**Problem 1B**

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Chart

Description automatically generated

Chart, scatter chart

Description automatically generated

#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 1, Part B

import numpy as np

import matplotlib.pyplot as plt

numPosCoeffs = 50

totalNumCoeffs = 2\*numPosCoeffs + 1

omega0 = np.pi/2.0

c0 = 1j\*omega0

cneg = -1j\*omega0

coeffs = np.zeros(totalNumCoeffs, dtype=complex)

harmNum = np.arange(-numPosCoeffs, numPosCoeffs + 1)

for k in harmNum:

index = k + numPosCoeffs

if k != 0:

expFactor = np.exp((c0\*k))

expFactorNeg = np.exp(cneg\*k)

coeffs[index] = 2.0/(c0\*k) \* expFactor - np.multiply(1/(c0\*k),1/(c0\*k)) \* (1-expFactor) - 3\*(1/(c0\*k) \* expFactorNeg) + 2 \* np.multiply(1/(c0\*k),1/(c0\*k)) \* (expFactorNeg - 1)

if k == 0:

coeffs[index] = 7.0/8.0 + 0.0\*1j

minIdx = -20 + numPosCoeffs

maxIdx = 20 + numPosCoeffs + 1

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(harmNum[minIdx:maxIdx], np.real(coeffs[minIdx:maxIdx]))

ax.set\_title("Real Part of $c\_{x}[k]$")

ax.set\_xlabel("Harmonic Number $k$")

ax.set\_ylabel("$Re\\{c\_{x}[k]\\}$")

ax.grid("True")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(harmNum[minIdx:maxIdx], np.imag(coeffs[minIdx:maxIdx]))

ax.set\_title("Imaginary Part of $c\_{x}[k]$")

ax.set\_xlabel("Harmonic Number $k$")

ax.set\_ylabel("$Im\\{c\_{x}[k]\\}$")

ax.grid("True")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(harmNum[minIdx:maxIdx], np.abs(coeffs[minIdx:maxIdx]))

ax.set\_title("Magnitude of $c\_{x}[k]$")

ax.set\_xlabel("Harmonic Number $k$")

ax.set\_ylabel("$|c\_{x}[k]|$")

ax.grid("True")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(harmNum[minIdx:maxIdx], np.angle(coeffs[minIdx:maxIdx]))

ax.set\_title("Phase of $c\_{x}[k]$")

ax.set\_xlabel("Harmonic Number $k$")

ax.set\_ylabel("$\\angle c\_{x}[k]$")

ax.grid("True")

plt.show()

**Problem 1C**

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

import numpy as np

import matplotlib.pyplot as plt

# Calculate the CTFS coefficients.

NCoeffs = 50

totalNCoeffs = 2\*NCoeffs + 1

omega0 = np.pi/2.0

coeffs = np.zeros(totalNCoeffs, dtype=complex)

harmonicNum = np.arange(-NCoeffs, NCoeffs + 1)

for k in harmonicNum:

index = k + NCoeffs

if k != 0:

expFactor = np.exp(1j\*omega0\*k)

expFactor2 = np.exp(-1j\*omega0\*k)

coeffs[index] = 2.0\*expFactor2/(k\*\*2\*np.pi\*\*2)+expFactor/(k\*\*2\*np.pi\*\*2)-3/(k\*\*2\*np.pi\*\*2)+3.0\*1j\*expFactor2/(2.0\*k\*np.pi)-1j\*expFactor/(k\*np.pi)

if k == 0:

coeffs[index] = 7.0/8.0

# Calculate the values of the original signal.

tx = np.linspace(0.0, 1.0, 400)

tx1 = np.linspace(-1.0, 0.0, 400)

oneVec = np.ones(np.size(tx))

oneVec = np.ones(np.size(tx1))

x = 2.0\*tx + 1.0\*oneVec

x = 1.0\*oneVec - tx1

# Calculate the values of the partial sums.

t = np.linspace(-2.0, 2.0, 500)

omega0jt = (1j\*np.pi/2.0)\*t

print("")

for N in [1]: #manually adjust for N values we need

xN = np.zeros(np.size(t), dtype=complex)

for k in np.arange(-N, N + 1):

xN += coeffs[k + NCoeffs]\*np.exp(k\*omega0jt)

print("The maximum absolute value of the imaginary part of x{0}(t) is {1}."\

.format(N, np.max(np.abs(np.imag(xN)))))

# Plot the Nth partial sum.

fig, ax = plt.subplots(1, figsize=(12,4))

ax.plot(t, np.real(xN)) # The imaginary part should be 0.

ax.set\_title("Partial sum $x\_N(t)$ of CTFS for $x(t)$ when N = {0}".format(N))

ax.set\_ylim([-1.0, 4.0])

ax.set\_xlabel("Time t")

ax.set\_ylabel("$x\_{N}(t)$")

