
Signal
Fourier
Problems
MATLAB
systems
z-Transform
data
signal
book
Transform
discrete-time
continuous-time
Control
FFT
Haar
course
Systems
wavelet
Analysis
Solutions
analysis
Discrete-Time
system
edition
text
time
speech
transform
representation
DTFT
coeficients
website
Sampling
State
Web
function
used
use
Representation
demos
students
digital
version
third
material
Section
shown
covered
various
Filters
Models
Design
downloaded
Professor
transfer
Laplace
previous
Toolbox
model
waveforms
aspects
linear
electrical
University
begins
DFT
properties
Prentice
Version
illustrations
control
Continuous-Time
new
devices
one-quarter
problems
theory
Simulink
treatment
Application
Preface

Contents

Chapter 1

Fourier Series

The frequency spectrum is a complex-valued function of the frequency variable, and thus it is usually specified in terms of an amplitude spectrum and a phase spectrum [1]. The complex exponential form is given by:

$$\begin{aligned} x(t) &= \sum_{-\infty}^{\infty} X_n e^{jn\omega_0 t} \\ X_n &= \int_{-T/2}^{T/2} x(t) e^{-jn\omega_0 t}\, dt \end{aligned} \tag{1.1}$$

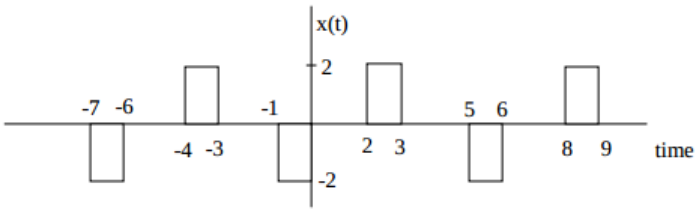
In the next exercises we will also be using the first and/or the second derivative of the previous expressions (**??**), and therefore we write them here explicitly:

$$\begin{aligned} \dot{x}(t) &= \sum_{-\infty}^{\infty} jn\omega_0 X_n e^{jn\omega_0 t} \\ jn\omega_0 X_n &= \int_{-T/2}^{T/2} \dot{x}(t) e^{-jn\omega_0 t}\, dt \end{aligned} \tag{1.2}$$

$$\ddot{x}(t) = \sum_{-\infty}^{\infty} -n^2\omega_0^2 X_n e^{jn\omega_0 t}$$
$$-n^2\omega_0^2 X_n = \int_{-T/2}^{T/2} \dot{x}(t) e^{-jn\omega_0 t}\, dt \tag{1.3}$$

Problem 1

For the following signal:



a) Find the Fourier series.

b) Plot the spectra versus frequency, $\omega = n\omega_0$.

Solution

The period of the shown signal is $T = 6$ and therefore $\omega_0 = \frac{2\pi}{T} = \frac{\pi}{3}$.

Taking the derivative of $x(t)$ we get:

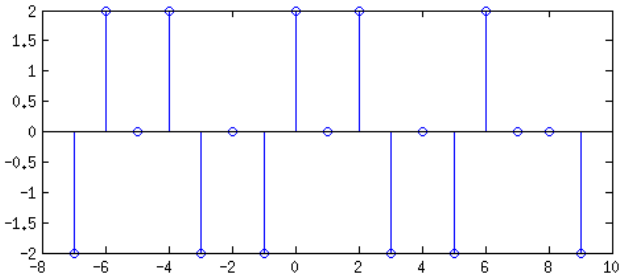


Figure 1.1: Derivative $\dot{x}$

The range t = [-3,3] contains one complete period of the signal. Usign (1.2) we have:

$$\dot{x}(t) = 2(-\delta(t+1) + \delta(t) + \delta(t-2) - \delta(t-3))$$

The Fourier coefficients are obtained with:

$$\begin{aligned} jn\omega_0 X_n &= \frac{2}{6}\int_{-3}^{3}(-\delta(t+1)+\delta(t)+\delta(t-2)-\delta(t-3))e^{-jn\omega_0 t}\,dt \\ &= \frac{1}{3}(-e^{jn\omega_0}+1+e^{-2jn\omega_0}-e^{-3jn\omega_0}) \\ &= \frac{1}{3}[e^{\frac{jn\omega_0}{2}}(e^{\frac{-jn\omega_0}{2}}-e^{\frac{jn\omega_0}{2}})-e^{\frac{-5jn\omega_0}{2}}(e^{\frac{-jn\omega_0}{2}}-e^{\frac{jn\omega_0}{2}})] \\ &= \frac{1}{3}[(e^{\frac{-jn\omega_0}{2}}-e^{\frac{jn\omega_0}{2}})(e^{-jn\omega_0}(e^{\frac{3jn\omega_0}{2}}-e^{\frac{-3jn\omega_0}{2}}))] \\ &= \frac{4j}{3}[\sin\frac{n\omega_0}{2}\sin\frac{3n\omega_0}{2}e^{-jn\omega_0}] \\ X_n &= \frac{-4j}{n\pi}[\sin(n\frac{\pi}{6})\sin(n\frac{\pi}{2})e^{-jn\frac{\pi}{3}}] \end{aligned}$$

Next we use Matlab to plot the magnitude and phase of the spectra using the script given in [2]

Listing 1.1: Calculate and plot magnitude and phase of Xn

```
n=1:15;
Xn=-4*j./n/pi.*sin(pi*n/6).*sin(n*pi/2).*exp(-j*n*pi/3);
n=-15:-1;
X_n=-4*j./n/pi.*sin(pi*n/6).*sin(n*pi/2).*exp(-j*n*pi/3);
Xn=[X_n 0 Xn];
n=-15:15;
```

```
subplot(211),stem(n,abs(Xn));
title('|X_n|')
subplot(212),stem(n,angle(Xn))
title('angle(X_n) in rad')
```

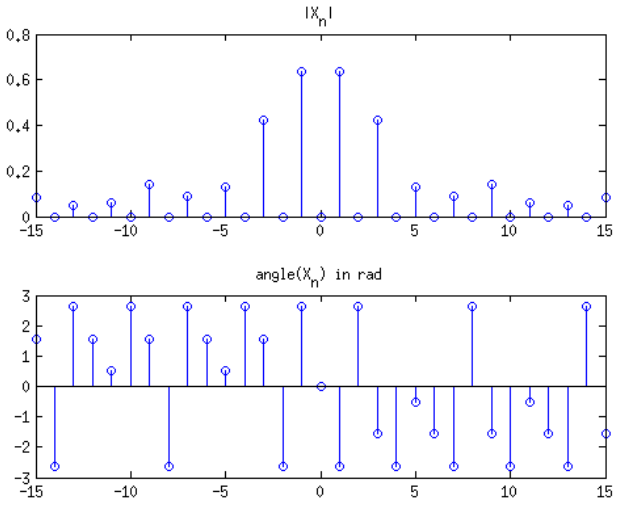


Figure 1.2: Magnitude and Angle X_n

We then plot the approximation of the function using its Fourier coefficients [2].

Listing 1.2: Approximation of x(t) with Fourier coefficients

```
function [x,t] = fapprox(N,T)
    t = -1.5*T:T/1000:1.5*T;
    w0 = 2*pi/T;
    X0 = 0;
    x = X0*ones(1,length(t)); % dc component
    for n=1:N,
     Xn = -4*j/n/pi*sin(pi*n/6)*sin(n*pi/2)*exp(-j*n*pi/3);
     X_n = conj(Xn);
     x = x + Xn*exp(j*n*w0*t) + X_n*exp(-j*n*w0*t);
    end
end
```

We do this for N=5 and N=50:

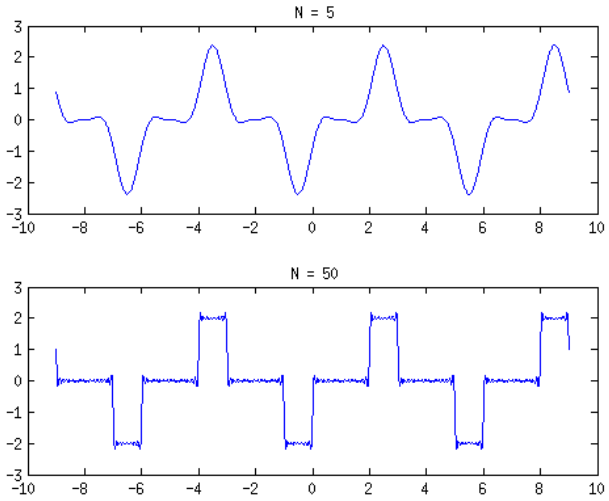
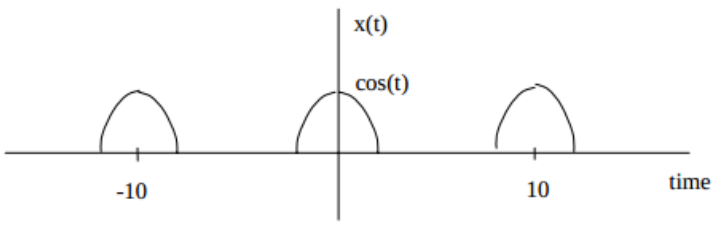


Figure 1.3: Approximation of x(t) by Xn

As we can see, the larger the N the more close to the original function we get. However, as a consequence of the Gibbs effect, we can't say that it'll be equal.

Problem 2

Repeat problem 1 for the following signal:



Solution

The period of the shown signal is $T = 10$ and therefore $\omega_0 = \frac{2\pi}{T} = \frac{\pi}{5}$.

