**UNIVERSIDAD DE SANTIAGO DE CHILE**
**FACULTAD DE INGENIERÍA**
**DEPARTAMENTO DE INGENIERÍA INFORMÁTICA**


# Señales ánalogas y digitales

Integrantes: Gabriel Bustamante Toledo

Curso: Redes de computadores

Profesor: Carlos González

Ayudante: Nicole Reyes

19 de Mayo de 2020

# Tabla de contenidos

# Índice de figuras

# 1. Introducción

Entre los distintos modelos de redes que se utilizan para estudiar su funcionamiento, existen ciertas capas que realizan operaciones fundamentales en el funcionamiento de una red, una de estas capas importantes es la capa física, que se focaliza en como transportar los datos por la red y entre los computadores. Para los efectos de esta experiencia, se estudiará y se dará un acercamiento a como son tratadas las señales utilizando herramientas computacionales, en este caso se hará con la ayuda de Python3 y sus librerías para la programación científica.

Se procesará un audio, al cual se le obtendrá su transformada de Fourier con ayuda de Scipy, adicionalmente se obtendrá el espectrograma del audio ingresado. Para poder "limpiar"la señal se probará utilizar distintos filtros con diferentes parámetros, y así obtener un audio lo mas limpio posible.

## 1.1. Objetivos

1. Reforzar de forma practica el procesamiento de señales.
2. Comprender el funcionamiento de los filtros.
3. Analizar una señal en función del tiempo y de la frecuencia.
4. Utilización de herramientas computacionales para manipular señales.

# 2. Marco teórico

Para comprender de que forma se llegará a la solución, hay que comprender una parte teórica, para posteriormente poder utilizar herramientas informáticas. En este caso se pondrá en contexto que es la transformada de Fourier y que representa, en cuanto a lo practico se dará a conocer las librerías de Python y sus funciones junto a sus parámetros que se utilizaran.

## 2.1. Transformada de Fourier

Es una transformación matemática utilizada para transformar señales desde el dominio del tiempo hacia el dominio de la frecuencia o viceversa (en el caso de usarla de forma inversa), posee mas aplicaciones en física e ingeniería, pero para nosotros es mas relevante la propiedad de transformar las señales.

## 2.2. Herramientas Computacionales

Para el desarrollo de esta actividad es esencial la utilización de herramientas computacionales, tanto para la precisión, como para hacer extensos cálculos. A continuación se describirán las principales funciones a utilizar en el código, dando una explicación a las entradas que se utilizarán y a las salidas de cada función.
Para el calculo de la transformada de fourier se utilizará scipy.fftpack.fft(), esta función recibe como primer argumento un array de números que serán a los cuales se le aplique la transformada, como resultado esta función entrega un array de números complejos.
Para la transformada inversa de fourier se hará utilización de scipy.fftpack.ifft(), que recibe un array de numero y retorna un array de flotantes.

Para crear el filtro se usará la función scipy.signal.butter(), el primer argumento hace referencia al orden que tendrá el filtro o que tan recto será, el segundo argumento es la frecuencia en la que se realizará el corte, el tercer argumento es opcional, pero indica de que tipo será el filtro, pasa bajos, pasa altos o pasa banda.
Luego de crear el filtro se debe aplicar a la señal, para realizar este paso se utiliza la función

scipy.signal.lfilter(), que recibe como argumentos las salidas de la función previamente realizada y la señal que se quiere filtrar, como resultado retorna un array que representa la nueva señal ya filtrada.

# 3. Desarrollo

## 3.1. Audio en el tiempo

Luego de leer el archivo de audio .wav con scipy.io.wavfile.read(), con los parámetros retornados se crea un array que representará una linea de tiempo de 0 a la duración del audio. Con este array de flotantes y con el array de la señal, es posible realizar la gráfica del audio en función del tiempo (figura [1]).

