# EE092IU
# DIGITAL SIGNAL PROCESSING LABORATORY

## Lab 6

# FOURIER ANALYSIS OF DISCRETE-TIME SIGNALS

**Full name:………………………………..**

**Student number:………………………...**

**Class:……………………………………..**

**Date:……………………………………...**

# I. OBJECTIVES

Students know how to write m-file in Matlab to illustrate the properties of two types of Fourier Transform of discrete time system: Discrete-Time Fourier Transform.

# II. REQUIRED EQUIPMENT

1. Computer
2. Matlab software

# III. INTRODUCTION

## 3.1. Discrete-Time Fourier Transform

Discrete-time Fourier transform (DTFT) is the counterpart transform to the continuous-time Fourier transform (CTFT) when dealing with discrete-time signals, i.e., when dealing with signals described by a function $\boldsymbol{x}[\boldsymbol{n}], \boldsymbol{n} \in \boldsymbol{Z}$. The mathematical expression of the DTFT is

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \tag{1}$$

The DTFT is a continuous complex-valued function of the cyclic frequency $\omega$. A sufficient condition for the existence of the DTFT $\boldsymbol{X}(\boldsymbol{\omega})$ of a signal $\boldsymbol{x}[\boldsymbol{n}]$ is

$$\sum_{n=-\infty}^{\infty} x[n] < \infty \tag{2}$$

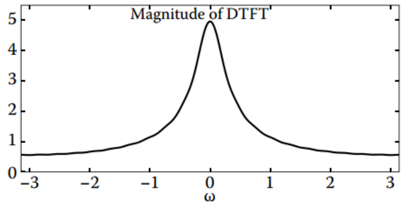
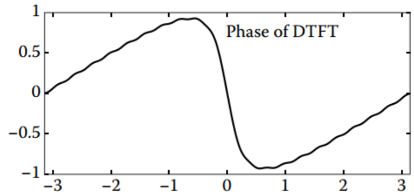
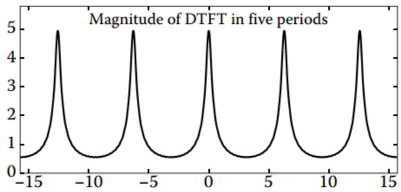
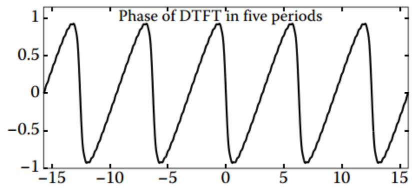
Moreover, the DTFT is always a *periodic* function with period $2\pi$. In order to return from the frequency domain $\omega$ back to the discrete-time domain $n$, we apply inverse DTFT. The mathematical expression of the inverse DTFT is

$$x[n] = \frac{1}{2\pi}\int X(\omega)e^{j\omega n}d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}d\omega \tag{3}$$

The computation of the DTFT $X(\omega)$ of a signal $\boldsymbol{x}[\boldsymbol{n}]$ is easily obtained directly from (1)

**Example 1**

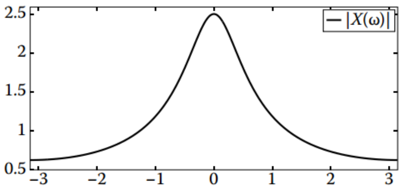
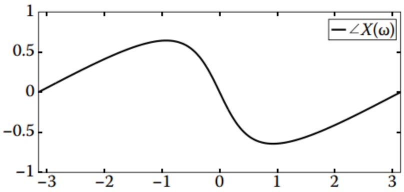
Compute the DTFT $X(\omega)$ of the signal $\boldsymbol{x}[\boldsymbol{n}] = 0.8^n, 0 \leq n \leq 20$ and plot $X(\omega)$ over the frequency intervals $-\pi \leq \omega \leq \pi$ and $-5\pi \leq \omega \leq 5\pi$.