# Plot the original signal

ax.plot([-1.5, -1.0], [0.0, 0.0], color="tab:red")

ax.plot([-1.0, -1.0], [0.0, 2.0 ],color="tab:red")

ax.plot([-1.0, 0.0], [2.0, 1.0 ], color="tab:red")

ax.plot([-1.0 , 0.0], [2.0, 1.0], color="tab:red")

ax.plot([0.0, 1.0], [1.0, 3.0], color="tab:red")

ax.plot([1.0, 1.0], [0.0, 3.0], color="tab:red")

ax.plot([1.0 , 2.0], [0.0, 0.0], color="tab:red")

plt.show()

**Problem 3B**

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart

Description automatically generated

Chart, line chart

Description automatically generated

#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 3, Part B

import numpy as np

import matplotlib.pyplot as plt

# F{x(t)} = 6/(j\*omega + 1)^4

omega = np.linspace(-8.0, 8.0, 2000)

oneVec = np.ones(np.size(omega))

factor = 1.0 + 1j\*omega

X = np.divide(6.0\*oneVec,np.multiply(np.multiply(factor,factor),np.multiply(factor,factor)))

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.real(X))

ax.set\_title("Real Part of $X(j\\omega)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$Re\\{X(j\\omega)\\}$")

plt.savefig("prob3b\_real\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.imag(X))

ax.set\_title("Imaginary Part of $X(j\\omega)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$Im\\{X(j\\omega)\\}$")

plt.savefig("prob3b\_imag\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.abs(X))

ax.set\_title("Magnitude of $X(j\\omega)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$|X(j\\omega)|$")

plt.savefig("prob3b\_mag\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.angle(X))

ax.set\_title("Phase of $X(j\\omega)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$\\angle X(j\\omega)$")

plt.savefig("prob3b\_phase\_python.png")

plt.show()

**Problem 3C**

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart

Description automatically generated

Chart, line chart

Description automatically generated

#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 3, Part C

import numpy as np

import matplotlib.pyplot as plt

from scipy.fft import fft, fftshift

N = 262144

T = 100.0

Ts = T/N # sample period

fs = 1.0/Ts # sample rate

df = fs/N # = 1/T = frequency increment

n = np.arange(0,N)

t = Ts\*n

x = np.multiply(t\*t\*t,np.exp(-t))

X = fftshift(Ts\*fft(x))

k = np.arange(-N/2,N/2)

omega = 2.0\*np.pi\*df\*k

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.real(X))

ax.set\_xlim([-8.75, 8.75])

ax.set\_title("Real Part of the DFT of $x(t)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$Re\\{X(j\\omega)\\}$")

plt.savefig("prob3c\_real\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.imag(X))

ax.set\_xlim([-8.75, 8.75])

ax.set\_title("Imaginary Part of the DFT of $x(t)$")

ax.set\_xlabel("Angular Frequency \omega")

ax.set\_ylabel("$Im\\{X(j\omega)\\})$")

plt.savefig("prob3c\_imag\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.abs(X))

ax.set\_xlim([-8.75, 8.75])

ax.set\_title("Magnitude of the DFT of $x(t)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$|X(j\\omega)|$")

plt.savefig("prob3c\_mag\_python.png")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(omega, np.angle(X))

ax.set\_xlim([-8.75, 8.75])

ax.set\_title("Phase of the DFT of $x(t)$")

ax.set\_xlabel("Angular Frequency $\\omega$")

ax.set\_ylabel("$\\angle X(j\\omega)$")

plt.savefig("prob3c\_phase\_python.png")

plt.show()

**Problem 5A**

Chart

Description automatically generated

Chart, box and whisker chart

Description automatically generatedChart

Description automatically generated

Chart

Description automatically generated

#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 5, Part A

import numpy as np

import matplotlib.pyplot as plt

Omega0 = (2.0\*np.pi)/5.0 # fundamental angular frequency of x[n]

k = np.arange(-10, 10) # frequency indices

Omega0k = Omega0\*k # Defined to prevent this from being calculated twice.

cx =(1/5.0)\*(2.0 - np.exp(-1j\*Omega0k)+np.exp(2.0\*-1j\*Omega0k) - 2.0\*np.exp(3.0\*-1j\*Omega0k)-np.exp(4.0\*-1j\*Omega0k))