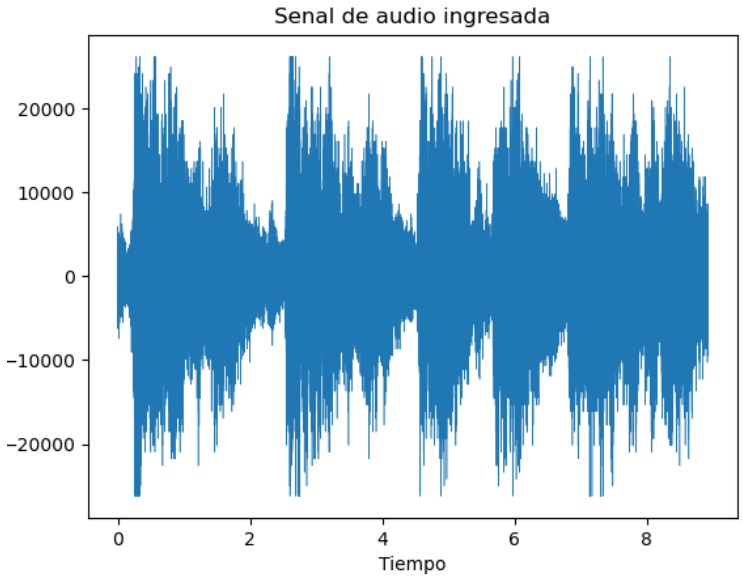


Figura 1: Audio en función del tiempo.

## 3.2. Transformada de Fourier del audio

En el caso de realizar el gráfico de la transformada de Fourier del audio se necesitan dos elementos, uno es la transformada misma y el otro es la frecuencia. En estos dos casos se hace utilización se scipy.fftpack, para la transformada se utiliza scipy.fftpack.fft() y para obtener la frecuencia se utiliza scipy.fftpack.fftfreq(). Con el retorno de estas dos funciones de puede realizar el gráfico tanto de la señal original (figura [2]) como de la señal filtrada (figura [6]).

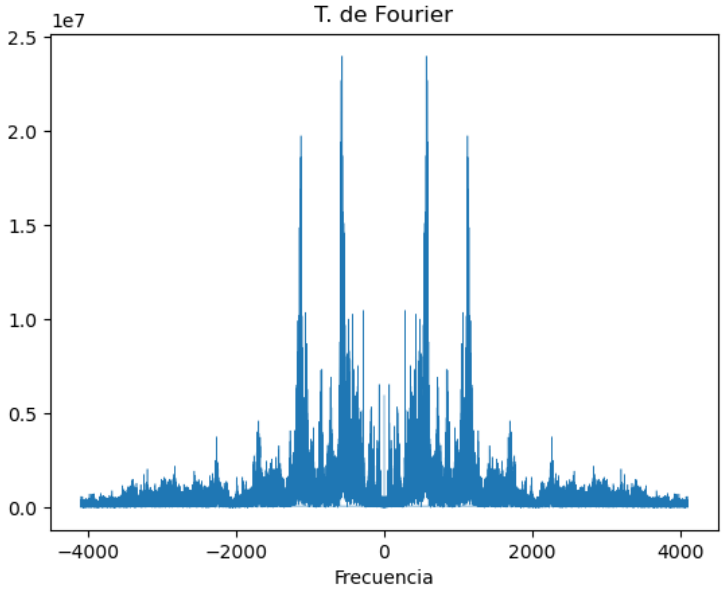


Figura 2: Transformada de Fourier del audio.

## 3.3. Transformada inversa de Fourier del audio

Para calcular la transformada inversa se utiliza la transformada calculada anteriormente, esta es ingresada a la función scipy.fftpack.ifft(), la cual entregaría la transformada inversa, junto con esto utilizaremos el array del tiempo obtenido inicialmente para poder hacer el gráfico (figura [3]).

## 3.4. Espectrograma del audio

Un espectrograma es un gráfico que contiene tres variables, para obtener estas variables haremos utilización de la función signal.stft(), la cual retorna los valores de la frecuencia, el tiempo y el valor de cada frecuencia en función del tiempo, lo cual da como resultado el espectrograma tanto de la señal original (figura [4]) como de la señal filtrada (figura [7]).

## 3.5. Audio filtrado

Para poder filtrar un audio de forma correcta se deben hacer dos pasos, que ha grandes rasgos son primero crear una función que será el filtro y como segundo aplicar esta

