Práctica 1 Señales

APELLIDOS, NOMBRE: _____

Objetivo

El objetivo de este taller es la creación de sonidos utilizando principios básicos de síntesis aditiva, donde una señal es generada como la suma tonos puros y armónicos. Para crear la señal, parta del siguiente código que permite generar la señal, representarla y escucharla.

```
import matplotlib.pyplot as plt
import numpy as np
import scipy.io.wavfile
import sounddevice as sd

"================================================"
" Senal coseno"
valini = 0 # Valor inicial
fs = 44100  # Hz
duracion = 44100  #en muestras
n =  np.arange(valini,duracion)/fs


"================================================"
" Creación de coseno "
f = 200

xcos = np.cos(2 * np.pi * f * n)

plt.xlabel('tiempo')
plt.ylabel('amplitud')
plt.title('Señal coseno')
plt.stem(n[1:100], xcos[1:100])
plt.show()

#sd.play(xcos, fs)   #escucha la señal si tienes cascos
```

EJERCICIO 1:

Para generar un sonido con el método de síntesis aditiva, cree una señal que sea la suma de una señal principal y Na armónicos, es decir

$$x(n) = \cos cos\ (2\pi fn)\ + \sum_{k=2}^{Na} cos(2\pi\, kfn)$$

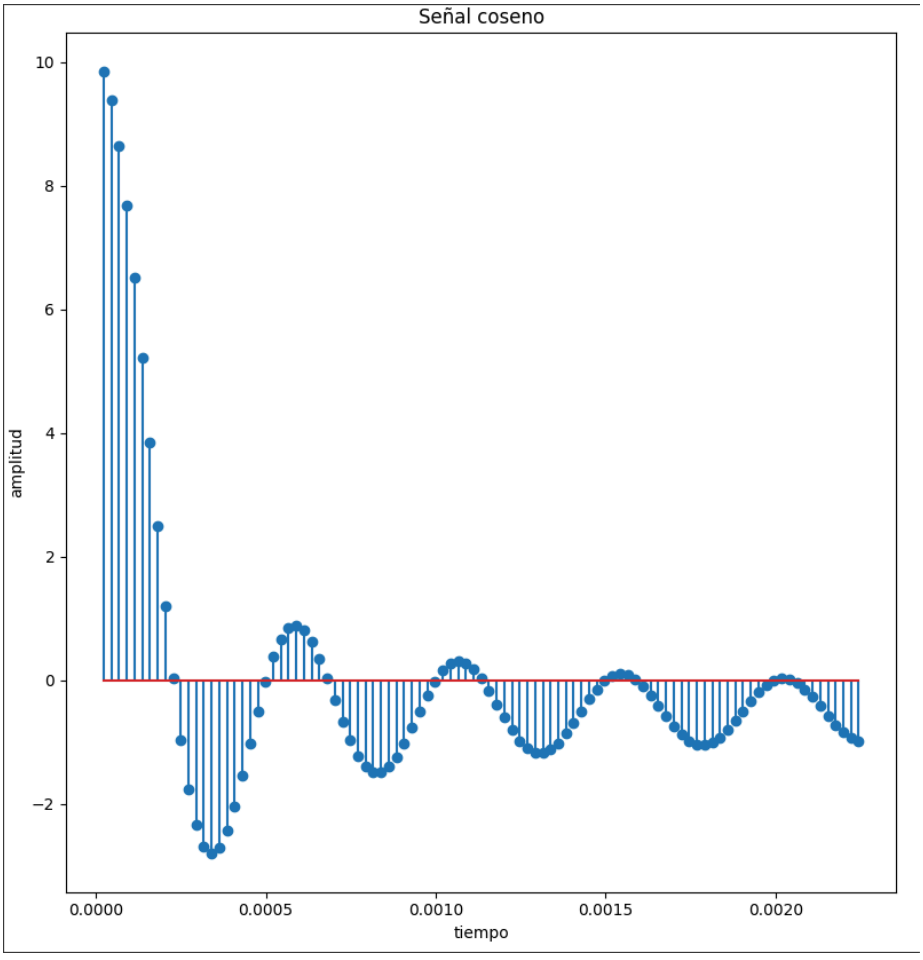
IMPORTANTE: Fíjese que el tono principal tiene frecuencia f (en Hz), y $k = 2, \ldots, Na$ da los armónicos con frecuencia kf (en Hz).

