9.8 Laboratory Experiment on Signal Processing

**Purpose**: By performing this experiment, the student will receive a better understanding of the use and power of the FFT algorithm in evaluating the corresponding discrete (time) Fourier transforms, continuous-time Fourier transform, and discrete-time convolution. As the computational tool, we will use the MATLAB functions `fft` and `ifft`.

**Part 1**. In this part we use the FFT algorithm, as implemented in MATLAB, to find the DFT of some discrete-time signals. In addition, we demonstrate the use of the IFFT in recovering original discrete-time signals.

(a) Consider the discrete-time signal
\[ x[k] = \begin{cases} 
1, & k = 0, 1, 2, 3 \\
0, & \text{otherwise} 
\end{cases} \]
and find analytically its DTFT.

(b) Use the MATLAB function `X=fft(x,N)` to find the DFT of the preceding signal for \( N = 4, 8, 12, 16, 24, 32 \). Use the MATLAB function `x=ifft(X,N)` to recover the original discrete-time signal. Plot the DFTs and IDFTs, and comment on the results obtained.

(c) Consider the signal
\[ x[k] = \begin{cases} 
1, & k = 0, 1, 2, \ldots, 11 \\
0, & \text{otherwise} 
\end{cases} \]
and repeat parts (a) and (b).

(d) Consider the signal whose nonzero values are between \( k = 0 \) and \( k = 8 \), respectively defined by \( x = [1, 2, 3, 4, 5, 4, 3, 2, 1] \), \( L = 9 \), and repeat parts (a) and (b). Comment on the results obtained.

**Part 2**. Formula (9.71), \( X(j\omega) \approx T_s X_s(j\Omega) \), can be used for an approximate evaluation of the continuous-time Fourier transform. In this formula, \( T_s \) is the sampling interval used for sampling the continuous-time signal \( x(t) \) into \( x(kT_s) \). \( X_s(j\Omega) \) is the corresponding DFT.

(a) Consider the continuous-time signals presented in Figures 3.22 and 3.23. Sample these signals with \( T_s = 0.1 \) and find DFTs of the obtained discrete-time signals. Calculate and plot the corresponding magnitude spectra for the approximate Fourier transforms and compare them to the results obtained analytically.

(b) Repeat part (a) with \( T_s = 0.01 \).

**Part 3**. Discrete-time signal convolution can be efficiently evaluated via the DTFT and its convolution property. The relation \( y[k] = x[k] * h[k] \) implies \( Y(j\Omega) = X(j\Omega)H(j\Omega) \). Hence, discrete-time convolution via DFT can be evaluated as \( y[k] = \text{IDFT}\{\text{DFT}(x[k])\text{DFT}(h[k])\} \). Note that such an obtained signal \( y[k] \) is, in general,
the wrapped signal, so that the corresponding convolution is called mod-$N$ circular convolution [1].

Use the formula to find the convolution of the discrete-time signals defined in Problems 6.15 and 6.16. Do the results obtained represent unwrapped or wrapped signals?

SUPPLEMENT:

Discrete-time signals in Problem 6.15 are defined as
\[
f_1[k] = \begin{cases} 
1 & k = 0 \\
-1 & k = 1 \\
1 & k = 2 \\
0 & \text{otherwise}
\end{cases}, \quad f_2[k] = \begin{cases} 
3 & k = 0 \\
-2 & k = 2 \\
0 & \text{otherwise}
\end{cases}
\]

Discrete-time signals in Problem 6.16 are defined as
\[
f_1[k] = \begin{cases} 
-2 & k = -1 \\
2 & k = 0 \\
1 & k = 1 \\
-1 & k = 2 \\
4 & k = 3 \\
0 & \text{otherwise}
\end{cases}, \quad f_2[k] = \begin{cases} 
1 & k = 0 \\
2 & k = 1 \\
3 & k = 2 \\
2 & k = 3 \\
0 & \text{otherwise}
\end{cases}
\]