ECE 535 — Digital Signal Processing
Matlab Project 2: Convolution and the DFT
George Mason University
Fall 2020
Due: Tuesday, December 8 by 11:59 pm
Submit via Blackboard.

- Matlab project submissions must be your own individual work. This includes code, plots, answers to questions, and all other report components. You may discuss the project with others at a high level. However, the coding and the write-up (answering questions, explanations, etc.) are to be completed individually.
- Matlab projects **must be submitted a single pdf file.**
- Matlab project submissions must be typed or written neatly. All parts must appear in the order they are assigned.
- All plots must be neatly labeled with x-axis and y-axis labels. Any plot that is not fully labeled will not be graded.
- All plots must be fully referenced. I will not try to figure out what plots correspond to what parts of the project when grading. You can either assign figure numbers to the plots (i.e. The resulting plot is shown in Figure 1) or cite the location in which the plot appears (i.e. The resulting plot appears at the top of page 5).
- Project submissions should include all analytical work (paper and pencil calculations), all relevant explanations and answers to questions (in complete sentences), and all Matlab code. Matlab code must be well-documented. It may be included in an appendix at the end of the report and appropriately referenced as needed. Alternatively, the relevant code may be included in each section of the report.
- Remember that you can use the help command in Matlab to find out more about any command.

1 Instructions

We discussed that linear convolution of a stream of data with a fixed impulse response can be performed efficiently using the DFT (more specifically, a fast implementation of the DFT, i.e., a FFT). The goal of this project is to implement the *overlap-add* method using MATLAB's FFT functions.

1. Write a Matlab function called `circonv` that implements circular convolution **using the DFT**. The DFT should be implemented via the FFT, and hence your function `circonv` should include two calls to the Matlab function `fft` and one call to the Matlab function `ifft`. Your function should allow the user to specify the length of the circular convolution. Alternatively, the user should be able to specify that the desired output is the linear convolution of the two input sequences. In this case, the function should automatically compute the required length of the circular convolution such that the result is the linear convolution of the input signals.

   Include the following components in your write-up:

   - Your Matlab function `circonv`
   - A thorough description of each of the inputs to and outputs of your function

2. Test your function `circonv` using the following inputs:

   (a) $x[n] = u[n] - u[n-4]$, $h[n] = (n+1)(u[n] - u[n-3])$, circular convolution of length $N = 4$.

   (b) $x[n] = u[n] - u[n-4]$, $h[n] = (n+1)(u[n] - u[n-3])$, linear convolution.

   (c) $x[n] = \{1, 3, 6, 2, 8\}$, $h[n] = \{2, 9, 5, 3\}$, convolution of length $N = 6$.

   For each set of inputs, include the following in your write-up:

   - The values of the input parameters to your function `circonv`
   - A *stem* plot of the output of your function `circonv`

3. Write a Matlab function called `ola` that implements linear convolution using the overlap-add method of block convolution. Your function should take as inputs the two sequences to be convolved and the size of the DFT to be used. This function should also use Matlab's `fft` function to compute the DFT.

   Include the following components in your write-up:

   - Your Matlab function `ola`
   - A thorough description of each of the inputs to and outputs of your function

4. The Matlab data file `MP2data.mat` is available on the course website. It includes a filter impulse response `h` and data sequence `x` which will be used to test your block convolution functions.

   (a) What kind of filter is $h[n]$, e.g. what kind of ideal filter does it best approximate? Does $h[n]$ have generalized linear phase? If so, what type? Justify your conclusions.

   (b) What range of DFT sizes are valid inputs for `ola` for the `h` and `x` given?

   (c) Compute $h[n] * x[n]$ using `ola`. Choose a DFT size no larger than 100. To confirm that your results are correct, use Matlab's `conv` function to compute $h[n] * x[n]$.

   Include the following components in your write-up:

   - The values of the input parameters to your function `ola`
   - A *stem* plot of $x[n]$ and of $h[n] * x[n]$; include both on one page using the subplot command in Matlab.