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(k, np.real(cx))

ax.set\_title("Four Periods of the Real Part of $c\_x[k]$")

ax.set\_xlabel("$k$")

ax.set\_ylabel("$Re\{c\_x[k]\}$")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(k, np.imag(cx))

ax.set\_title("Four Periods of the Imaginary Part of $c\_x[k]$")

ax.set\_xlabel("$k$")

ax.set\_ylabel("$Im\{c\_x[k]\}$")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(k, np.abs(cx))

ax.set\_title("Four Periods of $|c\_x[k]|$")

ax.set\_xlabel("$k$")

ax.set\_ylabel("$|c\_x[k]|$")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.stem(k, np.angle(cx))

ax.set\_title("Four Periods of the Phase of $c\_x[k]$")

ax.set\_xlabel("$k$")

ax.set\_ylabel("$\\angle c\_x[k]$")

plt.show()

**Problem 5B**

A picture containing chart

Description automatically generated

**The max absolute value of the imaginary part of x[n] is 0.0**  
  
#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 5, Part B

import numpy as np

import matplotlib.pyplot as plt

N0 = 5 # fundamental period of x[n]

Omega0 = 2.0\*np.pi/N0 # fundamental angular frequency of x[n]

k0 =-2 # initial frequency index

k = np.arange(k0, k0 + N0) # frequency indices

Omega0k = Omega0\*k # Defined to prevent this from being calculated three times.

cx =(1/5.0)\*(2.0 - np.exp(-1j\*Omega0k)+np.exp(2.0\*-1j\*Omega0k) - 2.0\*np.exp(3.0\*-1j\*Omega0k)-np.exp(4.0\*-1j\*Omega0k))

w = np.exp(-1j\*Omega0k)

x = np.zeros(N0, dtype=complex)

for n in np.arange(k0, k0 + N0):

wn = np.power(w, n)

idx = n - k0

x[idx] = np.dot(cx, wn)

print("The max absolute value of the imaginary part of x[n] is {0}"\

.format(np.max(np.abs(np.imag(x)))))

fig, ax = plt.subplots(1, figsize=(12,4))

ax.stem(k, np.real(x)) # the time indices n are the same as the freq indices k

ax.set\_title("One Period of the Signal $x[n]$ with DTFS Coefficients $c\_x[k]$")

ax.set\_xlabel("$n$")

ax.set\_ylabel("$x[n]$")

ax.grid(True)

plt.show()

**Problem 6**

Chart, line chart

Description automatically generated

Chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

#Phil Nevins

#ECE 315 Signals and Systems

#11/28/2022

#Problem 6, Part C

import numpy as np

import matplotlib.pyplot as plt

from scipy.fft import fft, fftshift

Omega = np.linspace(-np.pi, np.pi, 1000)

oneVec = np.ones(np.size(Omega))

X = 2.0\*oneVec - np.cos(Omega) - 3j\*np.sin(Omega)

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(Omega, np.abs(X))

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("$|X(e^{j\\Omega})|$")

ax.set\_title("Magnitude of the Fourier Transform of $x[n]$")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(Omega, np.angle(X))

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("$\\angle X(e^{j\\Omega})$")

ax.set\_title("Phase of the Fourier Transform of $x[n]$")

plt.show()

L = 1024

x = [-2.0,2.0,1.0]

z = np.zeros(L, dtype=complex)

for n in np.arange(0,3):

z[n] = x[n]

OmegaL = -np.pi + np.arange(0,L)\*(2.0\*np.pi/L)

xfft = fftshift(fft(z))

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(OmegaL, np.abs(xfft))

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("Magnitude of the FFT of $x$")

ax.set\_title("Magnitude of the FFT of $x$ vs $\\Omega$")

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

ax.plot(OmegaL, np.angle(xfft))

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("Phase of the FFT of $x$")

ax.set\_title("Phase of the FFT")

plt.show()

Y = np.multiply(np.exp(2j\*OmegaL),xfft)

fig, ax = plt.subplots(1, figsize=(12,3))

magX, = ax.plot(Omega, np.abs(X), label="$|X(e^{j\\Omega})|$", linestyle="-")

magY, = ax.plot(OmegaL, np.abs(Y), label="$|Y(e^{j\\Omega})|$", linestyle=":")

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("$|Y(e^{j\\Omega})|$")

ax.set\_title("Magnitude of the Frequency Shifted FFT")

ax.legend(handles=[magX, magY])

plt.show()

fig, ax = plt.subplots(1, figsize=(12,3))

angX, = ax.plot(Omega, np.angle(X), label="$\\angle X(e^{j\\Omega})$",\

linestyle="-")

angY, = ax.plot(OmegaL, np.angle(Y), label="$\\angle Y(e^{j\\Omega})$",\

linestyle=":")

ax.set\_ylim([-4, 4])

ax.set\_xlabel("Angular Frequency $\\Omega$")

ax.set\_ylabel("$\\angle Y(e^{j\\Omega})$")

ax.set\_title("Phase of the Frequency Shifted FFT")

ax.legend(handles=[angX, angY])

plt.show()