Procesamiento de Señales - Fundamentos

Trabajo práctico N2

Alumno: Ing. Jacobo Salvador

Profesor: Ing. Pablo Slavkin

1. Grafique las siguientes señales lado a lado con su respectivo espectro en frecuencias:

Indicando en cada caso los parámetros destacados como:Frecuencia, amplitud , densidad espectral de potencia, Fs, N, B.

En cada uno de los gráficos se visualiza en orden decreciente hacia abajo, (1) la señal temporal en función del tiempo o de la cantidad de muestras. (2) El espectro de densidad de potencia en función de la frecuencia (FFT). En (3) lo que se hizo fue tomar el espectro de potencia en función de la frecuencia y acortarlo para calcular la potencia de señal en frecuencia y de esa manera determinar el ancho de banda efectivo. En (4) se gráfica la IFFT.

Senoidal:

La función seno tiene bien definido dos picos de señal en el espectro en frecuencia en -fo y +fo donde f0 es la frecuencia fundamental.

Los datos de la senoidal podemos sacarlo directamento de los gráficos.

frecuencia =100 Hz

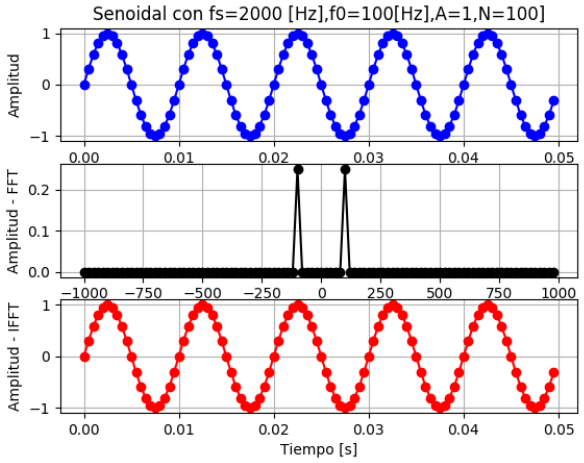
Amplitud =1

densidad espectral de potencia 0.5W /(20 Hz/N)

Fs =2000 Hz

N =100

B=100 Hz



Cuadrada

La potencia de la señal en t es: 1.0

La potencia de la señal en f es: 0.91 recorte la señal en ambos extremos a 20. Aproximadamente el 90 % de la señal esta contenido hasta una frecuencia de 300 Hz, tres veces más que la frecuencia fundamental de la cuadrada.

frecuencia =100 Hz

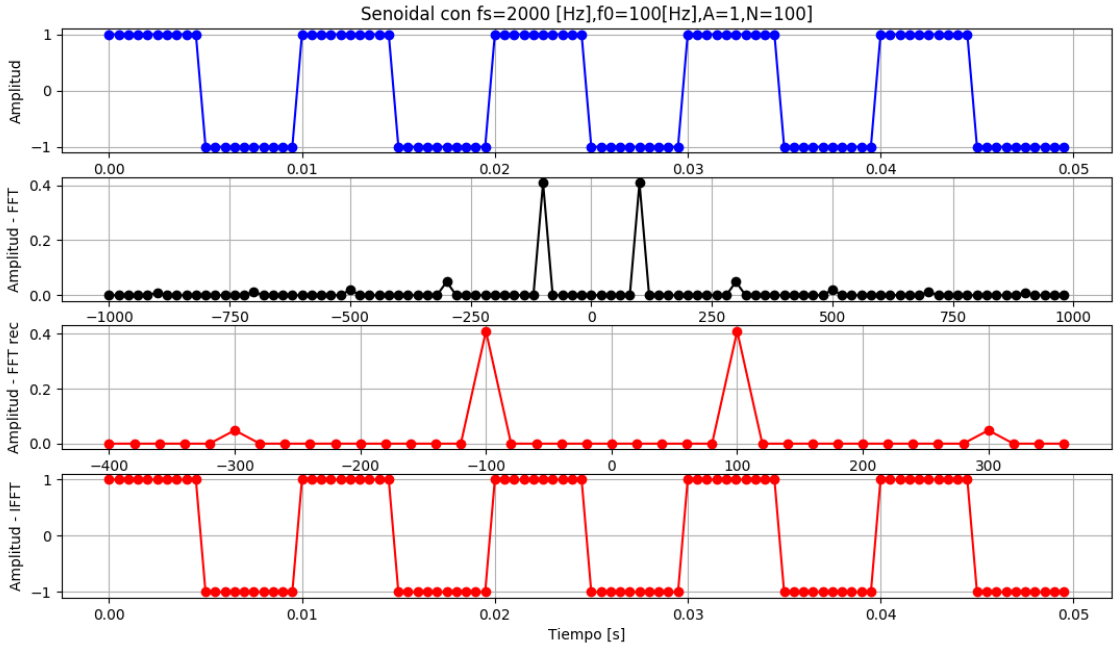
Amplitud =1

densidad espectral de potencia 20 Hz/N

Fs =2000 Hz

N =100

B=300 Hz



Triangular

La potencia de la señal en t es: 0.335

La potencia de la señal en f es: 0.290. Tome un 87% del espectro de potencia de la señal en frecuencia que se encuentra hasta una frecuencia de 350 Hz.