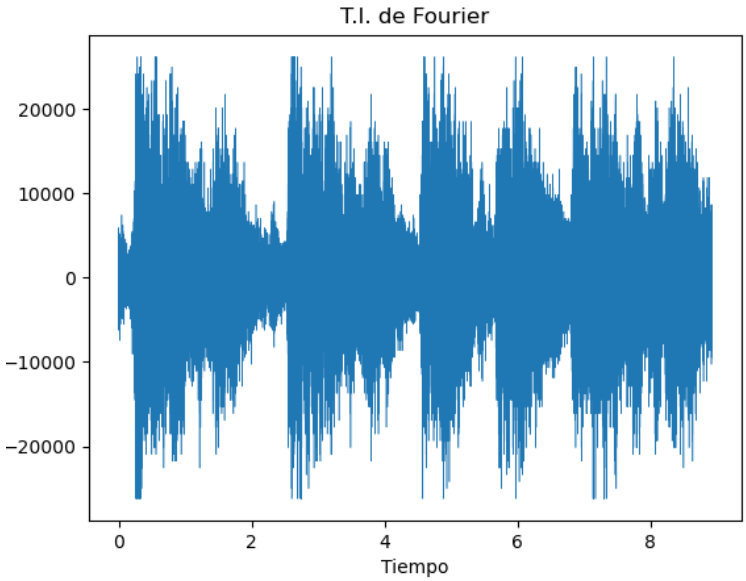


Figura 3: Transformada inversa de Fourier del audio.

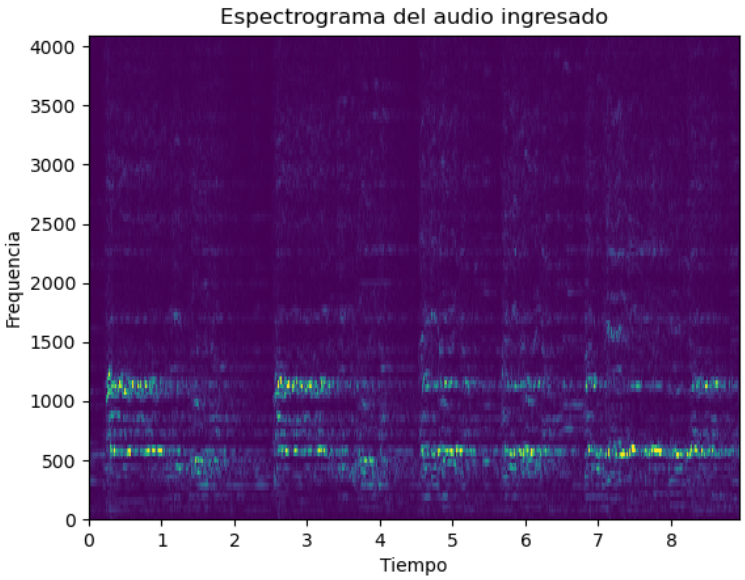


Figura 4: Espectrograma del audio.

función filtro a la función que se desea filtrar. Para crear el filtro se utilizará la función signal.butter(), que creará el filtro según la frecuencia y el tipo de filtro que necesitemos, para aplicar el filtro se utilizará la función signal.lfilter(), con esto se obtiene el siguiente gráfico (figura [5]).

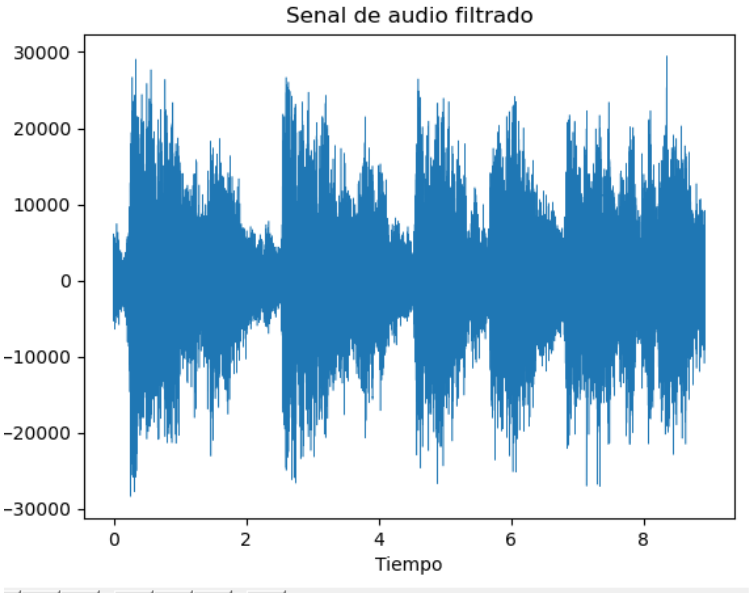


Figura 5: Audio filtrado en función del tiempo.

