



# **UNIVERSIDAD NACIONAL DE TRUJILLO**

Facultad de Ingeniería  
Programa de Ingeniería Mecatrónica

LABORATORIO N° 1

“FILTRO MEDIA MÓVIL Y FILTRO GAUSSIANO”

## **DESARROLLO DE GUIA DE LABORATORIO**

Procesamiento Digital de Señales e Imágenes

**ESTUDIANTE(S) :**

1. Rios Saldaña Hootie Baxter
2. Robles Gonzales Richard Daniel
3. Rodriguez Morales Joseph Daniel

**DOCENTE :**

**ASTO RODRIGUEZ EMERSON MAXIMO**

**CICLO :**

**2023 - II**

Trujillo, Perú  
2023

# INDICE

INDICE	ii
RESUMEN	1
DESARROLLO DEL LABORATORIO	2
1.1. Resultados de la experiencia	2
a) Filtro media móvil	2
b) Filtro gaussiano	5
1.2. Desarrollo de test de comprobación	7
a) ¿Cuál es la diferencia entre una señal estocástica de una señal determinística? Explique.	7
b) ¿Qué es la tasa de muestreo de la señal? Explique.	8
c) ¿El filtro media móvil implementado es causal? Explique.	10
1.3. Recomendaciones	11
1.4. Conclusiones	11
REFERENCIAS BIBLIOGRÁFICAS	1

## **RESUMEN**

En este informe se ha realizado el diseño y la implementación de filtro gaussiano y media móvil implementado en COLAB aplicado a una señal de audio, acompañado de una simulación previa, teniendo como objetivo conocer algoritmos de transformación de punto para modificar las propiedades de una imagen digital en el dominio del espacio; llegando a conclusiones fundamentadas, verificando así el funcionamiento del filtro de señales , así como el comportamiento de ellas mediante un análisis mediante simulación.


## DESARROLLO DEL LABORATORIO

### 1.1.Resultados de la experiencia

#### a) Filtro media móvil

Para esta primera experiencia importamos un archivo de audio utilizando la librería pysoundfile como se muestra en la figura 1 y cuya señal generada por dicho audio se muestra en la figura 2.

```
[2] import IPython
    IPython.display.Audio('Im_Superman.wav')
```



```
[4] import soundfile
    import matplotlib.pyplot as plt

    audio_signal, sampling_rate = soundfile.read('Im_Superman.wav')

    type(audio_signal)
```

Figura 1 Importación del audio en formato wav.

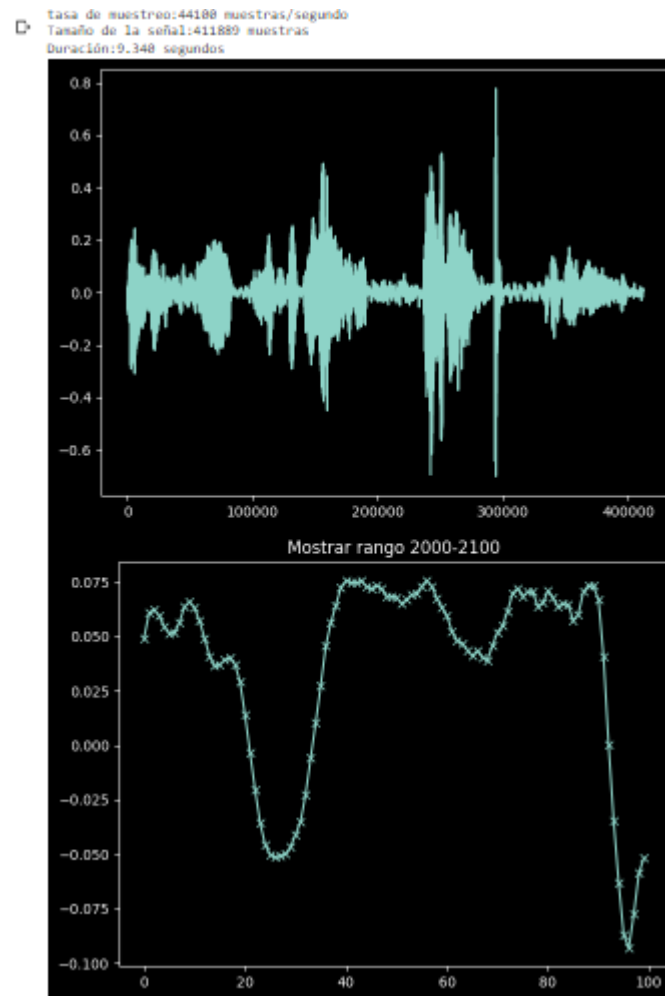


Figura 2 Señal generada por el audio.