| Commands | Results | Comments |
|---|---|---|
| `syms w`<br>`n= 0:20;`<br>`x=0.8.^n;` | | Definition of $x[n]=0.8^n$, $0 \le n \le 20$. |
| `X=sum(x.*exp(-j*w*n));` | | Computation of the DTFT of $x[n]$ according to (7.1). |
| `ezplot(abs(X),[-pi pi])`<br>`title('Magnitude of DTFT')`<br>`ylim([0 5.4])` |  | Graph of the magnitude of the DTFT $X(\omega)$ for $-\pi \le \omega \le \pi$. |
| `w1=-pi:.01:pi;`<br>`XX=subs(X,w,w1)`<br>`plot(w1,angle(XX));`<br>`xlim([-pi pi])`<br>`title('Phase of DTFT')` |  | Graph of the phase of the DTFT $X(\omega)$ for $-\pi \le \omega \le \pi$. |
| `ezplot(abs(X),[-5*pi 5*pi]);`<br>`title('Magnitude of DTFT in 5 periods')`<br>`ylim([0 5.8])` |  | Graph of the magnitude of the DTFT $X(\omega)$ for $-5\pi \le \omega \le 5\pi$. |
| `w1=-5*pi:.01:5*pi;`<br>`XX=subs(X,w,w1);`<br>`plot(w1,angle(XX));`<br>`xlim([-5*pi 5*pi])`<br>`title(' Phase of DTFT in 5 periods')` |  | Graph of the phase of the DTFT $X(\omega)$ for $-5\pi \le \omega \le 5\pi$. |

### Example 2

Compute and plot the DTFT of the signal $\boldsymbol{x}[\boldsymbol{n}] = 0.6^n u[n]$.

The signal $x[n]$ is defined over the time interval $[0,+\infty)$ as $x[n]=0.6^n u[n]$, which is equivalent to $x[n]=0.6^n$, $n \ge 0$. The length of the sequence $x[n]$ is not finite; hence, the DTFT $X(\omega)$ of $x[n]$ has to be computed with use of the command *symsum*.

<table>
<tr><th>Commands</th><th>Results</th><th>Comments</th></tr>
<tr><td><code>syms n w</code><br><code>x=0.6^n</code></td><td><code>x = (3/5)^n</code></td><td>Definition of $x[n] = 0.6^n u[n]$.</td></tr>
<tr><td><code>X=symsum(x*exp(-j*w*n),n,0,inf)</code></td><td><code>X = 5*exp(i*w)/(3+5*exp(i*w))</code></td><td>The DTFT of $x[n]$ is computed according to (7.1) with use of the command symsum.</td></tr>
<tr><td><code>w1=-pi:.01:pi;</code><br><code>X_=subs(X,w,w1);</code><br><code>plot(w1,abs(X_));</code><br><code>legend('|X(\omega)|')</code><br><code>xlim([-pi pi])</code></td><td>
</td><td>Graph of the magnitude of the DTFT $X(\omega)$ for $-\pi \le \omega \le \pi$.</td></tr>
<tr><td><code>plot(w1,angle(X_));</code><br><code>xlim([-pi pi])</code><br><code>legend('\angle X(\omega)')</code></td><td>
</td><td>Graph of the phase of the DTFT $X(\omega)$ for $-\pi \le \omega \le \pi$.</td></tr>
</table>

### 3.2. Discrete Fourier Transform

In order to implement DTFT analysis of a signal on a computer, it is necessary to obtain samples $\omega_k$ from the frequency $\omega$. This leads to a second type of Fourier transform appropriate for discrete-time signals, the discrete Fourier transform (DFT). The N -point DFT of a discrete-time signal $x[n]$ is denoted by $X_k$ or $X(k)$ or $X(\omega_k)$. It is defined in the discrete-time interval $0 \le n \le N-1$ and is computed according to

$$X_k = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi k}{N}}, k = 0,1,\dots,N-1 \tag{4}$$

The DFT $X_k$ of a sequence $x[n]$ is a function of k and is completely specified by its values for $k = 0,1,\dots,N-1$ that is, from the $N$ values $X_1, X_2, \dots, X_{N-1}$. Typically, $X_k$ are complex numbers. Hence, they can be expressed in polar form as

$$X_k = |X_k| e^{-j\angle X_k}, k = 0,1,\dots,N-1 \tag{5}$$

Where

$|X_k|$ is the magnitude of $X_k$, and $\angle X_k$ is the phase of $X_k$.