frecuencia =100 Hz

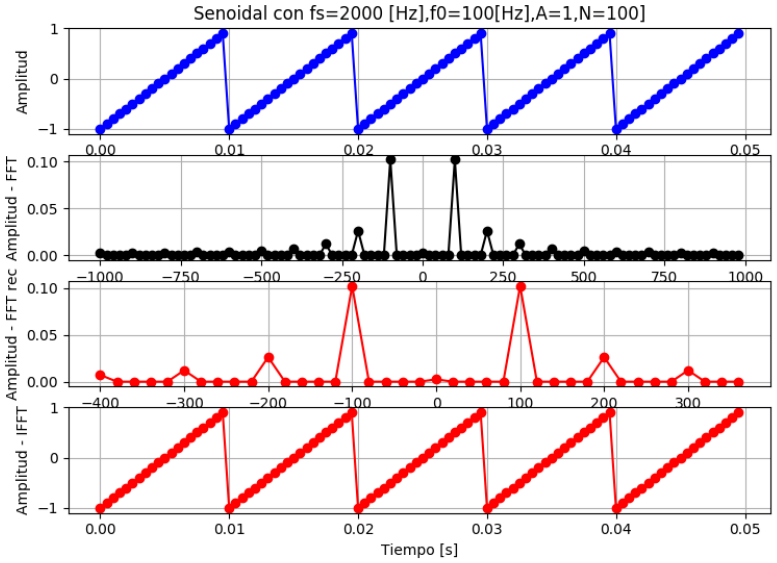
Amplitud =1

densidad espectral de potencia 20 Hz/N

Fs =2000 Hz

N =100

B=350 Hz



Impulso

La potencia de la señal en t es: 0.01

La potencia de la señal en f es: 0.0038. El impulso es un poco mas complicado porque el espectro se encuentra uniformente distribuido en frecuencia. Si se toma 350 Hz del impulso unitario, solamente tenemos un 30% del espectro de potencia de señal. Tomando mas ancho el espectro de frecuencia se llega al caso descripto en impulso 2.