Ahora creamos la señal

```
fm = sampling_rate  
t = np.arange(0, len(audio_signal)) / fm  
n = len(t)  
amp = audio_signal  
sen_ruido = amp
```

Figura 3 Código para generar la señal de audio.

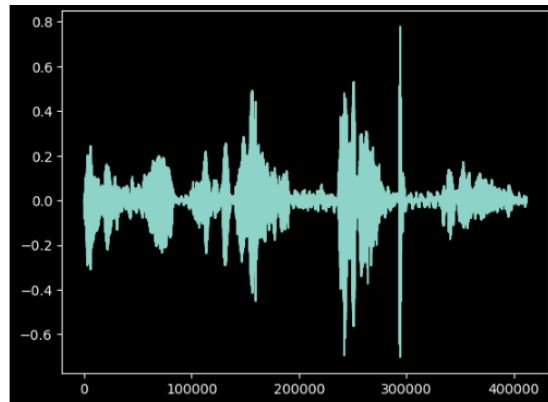


Figura 4 Señal de audio generada.

El siguiente paso es la aplicación del filtro media móvil utilizando el siguiente como se puede apreciar en la figura 5 y 6

```
[9] k=20
    senial_filtrada = np.zeros(n)

    for i in range(k, n-k-1):
        senial_filtrada[i]=np.mean(sen_ruido[i-k:i+k])
    tama = 1000*(2*k+1)/fm
```

Figura 5 Código para calcular el tamaño de la ventana en milisegundos.

```

plt.subplots(1,2, figsize = (25,5))
plt.subplot(121)
plt.plot(t,sen_ruido,label="Señal original")
plt.plot(t,sen_2,label="Señal filtrada")
plt.title(f"Filtro media movil con k={tama}-ms")
plt.xlabel("Tiempo(ms)")
plt.ylabel("Amplitud")
plt.grid()
plt.legend()

plt.subplot(122)
plt.plot(t,amp, "y", label="Señal original")
plt.plot(t,sen_2,"r",label="Señal filtrada")
plt.title(f"Efecto de borde con el filtro media movil k={tama}-ms")
plt.xlabel("Tiempo(ms)")
plt.ylabel("Amplitud")
plt.axis([2, 2.5, -0.025,0.025])
plt.grid()
plt.legend()

plt.show()

```

Figura 6 Código para plotear los gráficos de las señales.

En la siguiente figura se muestra la comparación de la señal original y la señal filtrada.

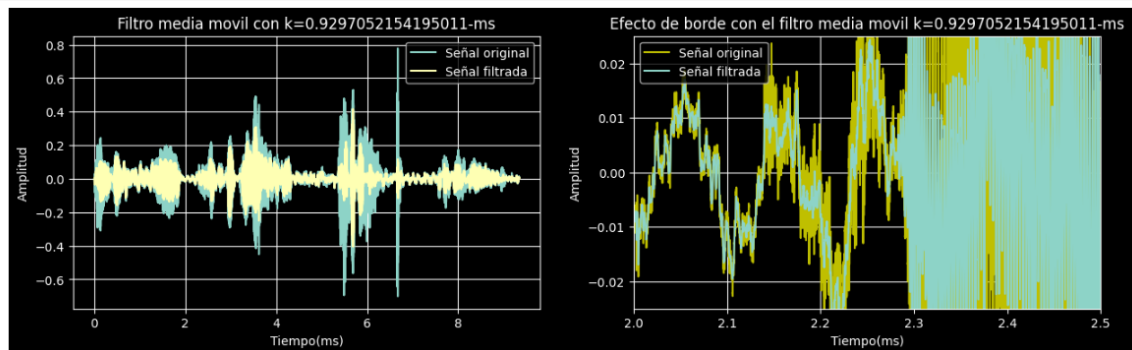


Figura 7 Comparación de la señal original y la señal filtrada.