Alternatively, $X_k$ is expressed as

$$X_k = \mathrm{Re}\{X_k\} + j\mathrm{Im}\{X_k\}, k = 0,1,\dots,N-1 \tag{6}$$

Where $\text{Re}\{X_k\}$ is the real part of $X_k$ and is given by:

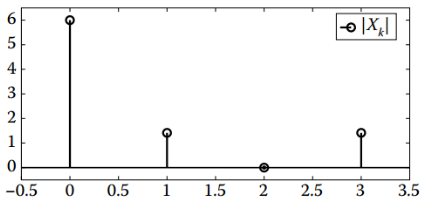
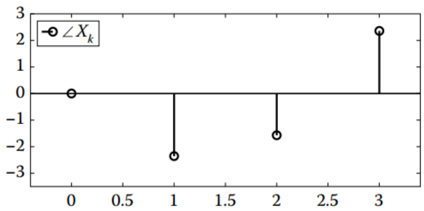
$$\text{Re}\{X_k\} = x(0) + \sum_{n=1}^{N-1} x[n]cos\left(\frac{2\pi nk}{N}\right) = \sum_{n=0}^{N-1} x[n]cos\left(\frac{2\pi nk}{N}\right) \tag{7}$$

where the second equality is derived from the fact that $x(0)cos(0) = x(0)$. The imaginary part $\text{Im}\{X_k\}$ of $X_k$ is given by

$$\text{Im}\{X_k\} = -\sum_{n=0}^{N-1} x[n]sin\left(\frac{2\pi nk}{N}\right) \tag{8}$$

**Example 3:**

Compute the DFT $X_k$ of the sequence $x[n] = [1,2,2,1], 0 \leq n \leq 3$. Plot the magnitude, the phase, the real part, and the imaginary part of $X_k$.

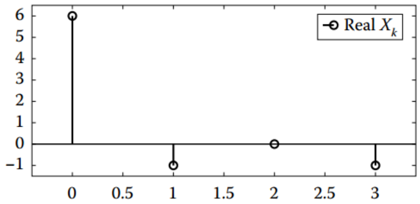
<table>
<tr><th>Commands</th><th>Results</th><th>Comments</th></tr>
<tr><td><pre>x=[1 2 2 1];
N=length(x);
for k=0:N-1
  for n=0:N-1
    X(n+1)=x(n+1)*exp(-j*2*pi*k*n/N);
  end
  Xk(k+1)=sum(X);
end
Xk</pre></td><td><pre>Xk=6.0000
      -1.0000-1.0000i
      0-0.0000i
      -1.0000+1.0000i</pre></td><td>The DFT $X_k$ of the sequence $x[n]$ is computed according to Equation 7.12. Notice that $X_k$ is a complex-valued sequence.</td></tr>
<tr><td><pre>mag=abs(Xk);
stem(0:N-1,mag);
legend ('|X_k|')
xlim([-.5  3.5]);
ylim([-.5  6.5]);</pre></td><td>
</td><td>Graph of the magnitude $|X_k|$ of the DFT $X_k$.</td></tr>
<tr><td><pre>phas=angle(Xk);
stem(0:N-1,phas);
legend ('\angle Xk')
xlim([-.4  3.4]);
ylim([-3.5  3]);</pre></td><td>
</td><td>Graph of the phase $\angle X_k$ of the DFT $X_k$.</td></tr>
</table>

```
mag.*exp(j*phas)
```

```
ans =6.0000
     -1.0000-1.0000i
     0.0000-0.0000i
     -1.0000+1.0000i
```

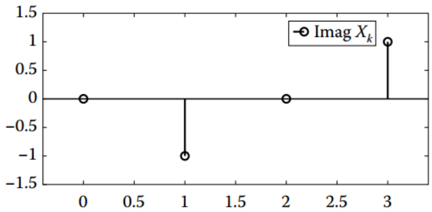
Confirmation of Equation 7.13.

```
re=real(Xk);
stem(0:N-1,re);
xlim([-.4  3.4]);
ylim([-1.5  6.5]);
legend ('real Xk')
```




Graph of the real part of $X_k$.

```
im=imag(Xk);
stem(0:N-1,im);
xlim([-.4  3.4]);
ylim([-1.5  1.5]);
legend ('imag Xk')
```




Graph of the imaginary part of $X_k$.

**Example 4:**

Write a function that computes the DFT $X_k$ of a sequence $x[n]$. Compute through your function the DFT of the discrete-time signal $x[n] = [1,2,2,1], 0 \leq n \leq 3$.

