LABORATORY MANUAL

CE3007: Digital Signal Processing
Hardware Lab 1 (Location: N4-01a-03)

SESSION 2019/2020
SEMESTER 2
COMPUTER ENGINEERING COURSE

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

LAB – 2

Discrete time - Linear Time Invariant system

1. OBJECTIVE

In this laboratory exercise, we will continue to use python to examine linear time invariant (LTI) system. We will explore the Input/Output relationship of LTI system with respect to impulse response and constant coefficient difference equation.

You will be asked to write routines to generate the output of LTI system by convolution and difference equation. You will also examine how LTI system affect Eigen functions (sinusoid and complex exponential). You will also use SciPy's spectrogram module to visualize the frequency content of the discrete time sequence.

Snippets of python code which can help in this laboratory is provided in Lab2Example.py

The expectations of students taking this subject remain the same as in Laboratory 1.

2. Tasks

1. What is the impulse response of an LTI system?
   a. State the convolution equation and discuss its application.
   b. What happens to a pure sinusoid when it is passes through an LTI system? What does Eigen function of an LTI system mean? Hint – read [7].

      Generate $x[n] = 1 * \cos(0.1\pi\, n)$ for 5 cycles of the signal. Convolve this signal with $h[n] = \{0.2, 0.3, -0.5\}$, n = 0,1,2. Plot the input and output signal together on the same graph, and comment on their relationship.

      Hint python implementation:
      ```
      h = np.array([0.2,0.3,-0.5])
      y  = np.convolve(x,h)
      ```

2. For the LTI system, how is the convolution equation related to the linear constant coefficient difference equation? Give an example when the special case of impulse response is finite duration?

3. You are given an input wave file (testIp_16bit.wav) which is a 16KHz-16bit sampled wave file and an LTI system with impulse response defined by

   ```
   impulseH = np.zeros(8000)
   impulseH[1] = 1
   impulseH[4000] = 0.5
   impulseH[7900] = 0.3
   ```

The above impulse response typifies an echo filter [1]. This is different to room impulse response which is not as sparse, see [2].

a. Sketch the impulse response. Write your own routine to generate the output y[n] by convolving the input wave file sequence with the given system. You can check the correctness of your implementation by comparing it against numpy's convolve function[8] output to ensure correctness.

b. Save y[n] as a wave file (16KHz, 16bits) and listen to it. What did the filter do to the input wave file? Explore the effects of the system on the given input by setting different values to impulseH.

c. Convolution is an expensive process. Is there a simpler way to generate the output? Hint – since most of impulse values are zero.

4. Given two discrete time systems (h1 and h2) with finite impulse response given by

   h1[n] = [0.06523, 0.14936,0.21529,0.2402,0.21529,0.14936,0.06523]
   h2[n] = [-0.06523, -0.14936,-0.21529,0.7598,-0.21529,-0.14936,-0.06523]

   a. Plot these two system's impulse response.

   b. Implement your own code as a module to generate the output for the above system for input $x[n] = \delta[n] - 2\delta[n-15]$. See Lab2Example.py for possible realisation. Plot the system's output. A suggested function prototype will be

      Y = myFilter_Convolve(H,X)

      where H = impulse response of the LTI system, X = impulse sequence of desired length. You can compare your results using SciPy's lfilter module[6] and numpy convolve function. What is the relationship between the output of the system to the impulse response?

   c. Generate an input sequence (1sec duration) consisting of two sinusoid of frequencies 0.7KHz and 3.333 KHz sinusoid signal with amplitude 0.1 sampled at 16 KHz. This input sequence is passed through the two given LTI system.

      i. Plot this signal's spectrogram (scale Sxx by 10*log10(Sxx)) using scipy's spectrogram function [3]. See Lab2Example.py
      ii. For each system, generate the corresponding output.
      iii. Examine each system's output signal by plotting it in time domain, its spectrogram, by listening to it, and suggest what had happened to the input signal after it has passed through each system?

5. Load a wave file "helloworld_noisy_16bit.wav". Listen to it. It is a speech utterance corrupted with a sinusoid at 3KHz. Send this wave file sequence through a system described by a constant coefficient difference equation [5,9] with the following coefficients: B= [1 -0.7653668, 0.99999], A = [1, -0.722744, 0.888622]
   a. Plot the input signal in time domain as well as its spectrogram (scale Sxx by 10*log10(Sxx)) using scipy's spectrogram function [3]. See Lab2Example.py.
   b. Sketch the direct form 1 representation [9] and write the difference equation of the system.
   c. Implement your own code to generate the output for the above system, you can compare your results using SciPy's lfilter module[6]. See Lab2ExampleFilter.py

d. Generate the output for the given input. Listen to the output signal, plot the input and output signal in time domain. Compare the spectrogram of the output signal to the input signal. Hence comment what effects the system has on the input. Hint: read about notch filter [4]
e. Suggest how to determine if the system is stable, and hence determine the above system's stability?
f. Find the close form equation representing the impulse response. Generate 10 values of the impulse response and compare it against Matlab's fvtool(B,A) realisation.

4. References

[1] "Echo and reverberation", http://personal.ee.surrey.ac.uk/Personal/P.Jackson/ee1.lab/D3_echo/D3_EchoAndReverberationExpt.pdf

[2] "Room impulse response generator" , https://www.audiolabs-erlangen.de/fau/professor/habets/software/rir-generator

[3] SciPy's Spectrogram, https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.spectrogram.html

[4] Notch filter - http://sepwww.stanford.edu/sep/prof/pvi/zp/paper_html/node30.html

[5] Constant coefficient difference equation LTI system, https://ocw.mit.edu/resources/res-6-007-signals-and-systems-spring-2011/lecture-notes/MITRES_6_007S11_lec06.pdf

[6] SciPy's lfilter https://docs.scipy.org/doc/scipy-0.19.0/reference/generated/scipy.signal.lfilter.html

[7] LTI system and Eigen function, https://ptolemy.eecs.berkeley.edu/eecs20/week9/lti.html

[8] Numpy's convolution function https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.convolve.html

[9] Direct form 1 realization, https://ccrma.stanford.edu/~jos/fp/Direct_Form_I.html