If we take the first and second derivative of $x(t)$ we get:

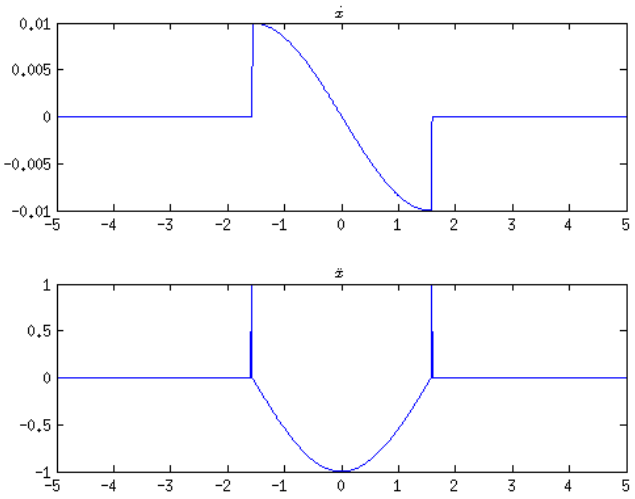


Figure 1.4: Derivative $\dot{x}$

The range t = [-5,5] contains one complete period of the signal. Applying (1.2) we have:

$$\ddot{x}(t) = -x(t) + \delta(t + \pi/2) + \delta(t - \pi/2)$$

$$\sum_{-\infty}^{\infty} -n^2\omega_0^2 X_n e^{jn\omega_0 t} = \sum_{-\infty}^{\infty} X_n e^{jn\omega_0 t} + \delta(t + \pi/2) + \delta(t - \pi/2)$$

$$\sum_{-\infty}^{\infty} (1 - n^2\omega_0^2) X_n e^{jn\omega_0 t} = \delta(t + \pi/2) + \delta(t - \pi/2)$$

Whe can now obtain X_n with:

$$
\begin{aligned}
(1-n^2\omega_0^2)X_n &= \frac{1}{T}\int_{-5}^{5} \delta(t+\pi/2)+\delta(t-\pi/2)e^{-jn\omega_0 t}\,dt \\
&= \frac{1}{T}(e^{jn\frac{\omega_0}{2}}+e^{-jn\frac{\omega_0}{2}}) \\
X_n &= \frac{1}{5(1-\frac{n^2\pi^2}{25})}\cos(\frac{n\pi^2}{10})
\end{aligned}
$$

Next we use Matlab to plot the magnitude and phase of the spectra using the script given in [2]

Listing 1.3: Calculate and plot magnitude and phase of Xn

```
n=-10:10;
Xn=cos(pi/2*n*w0)/5./(1-(n*w0).^2);
subplot(121),stem(n,abs(Xn))
title('|X_n|')
subplot(122),stem(n,angle(Xn))
title('angle(X_n) in rad')
```

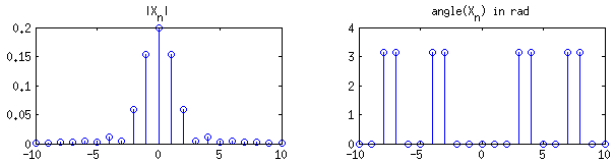


Figure 1.5: Magnitude and Angle X_n

We then plot the approximation of the function using its Fourier coefficients [2].

Listing 1.4: Approximation of x(t) with Fourier coefficients

```
function [x,t] = fapprox2(N,T)
    w0 = 2*pi/T;
    t = -1.5*T:T/1000:1.5*T;
    c0 = 1/5;
    x = c0*ones(1,length(t)); % dc component
    for n=1:N,
     cn = cos(pi/2*n*w0)/5/(1-(n*w0)^2);
     c_n = cn;
     x = x + cn*exp(j*n*w0*t) + c_n*exp(-j*n*w0*t);
    end
    plot(t,x)
    title([' N = ',num2str(N)])
end
```

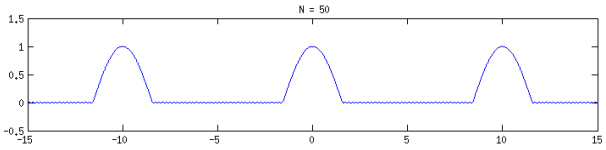


Figure 1.6: Approximation of x(t) by Xn

Problem 3

Compute the Fourier series for the following signals:

1. $x(t) = 2 + 4\cos(50t + \pi/2) + 12\cos(100t - \pi/3)$

Solution

Lets recall from [1] that:

$$x(t) = a_0 + \sum_{k=1}^{\infty} A_k \cos(k\omega_0 t + \theta_k) \ \ -\infty < t < \infty$$

And the equivalent coefficients between the trigonometric series and the exponential form are:

$$\begin{aligned} X_0 &= a_0 \\ |X_n| &= \frac{1}{2} A_k \ , k = 1, 2, .. \\ \angle X_n &= \theta_k \ , k = 1, 2, .. \end{aligned} \tag{1.4}$$

From (1.4) we can immediately calculate the X_n coefficients:

Listing 1.5: Plot Magnitude and Angle of X_n

```
fe = [-100 -50 0 50 100];
Ae = [6 2 2 2 6];
pe = [pi/3 -pi/2 0 pi/2 -pi/3];

figure(1);
subplot(2,1,1), stem(fe,Ae);
grid on;
```

```
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|X_n|', 'fontweight', 'bold');

subplot(2,1,2), stem(fe,pe);
grid on;
xlabel('Frec [rad/s]');
ylabel('Phase');
title('$\angle x$','interpreter','latex')
```

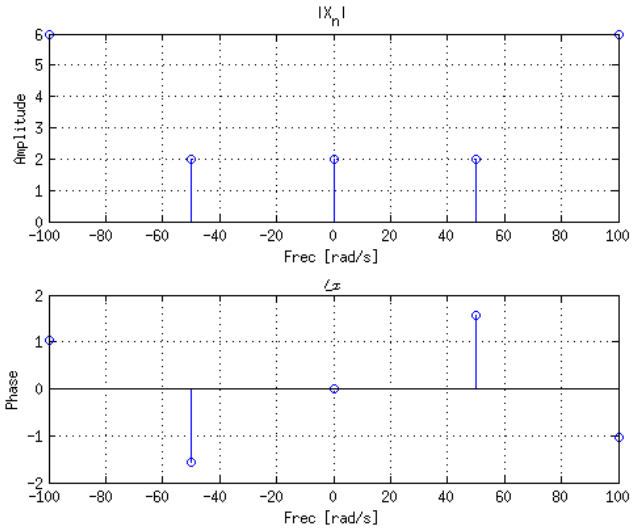


Figure 1.7: Magnitude and Angle of X_n

2. $x(t) = 4\cos(2\pi(1000)t)\cos(2\pi(750000)t)$

Solution

$$x(t) = 4\left(\frac{e^{j2\pi(1000)t} + e^{-j2\pi(1000)t}}{2}\right)\left(\frac{e^{j2\pi(750000)t} + e^{-j2\pi(750000)t}}{2}\right)$$
$$= e^{j2\pi(751000)t} + e^{j2\pi(-749000)t} + e^{j2\pi(749000)t} + e^{j2\pi(-751000)t}$$

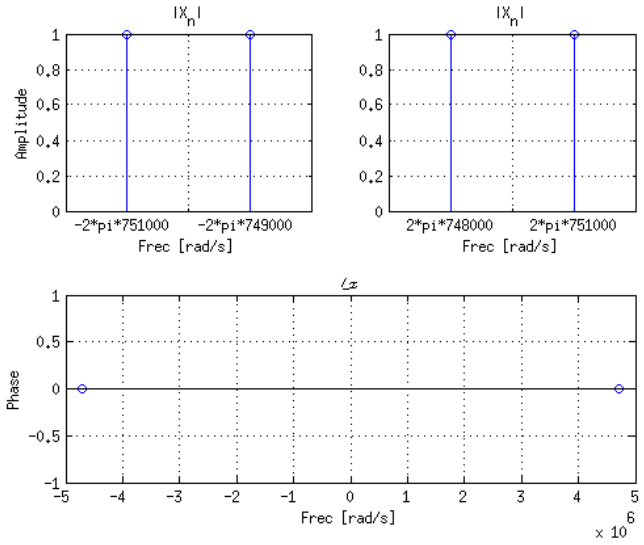
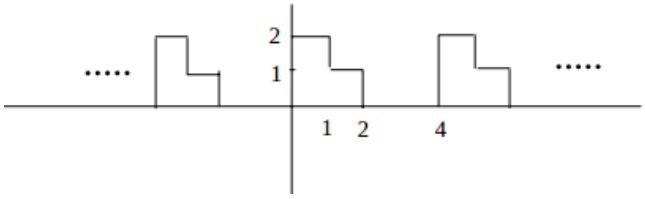


Figure 1.8: Magnitude and Angle of X_n

3. The function:



Solution

The period of the shown signal is $T = 4$ and therefore $\omega_0 = \frac{2\pi}{T} = \frac{\pi}{2}$.

Taking the derivative of the function we get:

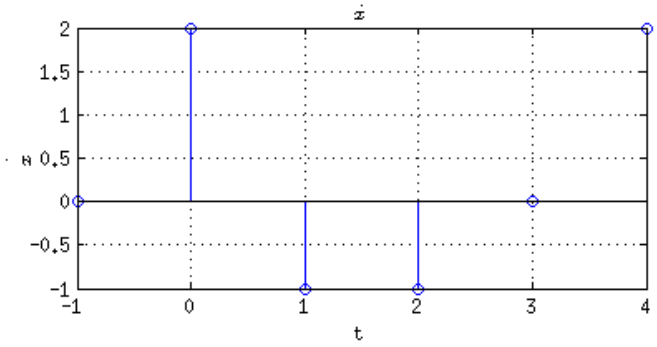


Figure 1.9: Derivative $\dot{x}$

In the range $[-1, 3]$ we have:

$$\dot{x}(t) = 2\delta(t) - \delta(t-1) - \delta(t-2)$$

Applying (1.2) we have:

$$\begin{aligned} jn\omega_0 X_n &= \frac{1}{T}\int_{-T/2}^{T/2} [2\delta(t) - \delta(t-1) - \delta(t-2)]e^{-jn\omega_0 t}\, dt \\ &= \frac{1}{4}[2 - e^{-jn\frac{\pi}{2}} - e^{nj\pi}] \end{aligned}$$

We can use the following properties:

$$\begin{aligned} e^{-jn\pi} &= (e^{-j\pi})^n = [\cos(\pi) - j\sin(\pi)]^n = (-1)^n \\ e^{-jn\frac{\pi}{2}} &= (e^{-j\frac{\pi}{2}})^n = [\cos(\frac{\pi}{2}) - j\sin(\frac{\pi}{2})]^n = (-j)^n \end{aligned}$$

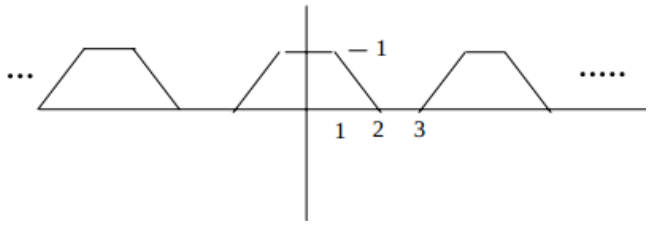
And reduce the equation to:

$$X_n = \frac{1}{2jn\pi}[2 - (-1)^n - (-j)^n]$$

where

$$X_0 = \frac{1}{4}\int_{-1}^{3} x(t)\, dt = \frac{3}{4}$$

4. The function:



Solution

The period of the shown signal is $T = 5$ and therefore $\omega_0 = \frac{2\pi}{T} = \frac{2\pi}{5}$.