**Commands**

```
function Xk=dft(x);
N=length(x);
  for k=0:N-1
  for n=0:N-1
    X(n+1)=x(n+1)*exp(-j*2*pi*k*n/N);
  end
  Xk(k+1)=sum(X);
end
```

**Results/Comments**

The function *dft.m* is based on the code written in the previous example. The sequence $x[n]$ is the input argument, while the DFT $X_k$ is the output argument of the function.

```
x=[1 -2 2 1];
Xk=dft(x)
```

The function *dft.m* is executed from the command prompt. The DFT $X_k$ of $x[n]$ is

```
Xk = 2.0000     -1.0000+3.0000i
     4.0000+0.0000i     -1.0000-3.0000i
```

### 3.3. Inverse Discrete Fourier Transform

Suppose that the DFT $X_k$ of a discrete-time signal $x[n]$ is known. The signal $x[n]$ can be derived from the N DFT points $X_k, k = 0,1, \ldots, N-1$ by applying the inverse discrete Fourier transform (IDFT). The IDFT of a sequence $X_k$ is given by

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{j\frac{2\pi nk}{N}}, n = 0,1, \ldots, N-1 \qquad (9)$$

**Example 5:**

Compute the IDFT of the sequence $\boldsymbol{X_k = [6, -1-j, 0, -1+j], 0 \leq k \leq 3}$.

| Commands | Results | Comments |
|---|---|---|
| `Xk=[6, −1−j, 0, −1+j];` | | Definition of $X_k$. |
| `N=length(Xk);`<br>`for n=0:N−1`<br>`for k=0:N−1`<br>`xn(k+1)=Xk(k+1)*exp(j*2*pi*n*k/N);`<br>`end`<br>`x(n+1)=sum(xn);`<br>`end`<br>`x=(1/N)*x` | `x = 1.0000`<br>`2.0000 − 0.0000i`<br>`2.0000 − 0.0000i`<br>`1.0000 + 0.0000i` | The IDFT of $X_k$ is computed according to Equation 7.21. The result is the discrete-time signal $x[n]=[1, 2, 2, 1]$, $0 \leq n \leq 3$. |

**Write a function that computes the IDFT of a sequence**

| Commands | Results | Comments |
|---|---|---|
| `function x=idft(Xk)`<br>`N=length(Xk);`<br>`for n=0:N−1`<br>`  for k=0:N−1`<br>`    xn(k+1)=Xk(k+1)*exp(j*2*pi*n*k/N);`<br>`  end`<br>`x(n+1)=sum(xn);`<br>`end`<br>`x=(1/N)*x;` | | The function *idft.m* is based on the code written in the previous example. The sequence $X_k$ is the input argument, while the IDFT of $X_k$ is the output argument of the function. |
| `Xk=[6, −1−j, 0, −1+j];`<br>`x=idft(Xk)` | `x = 1.0000`<br>`2.0000 − 0.0000i`<br>`2.0000 − 0.0000i`<br>`1.0000 + 0.0000i` | The function *idft.m* is executed from the command prompt and the IDFT $x[n]$ of $X_k$ is computed. |

### 3.4. Fast Fourier Transform

Computing an N-point DFT or IDFT directly from its definition can be a computationally expensive process. More specifically, looking into Equation (4) we notice that for every value that k takes, i.e., for every $X_k, k = 0, \dots, N-1$ multiplications must be performed. Thus, in order to compute the entire sequence $X_k, k = 0, \dots, N-1$ we must perform $N^2$ multiplications. If the discrete-time signal $x[n]$ is complex valued, things get difficult, since one multiplication of two complex numbers requires four multiplications between real numbers. Hence, computing the DFT (or IDFT) directly from the definition is usually too slow for real-time applications. In order to reduce the computational effort needed, an efficient algorithm (with many variants) is available for the DFT computation. This algorithm is called fast Fourier transform algorithm or FFT algorithm. FFT is based on a ‘‘divide and conquer’’ technique; that is, the original problem of N points is divided in two symmetric subproblems of N/2 points. If N/2 is even number, the problem of N/2 points is divided in two subproblems of N/4 points.

