UNIVERSIDAD SANTIAGO DE CALI
Facultad de Ingeniería
Programa de Ingeniería Electrónica

GUÍA PRÁCTICA DE LABORATORIO

Tema: Diseño de filtros digitales FIR.
Curso: Procesamiento digital de señales
Profesor: Milton Orlando Sarria P.
Correo Electrónico: milton.sarria00@usc.edu.co

Objetivos:

1. Aprender a diseñar filtros digitales usando herramientas de software.
2. Usar herramientas de visualización para mostrar la respuesta en frecuencia de un filtro digital.
3. Emplear un filtro digital para reducir el nivel de ruido en señales de audio.

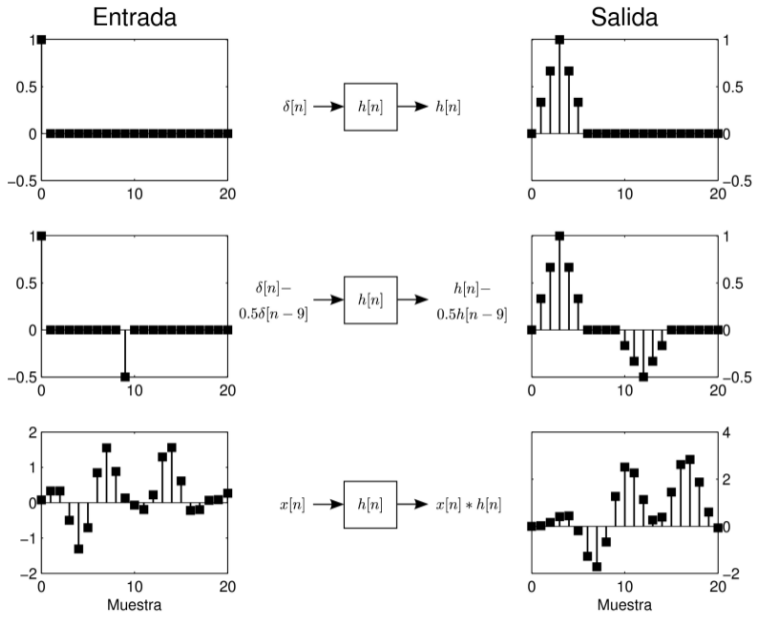
Información Preliminar:

Un filtro digital, es un filtro que opera sobre señales digitales. Es una operación matemática que toma una secuencia de números (la señal de entrada) y la modifica produciendo otra secuencia de números (la señal de salida) con el objetivo de resaltar o atenuar ciertas características.
Entre las múltiples aplicaciones se puede mencionar por ejemplo:

1) Separación de señales que fueron combinadas (ruido, interferencias provenientes de otros sistemas)

2) Recuperación de señales distorsionadas de alguna forma (por ejemplo, al ser trasmitidas)

Conociendo la respuesta al impulso, se puede calcular la respuesta del filtro a cualquier entrada (principio de superposición)



La respuesta en frecuencia es la Transformada de Fourier de Tiempo Discreto de la respuesta al impulso.

$$h[n] \overset{\text{DTFT}}{\longleftrightarrow} H(e^{j\theta})$$

Las transformadas de Fourier de la entrada y la salida del sistema se relacionan por

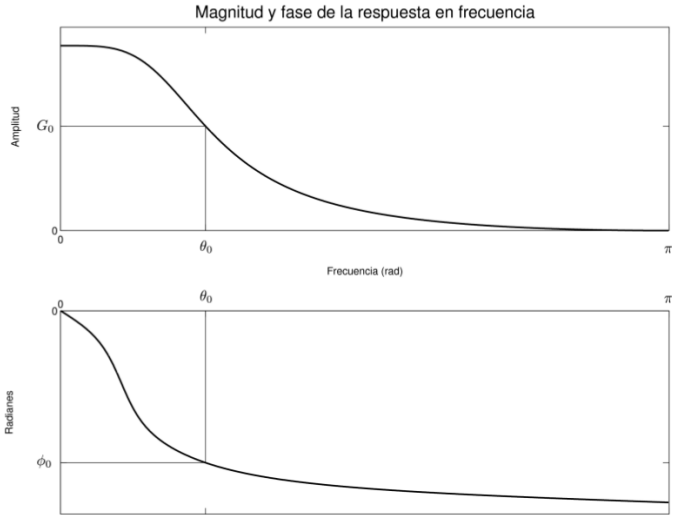
$$Y(e^{j\theta}) = H(e^{j\theta})X(e^{j\theta})$$