Taking the derivative of the function we get:

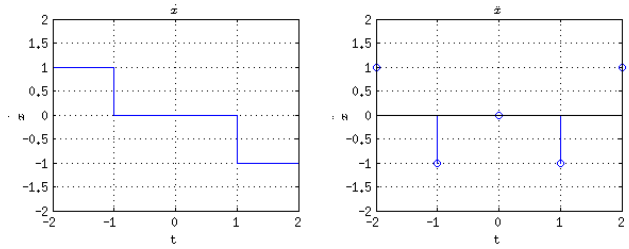


Figure 1.10: First and Second Derivatives $\dot{x}$ $\ddot{x}$

In the range $[-2, 2]$ we have:

$$\ddot{x}(t) = \delta(t+2) - \delta(t+1) - \delta(t-1) + \delta(t-2)$$

Applying (1.3) we have:

$$
\begin{aligned}
-n^2\omega_0^2 X_n &= \frac{1}{T}\int_{-T/2}^{T/2}[\delta(t+2)-\delta(t+1)-\delta(t-1)+\delta(t-2)]e^{-jn\omega_0 t}\,dt \\
&= \frac{1}{5}[(e^{2jn\omega_0}+e^{-2jn\omega_0})-(e^{jn\omega_0}+e^{-jn\omega_0})] \\
X_n &= \frac{2}{5n^2\omega_0^2}[\cos(n\omega_0)-\cos(2n\omega_0)]
\end{aligned}
$$

We can calculate the dc component by finding the area of the trapezoid:

$$
\begin{aligned}
X_0 &= \frac{1}{T}\int_{-T/2}^{T/2} x(t)\,dt \\
&= \frac{1}{5}\frac{B.b}{2}h = \frac{3}{5}
\end{aligned}
$$

Problem 4

For the signals given in Problem 3c) and 3d), use Matlab to plot the truncated Fourier series for N = 3, N = 10 and N= 40. (Use subplot to save paper).

Solution

1. For problem 3c:

Listing 1.6: Approximation of x(t) with Fourier coefficients

```
function [x,t] = fapprox3(N,T)
    t = -1.5*T:T/1000:1.5*T;
    w0 = 2*pi/T;
    X0 = 3/4;

    n_p = [1:N];
    n_n = [-N:-1];
    Xn = (2 - (-1).^n_p - (-j).^n_p) ./ (2*j*pi.*n_p);
    X_n = (2 - (-1).^n_n - (-j).^n_n) ./ (2*j*pi.*n_n);

    Xn = [X_n X0 Xn];
    n = [n_n 0 n_p];
```

```
13
14      x = Xn*exp(j*w0*n'*t);
15      x = real(x);
16  end
```

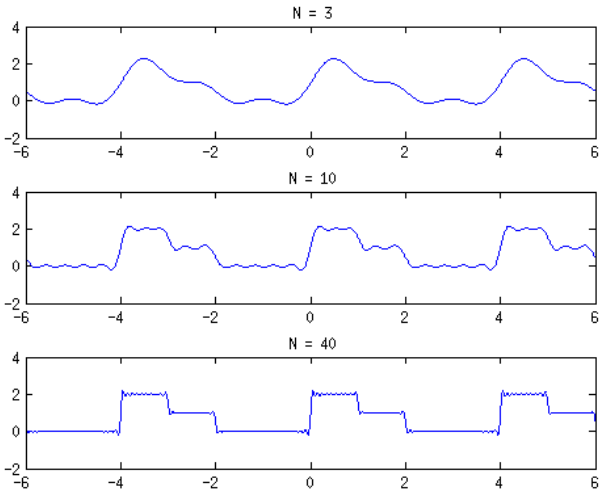


Figure 1.11: Approximation of x(t) by Xn for N=[3,10,30]

2. For problem 3d:

Listing 1.7: Approximation of x(t) with Fourier coefficients

```
1   function [x,t] = fapprox4(N,T)
2       t = -1.5*T:T/1000:1.5*T;
3       w0 = 2*pi/T;
4       X0 = 3/5;
5
6       n_p = [1:N];
7       n_n = [-N:-1];
8       Xn = (cos(n_p * w0) - cos(2*n_p * w0) ) * 2 ./ (5*w0
            ^2*n_p.^2);
9       X_n = (cos(n_n * w0) - cos(2*n_n * w0) ) * 2 ./ (5*w0
            ^2*n_n.^2);
10
11
12      Xn = [X_n X0 Xn];
13      n = [n_n 0 n_p];
14
15      x = Xn*exp(j*w0*n'*t);
16      x = real(x);
17  end
```

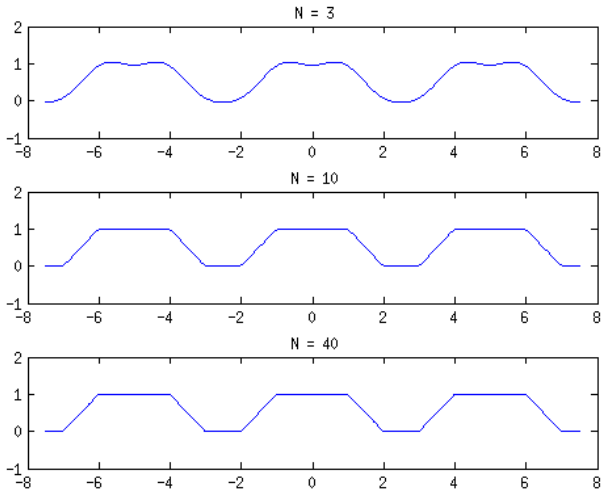
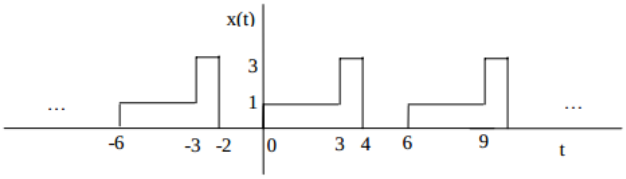


Figure 1.12: Approximation of x(t) by Xn for N=[3,10,30]

Problem 5

Find the Fourier series for the following signal.



Also, sketch the approximation if a large number of terms are kept in the series (say N=30).

Solution

The period of the shown signal is $T = 6$ and therefore $\omega_0 = \frac{2\pi}{T} = \frac{\pi}{3}$.

Taking the derivative of the function we get:

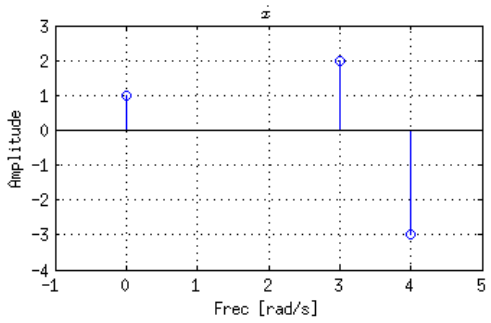


Figure 1.13: Derivative $\dot{x}$

In the range $[-1, 5]$ we have:

$$\dot{x}(t) = \delta(t) + 2\delta(t-3) - 3\delta(t-4)$$

Applying (1.2) we have:

$$\begin{aligned} jn\omega_0 X_n &= \frac{1}{T}\int_{-T/2}^{T/2} [\delta(t) + 2\delta(t-3) - 3\delta(t-4)]e^{-jn\omega_0 t}\, dt \\ &= \frac{1}{6}[1 + 2e^{-2jn\omega_0} - 3e^{-4nj\omega_0}] \\ &= \frac{1}{6}[1 + 2(-1)^n - 3e^{-\frac{4}{3}jn\pi}] \\ X_n &= \frac{1}{2jn\pi}[1 + 2(-1)^n - 3e^{-\frac{4}{3}jn\pi}] \end{aligned}$$

where

$$X_0 = \frac{1}{6}\int_{-1}^{3} x(t)\, dt = 1$$

The plot for approximating the function using its Fourier coefficients is:

Listing 1.8: Approximation of x(t) with Fourier coefficients

```
function [x,t] = fapprox5(N,T)
    t = -1.5*T:T/1000:1.5*T;
    w0 = 2*pi/T;
    X0 = 1;
```

```
5
6       n_p = [1:N];
7       n_n = [-N:-1];
8       Xn = (1 + 2*(-1).^n_p - 3*exp(-4*j*n_p*w0) ) ./ (6*j*n_p*w0
            );
9       X_n = (1 + 2*(-1).^n_n - 3*exp(-4*j*n_n*w0) ) ./ (6*j*n_n*
            w0);
10
11      Xn = [X_n X0 Xn];
12      n = [n_n 0 n_p];
13
14      x = Xn*exp(j*w0*n'*t);
15      x = real(x);
16  end
```

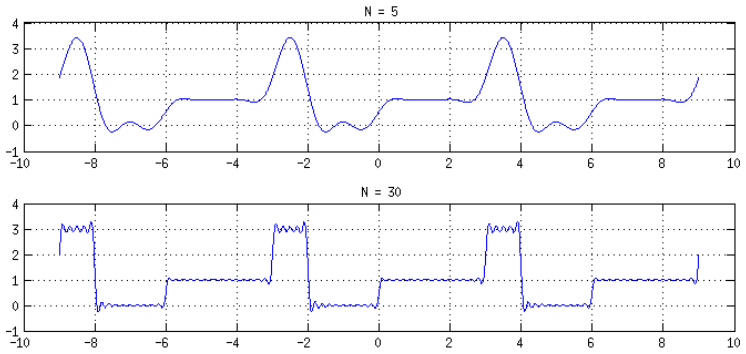


Figure 1.14: Approximation of x(t) by Xn

Chapter 2

Fourier Transform

When we want to know the description of aperiodic signals in terms of the frecuency content we need to use the Fourier Transform. The frecuency components of this non-periodic signals are defined for all reall values of the frecuency variable of ω and not just for discrete values as in the case of periodic ones in which we used the Fourier Series [1].