Pruebe con f = 200 Na =10 fs = 44100 y escuche el sonido

¿Cuál es la duración en segundos para fs = 44100 Hz y duración 44100 muestras?

La duración la calculadora con la duración en muestras entre la frecuencia, dando 1 segundo

Copie aquí la figura que obtiene.

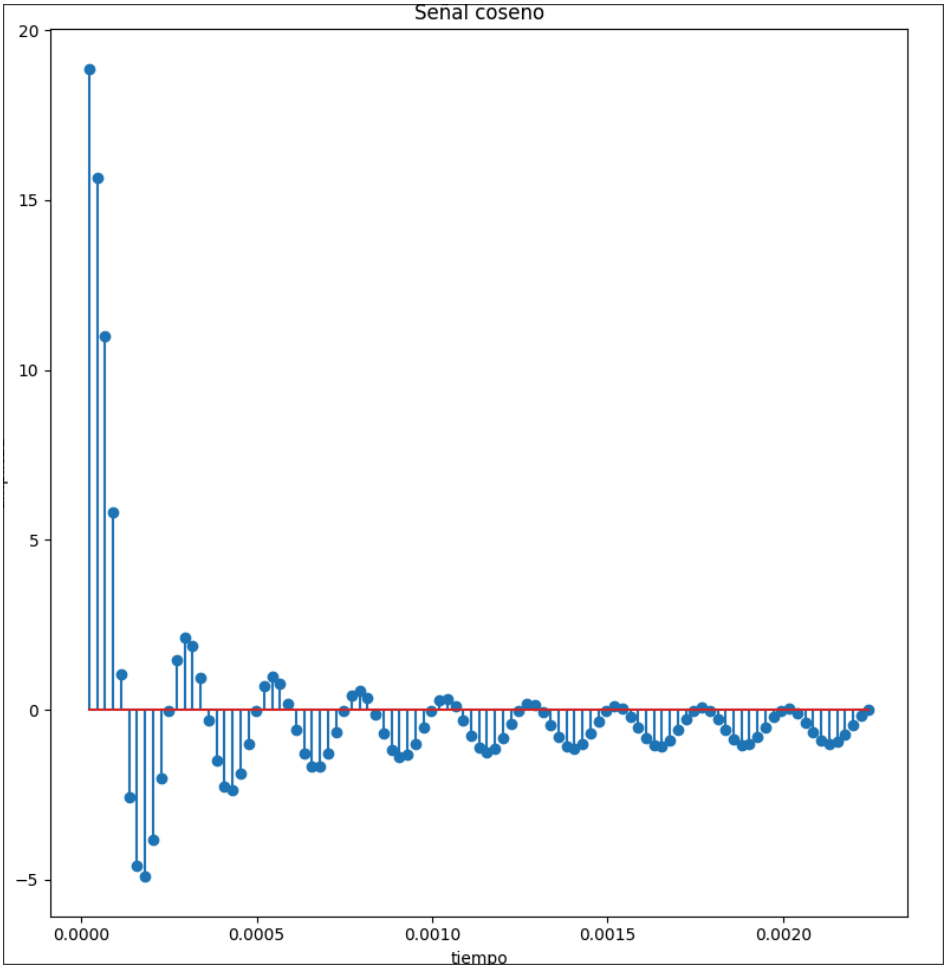


¿Qué opina del sonido?

El sonido resultante es más agudo y estridente

Pruebe con f = 200 Na =20 fs = 44100 y escuche el sonido

Copie aquí la figura que obtiene.



¿Qué opina del sonido?

El sonido resultante es aún más agudo ya que estamos aumentando la cantidad de armónicos sumados a la señal principal

```
import matplotlib.pyplot as plt
import numpy as np
import scipy.io.wavfile
import sounddevice as sd

"================================================"
" Senal coseno"
valini = 0 # Valor inicial
fs = 44100  # Hz
duracion = 44100  # en muestras
n =  np.arange(valini,duracion)/fs


"================================================"
" Creación de coseno "
f = 200
Na = 20

xcos = np.cos(2 * np.pi * f * n)
# rango de 2 hasta Na + 1 porque el rango es excluyente en el limite superior
en python
for k in range(2, Na + 1):
    xcos += np.cos(2 * np.pi * k * f * n)

plt.xlabel('tiempo')
plt.ylabel('amplitud')
plt.title('Señal coseno')
plt.stem(n[1:100], xcos[1:100])
plt.show()

duracion_seg = duracion / fs
print(f"Duración: {duracion_seg:.2f} segundos")

sd.play(xcos, fs)   #escucha la señal si tienes cascos
```