Observaciones

● En el caso general, es una función que toma valores complejos.
● Es periódica de período 2pi.
● Al ser una función compleja, se puede representar en notación cartesiana como la parte real y la parte imaginaria o en notación polar como la magnitud y la fase.
●La representación en notación polar es mas útil porque muestra directamente las propiedades del sistema.

Respuesta en frecuencia

$$|H(e^{j\theta_0})| = G_0$$
$$\angle H(e^{j\theta_0}) = \phi_0$$



Entrada

$$x[n] = \text{sen}(\theta_0 n)$$

Salida

$$y[n] = G_0 \text{sen}(\theta_0 n + \phi_0)$$

Implementación de un filtro
Convolución: Convolución de la señal de entrada con la respuesta al impulso del filtro. En este caso, la salida del filtro en cada instante es un promedio ponderado de la muestra actual y muestras pasadas de la entrada. Respuesta al impulso finita (FIR)

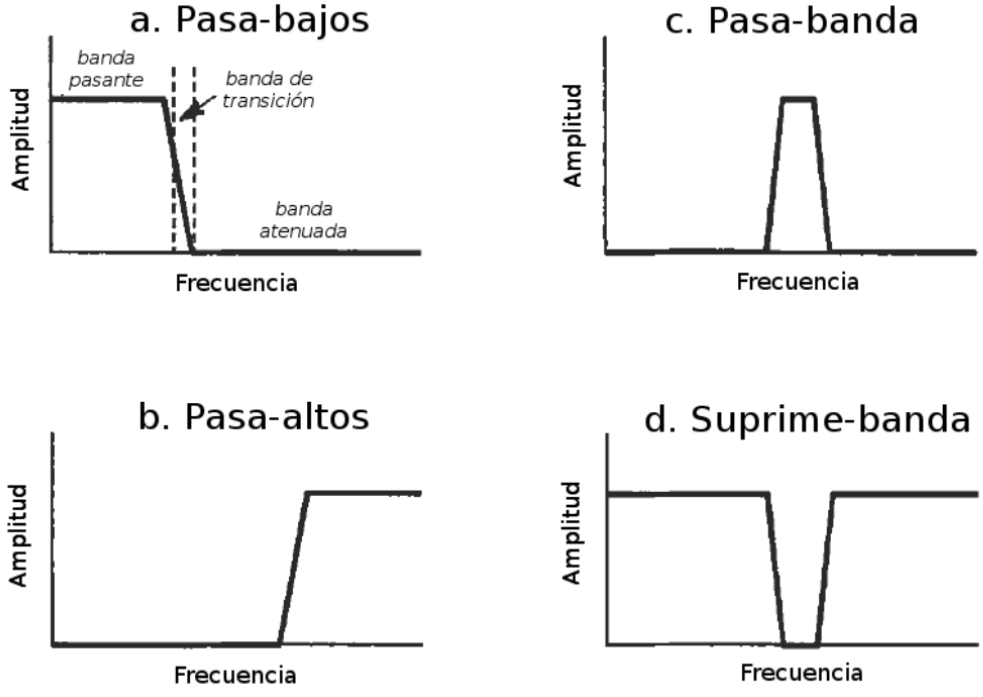
$$\begin{aligned} y[n] &= (x * h)[n] \\ &= \sum_k x[k]h[n-k] \end{aligned}$$

Coeficientes de filtro FIR

En un filtro FIR, los coeficientes de prealimentación de la ecuación de recurrencia son los coeficientes de la respuesta al impulso y los coeficientes de realim

$$\begin{aligned} y[n] &= (x * h)[n] \\ &= \sum_{i=0}^{M} h[k]x[n-k] \\ &= h[0]x[n] + h[1]x[n-1] + h[2]x[n-2] + \cdots + h[M]x[n-M] \\ &= b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] + \cdots + b_M x[n-M] \end{aligned}$$

Filtros selectores de frecuencias
El objetivo es permitir pasar inalterada cierta banda de frecuencias y bloquear completamente el resto. Hay cuatro tipos básicos: pasabjos, pasaltos, pasabanda y suprimebanda.