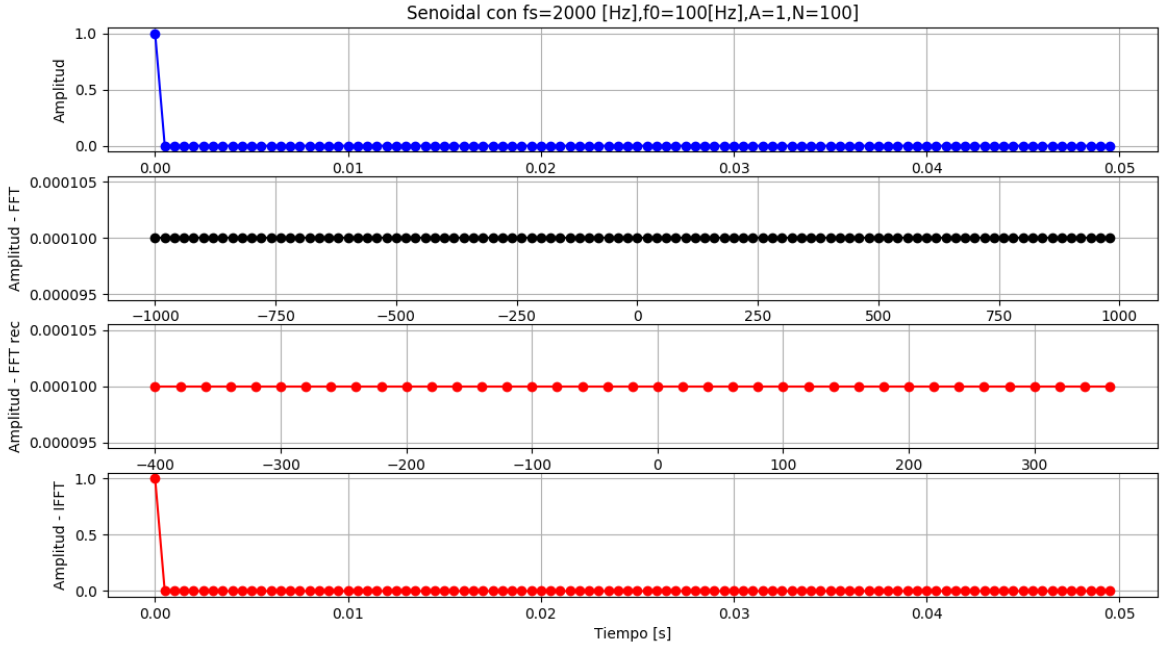
Amplitud =1

densidad espectral de potencia 20 Hz/N

Fs =2000 Hz

N =100

B=350 Hz



Impulso caso 2

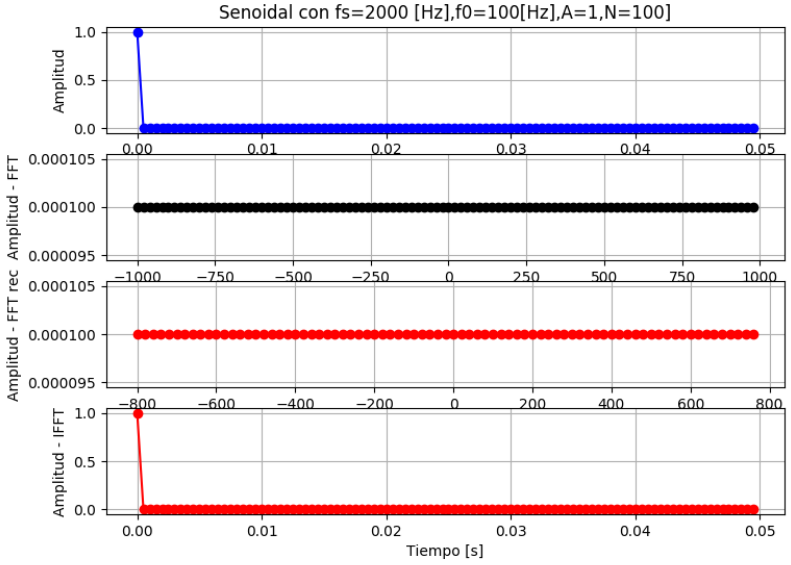
La potencia de la señal en t es: 0.01

La potencia de la señal en f es: 0.0079. Acá tome 750 Hz de ancho de banda del impulso unitario y llegó a obtener un 78% del espectro de la señal original lo cual consifero suficiente para limitar el ancho de banda.

Amplitud =1, densidad espectral de potencia 20 Hz/N, Fs =2000 Hz

N =100

B=750 Hz



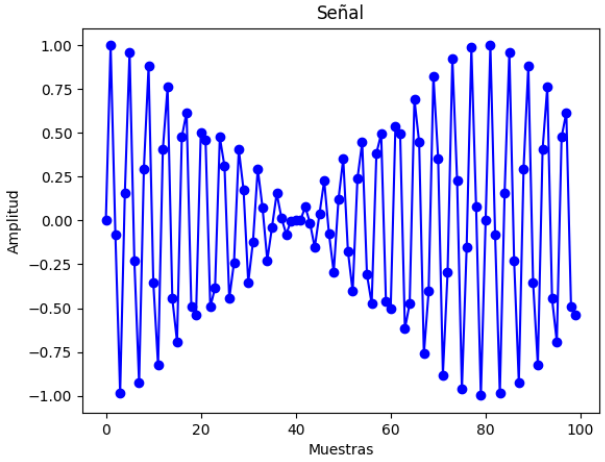
2. Dada la siguiente secuencia de números con N=100 y Fs=200, indique:

• Resolución espectral

• Obtenga el contenido espectral

• Que técnica conoce para mejorar la resolución en frecuencia?