EJERCICIO 2:

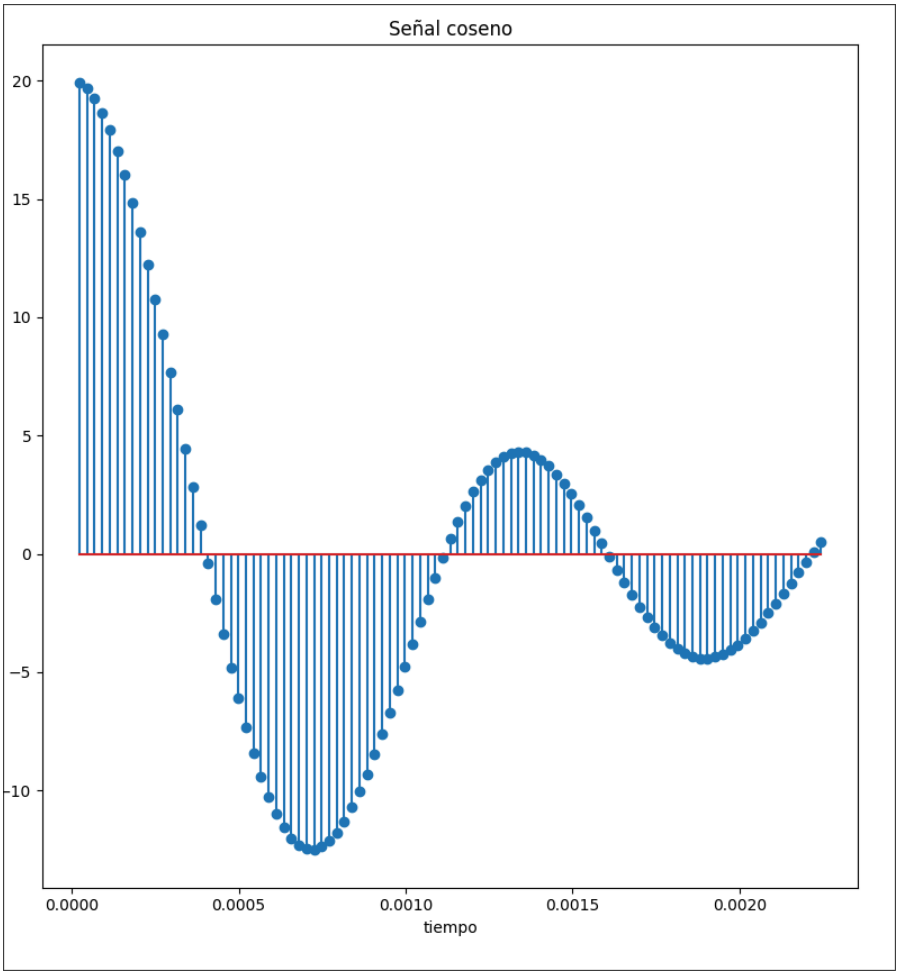
Repita lo anterior considerando otras frecuencias de los armónicos:

$$x(n) = \cos cos\,(2\pi fn) + \sum_{k=2}^{Na} cos(2\pi\sqrt[p]{k}fn)$$

En este caso el tono principal tiene frecuencia f, y los armónicos tienen frecuencia $\sqrt[p]{k}f$. Para hacer la potencia, en Python, puede utilizar k ** (1/p)