If N is a power of 2, i.e, $N = 2^P$ then only a 2-point DFT has to be computed. The DFT $X_k$ of a sequence $x[n]$ is computed in MATLAB through a FFT algorithm, with the command fft. The syntax is X = fft(x), where $x$ is the sequence $x[n]$ and $X$ is the DFT $X_k$.

| Commands | Results | Comments |
|---|---|---|
| `x=[1 2 3];`<br>`Xk=fft(x)` | `Xk = 6.0000-1.0000i`<br>`1.0981+1.3660i`<br>`-4.0981-0.3660i` | The DFT of the sequence $x[n] = [1,2,3]$, $0 \le n \le 2$ computed by the `fft` command. |
| `Xk=dft(x)` | `Xk = 6.0000-1.0000i`<br>`1.0981+1.3660i`<br>`-4.0981-0.3660i` | The DFT of the same sequence computed by the function *dft.m* that was created in Section 7.4. Of course, the result in both cases is the same. |

## Remark

FFT must not be confused with DFT. FFT is an algorithm that computes the DFT of a sequence.

An alternative syntax of the fft command is X = fft(x,N). Using this syntax we derive the N-point DFT of an M-point sequence. If $M > N$, the sequence x[n] is truncated; while if M < N, the sequence $x[n]$ is zero-padded.

| Commands | Results | Comments |
|---|---|---|
| `x=[1 2 3 4];`<br>`fft(x)` | `ans = 10.0000  -2.0000+2.0000i`<br>`-2.0000  -2.0000-2.0000i` | 4-point DFT of the sequence $x[n] = [1,2,3,4]$, $0 \le n \le M-1$, where $M = 4$. |
| `fft(x,6)` | `ans =10.0000  -3.5000-4.3301i`<br>`2.5000+0.8660i  -2.0000`<br>`2.5000-0.8660i`<br>`-3.5000+4.3301i` | Case $M < N$. 6-point DFT of the sequence $x[n] = [1,2,3,4], 0 \le n \le 3$. |
| `xx=[x 0 0];`<br>`fft(xx)` | `ans = 10.0000  -3.5000-4.3301i`<br>`2.5000+0.8660i -2.0000`<br>`2.5000-0.8660i`<br>`-3.5000+4.3301i` | The sequence $x[n]$ is padded with two zeros and its FFT is computed. In this way, we verify that by using the command `fft(x,6)` $N - M = 2$ zeros are padded at the end of $x[n]$ to convert it into a 6-point sequence. |
| `fft(x,3)` | `ans = 6.0000  -1.5000 +0.8660i`<br>`-1.5000-0.8660i` | Case $M > N$. 3-point DFT of the sequence $x[n] = [1,2,3,4], 0 \le n \le 3$. |
| `xx=[1 2 3];`<br>`fft(xx)` | `ans =6.0000  -1.5000+0.8660i`<br>`-1.5000-0.8660i` | The sequence $x[n]$ is truncated by one sample and its FFT is computed. Indeed by using the command `fft(x,3)`, the last point of $x[n]$ is discarded to convert $x[n]$ into a 3-point sequence. |

The IDFT of a sequence $X_k$ is computed by the MATLAB command *ifft*. The syntax is x = ifft(X) or x = ifft(X, N) if an N-point IDFT is required.

| Commands | Results | Comments |
|---|---|---|
| `X= [10, -2+2j, -2, -2-2j];` `ifft(X)` | `ans = 1  2  3  4` | IDFT of the sequence $X_k = [10, -2+2i, -2, -2-2i]$, $0 \le k \le N-1$, where $N = 4$. |
| `idft(X)` | `ans = 1.0000  2.0000+0.0000i  3.0000-0.0000i  4.0000-0.0000i` | The IDFT of $X_k$ is computed through the function *idft.m* that was created in Section 7.6. As expected, the two results are same. |
| `ifft(X,6)` | `ans = 0.6667  1.7113-0.0774i  1.3780-0.5000i  2.0000  1.9553-0.5000i  2.2887+1.0774i` | 6-point IDFT of $X_k$. |
| `X(6)=0` | `X = 10  -2+2i  -2  -2-2i  0  0` | $X_k$ is zero padded in order to convert it into a 6-point sequence and its IDFT is computed below. |
| `ifft(X)` | `ans = 0.6667  1.7113-0.0774i  1.3780-0.5000i  2.0000  1.9553-0.5000i  2.2887+1.0774i` | Indeed, by using the command `ifft(X,6)` $N-M=2$, zeros are padded at the end of $X_k$ to convert it into a 6-point sequence. |

## 3.5. Relationship between DFT and DTFT

In this section, we establish the relationship between the two types of Fourier transform