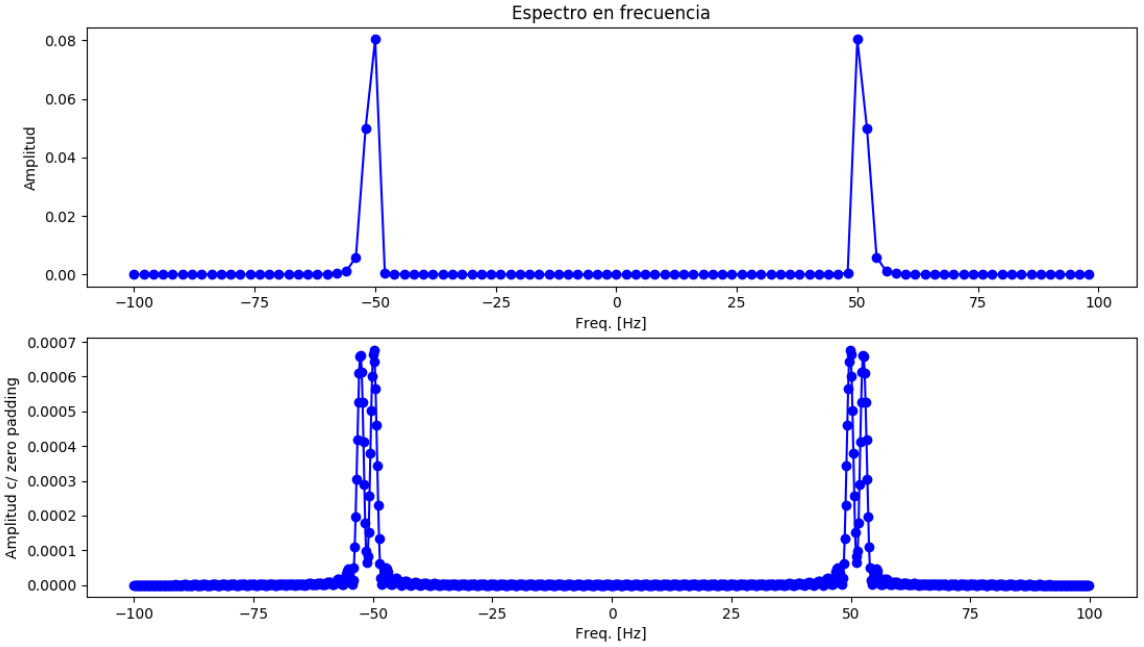
• Aplique la técnica, grafique y comente los resultados



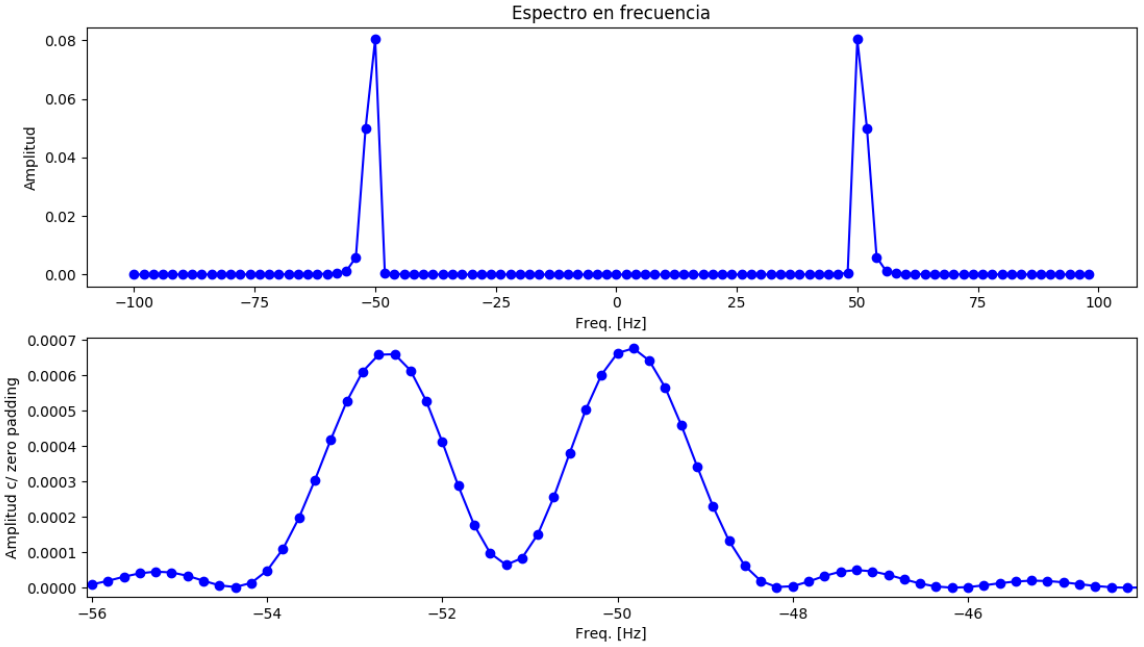
Señal en función de la cantidad total de muestras

Espectro con zero padding. Está técnica consiste en ampliar el vector con valores nulos, con ello se logra aumentar la resolución espectral. Se debe tener en cuenta que a medida que se adicionan ceros y se mejora la resolución la amplitud de la señal en frecuencia disminuye. Para mantener la amplitud constante se debería multiplicar por un factor constante.