The Fourier Transform and its inverse of an aperiodic signal are defined as:

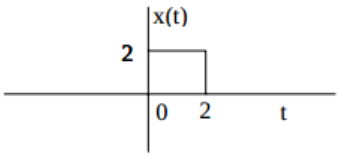
$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} x(t) e^{-j\omega t}\, dt \\ x(t) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t}\, d\omega \end{aligned} \tag{2.1}$$

In the next exercises we will also be using the following properties:

$$x(t) \Leftrightarrow X(\omega) \tag{2.2a}$$
$$x(t - t_0) \Leftrightarrow e^{-j\omega t_0} X(\omega) \tag{2.2b}$$
$$e^{j\omega_0 t} x(t) \Leftrightarrow X(\omega - \omega_0) \tag{2.2c}$$
$$\frac{dx(t)}{dt} \Leftrightarrow j\omega X(\omega) \tag{2.2d}$$
$$x(\alpha t) \Leftrightarrow \frac{1}{|\alpha|} X\left(\frac{\omega}{a}\right) \tag{2.2e}$$

For each signal, find the Fourier transform, $X(\omega)$, and then plot $|X(\omega)|$ (note, you may want to use MATLAB for the plot in 3.)

Problem 1



Solution

Taking the derivative of $x(t)$ we get:

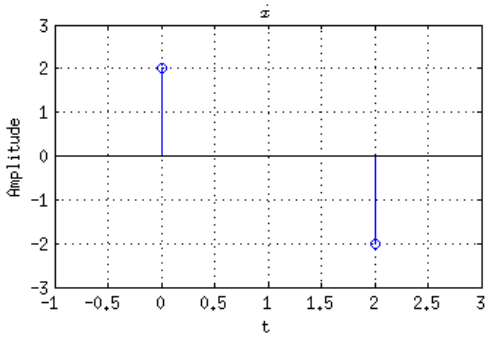


Figure 2.1: Derivative $\dot{x}$

Applying 2.1 and 2.2c to $\dot{x}$ we have:

$$\begin{aligned} j\omega X(\omega) &= 2\int_{-\infty}^{\infty} (\delta(t) - \delta(t-2))e^{-j\omega t}\, dt \\ &= 2[1 - e^{-2j\omega}] \\ &= 2e^{-j\omega}[e^{-j\omega} - e^{-j\omega}] \\ &= 4je^{-j\omega}\sin(\omega) \\ X(\omega) &= 4\frac{\sin(\omega)}{\omega}e^{-j\omega} \\ &= 4Sa(\omega)e^{-j\omega} \end{aligned}$$

The plot of the magintude and angle of $X(\omega)$ is:

Listing 2.1: Plot of Magnitude and Angle

```
LIM = 3*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 4 * sin(w)./w .* exp(-j * w);

subplot(1,2,1), plot(w,abs(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');

subplot(1,2,2), plot(w,angle(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

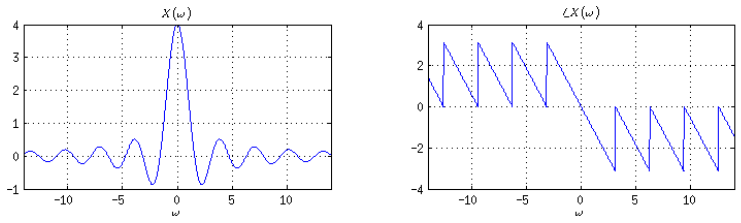
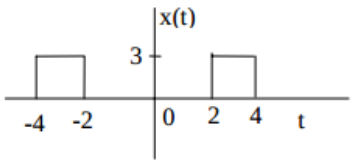


Figure 2.2: Magnitude $|X(\omega)|$ and Angle

Problem 2



Solution

Lets call $x_{p1}(t)$ and $X_{p1}(\omega)$ the function in time domain of the first point and its Fourier Transform respecively. We can express our function in terms of such function as:

$$x(t) = \frac{3}{2}[x_{p1}(t-2) + x_{p1}(t+4)]$$

As we can see from (2.2b) the displacement in time is reflected in the frecuency domain as a multiplication by an exponential.

$$\begin{aligned} X(\omega) &= \frac{3}{2}X_{p1}(\omega)[e^{-2j\omega} + e^{4j\omega}] \\ &= \frac{3}{2}X_{p1}(\omega)e^{j\omega}[e^{-3j\omega} + e^{3j\omega}] \\ &= 12Sa(\omega)\cos(3\omega) \end{aligned}$$

The plot of the magintude and angle of $X(\omega)$ is:

Listing 2.2: Plot of Magnitude and Angle

```
LIM = 1.5*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 12 * sin(w)./w .* cos(3*w);

subplot(1,2,1), plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');

subplot(1,2,2), plot(w,angle(Xw));
```

```
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

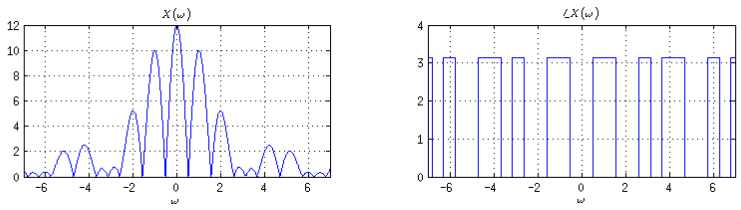
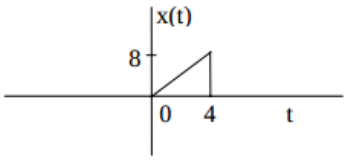


Figure 2.3: Magnitude $|X(\omega)|$ and Angle

Problem 3



Solution

Taking the derivative of $x(t)$ we get:

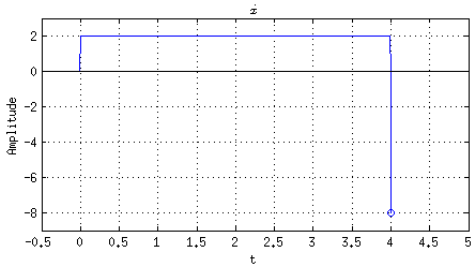


Figure 2.4: Derivative $\dot{x}$

Lets call $x_{p1}(t)$ and $X_{p1}(\omega)$ the function in time domain of the first point and its Fourier Transform respecively. We can express the first derivative of our function in terms of such function as:

$$\dot{x}(t) = x_{p1}(t/2) - 8\delta(t-4)$$

As we can see from (2.2e) the scale in time is reflected inversely in the frecuency domain.

$$\begin{aligned} j\omega X(\omega) &= X_{p1}(2\omega) - 8e^{-4j\omega} \\ &= 4Sa(2\omega)e^{-2j\omega} - 8e^{-4j\omega} \\ X(\omega) &= \frac{4j}{\omega}e^{-2j\omega}[2e^{-2j\omega} - Sa(2\omega)] \end{aligned}$$

The plot of the magintude and angle of $X(\omega)$ is:

Listing 2.3: Plot of Magnitude and Angle

```
LIM = 2*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 4*j./w .* exp(-2*j*w) .* (2*exp(-2*j*w) - sin(2*w)./(2*w)
    );

subplot(1,2,1), plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');

subplot(1,2,2), plot(w,angle(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

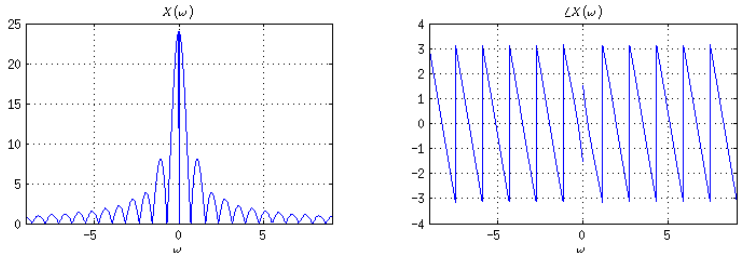


Figure 2.5: Magnitude $|X(\omega)|$ and Angle

Problem 4

$x(t) = cos(200t)G_4(t)$

Solution

The generic Fourier Transform for the gate function $G_\tau(t)$ is:

$$\begin{aligned} \mathfrak{F}\{G_\tau(t)\} &= \tau Sa\left(\frac{\omega\tau}{2}\right) \\ \mathfrak{F}\{G_4(t)\} &= 4Sa(2\omega) \end{aligned} \tag{2.3}$$

We can express our function as:

$$x(t) = \frac{1}{2}[e^{200t} + e^{-200t}]G_4(t)$$

As we can see from (2.2c) the displacement in frecuency is reflected in the time domain as a multiplication by an exponential.

$$X(\omega) = 2[Sa(2(\omega - 200)) + Sa(2(\omega + 200))]$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.4: Plot of Magnitude

```
1  Sa=@(x) sin(x)./x;
2
3  LIM = 100*pi;
4  w = -1.5*LIM:LIM/1000:1.5*LIM;
5  Xw = 2 * ( Sa(2*(w-200)) + Sa(2*(w+200)) );
6
7  plot(w,abs(real(Xw)));
8  grid on;
9  xlim([-round(1.5*LIM) round(1.5*LIM)]);
10 xlabel('$\omega$','interpreter','latex');
11 title('$|X(\omega)|$','interpreter','latex');
```

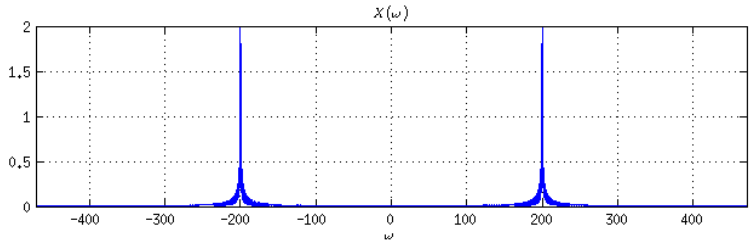


Figure 2.6: Magnitude $|X(\omega)|$

Problem 5

$x(t) = e^{-3t}cos(10t)u(t)$

Solution

From the 1st result of the Fourier Transforms table we have:

$$\mathfrak{F}\{e^{-\alpha t}u(t)\} = \frac{1}{\alpha + j\omega} \tag{2.4}$$

Rearranging the equation for $x(t)$ we have:

$$\begin{aligned} x(t) &= \frac{1}{2}[e^{-3t}e^{10jt} + e^{-3t}e^{-10jt}]u(t) \\ &= \frac{1}{2}[e^{-t(3-10j)} + e^{-t(3+10j)}]u(t) \end{aligned}$$

We can now apply directly the result from (2.5):

$$\begin{aligned} X(\omega) &= \frac{1}{2}\left[\frac{1}{(3-10j)+j\omega} + \frac{1}{(3+10j)+j\omega}\right] \\ &= \frac{1}{2}\left[\frac{1}{3+(\omega-10)j} + \frac{1}{3+(\omega+10)j}\right] \end{aligned}$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.5: Plot of Magnitude

