Lab 2 - Discrete Time Fourier Analysis and Sampling

Section	L01	L02	L03	L04
Lab Date	Oct. 16	Oct. 2	Oct. 17	Oct. 3
Due Date for Report	Oct. 22	Oct. 22	Oct. 22	Oct. 22

Assessment: 5% of the total course mark.

Objectives:

- To gain experience in implementing and analyzing digital systems within `MATLAB`. In particular, you will look at time domain convolution, system impulse responses, and the discrete-time Fourier Transform DTFT.
- To gain experience in creating projects in `CCS5` and running programs on the TMS 320 DSP processor. You need to understand the cause of aliasing and be able to observe the aliasing effects.

Assessment:

- Your grade for this lab will be based on your ability to create and work with digital signals within `MATLAB` and the TMS 320 DSP processor, and on your reporting of the results.
- Clearly label all plots and their axes (points for style will be deducted otherwise).
- Please attend the lab section to which you have been assigned.
- You can complete this lab with one lab partner.
- By the end of the lab session, you must demonstrate to your TA the MATLAB code and the aliasing effects of the sampled audio signal on the TMS 320 DSP processor.
- Each pair of students should complete one lab report together. The source code and the report have to be submitted by **11:59 pm on Oct. 22, 2023**. One of the group members can submit the source code and the report.

Pre-lab:

- Carefully read through this lab description so that you know what is required.
- Read through the lecture notes so that you know how to answer the questions.
- Familiarize yourself with the `MATLAB` commands that may be required for this lab – see the list at the end of this lab description for some hints.

Experiments:

1. **Introduction to convolution**

   (a) Create the discrete-time sequence:

   $$x[n] = u[n] - u[n-10]$$

   where $u[n]$ is the unit-step function (i.e., $u[n] = 1$ for $n \geq 0$ and $u[n] = 0$ for $n < 0$). You do not have to zero-pad $x[n]$ (i.e., the vector you have should contain no zero elements). Plot $x[n]$ using `stem()` function.

(b) Now, convolve $x[n]$ over and over:

$$\begin{aligned} a[n] &= x[n] * x[n] \\ b[n] &= a[n] * x[n] \\ c[n] &= b[n] * x[n] \\ d[n] &= c[n] * x[n] \end{aligned}$$

You can use the `MATLAB` function `conv()` with proper input parameters.

(c) Plot $a[n], b[n], c[n], d[n]$ using `stem()` function.

2. **Convolution of signals and system impulse responses:**

   (a) Load the supplied acoustic impulse response of a room into `MATLAB` using the command:
   `[impr,fs] = audioread('impr.wav');`
   This impulse response was obtained by creating an impulsive noise at one position in the room and recording (and digitizing) the sounds arriving at another position in the room.

   (b) Plot the impulse-response waveform `impr` using the `plot()` command and listen to it using the `soundsc()` command. What can you see and hear in the impulse response?

   (c) Load the supplied speech signal into MATLAB using the command:
   `[y,fs] = audioread('oilyrag.wav');`

   (d) Convolve the speech signal with the impulse response, and plot and listen to the resulting signal. Describe what you see and hear, comparing it to the original speech signal `y`. Explain what the convolved signal is physically equivalent to, according to the impulse-response theory.

3. **The Discrete-Time Fourier Transform (DTFT)**

   (a) Create a `MATLAB` function `output_dtft = calculate_dtft(x,w)` to calculate the DTFT for a given input signal vector `x` and a given discrete-time frequency vector `w`. For the sample index `n`, create the vector:
   `n = -ceil((length(x)-1)/2):floor((length(x)-1)/2);`

   (b) Calculate the DTFTs of the impulse responses $h_1[n]$ and $h_2[n]$ for frequencies `w` covering the range $[-3\pi, 3\pi]$. Consider the two impulse responses, $h_1[n]$ and $h_2[n]$

$$\begin{aligned} h_1[n] &= \frac{1}{4}\delta[n] + \frac{1}{2}\delta[n-1] + \frac{1}{4}\delta[n-2] \\ h_2[n] &= -\frac{1}{4}\delta[n] + \frac{1}{2}\delta[n-1] - \frac{1}{4}\delta[n-2] \end{aligned}$$

   Are the resulting DTFTs real or complex valued? What should they be?

   (c) Plot the magnitude of each of the DTFTs from part (b) versus frequency `w`. Do you see periodicity in these spectra? If so, what is the period and why?