Espectro con zero padding. Se agregaron 500 valores a ambos lados de la señal. En el grafico debajo se visualiza el espectro original en la parte superior sin aplicar ninguna técnica de mejora en su resolución. En el espectro 2) se agregar 500 puntos a ambos lados de la señal totalizando 1000 puntos. Se concluye que la mejora en la resolución espectral permite visualizar dos frecuencias próximas.



Se visualiza en detalle el espectro al que se le aplico el método de zero padding y se puede ver en detalle dos frecuencias próximas alrededor de 53 y 49 Hz.



En general se aprecia como el agregado de ceros mejora la resolución a costa de disminuir la amplitud del espectro. Una aplicación en los sistemas embebidos podría ser en el caso de tener una señal muy larga que se analizará por tramos, se podría generar tramos de valores cercanos a una potencia de 2, para el calculo de la FFT y la diferencia de valores completarlos con ceros. Ejemplo: tomo 200 valores de una señal y relleno los 55 restantes con ceros.

3.Dado el siguiente espectro extraído del archivo fft_hjs.npy, indique:

• Que cree que representa esta señal? tip: grafique en 2d la idft

• Hasta que punto podría limitar el ancho de banda y que se siga interpretando su significado

• Grafique para mostrar los resultados

En la figura 3a se muestra las componentes del espectro en real e imaginario superpuestas en función de la frecuencia en colores rojo y azul

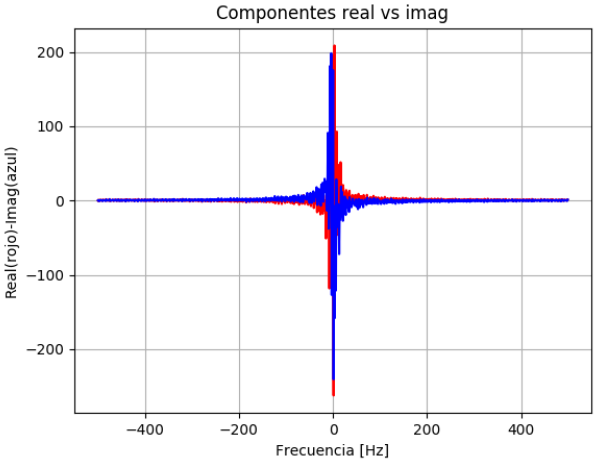


Figura 3a: Visualización del espectro por componentes.

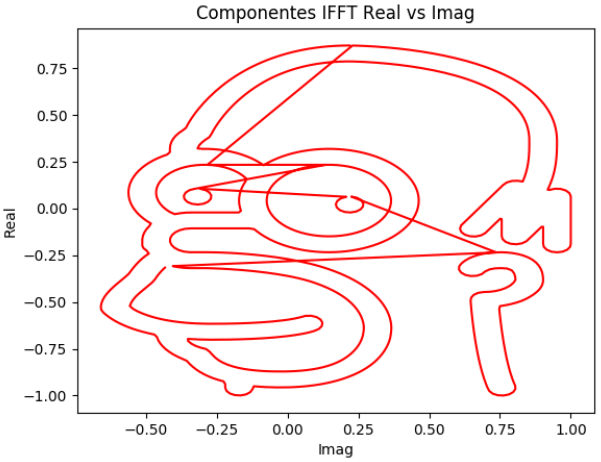


Figura 3b: Visualización de las componentes de la IFFT.

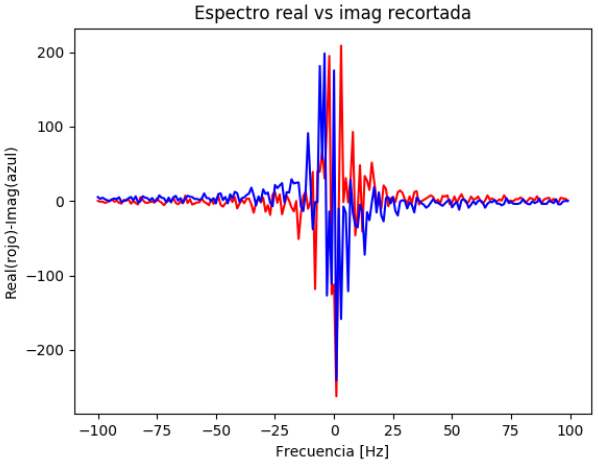


Figura 3c: Visualización del espectro por componentes recortado en puntos en ambos extremos. Se le recorto 100 puntos cada punto equivale a 1 Hz.