```
LIM = 10*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 0.5 * ( 1./(3+(w-10)*j) + 1./(3+(w+10)*j));

plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\|X(\omega)\|$','interpreter','latex');
```

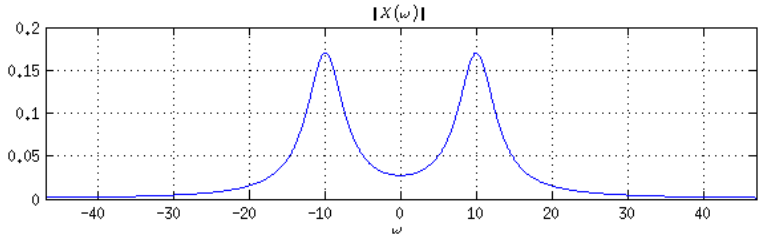
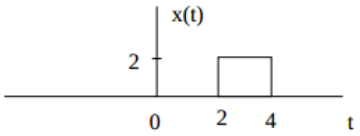


Figure 2.7: Magnitude $|X(\omega)|$

Problem 6

Find the Fourier transform of the following signals. Sketch $|X(\omega)|$ in each case.

1. :



Solution

Lets call $x_{p1}(t)$ and $X_{p1}(\omega)$ the function in time domain of the first point and its Fourier Transform respecively. We can express our function in terms of such function as:

$$x(t) = x_{p1}(t-2)$$

As we can see from (2.2b) the displacement in time is reflected in the frecuency domain as a multiplication by an exponential.

$$\begin{aligned} X(\omega) &= X_{p1}(\omega)e^{-2j\omega} \\ &= 4Sa(\omega)e^{-3j\omega} \end{aligned}$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.6: Plot of Magnitude

```
1  Sa=@(x) sin(x)./x;
2
3  LIM = -0.8*pi;
4  w = -1.5*LIM:LIM/1000:1.5*LIM;
5  Xw = 4 * Sa(w) .* exp(-3*j*w);
6
7  plot(w,abs(real(Xw)));
8  grid on;
9  xlim([-round(1.5*LIM) round(1.5*LIM)]);
10 xlabel('$\omega$','interpreter','latex');
11 title('$|X(\omega)|$','interpreter','latex');
```

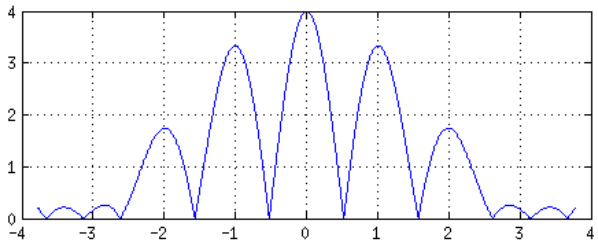


Figure 2.8: Magnitude $|X(\omega)|$

2. $x(t) = 2e^{-2t}u(t)$

Solution

Directly from (2.5) we have:

$$X(\omega) = \frac{2}{2 + j\omega}$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.7: Plot of Magnitude

```
LIM = 2*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 2./(2 + j*w);

plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');
```

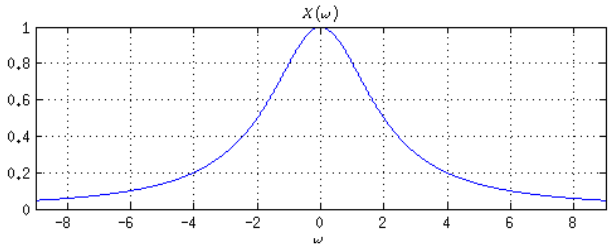


Figure 2.9: Magnitude $|X(\omega)|$

3. $x(t) = 5e^{-5t}u(t)$

Solution

Directly from (2.5) we have:

$$X(\omega) = \frac{5}{5 + j\omega}$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.8: Plot of Magnitude

```
LIM = 3*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 5./(5 + j*w);

plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');
```

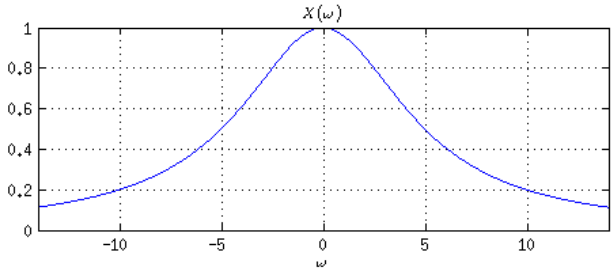


Figure 2.10: Magnitude $|X(\omega)|$

4. $x(t) = e^{-2t}\cos(4t)u(t)$

Solution

$$x(t) = \frac{1}{2}\left[e^{-2t}e^{4jt}u(t) + e^{-2t}e^{-4jt}u(t)\right]$$

Directly from (2.5) we have:

$$\begin{aligned} X(\omega) &= \frac{1}{2}\left[\frac{1}{(2-4j)+j\omega} + \frac{1}{(2+4j)+j\omega}\right] \\ &= \frac{1}{2}\left[\frac{1}{2+(\omega-4)j} + \frac{1}{2+(\omega+4)j}\right] \end{aligned}$$

The plot of the magintude of $X(\omega)$ is:

Listing 2.9: Plot of Magnitude

```
LIM = 2*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
```

```
3  Xw = 0.5* (1./(2 + (w-4)*j) + 1./(2 + (w+4)*j));
4
5  plot(w,abs(real(Xw)));
6  grid on;
7  xlim([-round(1.5*LIM) round(1.5*LIM)]);
8  xlabel('$\omega$','interpreter','latex');
9  title('$|X(\omega)|$','interpreter','latex');
```

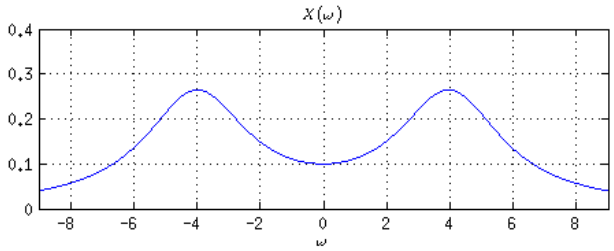
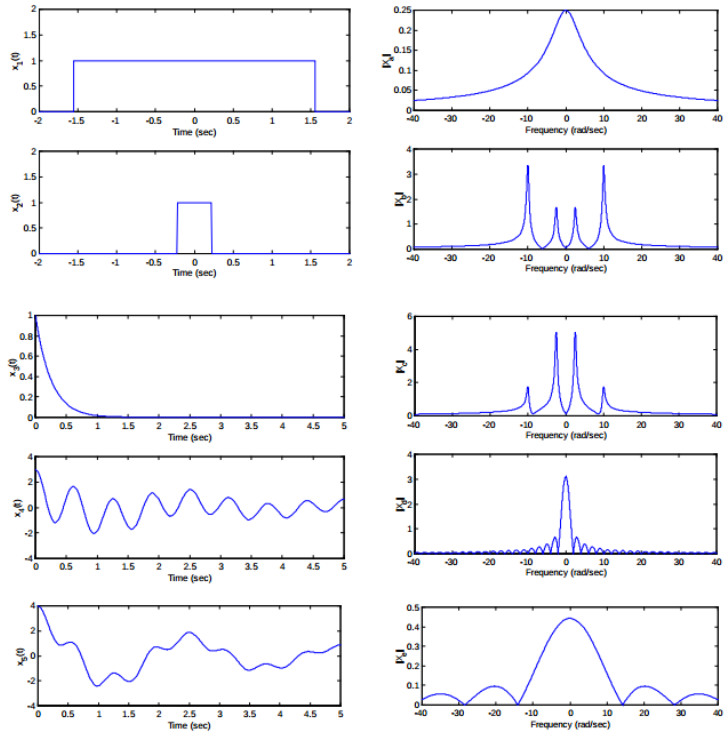


Figure 2.11: Magnitude $|X(\omega)|$

Problem 7

Match the time responses with the corresponding frequency responses.



Solution

As stated in (2.3) the Fourier Transform of a Gate function is a Sa. The 1st gate is more spread in time that the 2nd and therefore its corresponding transform must be more narrow.

$1 \Leftrightarrow d$

$2 \Leftrightarrow e$

The 3rd plot corresponds to a negative exponential truncated by a step function. We know for (2.5) and for Points 6b and 6c that the corresponding transform corresponds to a figure like (a).

$3 \Leftrightarrow a$

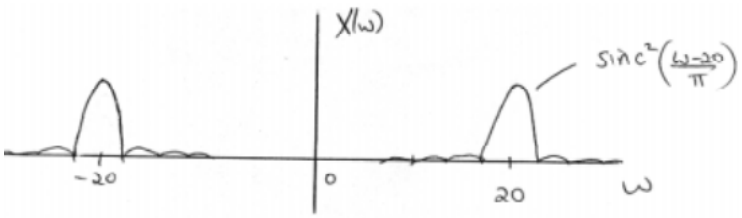
From the following plots we see that the frecuency of 5 (rad/s) is more dominant in the last one and the 10 (rad/s) is more dominant in the previous one.

$4 \Leftrightarrow b$

$5 \Leftrightarrow c$

Problem 8

Compute the inverse Fourier transform of the following signal



Solution

From the 14th result of the Fourier Transforms table we have:

$$Tr_\tau(t) = \begin{cases} 1 - \frac{|t|}{\tau}, & \text{if } |t| < \tau \\ 0, & \text{if } |t| > \tau \end{cases}$$

$$\begin{aligned} \mathfrak{F}\{Tr_\tau(t)\} &= \tau \left[Sa\left(\frac{\omega\tau}{2}\right)\right]^2 \\ \mathfrak{F}\{Tr_2(t)\} &= 2Sa(\omega)^2 \end{aligned} \tag{2.5}$$

First, we transform sinc into a known function like Sa:

$$sinc^2\left(\frac{\omega}{\pi}\right) = Sa^2(\omega)$$

As we saw in Point 4, when we multiply by a $\cos$ in time we are adding 2 components of the function displaced simetrically with half of its amplitude. The displacement in the plot is of 20, therefore we need to multiply by a $\cos(20t)$. We end up having:

$$x(t) = Tr_2(t)\cos(20t)$$

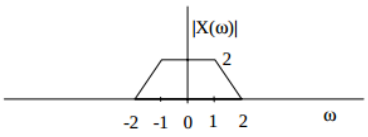
Chapter 3

Sampling and Reconstruction

Problem 1

Draw $|Xs(\omega)|$ for the following cases if $x_s(t) = x(t)p(t)$ with sampling period T.

$$p(t) = \sum_{n=-\infty}^{n=\infty} \delta(t - nT).$$



Solution

The minimum sampling period of the signal according to the Nyquist theorem is:

$$\omega_m = 2$$
$$\omega_{Ns} >= 2\omega_m >= 4$$
$$T_{Ns} = \frac{2\pi}{\omega_s} <= \frac{\pi}{2}$$

- $T_s = \pi/4sec$

This sampling period is lower than the minimum required and therefore no aliasing will occur as can be seen in (3.1).

$$\omega_s = \frac{2\pi}{T_s} = 8$$

Listing 3.1: Plot of $|X_s(\omega)|$

