

## LAB 3 - DISCRETE TIME FOURIER ANALYSIS AND FILTERING

Section	L01	L02	L03	L04
Lab Date	Oct. 27	Oct. 28	Oct. 29	Oct. 30
Due Date for Report	Nov. 7	Nov. 7	Nov. 7	Nov. 7

**Assessment:** 5% of the total course mark.

---

### OBJECTIVES:

- To gain experience in filtering and discrete-time Fourier Transform (DTFT) within **MATLAB**.
- To gain experience in applying a filter for radar data processing using convolution on the TMS 320 DSP processor.

### ASSESSMENT:

- Your grade for this lab will be based on your ability to 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 TMS 320 DSP processor part.
- All **MATLAB** source code and c code (updated “RadarProcessor\_lab3.c”) must be submitted on Avenue to Learn by the end of your designated lab session.
- Each pair of students should complete one lab report together. The report has to be submitted by **11:59 pm on Nov. 7, 2025**.

### EXPERIMENTS:

#### 1. Introduction to filtering:

- (a) Consider two impulse responses,  $h_1[n]$  and  $h_2[n]$

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

and six discrete time sinusoidal signals

$$x_a[n] = 5 \cos(0n)$$

$$x_b[n] = 5 \cos\left(\frac{\pi}{5}n\right)$$

$$x_c[n] = 5 \cos\left(\frac{2\pi}{5}n\right)$$

$$x_d[n] = 5 \cos\left(\frac{3\pi}{5}n\right)$$

$$x_e[n] = 5 \cos\left(\frac{4\pi}{5}n\right)$$

$$x_f[n] = 5 \cos(\pi n)$$

for  $n = 0, \dots, 50$ . Note that only the frequency is different in each signal.

- (b) Calculate the energy in each of the signals  $x_a$  through  $x_f$  by taking the root mean square (RMS) of each of them:

$$RMS = \sqrt{\frac{1}{N} \sum_{n=1}^N x^2[n]}$$

You **cannot** use MATLAB builtin functions to do RMS calculation.

- (c) Now, convolve  $h_1[n]$  with each of the signals  $x_a[n]$  through  $x_f[n]$ , and calculate the resulting RMS values of the six different output signals. Calculate the difference between output and input RMS values for each of the signals in terms of the system gain measured in decibels (dB):

$$Gain = 20 \log \left( \frac{RMS_{out}}{RMS_{in}} \right) dB$$

Plot the gain for each signal as a function of the signal frequency.

- (d) Repeat part c but with the impulse response  $h_2[n]$ .  
(e) What are the differences between the two plots? What are the differences between  $h_1[n]$  and  $h_2[n]$  in terms of the frequencies of the input signals they pass?

## 2. 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 = 0:length(x)-1;`  
(b) Using your function from part (a), calculate the DTFTs of the impulse responses  $h_1[n]$  and  $h_2[n]$  from section 1 above, for frequencies **w** covering the range  $[-3\pi, 3\pi]$ . 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?

## 3. Filtering white Gaussian noise

- (a) Create a signal 1000 samples long of white Gaussian noise. Convolve this signal with  $h_1[n]$  and  $h_2[n]$  from section 1 above, then compute the DTFT for each of the convolved signals using the function you created in section 2 above.

- (b) Plot the magnitude of each of the DTFTs from part (a) versus frequency  $\omega$ . Do the results agree with those of sections 1 and 2 above? Explain.

#### 4. Radar data target detection with convolution on the TMS 320 DSP processor

##### (a) Background Knowledge

In the previous lab, you loaded the transmitted signal and received signal of the FMCW radar and visualized them through the oscilloscope. However, without further signal processing, no useful information can be obtained. In this session, you will learn how to use convolution (a matched filter) to detect the targets and find their ranges. Matched filter can be implemented in both time-domain and frequency-domain. In this lab, you will explore the time-domain matched filter.

##### i. Transmitted Signal and Received Signal

The transmitted signal will be reflected back if there is a target in front. Depending on the range of the target to the radar, there is a delay  $\delta t$  between the transmitted signal and the received signal. The waveform of the received signal resembles the waveform of the transmitted signal. The frequency vs. time plot of the two signals are shown in figure 1. For simplicity, the received signal is the signal reflected back when there is only one target in the detection range.

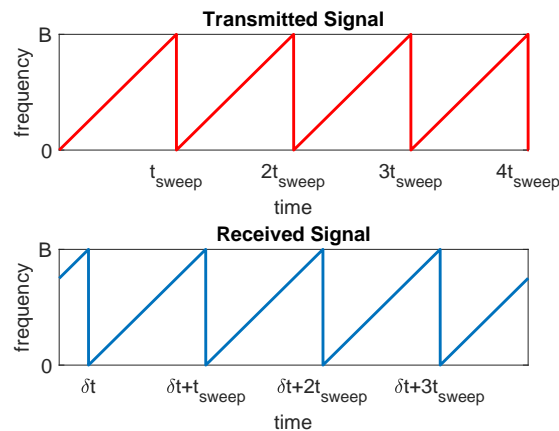


Figure 1: Frequency vs. time plots of the transmitted signal and the received signal.