Pruebe con f = 200 Na =20 fs = 44100 p=2

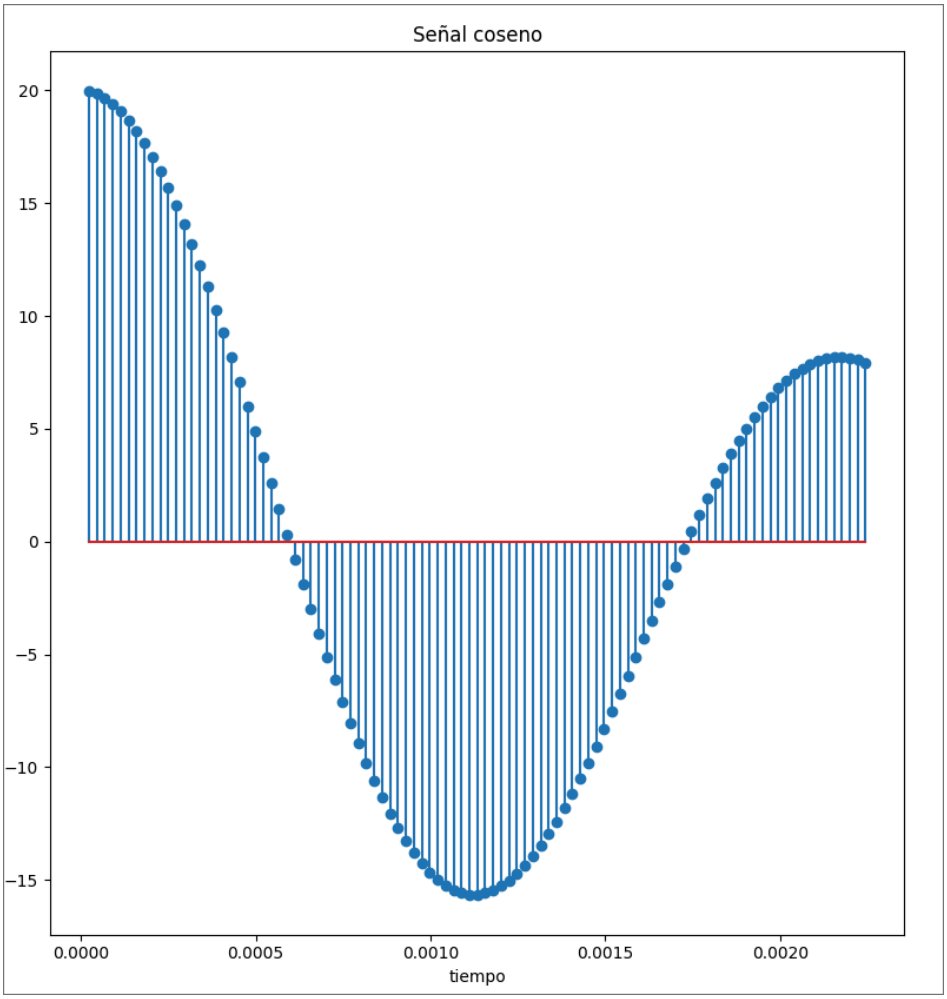
Copie aquí la figura que obtiene



¿Qué opina del sonido?
Es

Pruebe con f = 200 Na =20 fs = 44100 p=3

Copie aquí la figura que obtiene



¿Qué opina del sonido?
Es más grave que el anterior

```python
import matplotlib.pyplot as plt
import numpy as np
import scipy.io.wavfile
import sounddevice as sd

"================================================"
" Senal coseno"
valini = 0 # Valor inicial
fs = 44100  # Hz
duracion = 44100  # en muestras
n =  np.arange(valini,duracion)/fs


"================================================"
" Creación de coseno "
f = 200
Na = 20
p = 3

xcos = np.cos(2 * np.pi * f * n)
# rango de 2 hasta Na + 1 porque el rango es excluyente en el limite superior en python
for k in range(2, Na + 1):
    xcos += np.cos(2 * np.pi * k**(1/p) * f * n)

plt.xlabel('tiempo')
plt.ylabel('amplitud')
plt.title('Señal coseno')
plt.stem(n[1:100], xcos[1:100])
plt.show()

duracion_seg = duracion / fs
print(f"Duración: {duracion_seg:.2f} segundos")

sd.play(xcos, fs)   #escucha la señal si tienes cascos
```

