Michael Hickey

Professor Tom

Digital Signals Processing

Lab 9 Homemade Discrete Fourier Transform

July 19th, 2018

Code:

import numpy as np

import matplotlib.pyplot as plt

import time

from scipy.io import wavfile

times = []

def myOwnDFT(data):

# Start a clock timer to track how long the function takes.

start = time.time()

# The number of elements in signal u

N = len(data)

# Set up the for loop bounds

n = np.arange(0, N)

k = np.arange(0, N)

sum = 0

# The empty list, which will store the Fourier series.

Un = []

# Nesting for loops

# This Outer forloop dictates what array index we are working on.

for g in n:

# Reset the sum before it is appended.

sum = 0

# This forloop dictates what sum we input into the array.

for i in k:

# Perform Summation for value of n.

sum = sum + data[i]\*np.exp( (-2)\*np.pi\*1j\*i\*g/N )

# Append the result of the summation to the end of Un array.

Un = np.append(Un, sum)

Un = np.absolute(Un)

end = time.time()

print("Elapsed Time: " + str(end-start) )

# Store the elapsed time in an array.

times.append(end-start)

# Un is of the form np.array()

return Un

def makeSinWave(freq, freq\_length):

# Creating a list of frequency from 0 to freq\_length in increments of 1 Hz

t = np.linspace(0, 1, freq\_length)

# Create sin waves.

return np.sin(2\*np.pi\*freq\*t)

if \_\_name\_\_ == "\_\_main\_\_":

end\_freq = 1200

if 1 :

for i in range(200, end\_freq, 200):

# Make a sin wave

sin\_wave\_1 = makeSinWave(10, i)

# Get the length of the sin wave.

length\_of\_sin\_myDFT = np.arange(len(sin\_wave\_1))

# Jam the sin wave into my DFT function.

sin\_myDFT\_1 = myOwnDFT(sin\_wave\_1)

# Start Plotting

fig = plt.figure()

fig.suptitle('myDFT of Sin Waves, size ' + str(i) , fontsize = 15)

ax = plt.subplot(5,1,1)

ax.set\_title("Frequency 10")

# Plot the sin wave against a list of frequencies the length of the sin wave.

plt.plot(length\_of\_sin\_myDFT, sin\_myDFT\_1)

ax = plt.subplot(5,1,3)

ax.set\_title("Frequency 20")

sin\_wave\_2 = makeSinWave(20, i)

sin\_myDFT\_2 = myOwnDFT(sin\_wave\_2)

plt.plot(length\_of\_sin\_myDFT, sin\_myDFT\_2)

ax = plt.subplot(5,1,5)

ax.set\_title("Frequency 30")

sin\_wave\_3 = makeSinWave(30, i)

sin\_myDFT\_3 = myOwnDFT(sin\_wave\_3)

plt.plot(length\_of\_sin\_myDFT, sin\_myDFT\_3)

# Plot the times it takes to calculate all the functions

print(times)

print(len(times))

fig6 = plt.figure()

length\_of\_time\_list = np.linspace(0, 14, len(times))

plt.stem(length\_of\_time\_list, times)

plt.ylabel("Time for Completion in Seconds")

plt.xlabel("Iteration")

print("Running...")

fig7 = plt.figure()

# Perform the myOwnDFT on the wave file

fs, wav\_data = wavfile.read("kpt1note2k.wav")

length\_of\_data = len(wav\_data)

wav\_myOwnDFT = myOwnDFT(wav\_data)

plt.plot(np.arange(length\_of\_data), wav\_myOwnDFT)

plt.suptitle("kpt1note2k.wav through myOwnDFT")

plt.xlabel("Frequency in Hz")

plt.ylabel("Amplitude")

plt.show()

1. Plots against frequency for different frequencies and length of summation

A screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidence

1. How long each plot took to calculate.

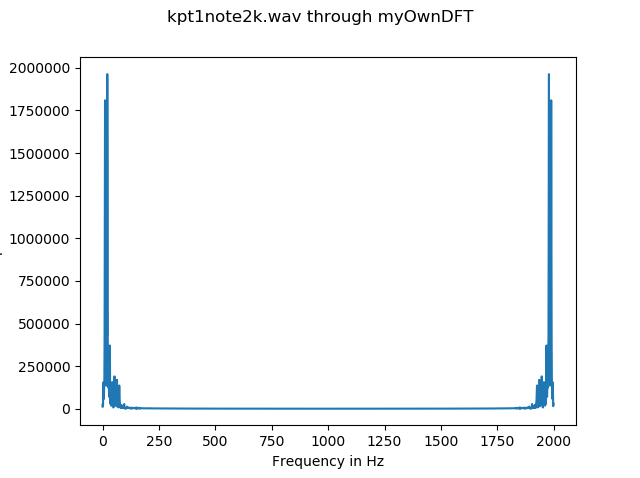
A screenshot of a cell phone

Description generated with very high confidence

1. Estimate how long it takes to perform our DFT on the lab 8 wav file.

I estimate a few months.

This is a graph of the lab 9 wav file after it has been passed through myOwnDFT



1. What was the note

I estimate B1