```
t1=[-2:0.01:-1];
x1=[2*t1+4];
t2=[-1:0.01:1];
x2=[0*t2+2];
t3=[1:0.01:2];
x3=[-2*t3+4];

t=[t1 t2 t3];
x=[x1 x2 x3];

ws = 8;
t = [t-ws t t+ws];
x = [x x x];

plot(t,x);
grid on;
ylim([-0.5 2.5]);
xlabel('$\omega$','interpreter','latex');
title('$|X_s(\omega)|$','interpreter','latex');
```

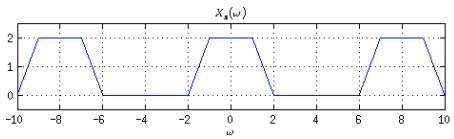


Figure 3.1: Sampling $|X_s(\omega)|$

- $T_s = \pi/2sec$

This sampling period is equal than the minimum required and therefore is in the limit of no aliasing as can be seen in (3.2).

$$\omega_s = \frac{2\pi}{T_s} = 4$$

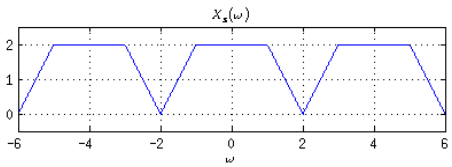


Figure 3.2: Sampling $|X_s(\omega)|$

- $T_s = 2\pi/3sec$

This sampling period is lower than the minimum required and therefore aliasing will occur as can be seen in (3.3).

$$\omega_s = \frac{2\pi}{T_s} = 4$$

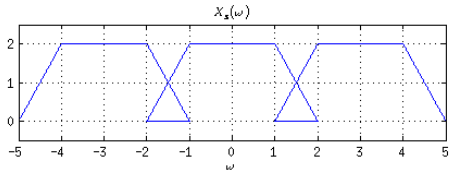


Figure 3.3: Sampling $|X_s(\omega)|$

Problem 2

Repeat Problem 1 where $x(t) = e^{-t/4}cos(t)u(t)$

In order to examine the effects of aliasing in the time domain, plot x(t) for each of the sampling times for t=0 to 15 sec. In MATLAB, this is done by defining your time vector with the time increment set to the desired sampling period. MATLAB then ”reconstructs” the signal by connecting the sampled points with straight lines (this is known as a linear interpolation). Compare your sampled/reconstructed signals with a signal that is more accurate, one that is created by using a very small sampling period (such as T = 0.05 sec) by plotting them on the same graph.

Solution

First we calculate $X(\omega)$.

$$x(t) = \frac{1}{2}\left[e^{-t/4}e^{jt}u(t) + e^{-t/4}e^{-jt}u(t)\right]$$

Directly from (2.5) we have:

$$\begin{aligned} X(\omega) &= \frac{1}{2}\left[\frac{1}{(1/4+j)+j\omega} + \frac{1}{(1/4-j)+j\omega}\right] \\ &= \frac{1}{2}\left[\frac{1}{1/4+(\omega+1)j} + \frac{1}{1/4+(\omega-1)j}\right] \end{aligned}$$

The plot of the magintude of $X(\omega)$ is:

Listing 3.2: Plot of Magnitude

```
LIM = 0.5*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 0.5* (1./(0.25 + (w-1)*j) + 1./(0.25 + (w+1)*j));

plot(w,abs(real(Xw)));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');
```

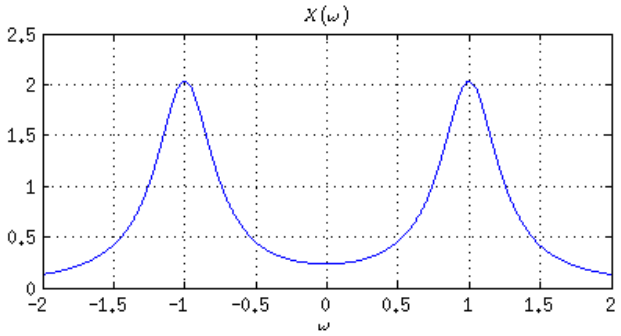


Figure 3.4: Magnitude $|X(\omega)|$

- $T_s = \pi/4 sec$

This sampling period is lower than the minimum required and therefore no aliasing will occur as can be seen in (3.5).

$$\omega_s = \frac{2\pi}{T_s} = 8$$

Listing 3.3: Plot of $|X_s(\omega)|$

```
LIM = 2.5*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 0.5* (1./(0.25 + (w-1)*j) + 1./(0.25 + (w+1)*j));
Xw = abs(real(Xw));

ws = 8;
w = [w-ws w w+ws];
Xw = [Xw Xw Xw];

plot(w,Xw);
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
ylim([-0.5 2.5]);
xlabel('$\omega$','interpreter','latex');
title('$|X_s(\omega)|$','interpreter','latex');
```

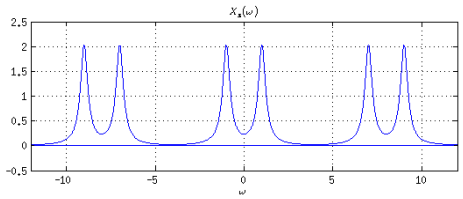


Figure 3.5: Sampling $|X_s(\omega)|$

- $T_s = \pi/2 sec$

This sampling period is equal than the minimum required and therefore is in the limit of no aliasing as can be seen in (3.6).

$$\omega_s = \frac{2\pi}{T_s} = 4$$

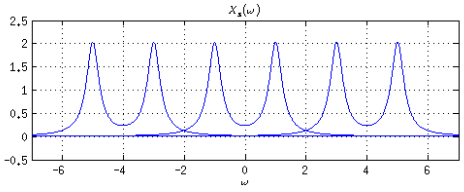


Figure 3.6: Sampling $|X_s(\omega)|$

- $T_s = 2\pi/3sec$

This sampling period is lower than the minimum required and therefore aliasing will occur as can be seen in (3.7).

$$\omega_s = \frac{2\pi}{T_s} = 4$$

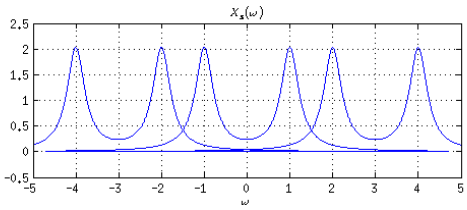


Figure 3.7: Sampling $|X_s(\omega)|$

Plot of ”reconstructed” x(t) with MATLAB:

Listing 3.4: Plot of x(t) for different T

```
u = @(x)(x>=0);

i=1;
for T=[0.05, pi/4, pi/2, 2*pi/3]
    t=[0:T:15];
    x=exp(-t/4).*cos(t).*u(t);

    subplot(4,1,i), plot(t,x);
    grid on;
    xlabel('t');
    title(['Reconstructed x(t) with T=' num2str(T)]);
    i = i + 1;
end
```

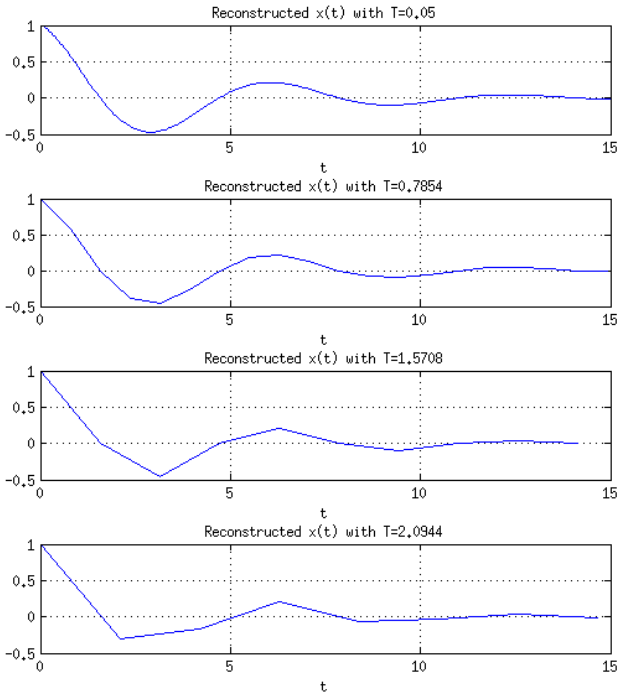
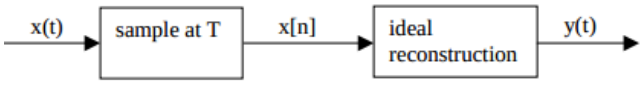


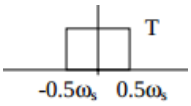
Figure 3.8: Reconstructed x(t)

Problem 3

Consider the following sampling and reconstruction configuration:



The output y(t) of the ideal reconstruction can be found by sending the sampled signal $x_s(t) = x(t)p(t)$ through an ideal lowpass filter:



Let $x(t) = 2 + cos(50\pi t)$ and T = 0.01 sec.

- Draw $|X_s(\omega)|$ where $x_s(t) = x(t)p(t)$. Determine if aliasing occurs.

Solution

First we calculate $X(\omega)$.

$$x(t) = 2(1) + \frac{1}{2}\left[e^{50\pi jt}(1) + e^{-50\pi jt}(1)\right]$$

Using this known result $\mathfrak{F}\{1\} = 2\pi\delta(\omega)$ and from (2.2c) we have:

$$X(\omega) = 4\pi\delta(\omega) + \pi\left[\delta(\omega - 50\pi) + \delta(\omega + 50\pi)\right]$$

The plot of the magintude of $X(\omega)$ is:

Listing 3.5: Plot of Magnitude