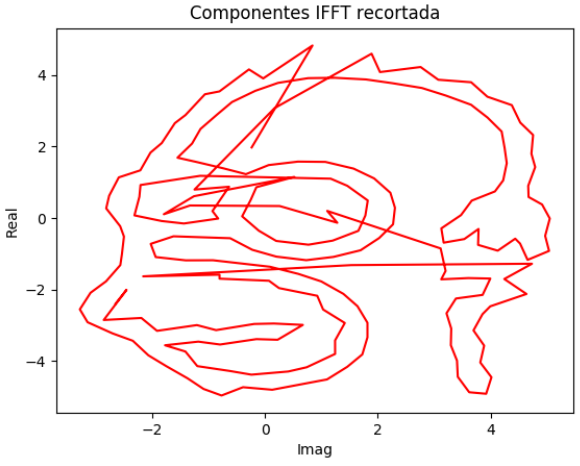


Figura 3d: Reconstrucción de la IFFT a partir del espectro recortado de las señales en 3c.

4 Dado el segmento de audio en el archivo chapu_noise.npy con fs=8000 y sumergido en ruido de alta frecuencia

resuelva:

• Diseñe un filtro que mitigue el efecto del ruido

• Grafique el espectro antes y después del filtro

• Reproduzca el segmento antes y después del filtrado

• Comente los resultados obtenidos

El diseño del filtro es pasabajos con respuesta finita al impulso (FIR) con la opción de orden mínimo ### quedo en 402, con atenuación en la banda de paso de 0,3 dB y en la banda de supresión de 60 dB. En ### frecuencia el filtro tiene una banda de transición entre 1600 Hz y 1650 Hz con un fs=8 KHz.