##### ii. Matched Filter in Time Domain

In order to detect targets and obtain their ranges, one of the common approaches is using a matched filter. The basic idea behind the matched filter is to “match” the transmitted signal, which is referred as “reference signal”, with the received signal using convolution.

In the matched filter, the reference signal slides over the received signal. Considering that the reference signal and the received signal have very similar waveform, the product of the signals during sliding should reach the maximum when the two signals “matches” with each other (Figure 2).

The signals that radar transmits and receives are essentially electromagnetic (EM) waves, so they can be entirely represented by complex numbers, which characterize both their magnitudes and phases. During the convolution evaluation, the conjugate of the reference signal is used. This is to ensure that when the signals match, the

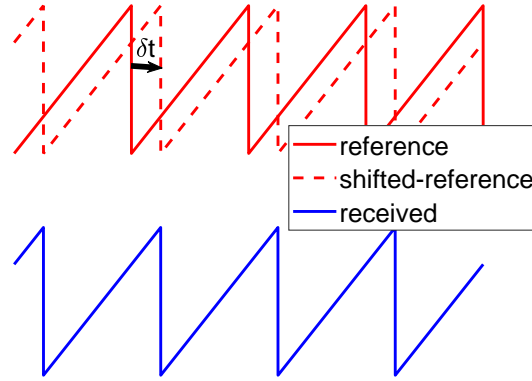


Figure 2: The point that the two signals match, and the convolution reaches maximum.

phase terms are cancelled out for all the products of the samples, and their sum is constructively summed in the convolution, which makes the norm of the result reach the maximum. However, if the two signals mismatch, the phase terms do not cancel out, and when sum them together, negative phases may get cancelled out with positive phases (the phase terms get interfered), so that the convolution norm decreases. The convolution equation is

$$y[n] = x[n] * s^*[-n] \quad (1)$$

where  $x[n]$  is the received signal and  $s[n]$  is the transmitted signal.

To summarize the operation we are doing here, we take the conjugate of the reference signal and invert it in time domain. Then calculate the convolution of it with the received signal. The convolution reaches the maximum when the two signals match with each other, which corresponds to the time delay between the two signals. The number of peaks is equal to the number of targets in the scenario. Note that noise can also lead to small peaks.

### iii. Range Equation

After finding a peak,  $n_{peak}$  in the discrete-time domain, it can be converted to the continuous time,  $t_p$ , using (2). The time delay is caused by the EM wave traveling forth and back between the radar and the target, hence the range,  $r$ , can be calculated with (3).

$$t_p = n_{peak}T_s = \frac{n_{peak}}{f_s} \quad (2)$$

$$r = \frac{t_p c}{2} = \frac{n_{peak}c}{2f_s} \quad (3)$$

where  $T_s$  and  $f_s$  represent the sample period and sample frequency, respectively, and  $c$  is the speed of light.

The sampling frequency is specified in the data file (“data\_rec.h”).

## (b) **Lab Procedure**

- i. Import the source file into the project:

- A. Create a copy of the project that you created in lab 2 and rename as “RadarProcessor\_Lab3” and open this project in CSS.
  - B. Delete “RadarProcessor.c” from this project.
  - C. Right click the project under the explorer, and choose “Add Files..”.
  - D. Direct to the folder “Y:/COE4TL4/RadarProcessor/src”, and add the source code “RadarProcessor\_lab3.c” to the project. **MAKE SURE TO CHOOSE THE OPTION OF “COPY FILES”, NOT “LINK FILES”.**
- ii. Understand the implementation of the matched filter:
    - A. Read the “convolution” function in the source code, and explain the code to the TA. Read the previous section to understand how the matched filter works.
  - iii. Read the interrupt function “c\_int11()”, and explain the code to the TA.
  - iv. Compile and run the code. Describe what you observe on the oscilloscope, and determine the number of targets.
  - v. Complete the function “save\_results(COMPLEX\* data, int N)” to save the output of the convolution function to a csv file.
  - vi. Write a simple script in MATLAB that reads the csv file you just created in the previous step. Plot the result in MATLAB. Find the peaks and use the range equation given in the previous section to estimate the ranges of the targets.

REPORT: The report should contain:

- **ALL** the MATLAB plots and the oscilloscope screenshots **WITH BRIEF DESCRIPTIONS**.
- Answer the questions 1(e), 2(c), 3(b), 4(b-iv), and 4(b-vi) with some discussions.

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