## b) Filtro gaussiano

Para la aplicación del filtro gaussiano usamos el código que se muestra en las figuras 8 y 9.

```
[32] FWHM = 25
     k = 100

     gauss_t = 1000*np.arange(-k, k)/fm

     filtro_gaussiano= np.exp(-4*(np.log(2)*gauss_t**2)/FWHM**2)
     filtro_gaussiano_normalizado = filtro_gaussiano/np.sum(filtro_gaussiano)
     filtro_gaussiano_normalizado = np.expand_dims(filtro_gaussiano_normalizado, axis=-1)
     flanco_subida=np.argmax((filtro_gaussiano-.5)**2)
     flanco_bajada= k+np.argmax((filtro_gaussiano[k:]-.5)**2)
     FWHM_calculado= gauss_t[flanco_subida]-gauss_t[flanco_bajada]
     plt.plot(filtro_gaussiano_normalizado)
```

Figura 8 Código para el filtro gaussiano.

```
plt.subplots(1,2, figsize=(15,5))
plt.subplot(121)
plt.plot(gauss_t,filtro_gaussiano,label="Filtro Gaussiano")
plt.plot([gauss_t[flanco_subida],gauss_t[flanco_bajada]],
         [filtro_gaussiano[flanco_subida],filtro_gaussiano[flanco_bajada]],
         label="FWHM")
plt.title(f"Filtro gaussiano con FWHM teorico de {FWHM}-ms. Logrado {FWHM_calculado} -ms")
plt.xlabel("Tiempo(ms)")
plt.ylabel("Ganancia")
plt.grid()
plt.legend()

plt.subplot(122)
plt.plot(gauss_t,filtro_gaussiano_normalizado,label="Filtro Gaussiano Normalizado")
plt.plot([gauss_t[flanco_subida],gauss_t[flanco_bajada]],
         [filtro_gaussiano_normalizado[flanco_subida],filtro_gaussiano_normalizado[flanco_bajada]],
         label="FWHM")
plt.title(f"Filtro gaussiano Normalizado FWHM teorico de {FWHM}-ms. Logrado {FWHM_calculado} -ms")
plt.xlabel("Tiempo(ms)")
plt.ylabel("Ganancia")
plt.grid()
plt.legend()
plt.show()
```

Figura 9 Código para plotear las gráficas del filtro gaussiano normalizado.

A continuación se muestra cómo funciona el filtro gaussiano:



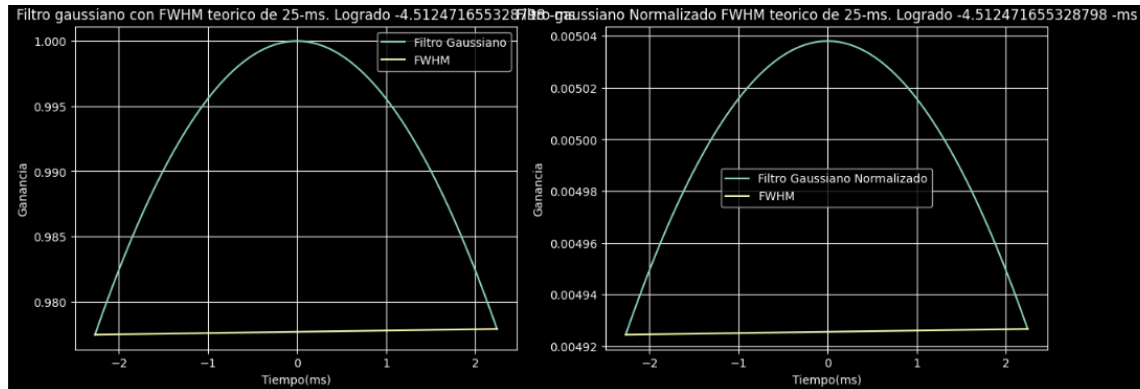


Figura 10 Filtro gaussiano normalizado.