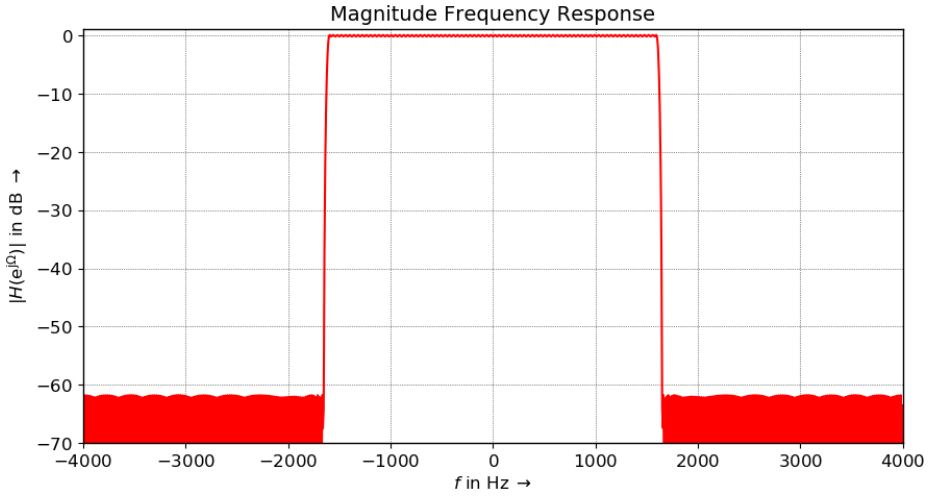


Figura 4a: Filtro pasabajos

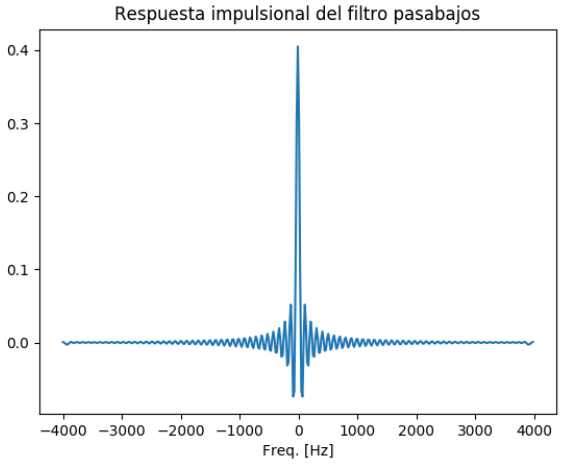


Figura 4b: Respuesta impulsional de un filtro pasabajos de mínimo orden

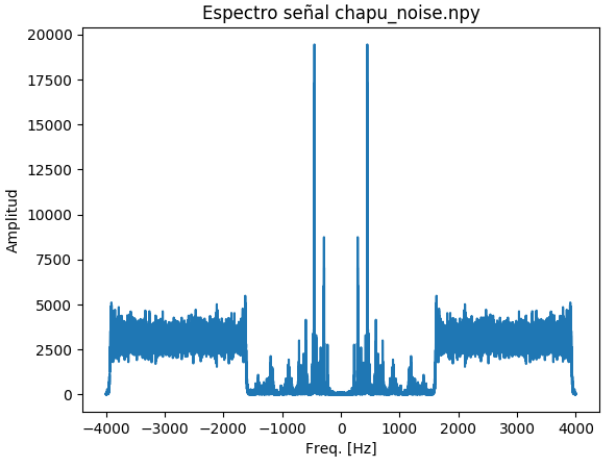


Figura 4c: Espectro de la señal con ruido incluído

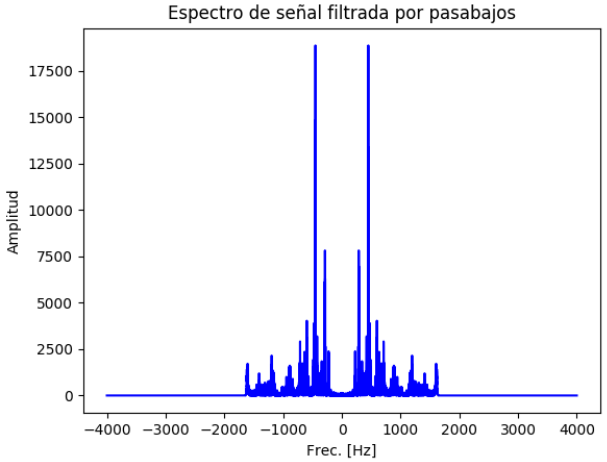


Figura 4d: Espectro de la señal luego de aplicarse el filtro pasabajos

Algunos comentarios: Para la realización del filtrado se utilizo la propiedad de convolución en tiempo. La función utilizada por el paquete Scipy de Python fue numpy.convolve con la opción full que da un rango de salida en N+M-1 donde N es el largo de la señal y M el largo de la respuesta impulsional del filtro.

En los dos ejemplos de abajo se muestar la señal original con y sin el ruido de fondo

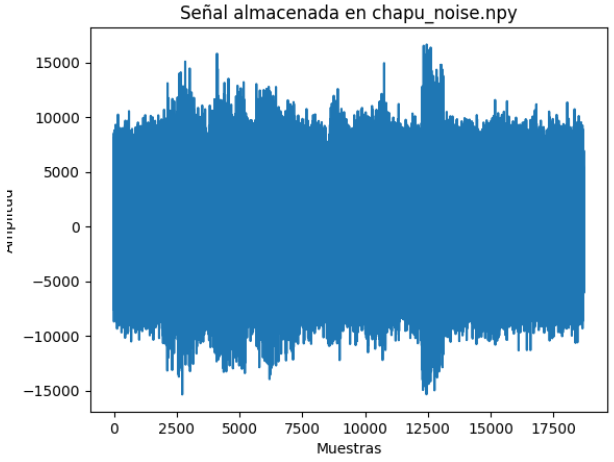
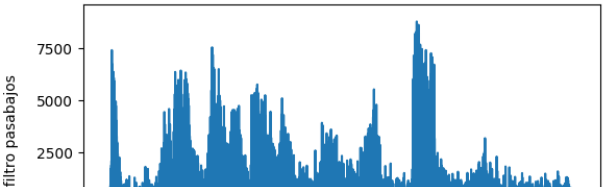