# 4. Análisis

En el gráfico 1 se puede apreciar cuál es la amplitud de la señal en cada instante de tiempo, esto nos puede decir que tan 'fuerte suena' el audio en esos instantes. Observando el gráfico 2 ya podemos rescatar información mas importante, podemos ver el rango de frecuencia que tiene el audio, cual de estas frecuencias son las mas importantes y que tiene mas 'presencia', y cuales de estas frecuencias no son de mucha utilidad, de lo cual se podría deducir que es ruido. En el presente caso es fácil percatarse que las frecuencias mas importantes son las bajas, en un rango del 0 al 2000 aprox. y las frecuencias altas a partir de 2000 pueden llegar a considerarse señales de ruido, para poder limpiar la señal de estas frecuencias se aplicará un filtro. Analizando el ultimo gráfico 4 del audio original nos encontramos con un espectrograma, el cual es en tres dimensiones que nos muestra como es que cambia el espectro de la frecuencia en función del tiempo, también nos ayuda a confirmar la aseveración previamente hecha con respecto a las frecuencias con mayor presencia y las frecuencias que pueden ser consideradas como ruidosas.

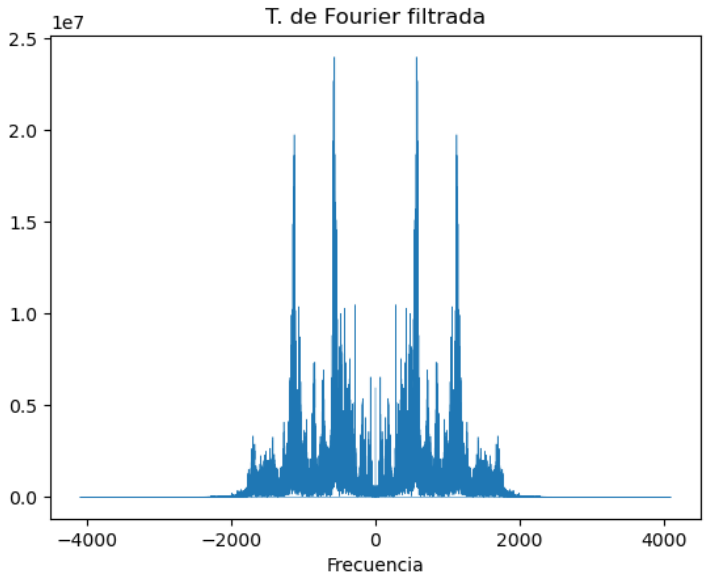


Figura 6: Transformada de Fourier del audio filtrado.

Para aplicar un filtro, como se señala en el párrafo anterior, hay que observar la señal en el dominio de la frecuencia e interpretar cuales con aquellas frecuencias mas dominantes en la señal. En el caso de esta experiencia es fácil darse cuenta que tienen mayor importancia

las frecuencias bajas, para ser mas especifico desde la 2000 hacia abajo, para esto se diseña un filtro pasa bajos con una frecuencia de corte de 1700, esto es que debido a que los filtros no llegan a ser del cien por ciento precisos, entonces para que logre hacer un corte justo en los 2000 se hace el filtro un poco antes. Una vez realizando el filtro se gráfica su resultado para detectar las diferencias y si el filtro se aplico de forma exitosa, como resultado tenemos la transformada de fourier filtrada (figura [6]), donde se ve que justo en la frecuencia 2000 es filtrada la señal y solo quedan las frecuencias bajas que son las mas importantes, adicionalmente se analiza el espectrograma (figura [7]), del cual se ve que aproximadamente en la frecuencia 2000 hacia las frecuencias altas ya no hay nada de presencia, con este filtro se logro eliminar por completo las frecuencias altas que en efectos de esta experiencia son frecuencias que no son muy útiles.

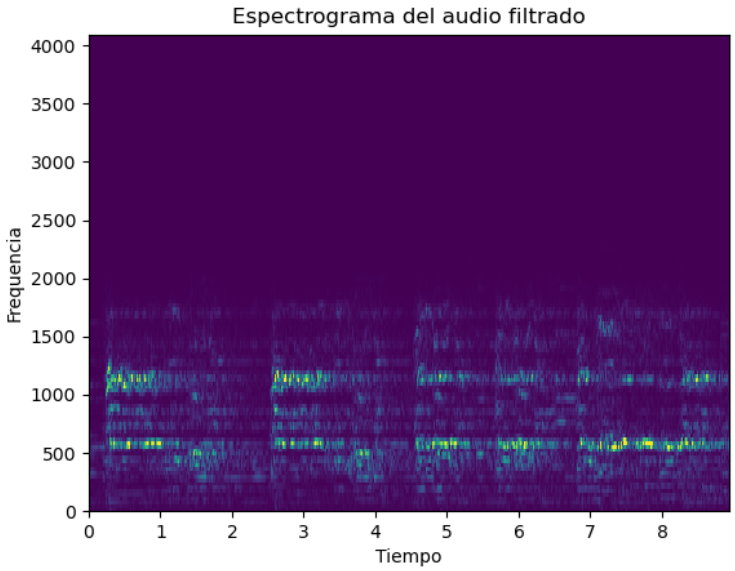


Figura 7: Espectrograma del audio filtrado.