Y finalmente aplicamos el filtro gaussiano empleando el segmento de código que se muestra en la figura 11:

```
senal_filtrada_gauss = np.zeros_like(sen_ruido)

for i in range(k+1, n-k-1):
    senal_filtrada_gauss[i]=np.sum(sen_ruido[i-k:i+k]*filtro_gaussiano_normalizado)
```

Figura 11 Código para aplicar el filtro gaussiano.

La siguiente figura muestra la comparación de la señal original con la señal filtrada:

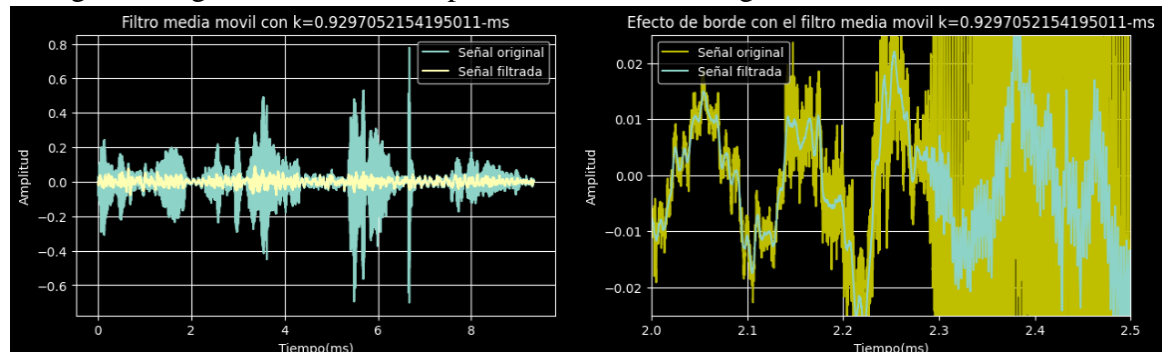


Figura 12 Comparación de la señal original y la señal con filtrado gaussiano.

## 1.2.Desarrollo de test de comprobación

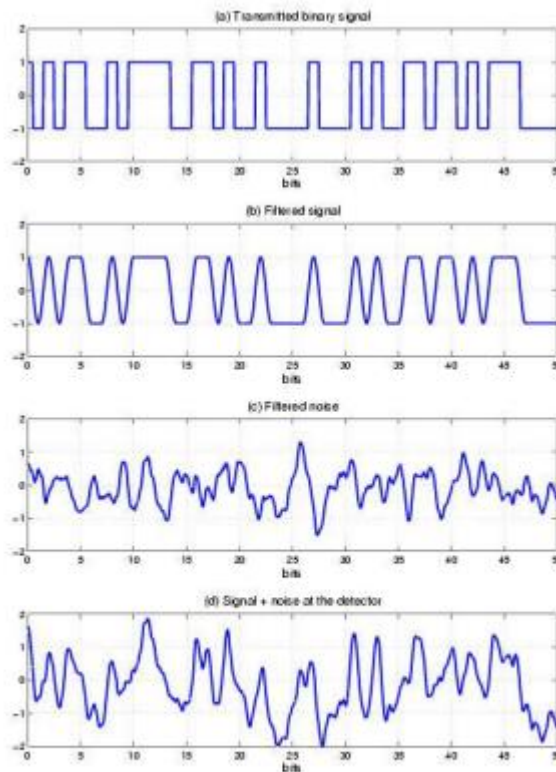
- a) **¿Cuál es la diferencia entre una señal estocástica de una señal determinística? Explique.**

Las **Señales Determinísticas** pueden ser especificadas completamente para cualquier instante de tiempo mediante funciones o sistemas de ecuaciones.

Según Lathi 1.3.2 pdf/p.63, Estándar ISO16, Frecuencias de afinación de un piano:“Señal determinística, conocida como una señal cuya descripción física es completamente conocida por su forma matemática o gráfica.”

Las **Señales Estocásticas** toman valores aleatorios y solo se pueden caracterizar estadísticamente.

Las señales en condiciones de operación real son estocásticas.