```
fe = [-50*pi 0 50*pi];
Ae = [pi 4*pi pi];

figure(1);
stem(fe,Ae);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|X_\omega|', 'fontweight', 'bold');
```

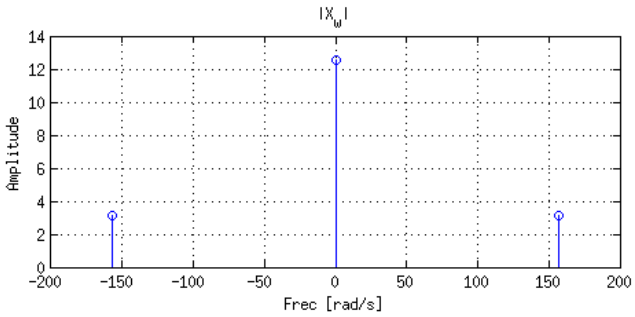


Figure 3.9: Magnitude $|X(\omega)|$

The minimum sampling period of the signal according to the Nyquist theorem is:

$$\begin{aligned} \omega_m &= 50\pi \\ \omega_{Ns} >= 2\omega_m &>= 100\pi \\ T_{Ns} = \frac{2\pi}{\omega_s} &<= \frac{1}{50} <= 0.02 \end{aligned}$$

Since $T_s = 0.01sec$ is lower than the minimum required no aliasing will occur as can be seen in (3.10).

$$\omega_s = \frac{2\pi}{T_s} = 200\pi$$

Listing 3.6: Plot of $|X_s(\omega)|$

```
w = [-50*pi 0 50*pi];
Xw = [pi 4*pi pi];

ws = 200*pi;
w = [w-ws w w+ws];
Xw = [Xw Xw Xw];

stem(w,Xw);
grid on;
%xlim([-50*pi-2 50*pi+2]);
xlabel('$\omega$','interpreter','latex');
title('$|X_s(\omega)|$','interpreter','latex');
```

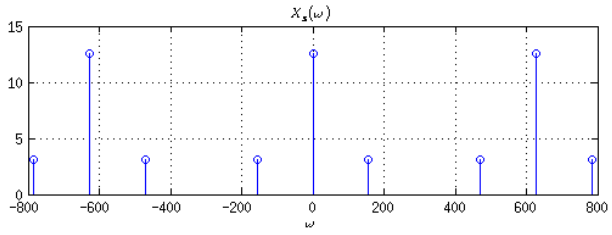


Figure 3.10: Sampling $|X_s(\omega)|$

- Determine the expression for y(t).

Solution

The limits of the lowpass filter are $-0.5\omega_s = -100\pi$ to $0.5\omega_s = 100\pi$. The Fourier transform of $Y(\omega) = H(\omega)X(\omega)$ is:

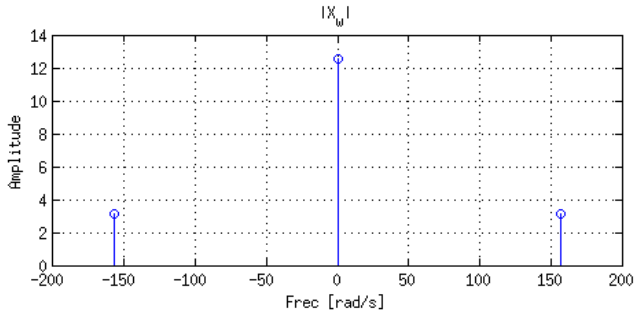


Figure 3.11: Magnitude $|X(\omega)|$

And therefore:

$$y(t) = 2 + cos(50\pi t)$$

- Determine an expression for x[n].

Solution

$$\begin{aligned} x(n) &= 2 + cos(50\pi n T_s) \\ &= 2 + cos(0.5\pi n) \end{aligned}$$

Problem 4

Repeat Problem 3 for $x(t) = 2 + \cos(50\pi t)$ and T = 0.025 sec.

- Draw $|X_s(\omega)|$ where $x_s(t) = x(t)p(t)$. Determine if aliasing occurs.

Solution

Since $T_s = 0.025sec$ is greater than the minimum required, aliasing will occur as can be seen in (3.12).

$$\omega_s = \frac{2\pi}{T_s} = 80\pi$$

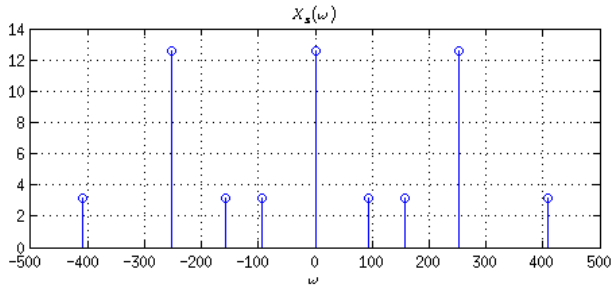


Figure 3.12: Sampling $|X_s(\omega)|$

- Determine the expression for y(t).

Solution

The limits of the lowpass filter are $-0.5\omega_s = -40\pi$ to $0.5\omega_s = 40\pi$. The Fourier transform of $Y(\omega) = H(\omega)X(\omega)$ is:

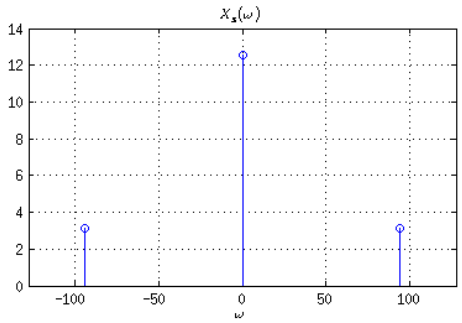


Figure 3.13: Magnitude $|X(\omega)|$

And therefore:

$$y(t) = 2 + \cos(30\pi t)$$

- Determine an expression for x[n].

Solution

$$\begin{aligned} x(n) &= 2 + \cos(50\pi n T_s) \\ &= 2 + \cos(1.25\pi n) \end{aligned}$$

Problem 5

Repeat Problem 3 for $x(t) = 1 + \cos(20\pi t) + \cos(60\pi t)$ and T = 0.01 sec.

- Draw $|X_s(\omega)|$ where $x_s(t) = x(t)p(t)$. Determine if aliasing occurs.

Solution

First we calculate $X(\omega)$.

$$X(\omega) = \pi[2\delta(\omega) + \delta(\omega - 20\pi) + \delta(\omega + 20\pi) + \delta(\omega - 60\pi) + \delta(\omega + 60\pi)]$$

The plot of the magintude of $X(\omega)$ is:

Listing 3.7: Plot of Magnitude

```
w = [-60*pi -20*pi 0 20*pi 60*pi];
Xw = [pi pi 2*pi pi pi];

figure(1);
stem(w,Xw);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|X_\omega|', 'fontweight', 'bold');
```

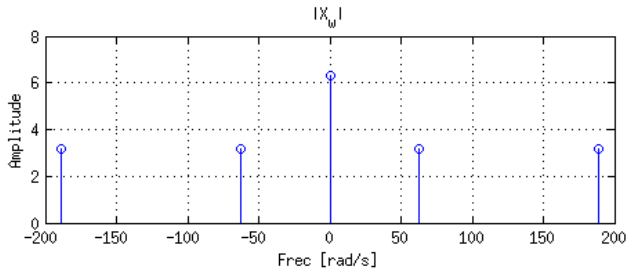


Figure 3.14: Magnitude $|X(\omega)|$

The minimum sampling period of the signal according to the Nyquist theorem is:

$$\begin{aligned} \omega_m &= 60\pi \\ \omega_{Ns} >&= 2\omega_m >= 120\pi \\ T_{Ns} &= \frac{2\pi}{\omega_s} <= \frac{1}{50} <= 0.01666 \end{aligned}$$

Since $T_s = 0.01sec$ is lower than the minimum required no aliasing will occur as can be seen in (3.15).

$$\omega_s = \frac{2\pi}{T_s} = 200\pi$$

Listing 3.8: Plot of $|X_s(\omega)|$