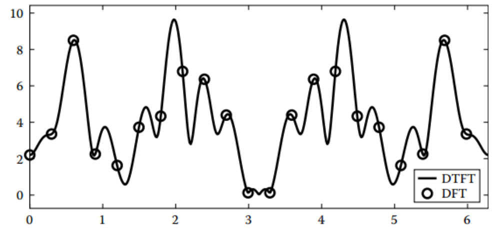
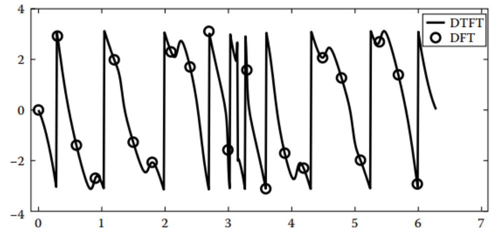
that are applicable to discrete-time signals, namely, the DFT and the DTFT. It states that the DFT $X_k$ is actually a sampling in the frequency of the DTFT $X(\omega)$, or in other words the DFT is a sampling of the continuous spectrum of a discrete-time signal. More precisely, we can state that the DFT sequence $X_k$ is equal to the DTFT $X(\omega)$ when $X(\omega)$ is evaluated at the points $\omega_k = {}^{2\pi k}/_{N}, k = 0,1,\dots,N-1$. To verify the relationship between DTFT and DFT, consider the truncated discrete-time signal $x[n] = 0.9^n, 0 \le n \le 7$. The DTFT is evaluated at the frequencies $\omega_k = {}^{2\pi k}/_{N}, k = 0,1,\dots,N-1$ and is compared to the DFT of $x[n]$.

| Commands | Results | Comments |
| --- | --- | --- |
| `n=0:7;`<br>`x=0.9.^n;`<br>`syms w`<br>`Xdtft=sum(x.*exp(-j*w*n));` | | Definition of $x[n]$ and computation of its DTFT $X(\omega)$. |
| `N=8;`<br>`k=0:N-1;`<br>`wk=2*pi*k/N;`<br>`XXdtft=subs(Xdtft,w,wk);`<br>`XXdtft.'` | `ans = 5.6953`<br>`0.3855-0.6747i`<br>`0.3147-0.2832i`<br>`0.3023-0.1176i`<br>`0.2998-0.0000i`<br>`0.3023+0.1176i`<br>`0.3147+0.2832i`<br>`0.3855+0.6747i` | The DTFT $X(\omega)$ of $x[n]$ is evaluated at the frequency points $\omega_k=2\pi k/N$, $k=0,1,\ldots,N-1$. |
| `X=fft(x)`<br>`X.'` | `ans = 5.6953`<br>`0.3855-0.6747i`<br>`0.3147-0.2832i`<br>`0.3023-0.1176i`<br>`0.2998-0.0000i`<br>`0.3023+0.1176i`<br>`0.3147+0.2832i`<br>`0.3855+0.6747i` | The DFT $X_k$ of $x[n]$ is computed, and is equal to the values that $X(\omega)$ takes for $\omega_k=2\pi k/N$, $k=0,1,\ldots,N-1$. |

## Example 6:

Let $x[n]$ be a random sequence of 21 elements. Plot in the same figure the DTFT $X(\omega)$ of $x[n]$ for $0 \le n \le 2\pi$ and the DFT of $x[n]$ versus the frequencies $\omega_k = {}^{2\pi k}/_{N}$, $k = 0,1,\ldots,N-1$.

| Commands | Results | Comments |
| --- | --- | --- |
| `x=randn(1,21);` | | Definition of a random sequence$x[n]$. |
| `n=0:20;`<br>`syms w`<br>`Xdtft=sum(x.*exp(-j*w*n));` | | The DTFT $X(\omega)$ of $x[n]$ is computed. |
| `Xdft=fft(x);` | | Computation of the DFT $X_k$ of $x[n]$. |
| `N=length(Xdft);`<br>`k=0:N-1;`<br>`wk=2*pi*k/N;` | | Computation of the frequency points $\omega_k=2\pi k/N$, $k=0,1,\ldots,N-1$. |

| Commands | Results | Comments |
|---|---|---|
| `ezplot(abs(Xdtft),[0 2*pi]);`<br>`hold on`<br>`plot(wk,abs(Xdft),'o')`<br>`hold off`<br>`legend('DTFT','DFT')` |  | First, the magnitude of DTFT is plotted over the interval $0 \le \omega \le 2\pi$. Next, the magnitude of the DFT is plotted versus the evaluated at the previous step frequency points $\omega_k$. |
| `w1=0:.01:2*pi;`<br>`XXdtft=subs(Xdtft,w,w1);`<br>`plot(w1,angle(XXdtft));`<br>`hold on`<br>`plot(wk,angle(Xdft),'o')`<br>`legend('DTFT','DFT')`<br>`hold off` |  | Graph of the phase of the two transforms. |