Señal + Ruido = Señal estocástica

Figura 13 Señal estocástica.

**b) ¿Qué es la tasa de muestreo de la señal? Explique.**

Al grabar sonido y pasar de una onda de sonido física a digital, el ordenador lo que hace es tomar un montón de muestras para convertir esa señal.

Es como en un vídeo, un vídeo se compone por muchos fotogramas o fotografías puestas de forma continua, una detrás de otra, de manera que al reproducirlas de forma constante vemos el movimiento. Pues esos fotogramas serían el equivalente a esas muestras que se toman a las ondas físicas de sonido.

La frecuencia de muestreo es una forma de medir que cantidad de muestras queremos tomar por segundo de esa onda. Cuantas más muestras de sonido más se acerca a la realidad.

Un claro ejemplo es la siguiente imagen que se muestra para que se entienda de forma gráfica que es una muestra.

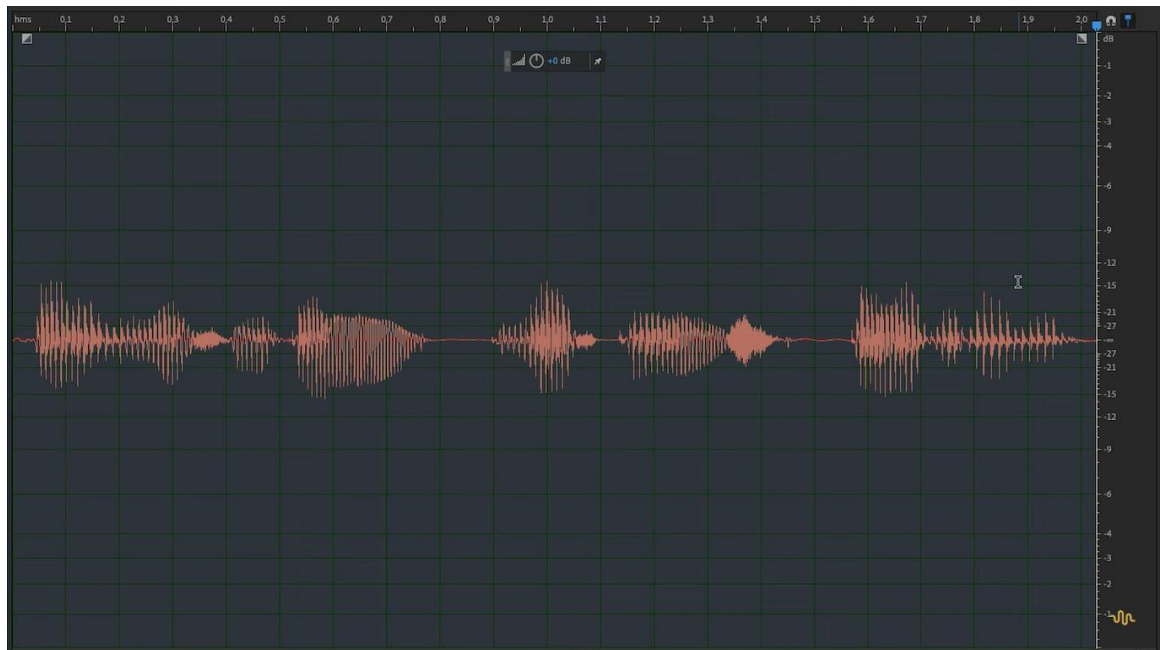


Figura 14 Muestreo de señal

Si ampliamos esta imagen al máximo, se puede ver cada una de las muestras representadas por puntitos.