Clasificación de las regiones de filtros selectores
● Banda pasante: Rango de frecuencias que el filtro permite pasar sin alterar.

● Banda atenuada: Rango de frecuencias que el filtro bloquea.
● Banda de transición: Región entre la banda pasante y la banda atenuada.
● Frecuencia de corte: Frecuencia entre la banda pasante y la banda de transición.

Equipos y materiales:

1. Computador con python instalado
2. Código fuente para análisis frecuencial y diseño de filtros
3. Paquetes de procesamiento de señales de python

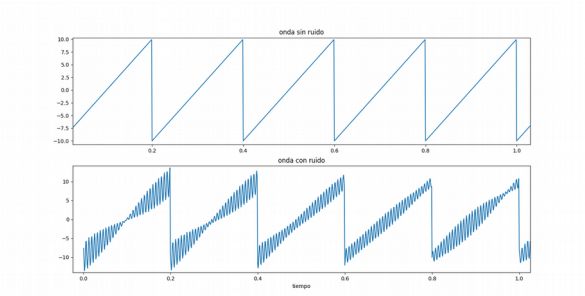
Procedimiento:

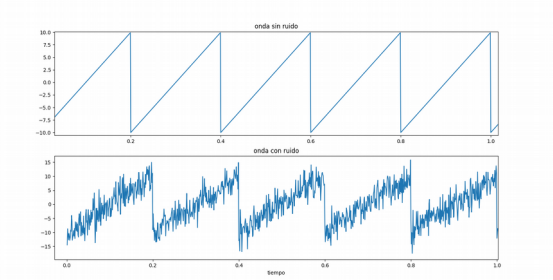
1) Usar el código en el programa **fir_design.py** para diseñar un filtro FIR y analizar su respuesta en frecuencia. Variar parámetros como el tipo de ventana, la frecuencia de corte, el numero de coeficientes, el tipo de filtro, pasa bajas o pasa altas.
Comparar y analizar que cambia cuando se modifican los parámetros mencionados

2) Usar el código en el programa **fourier_sine.py** para analizar las diferencias entre dos ondas, una sinusoidal pura de baja frecuencia, y una onda sinusoidal de baja frecuencia contaminada con ruido también sinusoidal pero de alta frecuencia.

3) Usar el código en los programa **filter_sine1.py** y **filter_sine2.py** para analizar la salida de un filtro FIR pasa bajas. Modificar los parámetros del filtro y explicar como se puede cambiar el filtro para que deje pasar únicamente el ruido y no la señal original.

4) Usar el codigo en el programa **display_onda.py** para visualizar una señal guardada en un archivo de texto y que ha sido contaminada con ruido, como se muestra en la figura. En la parte superior esta la onda original, en la parte inferior se encuentra la onda contaminada con ruido.





Se debe diseñar un filtro FIR para atenuar el ruido en la señal y mostrar el resultado final. Hay dos ondas guardadas en dos archivos diferentes, una onda ha sido contaminada con ruido sinusoidal y la otra con ruido aleatorio. Los archivos son:
sierra_ruido_sin.txt
sierra_ruido_ran.txt
Hacer el procedimiento para las dos señales.

5) Usar el código en el programa **fourier_audio.py** para visualizar el espectro de un archivo de audio antes y después de haber sido contaminado con ruido sinusoidal. Comparar los dos espectros y diseñar un filtro FIR que permita atenuar el ruido y mejorar la calidad del audio. Los audios son:

audio1.wav y audio1_noise.wav

6) Realizar los pasos 1 a 5, analizar el código y modificar cada archivo de acuerdo a las necesidades especificas que se requieren. Presentar conclusiones al final de la sesión y programas modificados mostrando lo que se esta pidiendo en cada punto.

Bibliografía:

1. Alan, V. O., Ronald, W. S., & John, R. (1989). Discrete-time signal processing. New Jersey, Printice Hall Inc.
2. Documentación scipy, numpy y matplotlib