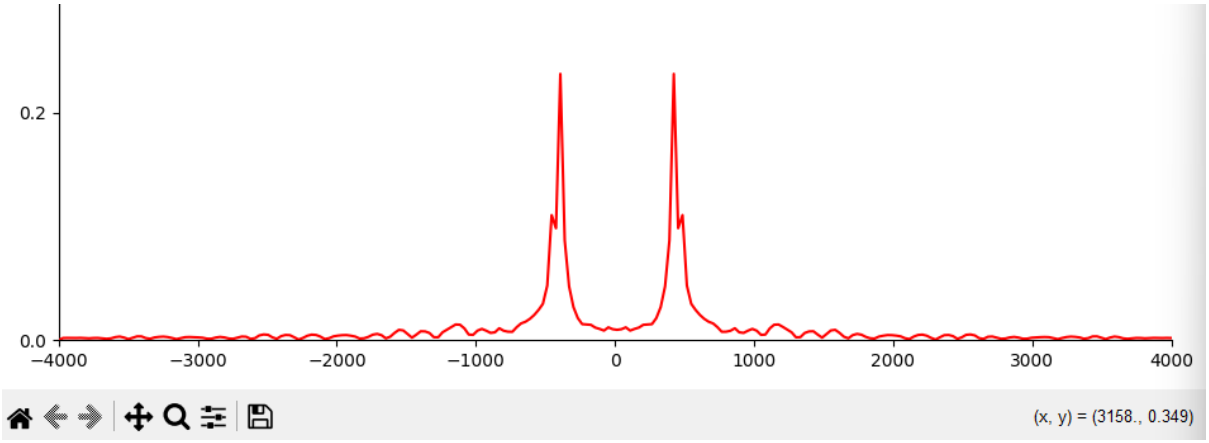
1024 muestras por bloque:



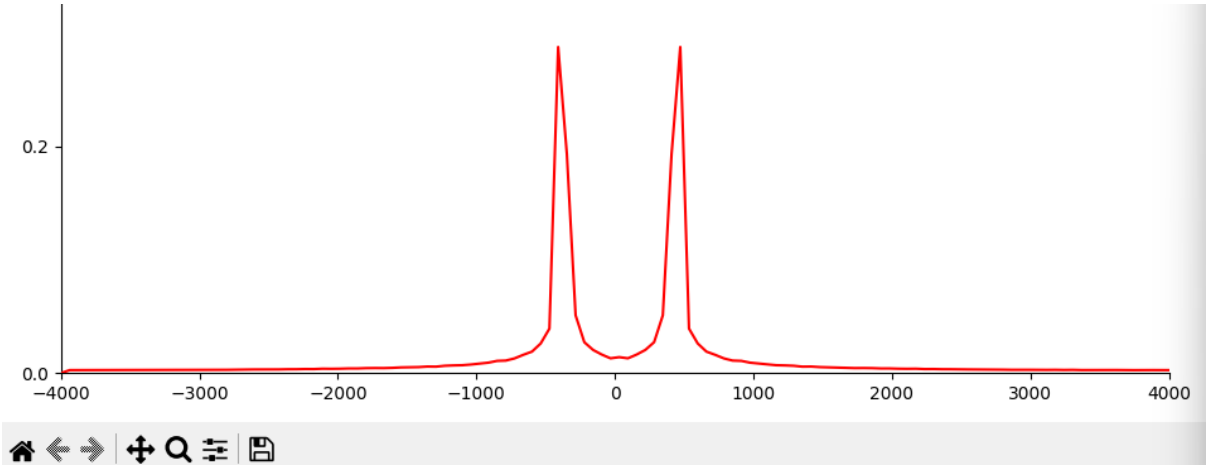
512 muestras por bloque:



256 muestras por bloque:



128 muestras por bloque:



Afecta de la siguiente manera, describiendo lo que ocurre en las imágenes, vemos un solapamiento entre ambas señales de 400 y 450 hz, esto se debe a que lo que vemos en pantalla en tiempo real es la FFT de la convolución de cada bloque, es decir, a menor tamaño tenga el bloque (menor número de muestras) menos será la precisión con la que se muestran en pantalla.

Tarea 6:

Sí, es completamente posible. Primero, vamos a partir del escenario donde es difícil poder hacerlo, esto es cuando la frecuencia que medimos no cae dentro de un bin exacto, cuando ocurre esto la frecuencia se dispersa en bins vecinos, esto produce que distinguir entre dos frecuencias separadas por tan pocos hz sea complicado, además, este efecto aumenta a medida que aumentamos el volumen, aunque nuestra frecuencia no caiga en un bin específico, si el volumen es bajo, es posible que la distinción entre ambas frecuencias sea posible(aquí estamos suponiendo que el número de bloques por muestras es lo suficientemente grande como para que pueda haber una separación física entre dos frecuencias separadas 20hz, si muestreamos por ejemplo a 8000 hz con un tamaño de bloque de 128, 8000/128 = 62.5, cada bin representa 62.5 hz, o sea que dos frecuencias separadas 20 hz estarían en el mismo bin, por supuesto, lo primero que tenemos que suponer es que el tamaño del bloque es lo suficientemente grande como para poder dejar

estas 2 muestras en bins distintos) ahora, si las frecuencias que queremos medir cada una cae en un bin especifico, entonces sin importar cuanto se suba el volumen, se puede distinguir perfectamente entre esas dos frecuencias separadas 20hz, supongamos que queremos distinguir entre 400hz y 420hz, un divisor común es el 10, entonces si multiplicamos 10 por un tamaño de muestra de 1024 tenemos 10240, o sea, si muestreamos a 10240hz con un tamaño de bloque de 1024 muestras podemos distinguir perfectamente entre estas dos muestras separadas 20hz, lo he probado y se puede perfectamente.