Mike Hickey

Professor Tom

Digital Signals Processing

Lab 7

Problem 1

import numpy as np

import scipy.signal as sig

import matplotlib.pyplot as plt

plt.style.use('seaborn-dark')

Tx = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

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

i = 0

summation = 0

for T in Tx:

n = T

snt = ((-2)/(np.pi))\*(np.cos(n\*np.pi) \* np.sin(n\*t))/n

summation = summation + snt

st = t/np.pi

plt.scatter(t, st, c="g", marker="o", label="st")

plt.scatter(t, summation, c="r", marker="x", label="summation")

plt.savefig("1/lab7\_1\_%i.png"%i)

plt.show()

i+=1

Problem 2

import numpy as np

import scipy.signal as sig

import matplotlib.pyplot as plt

plt.style.use('seaborn-dark')

Tx = np.array(range(1, 20, 1))

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

i = 0

summation = 0

for T in Tx:

n = T

L = 5

scalar = 4/(np.pi \* n)

snt = scalar \* ( np.sin(n\*np.pi\*t/L) )

# If you do not save the figure before you show it, the .png will be corrupted.

# Is i an even number?

if(T % 2 == 1):

summation = summation + snt

st = t/np.pi

plt.scatter(t, st, c="g", marker="o", label="st")

plt.scatter(t, summation, c="r", marker="x", label="summation")

plt.title('Problem 2, n = %s'%T)

plt.savefig('2/lab7\_2\_%i.png'%i)

i+=1

plt.show()

Problem 2.1

The waveform generated is a square wave.

Problem 3

import numpy as np

import scipy.signal as sig

import matplotlib.pyplot as plt

plt.style.use('seaborn-dark')

Tx = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20])

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

i = 1

summation = 0

for T in Tx:

n = T

if T % 2 != 0:

L = np.pi

scalar = (8)/((np.pi)\*\*2)

exponent = (n-1)/2

num = (-1)\*\*exponent

den = (n\*\*2)

snt = scalar \* num \* ( np.sin(n\*np.pi\*t/L) )/den

summation = summation + snt

st = t/np.pi

plt.scatter(t, st, c="g", marker="o", label="st")

plt.scatter(t, summation, c="r", marker="x", label="summation")

if n != 0 or n != 1:

plt.savefig('3/lab7\_2\_%i.png'%i)

i+=1

plt.show()

Problem 3.1

The wave generated is a triangle wave.

Problem 4

Problem 4.1

Fourier series 1 and 2 had some inaccuracies. The approximation is not exact.

Problem 4.2

This introduces all the inverse square. If we skip those in the summation then our wave changes from a square wave to a triangle wave. However, this is geometrically introduced to problem 3. Problems 1 and 2 do not have that geometric quality.

Problem 4.3

Overall in the end, the inaccuracies are decreasing. However, it’s inaccuracies are decreasing at the same rate. Problem 3 on the other hand, its inaccuracy is decreasing extremely quickly.

Problem 4.4

This phenomenon is called Roll-Off.

The triangle wave is a non-sinusoidal wave form. The harmonics of sine are odd; therefore, triangle waves build of odd harmonics. The higher harmonics will roll off faster because of the inverse square introduced to problem 3.

The phenomenon for inaccuracies is called Gibbs Phenomenon, where there is an overshoot/ringing when approximating a wave with Fourier series. One can reduce the Gibbs Phenomenon by implementing the Lanczos sigma Factor.