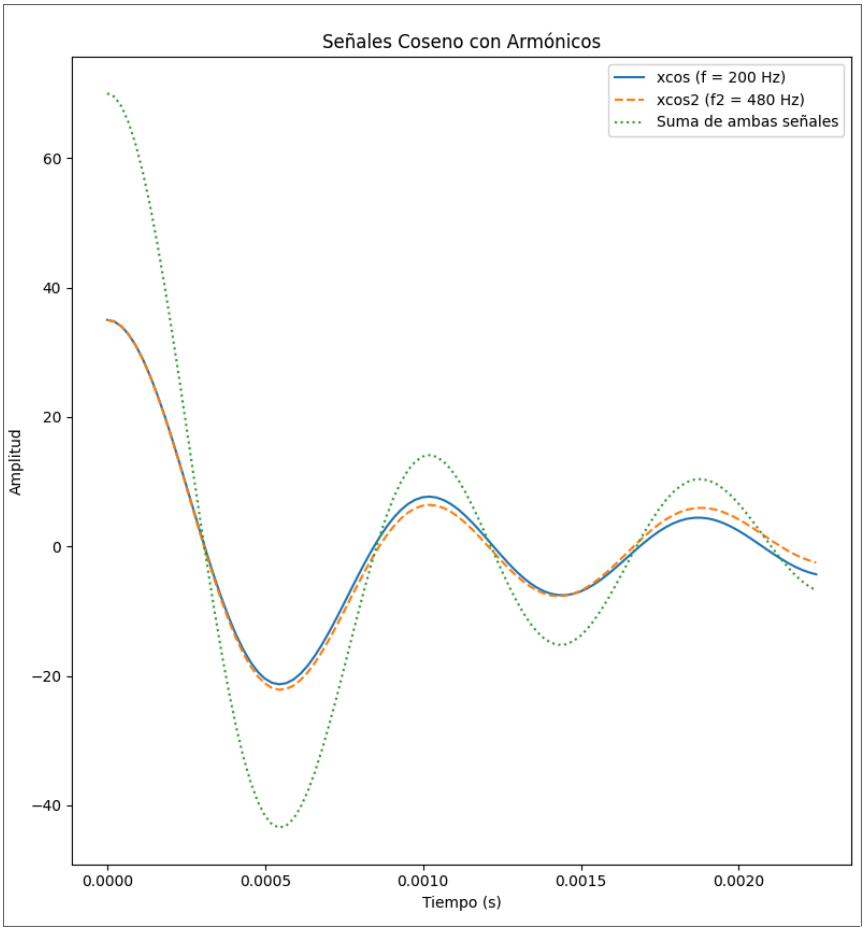
EJERCICIO 3:

En la época de los 80's, se utilizó el método anterior para generar sonidos para cine, televisión, etc. Así que vamos a intentar crear un sonido más agradable para nuestros oídos.

Realice lo siguiente:

1. Genere dos señales que tengan un tono simple (por ejemplo, f1 = 200 Hz y f2 = 480 Hz) y 35 armónicos obtenidos utilizando $\sqrt{k}$.
2. Sume ambas señales para obtener un sonido compuesto.
3. Represéntelo y escúchelo.

Copie aquí la figura.



¿Qué opina del sonido?

Suena parecido a los otros sonidos, pero más "burbujeante"

```python
import matplotlib.pyplot as plt
import numpy as np
import sounddevice as sd

"================================================"
" Senal coseno"
valini = 0 # Valor inicial
fs = 44100  # Hz
duracion = 44100  # en muestras
n =  np.arange(valini,duracion)/fs


"================================================"
" Creación de coseno "
f = 200
Na = 35
p = 3
f2 = 480

xcos = np.cos(2 * np.pi * f * n)
xcos2 = np.cos(2 * np.pi * f2 * n)

# rango de 2 hasta Na + 1 porque el rango es excluyente en el limite superior en python
for k in range(2, Na + 1):
    xcos += np.cos(2 * np.pi * k**0.5 * f * n)
    xcos2 += np.cos(2 * np.pi * k**0.5 * f * n)

xcos_suma = xcos + xcos2

plt.figure(figsize=(10, 5))

plt.plot(n[:100], xcos[:100], label=f"xcos (f = {f} Hz)")
plt.plot(n[:100], xcos2[:100], label=f"xcos2 (f2 = {f2} Hz)", linestyle="dashed")
plt.plot(n[:100], xcos_suma[:100], label="Suma de ambas señales", linestyle="dotted")

plt.xlabel("Tiempo (s)")
plt.ylabel("Amplitud")
plt.title("Señales Coseno con Armónicos")
plt.legend()
plt.show()

sd.play(xcos_suma, fs)   #escucha la señal si tienes cascos
```