Carnegie Mellon DSP Electrical & Computer ENGINEERING

Digital Signal Processing (18-491/18-691)
Spring Semester, 2023

Problem Set 6

Issued: 3/16/23

Due: 3/24/23 at midnight via Gradescope

Reading: In hte past week or so we discussed the DFS/DTFS and the DFT, following material in OSYP 8.0-8.6. We also discussed the differences between circular and linear convolution, including the overlap-add (OLA) and overlap-save (OLS) algorithms, which enable us to implement linear convolution using a series of circular convolutions. Next week we will introduce the fast Fourier transform (FFT) algorithm, following the presentation in OSYP Chapter 9. It is important that you know and understand the basic definitions of the DFS and the DFT, their properties, and the differences and relationships between linear and circular convolution.

We then went on to introduce the basic decimation-in-time FFT algorithm, following OSYP 9.2. Next week we will discuss alternative structures, and non-radix-2 FFTs. The relevant sections of OSYP are Secs. 8.7, 9.0, and 9.2 to 9.5. discuss the implementation procedures for discrete-time systems, following OSYP Secs. 6.0 through 6.5.

Note: Quiz 2 will be on April 5

Problem 6.1

The unit sample response of an LSI system is

$$h[n] = \begin{cases} \left(\frac{1}{4}\right)^n, & 0 \le n \le 7 \\ 0, & \text{otherwise} \end{cases}$$

The input to the system is

$$x[n] = \delta[n] - \delta[n-4]$$

(a) Determine and sketch the linear convolution of $x[n]$ and $h[n]$.

(b) Determine and sketch the 16-point circular convolution of $x[n]$ and $h[n]$ for $0 \le n \le 15$.

(c) Determine and sketch the 10-point circular convolution of $x[n]$ and $h[n]$ for $0 \le n \le 9$.

Problem 6.2:

We would like to implement the linear convolution of a 20,000-point sequence with an FIR impulse response that is 100 points long. The convolution is to be implemented by using DFTs and inverse DFTs of length 1024.

(a) Suppose that the overlap-add method is used.

- What is the minimum number of 1024-point DFTs and the minimum number of 1024-point inverse DFTs needed to implement the covolution for the entire 20,000-point sequence? Justify your answer.
- What are the values of n at the beginning and end of the first three subsequences of $x[n]$ that are used to implement the convolution?

(b) Now consider the implementation of the convolution described above using the overlap-save algorithm.

- What is the minimum number of 1024-point DFTs and the minimum number of 1024-point inverse DFTs needed to implement the convolution for the entire 20,000-point sequence? Justify your answer.
- What are the values of n at the beginning and end of the first three subsequences of $x[n]$ that are used to implement the convolution?
- What are the values of n that identify the first and last points of the points that are retained from the first three output sequences that result from the circular convolutions.

(c) Compare the number of multiplications incurred using the overlap-add method, the overlap-save method, and direct convolution when the DFT calculation is performed in the traditional fashion.

(d) As we will discuss in class, when N is a power of 2, an N-point DFT or inverse DFT can be implemented in $(N/2)\log_2(N)$ complex multiplications and $(N/2)\log_2(N)$ complex additions using the FFT. For the same filter and impulse response lengths considered in part (a), compare the number of multiplications in the overlap-add method and direct convolution, assuming that the FFT algorithm is used to implement DFTs for the overlap-add method.

Problem 6.3: Problem 9.6 in OSYP.

Problem 6.4: Problem 9.45 in OSYP.

Problem 6.5: Problem 9.58 in OSYP.

MATLAB Problems

For the MATLAB questions this week and in the future, please submit the following components of your answer to the **written** component of your homework submission on Gradescope. An easy way to handle the code part of this is to use the `publish` feature in MATLAB and submit the output .pdf to the Written assignment on gradescope.

- Answers to the written portions of the problems
- Your plots
- A pdf copy of your code

The MATLAB component of your submission should contain only your `.m` files. We appreciate your help in complying with these formatting requests as it makes your work much easier to grade. As before, we will deduct points for noncompliant submissions.

Problem C6.1:

Verify your answers to Problem 6.1 using the MATLAB routines `fft` and `conv`. Implement the circular convolutions in the problem using DFTs and their inverses, which you can compute using the `fft` and `ifft` commands in MATLAB.

Please complete this problem by filling in the details in the shell program `main_6_1.m` that is provided.

Problem C6.2:

You are given a shell script `main_6_2.m` that you must complete.

(a) Write a MATLAB routine that implements the overlap-save algorithm. It should function according to the following preamble:

```
function [y] = ovrlpsav(x,h,N)
% Overlap-Save method of block convolution
%
% [y] = ovrlpsav(x,h,N)
% y = output sequence
% x = input sequence
% h = unit sample response
% N = block length
%
```

Demonstrate that your routine works by convolving an input sequence of length 256 with a unit sample response of size 64, using a block size of 100, with the input and unit sample both consisting of a set of random numbers of the appropriate length. Show that results obtained using `ovrlpsav` are identical to those obtained using direct convolution.

(b) Now consider the case of an input sequence of length 20,000 and a unit sample response of length 100, as in Problem 6.2. Using your `ovrlpsav` program, implement the convolution using random inputs and unit sample responses of the appropriate size, and block sizes 512, 1024, 2048, and 4096. Measure the CPU time required to perform the convolution for each block size using the MATLAB routine `cputime` (see its `help` file for details on its use). Plot CPU time versus block size. Which block size enables this computation to be implemented with the least amount of CPU time?

What to turn in to Gradescope:

- The code in wrapper `main_6_2.m`
- The completed function `ovrlpsav.m`
- All of the plots