Figura 4e: Señal original con ruido



In [ ]:
```
#CÓDIGO UTILIZADO PARA EL EJERCICIO 1 Y PARA LOS DEMÁS EJERCICIOS

import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sc
from cmath import exp, pi


def senoidal(fs,f0,A,N,fase):
    # fs [Hz]
    # f0 [Hz]
    # A [a-1]
    # N {muestras}
    # fase [radianes]
    t=np.arange(0,N/fs,1/fs)
    y=A*np.sin(2 * np.pi * f0 * t + fase) #sin
    return t,y

def square(fs,f0,A,N,fase):
    # fs [Hz]
    # f0 [Hz]
    # A [a-1]
    # N {muestras}
    # fase [radianes]
    n = np.arange(0, N/fs, 1/fs)
    y=A*sc.square(2 * np.pi * f0 * n+fase,duty=0.5)
    return n,y

def sawtooth(fs,f0,A,N,fase):
    # fs [Hz]
    # f0 [Hz]
    # A [a-1]
    # N {muestras}
    # fase [radianes]
    n = np.arange(0, N/fs, 1/fs)
    y=A*sc.sawtooth(2 * np.pi * f0 * n + fase)
    return n,y

def delta(fs,A,N):
    # fs [Hz]
    # f0 [Hz]
    # A [a-1]
    # N {muestras}
    # fase [radianes]
    t=np.arange(0,N/fs,1/fs)
    y=A*sc.unit_impulse(len(t))
    return t,y

# The average power of a signal x is defined the energy per sample
#return a vector from -fs/2 to f2/2 -> -N/2 to N/2
def dft (xn):
    N=len(xn)
    ck=np.zeros(N,complex)

    for k in range(N):
        for i in range(N):
            ck[k]+=xn[i]*exp((-2j*pi*k*i)/N)

    #X = np.concatenate((ck[N//2:N],ck[0:N//2]))/N
    #(np.abs(X)**2) # return the average power X applying the Parseval theorem 1/N² Sum |Xk|¹
    return ck

def ifft (Xk):
    N=len(Xk)
    ci=np.zeros(N,complex)

    for i in range(N):
        for k in range(N):
            ci[i]+=Xk[k]*exp((2j*pi*k*i)/N)
    return ci/N

def Pk(ck):
    N=len(ck)
    X = np.concatenate((ck[N//2:N],ck[0:N//2]))/N
    return (np.abs(X)**2) # return the average power X applying the Parseval theorem 1/N² Sum |Xk|¹

def pn(xn):
    N=len(xn)
    return (np.sum(xn**2,dtype=float))/N

def Pk_sum(ck):
    N=len(ck)
    X = np.concatenate((ck[N//2:N],ck[0:N//2]))/N
    return (np.sum(np.abs(X)**2,dtype=float)) # return the average power X applying the Parseval theorem 1/N² Sum |Xk|¹

def Pk_sum_lim(ck,lim):
    Nk=len(ck)
    #X = np.concatenate((ck[N//2:N],ck[0:N//2]))/N
    X = np.concatenate((ck[Nk//2:Nk],ck[0:Nk//2]))
    X=X[lim:-1-lim]
    X=X/Nk
    return (np.sum(np.abs(X)**2,dtype=float)) # return the average power X applying the Parseval theorem 1/N² Sum |Xk|¹

#-------------------------------------
fig        = plt.figure()
fs         = 2000
N          = 100
f0 = 100
A=1

lim=10     #recorto espectro para calcular la potencia de la señal
#-------------------------------------
tData      = np.arange(0,N/fs,1/fs)
nData      = np.arange(0,N,1)
circleFrec = np.arange(-fs/2,fs/2,fs/N)

#------------SIGNAL-------------------------
#tData,signalData=senoidal(fs,f0,A,N,0)
#tData,signalData=square(fs,f0,A,N,0)
#tData,signalData=sawtooth(fs,f0,A,N,0)
tData,signalData=delta(fs,A,N)

potenciaSig_t=pn(signalData)
print('La potencia de la señal en t es:')
print(potenciaSig_t)

#------------FFT IFFT-----------------------
```

```
fftSignal=dft(signalData)
fftData  = Pk(fftSignal)

#Calculo la IFFT pero pero a partir del espectro recortado
Sig=ifft (fftSignal)


potSigF=Pk_sum_lim(fftSignal,lim)
print('La potencia de la señal en f es:')
print(potSigF)
#----------------Señal-----------------------
signalAxe  = fig.add_subplot(4,1,1)
ax=plt.subplot(411)
plt.plot(tData,signalData,'bo-')
signalAxe.grid(True)
ax.set(xlabel='Tiempo [s]', ylabel='Amplitud',title='Senoidal con fs=2000 [Hz],f0=100[Hz],A=1,N=100]')

#----------grafico FFT-----------------------
signalAxe  = fig.add_subplot(4,1,2)
ax=plt.subplot(412)
plt.plot(circleFrec,fftData,'ko-')
signalAxe.grid(True)
ax.set(xlabel='N', ylabel='Amplitud - FFT')

#----------grafico FFT recortado -------------
signalAxe  = fig.add_subplot(4,1,3)
ax=plt.subplot(413)
plt.plot(circleFrec[lim:-1-lim],fftData[lim:-1-lim],'ro-')
signalAxe.grid(True)
ax.set(xlabel='N', ylabel='Amplitud - FFT rec')

#----------señal IFFT reconstruida------------
signalAxe  = fig.add_subplot(4,1,4)
ax=plt.subplot(414)
plt.plot(tData,Sig,'ro-')
signalAxe.grid(True)
ax.set(xlabel='Tiempo [s]', ylabel='Amplitud - IFFT',)
plt.show()
```