### 3.6. Relationship between Fourier Transform and Discrete Fourier Transform

In this section, we discuss how the CTFT is approximated by the DFT. Suppose that $\boldsymbol{X(\omega)}$ denotes the Fourier transform of a continuous-time signal x(t). The procedure followed in order to approximate $\boldsymbol{X(\omega)}$ through samples obtained from the FFT algorithm is

- The signal $x(t)$ is sampled with sampling time T; that is, we obtain the discrete-time signal $x[nT], n = 0,1,\dots,N-1$.

- The DFT $X_k, k = 0,1,\dots,N-1$ of the discrete-time signal $x[nT]$ is computed.

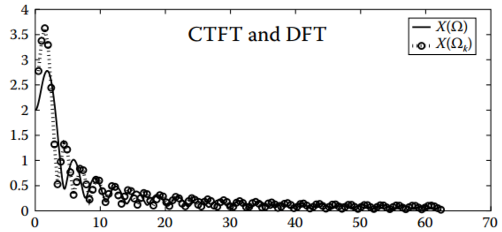
- The Fourier transform $X(\Omega)$ can be approximated at the frequencies $\Omega_k = {}^{2\pi k}/_{N}, k = 0,1,\dots,N-1$ from the DFT samples according to

$$X[\Omega_k] = NT\frac{1-e^{-j\frac{2\pi k}{N}}}{j2\pi k}X_k, k = 0,1,\dots,N-1 \qquad (10)$$

As N is getting larger or the sampling time $T$ is getting smaller, we obtain a better approximation of $X(\Omega)$ by the sequence $X(\Omega_k)$, which is computed according to (10).

**Example 7:**

Compute and plot the magnitude of the Fourier transform $\boldsymbol{X(\Omega)}$ of the continuous-time signal $\boldsymbol{x(t) = 2-3t, 0 \le t \le 2}$. Also compute and plot in the same graph the magnitude of the approximate sequence $\boldsymbol{X(\Omega_k)}$ for $\boldsymbol{N = 128}$ and $\boldsymbol{T = 0.1}$.

| Commands | Results/Comments |
|---|---|
| `T=.1;`<br>`N=128;`<br>`t=0:1/(N*T):2;`<br>`x=2-3*t;` | The sequence $x[nT]$, $n=0,1,\ldots,N-1$ is defined for $N=128$ and $T=0.1$. |
| `Xk=fft(x,N);` | Computation of the DFT $X_k$ of $x[nT]$. |
| `k=0:N-1;`<br>`wk=2*pi*k/(N*T);`<br>`Xwk=(N*T*(1-exp(-j*2*pi*k/N))./(j*2*pi*k)).*Xk` | The frequency points $\Omega_k$ are evaluated and the sequence $X(\Omega_k)$ is computed according to (7.32). |
| `syms t w`<br>`x=(2-3*t)*(heaviside(t)-heaviside(t-2));`<br>`X=fourier(x,w);` | The signal $x(t)$ is declared as symbolic expression and its Fourier transform $X(\Omega)$ is computed. |
| `ezplot(abs(X),[0 wk(N)]);`<br>`hold on`<br>`plot(wk,abs(Xwk),':o')`<br>`hold off`<br>`legend('X(\Omega)','X(\Omega_k)')`<br>`ylim([0 4])`<br>`title('CTFT and DFT')` | Graph of $X(\Omega)$ and of $X(\Omega_k)$ over the frequency interval $\Omega_0 \leq \Omega \leq \Omega_{N-1}$.<br> |



## 3.7. Linear Convolution Computation via Fast Fourier Transform

Suppose that $X_1(k)$ and $X_2(k)$ are the DFTs of the N1-point sequence $x_1[n]$ and N2-point sequence $x_2[n]$, respectively. The procedure of computing the linear convolution $x_1[n] * x_2[n]$ is