Realizando una comparación visual en función del tiempo, entre la señal original con la señal filtrada, es claro ver que los picos mas pronunciados de la señal original están de igual forma en la señal nueva pero con menor amplitud, sobre todo en los últimos 2 segundos donde es mayor la diferencia ya que la señal filtrada tiene amplitudes mas compactas.

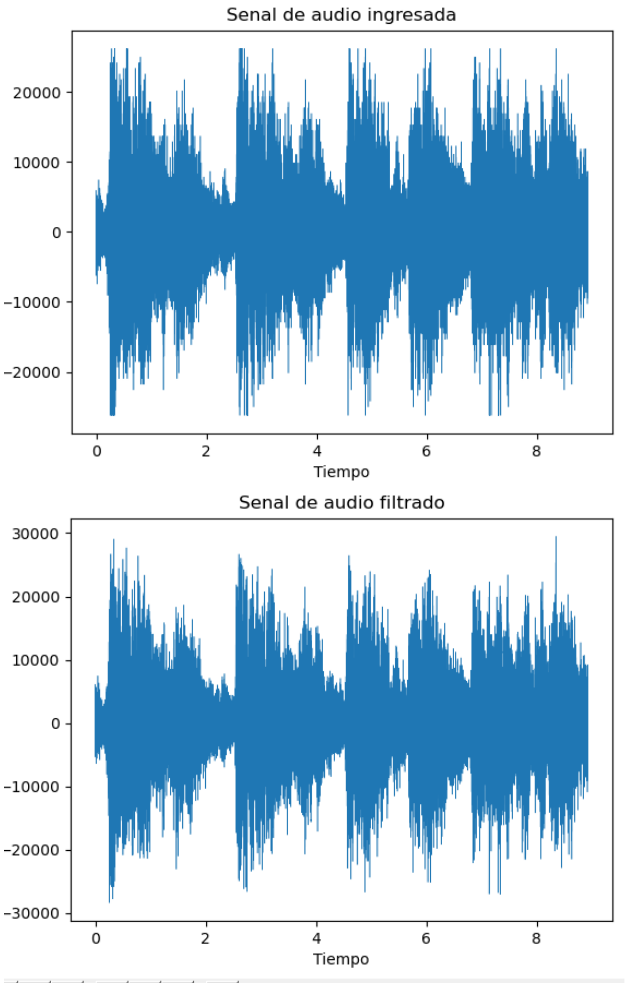


Figura 8: Comparación de señales en el tiempo.

# 5. Conclusión

A lo largo de esta experiencia se puede ver que uno de los principales objetivo, o en lo que estaba mas centrado el desarrollo de la actividad, era en torno a analizar la señal para crear un filtro y posteriormente comparar ambas señales. En el desarrollo se encontraron problemas mas bien relacionados con la herramientas computacionales utilizadas, esto debido a la poca experiencia con el procesamiento de señales con Python3, lo cual fue solucionado consultando a la documentación de las extensiones utilizadas.

Se puede ver como los filtros son efectivos solamente con un buen análisis previo del espectro de la señal a filtrar. Adicionalmente a esto es importante darse cuenta de la relación que existe entre aplicar un filtro con la convolución, ya que son temas bastante relacionados.

Personalmente rescato el hecho de poder aprender a utilizar las funciones mencionadas en el informe, y ver de 3 formas distintas como analizar una señal (en función del tiempo, en función de la frecuencia y en un espectrograma), y como son aplicados los filtros ya que previo a esta experiencia no tenia conocimientos acerca de este tema. En general la actividad pudo ser desarrollada en su totalidad analizando todos los puntos propuestos inicialmente.

# Bibliografía

[1] [Online] https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.fft.html#scipy.fft.fft.

[IFF] [Online] https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.ifft.html#scipy.fft.ifft.

[3] [Online] https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.fftfreq.html#scipy.fft.fftfreq.

[BUT] [Online] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.butter.html#scipy.signal.butter.

[LFI] [Online] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.lfilter.html#scipy.signal.lfilter.