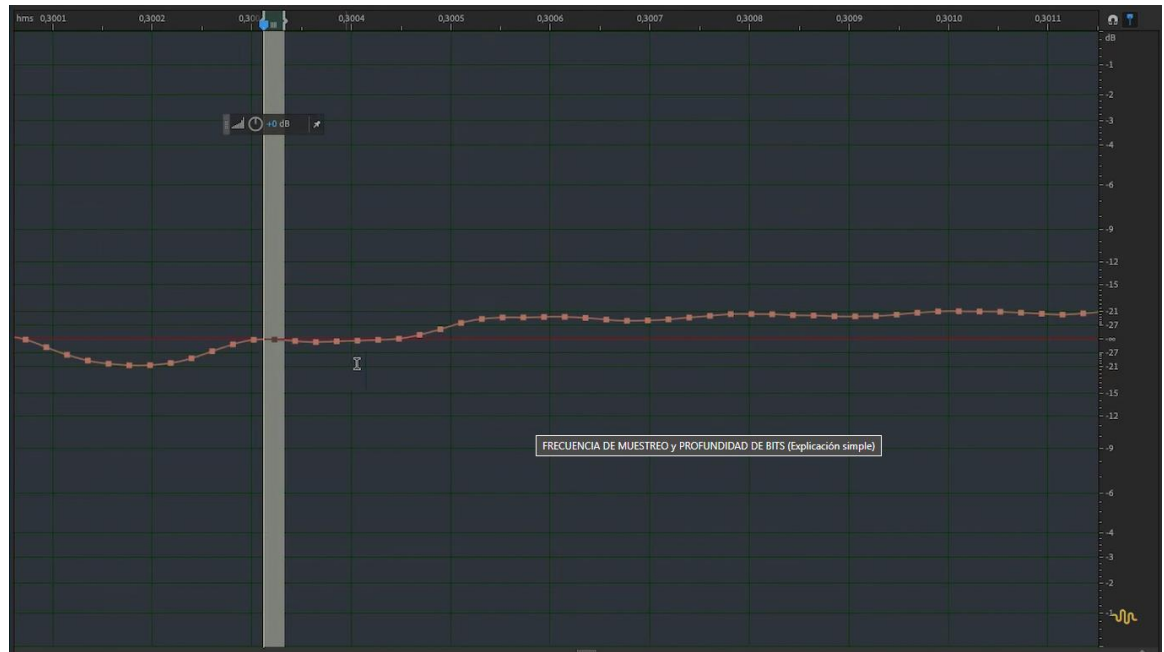


Figura 15 Acercamiento del muestreo de señal.

No confundir con profundidad de bits que sería algo parecido con la frecuencia de muestreo, pero en lugar de muestras tomadas a lo largo del tiempo, serían las muestras tomadas en amplitud de onda.

c) **¿El filtro media móvil implementado es causal? Explique.**

Primeramente, entendamos que un filtro es causal si cada efecto en la salida ocurre luego de la causa correspondiente en la entrada. Por lo tanto los filtros no causales son irrealizables en la práctica. No es posible construir un filtro no causal que opere en tiempo real.

Y en este caso el filtro media móvil es el más simple, intuitivo y fácil de implementar, razón por la que no se presta para ser causal.

### **1.3.Recomendaciones**

- a) Usualmente el filtro media móvil ofrece ciertas desventajas, ya sea por ser bastante sensible a cambios locales así como, crear nuevas intensidades que no aparecen en el ruido original, por lo que se recomienda manejarlo con cuidado.
- b) Se debería implementar el uso de archivos que contengan ambos filtros.
- c) El diseño podría ser adaptable a un sistema de redes neuronales, para que cada uno de los valores sea almacenado para su posterior evaluación y optimización del sistema.

### **1.4.Conclusiones**

- a) El filtro gaussiano es ligeramente mejor que el filtro media móvil, ya que muestra la señal filtrada más suavizada.
- b) El filtro de media móvil es una solución rápida y fácil para filtrar ruido de alta frecuencia, pero presenta dos debilidades: 1) su banda de rechazo tiene poca atenuación y 2) no permite ajustar el ancho de banda de manera independiente de la atenuación.

## REFERENCIAS BIBLIOGRÁFICAS

Kamen, Edward W., y Bonnie S. Heck. (2008). Fundamentos de señales y sistemas usando la Web y MATLAB® PEARSON EDUCACIÓN, México, ISBN: 978-970-26-1187-5

Oppenheim, A. V., Willsky, A. S., & Young, I. T. (1983). Signals and systems. Englewood Cliffs, N.J: Prentice-Hall. Sanz, J. (2005). *Técnicas y procesos en las instalaciones*. Madrid: Paraninfo.

## **ANEXOS**