> a. Compute the $N = N_1 + N_2 - 1$ point DFTs of $x_1[n]$ and $x_2[n]$.
> b. Multiply the two DFTs.
> c. The IDFT of the product is equal to the linear convolution of $x_1[n]$ and $x_2[n]$.

| Commands | Results | Comments |
|---|---|---|
| `x1=[1 2 0 5];`<br>`x2=[3 2 1];`<br>`N1=4;`<br>`N2=3;` | `x1 = 1 2 0 5`<br>`x2 = 3 2 1` | The sequences $x_1[n]=[1,2,0,5]$, $0 \leq n \leq N_1-1$, where $N_1=4$ and $x_2[n]=[3,2,1]$, $0 \leq n \leq N_2-1$, where $N_2=3$ are defined. |
| `N=N1+N2-1;`<br>`X1=fft(x1,N);`<br>`X2=fft(x2,N);` | | The $N=N_1+N_2-1=6$ point DFTs $X_1(k)$ and $X_2(k)$ of $x_1[n]$ and $x_2[n]$, respectively, are computed. |
| `PROD=X1.*X2;` | | The product $X_1(k) \cdot X_2(k)$ is calculated. |
| `CON=ifft(PROD)` | `CON = 3 8 5 17 10 5` | Applying IDFT at the product we obtain the result of the linear convolution between $x_1[n]$ and $x_2[n]$. |
| `y=conv(x1,x2)` | `y = 3 8 5 17 10 5` | The result is confirmed by computing the linear convolution of the two sequences via the command `conv`. |

# PROCEDURE

**Problem 1:**

Plot the magnitude and the phase of the DTFT $X(\omega)$ of the signal $x[n] = cos(\pi n/3), 0 \leq n \leq 10$ over the frequency intervals $-3\pi \leq \omega \leq 3\pi$ and $-7\pi \leq \omega \leq 7\pi$.

**Problem 2:**

Plot the magnitude, the angle, the real part and the imaginary of the DFT $X_k$ of the signal $x[n] = 3cos(2\pi n/3), 0 \leq n \leq 64$.

**Problem 3:**

Consider the sequence $x[n] = 5cos(2\pi n/3), 0 \leq n \leq 19$. Plot in the same graph the DTFT of $x[n]$ over the frequency interval $0 \leq \omega \leq 2\omega$ and the DFT of $x[n]$ versus the frequency points $\omega_k = {}^{2\pi k}/_{N}, k = 0,1, \ldots, N-1$.

**Problem 4:**

Given that

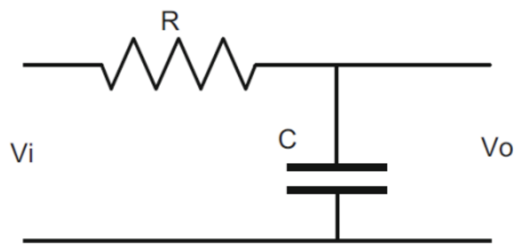



**Figure 1:**

Assume that $R = 1\ \Omega$ and $C = 0.1\ F$. Write the MATLAB code to

a) Plot the frequency response of the given circuit in Fig.1.
b) Interchange the position of $R$ to $C$ and $C$ to $R$ while keeping their values unchanged. Plot the frequency response of the new circuit. What happens?
c) Generate a square signal with frequency 7 Hz and sampling frequency as 2000 Hz. Consider this signal as input (Vi) to the circuit. Plot the response Vo.
d) Apply the same square signal in part c as input (Vi) to the circuit described in part b. Again, Plot the response Vo. Compare with the result in part c.

**Problem 5:**

Consider the sequence x[n] = $0.7^n$ , $0 \leq n \leq 49$.

**a.** Plot in the same graph the DTFT of x[n] over the frequency interval $0 \leq \omega \leq 2\pi$ and the DFT of x[n] versus the frequency points $\omega_k = 2\pi k = N, k = 0,1,\ldots,N-1$.

**b.** Compute the energy of the discrete-time signal x[n]

- At the discrete-time domain by the formula:

$$E = \sum_{n=0}^{N-1} |x[n]|^2$$

- From the DFT $X_k$ of $x[n]$ by the formula

$$E = \frac{1}{N} \sum_{k=0}^{N-1} |X_k|^2$$

- From the DTFT $X(\omega)$ of $x[n]$ by the formula:

$$E = \frac{1}{2\pi} \int_{2\pi} |X(\omega)|^2 d\omega$$