4. **Effect of Sampling and Aliasing on the TMS320C6713 DSK**
   The TMS320C6713 DSK board has an on-board 16-bit audio stereo codec (the Texas Instruments AIC23B) that serves both as an A/D and a D/A converter. The AIC23 codec can be programmed to sample audio inputs at the following sampling rates:

$$f_s = 8,\ 16,\ 24,\ 32,\ 44.1,\ 48,\ 96 \text{ kHz}$$

   The smallest sampling rate that can be defined is 8 kHz with a Nyquist interval of $[-4, 4]$ kHz. Thus, if a sinusoidal signal is generated (e.g., with MATLAB) with frequency outside this interval, e.g., $f = 5$ kHz, and played into the line-input of the DSK, one might expect that it would be aliased with $f_a = f_s - f = 8 - 5 = 3$ kHz. However, this will not work because the anti-aliasing oversampling decimation filters of the codec filter out any such out-of-band components before they are sent to the processor.

An alternative is to decimate the signal by a factor of 2, i.e., dropping every other sample. If the codec sampling rate is set to 8 kHz and every other sample is dropped, the effective sampling rate will be 4 kHz, with a Nyquist interval of $[-2, 2]$ kHz. A sinusoid whose frequency is outside the decimated Nyquist interval $[-2, 2]$ kHz, but inside the true Nyquist interval $[-4, 4]$ kHz, will not be cut off by the antialiasing filter and will be aliased. For example, if $f = 3$ kHz, the decimated sinusoid will be aliased with $f_a = 4 - 3 = 1$ kHz.

Lab Procedure

(a) Several steps are required to compile, link and load the code for the DSP processors. These steps are outlined in the document "Configuring CCS5.pdf".

(b) You initialize and run the program "PassThrough", which is indicated in the Config document and uses a sampling rate of 8 kHz. This program simply reads in values from the function generator into the DSP, and then immediately writes them back out again, to display on the scope.

(c) Open MATLAB and generate three sinusoids of frequencies $f_1 = 1$ kHz, $f_2 = 3$ kHz, and $f_3 = 1$ kHz, each of a duration of 1 second, and concatenate them to form a 3-second signal. Then play this out of the PC's sound card using the `sound()` function. You can use the following MATLAB code.

```
fs = 8000; f1 = 1000; f2 = 3000; f3 = 1000;
L = 8000; n = (0:L-1);
A = 1/5; % adjust playback volume

x1 = A * cos(2 * pi * n * f1 / fs);
x2 = A * cos(2 * pi * n * f2 / fs);
x3 = A * cos(2 * pi * n * f3 / fs);

sound([x1, x2, x3], fs);
```

(d) Connect the sound card's audio output to the line-input of the DSK and rebuild/run the "PassThrough" program after commenting out the line in `c\_int11()`:

```
pulse = (pulse == 0);
```

This disables the downsampling operation (default state). Send the above concatenated sinusoids to the DSK input and you should hear three distinct 1-second segments, with the middle one having a higher frequency.

(e) Next, uncomment the above line so that downsampling takes place and rebuild/run the program. Send the concatenated sinusoids to the DSK. Describe what you hear and explain the possible reasons.

(f) Replace the first line of the above MATLAB code in (c) with the following one:

```
fs = 8000; f1 = 1000; f2 = 5000; f3 = 1000;
```

Repeat steps (d) and (e). What do you expect to hear in this case? Explain the possible reasons.

(g) Replace the first two lines of the above MATLAB code in (c) with the following two:

```
fs = 16000; f1 = 1000; f2 = 5000; f3 = 1000;
```

```
L = 16000; n = (0:L-1);
```

Turn off the downsampling operation, rebuild and run your program and send this signal through the DSK. Describe what you hear and explain the possible reasons.

REPORT: The report should contain

- Any mathematical calculations or derivations carried out
- `MATLAB` plots of results with brief descriptions
- Answers to questions

You **do not** need to include the `MATLAB` code in the report. However, you have to submit the `MATLAB` code separately.

POTENTIALLY USEFUL `MATLAB` COMMANDS:

Note that this is not an exhaustive list! You are not required to incorporate all of these in your scripts.

help topic	helpwin	figure	plot	stem
histogram	subplot	hold on	xlabel	ylabel
legend	title	function	clear	close
clc	zeros	ones	cos	exp
abs	round	max	min	find
if	for	end	real	imag
angle	unwrap	phase	audioinfo	audioread
audiowrite	soundsc			