```
w = [-60*pi -20*pi 0 20*pi 60*pi];
Xw = [pi pi 2*pi pi pi];

ws = 200*pi;
w = [w-ws w w+ws];
Xw = [Xw Xw Xw];

figure(1);
stem(w,Xw);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|X_\omega|', 'fontweight', 'bold');
```

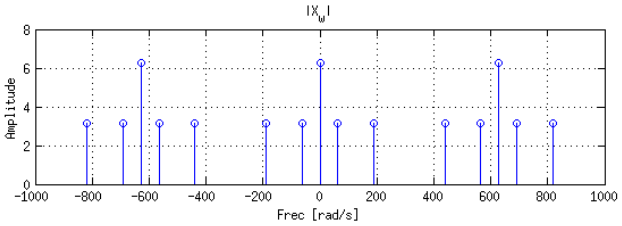


Figure 3.15: Sampling $|X_s(\omega)|$

- Determine the expression for y(t).

Solution

The limits of the lowpass filter are $-0.5\omega_s = -100\pi$ to $0.5\omega_s = 100\pi$. The Fourier transform of $Y(\omega) = H(\omega)X(\omega)$ is:

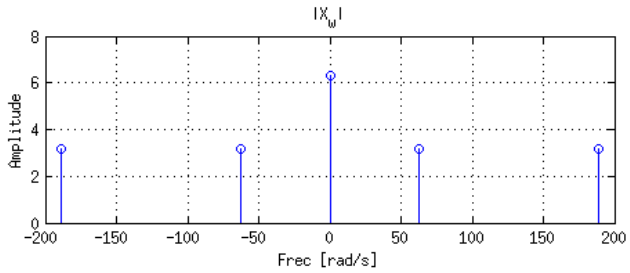


Figure 3.16: Magnitude $|X(\omega)|$

And therefore:

$$y(t) = x(t) = 1 + \cos(20\pi t) + \cos(60\pi t)$$

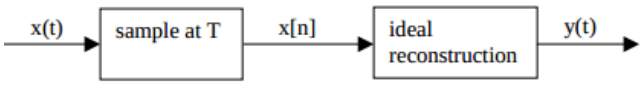
- Determine an expression for x[n].

Solution

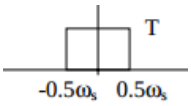
$$\begin{aligned} x(n) &= 1 + \cos(20\pi n T_s) + \cos(60\pi n T_s) \\ &= 2 + \cos(0.2\pi n) + \cos(0.6n) \end{aligned}$$

Problem 6

Consider the following sampling and reconstruction configuration:



The output y(t) of the ideal reconstruction can be found by sending the sampled signal $x_s(t) = x(t)p(t)$ through an ideal lowpass filter:



- Let $x(t) = 1 + cos(15\pi t)$ and T = 0.1 sec. Draw $|X_s(\omega)|$ where $x_s(t) = x(t)p(t)$. Determine the expression for y(t).

Solution

First we calculate $X(\omega)$.

$$X(\omega) = \pi[2\delta(\omega) + \delta(\omega - 15\pi) + \delta(\omega + 15\pi)]$$

The plot of the magintude of $X(\omega)$ is:

Listing 3.9: Plot of Magnitude

```
w = [-15*pi 0 15*pi];
Xw = [pi 2*pi pi];

figure(1);
stem(w,Xw);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|X_\omega|', 'fontweight', 'bold');
```

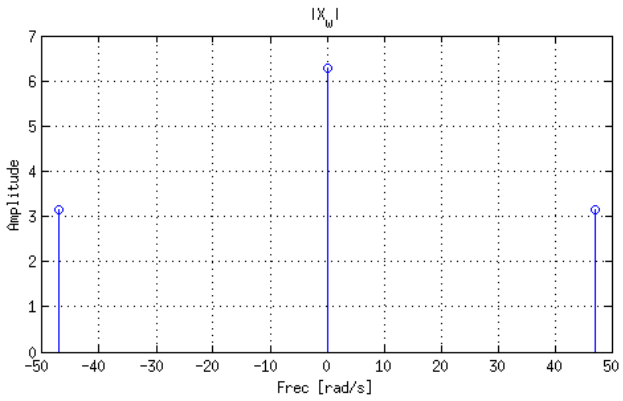


Figure 3.17: Magnitude $|X(\omega)|$

The minimum sampling period of the signal according to the Nyquist theorem is:

$$\begin{aligned} \omega_m &= 15\pi \\ \omega_{Ns} >= 2\omega_m &>= 30\pi \\ T_{Ns} = \frac{2\pi}{\omega_s} &<= \frac{1}{15} <= 0.066 \end{aligned}$$

Since $T_s = 0.1sec$ is greater than the minimum required, aliasing will occur as can be seen in (3.18).

$$\omega_s = \frac{2\pi}{T_s} = 20\pi$$

Listing 3.10: Plot of $|X_s(\omega)|$

```
w = [-15*pi 0 15*pi];
Xw = [pi 2*pi pi];

ws = 20*pi;
w = [w-ws w w+ws];
Xw = [Xw Xw Xw];

stem(w,Xw);
grid on;
%xlim([-10*pi-2 10*pi+2]);  %passband filter
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('|x_\omega|', 'fontweight', 'bold');
```

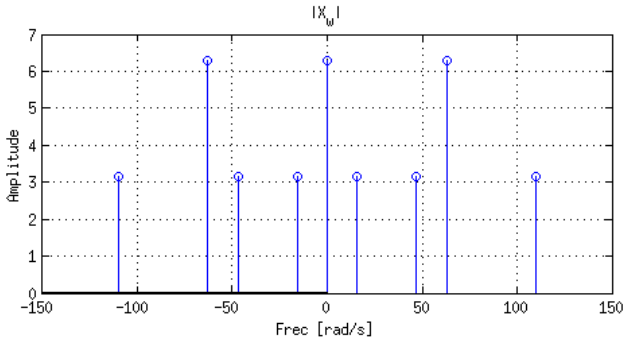


Figure 3.18: Sampling $|X_s(\omega)|$

The limits of the lowpass filter are $-0.5\omega_s = -10\pi$ to $0.5\omega_s = 10\pi$. The

Fourier transform of $Y(\omega) = H(\omega)X(\omega)$ is:

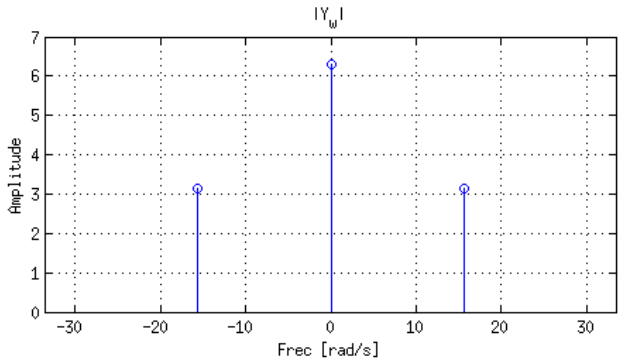


Figure 3.19: Magnitude $|X(\omega)|$

And therefore only the $\omega = 5\pi$ passes through it:

$$y(t) = 1 + cos(5\pi t)$$

- Let $X(\omega) = \frac{1}{(j\omega+1)}$ and T = 1 sec. Draw $|X_s(\omega)|$ where $x_s(t) = x(t)p(t)$. Does aliasing occur? (Justify your answer).

Solution

The magnitude of $X(\omega)$ is:

$$|X(\omega)| = \frac{1}{\omega^2 + 1}$$

From the previous equation we can see that the function never intesects the w axis, and therefore there will always be aliasing. However, setting w_m to its FWHM we have:

$$\frac{1}{\omega_m^2 + 1} = \frac{1}{2}$$
$$\omega_m = \sqrt{3}$$

Hipoteticaly taking this value as w_m we see that $w_s = 2\pi > 2w_m = 2\sqrt{3}$ and in this case there will not be aliasing.

The plot of the magintude of $X(\omega)$ and $X_s(\omega)$ is:

Listing 3.11: Plot of Magnitude

```
LIM = 1.5*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 1./(j*w + 1);

subplot(2,1,1), plot(w,abs(real(Xw)));
grid on;
xlabel('$\omega$','interpreter','latex');
title('$|X(\omega)|$','interpreter','latex');

%FWHM
hold on
stem(sqrt(3),1./(j*sqrt(3) + 1))

ws = 2*pi;
w = [w-ws w w+ws];
Xw = [Xw Xw Xw];

subplot(2,1,2), plot(w,abs(real(Xw)));
grid on;
%xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|X_s(\omega)|$','interpreter','latex');

%FWHM
hold on
stem(sqrt(3),1./(j*sqrt(3) + 1))
```

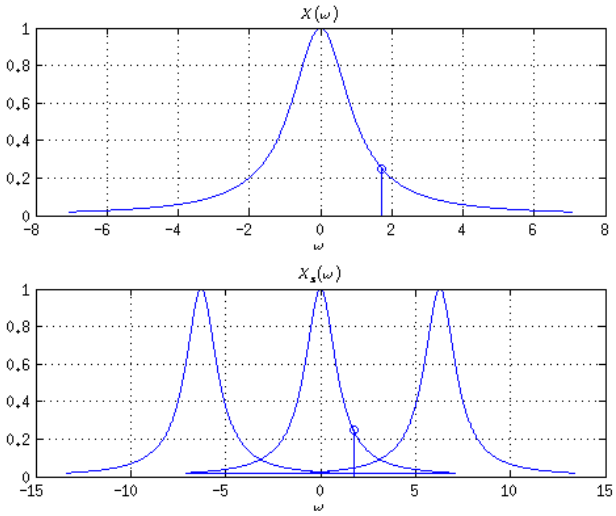


Figure 3.20: Magnitude

Chapter 4

DTFT and DFT

When we move into a discrete time domain, the Fourier Transform can be calculated using the DTFT or Discrete Time Fourier Transform which is defined as [1]:

$$X(\omega) = \mathfrak{F}\{x[n]\} = \sum_{-\infty}^{\infty} x[n]e^{-j\omega n} \tag{4.1}$$

As can be seen in (4.1), the frecuency domain obtained is continuous. However, in order to manipulate (save/restore) such information in a computer, the frecuency should also be discretized. For time-frecuency discrete domains, the DFT is defined over the time interval from $n = 0$ to $n = N$ as:

$$X_k = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi k}{N}n} \tag{4.2}$$

where the frecuency ω was redefined as $\omega = \frac{2\pi k}{N}$.

The following function was implemented to calculate the $k = [0, 1, ..N]$ values for the Transform. Additionally, an extra parameter N was added to force the input array to be of a bigger size. This will be used in some problems to extend the size of a discrete function $x[n]$ assuming that it fades to 0 for the missing values.

Listing 4.1: Fourier Discrete Transform with 0 padding implementation

```
%
% Discrete fourier transform <my version>
%
function xk = dft2(xn, N)

    %If N is given and is greater than xn, then do 0 padding
    if N > length(xn)
        xn = [xn zeros(1, N - length(xn))];
    else
        N = length(xn);
    end

    n = [0:N-1];
    k = [0:N-1];

    %create a matrix of nxk with all the combinations
    expnk = exp( (-2*pi*j/N) * (n'*k) );

    xk = xn*expnk;
end
```

Problem 1

Compute the DTFT of the following signals and sketch $X(\omega)$.

- $x[n] = \left[\begin{array}{cccc} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{array} \right]$

Solution

Aplying the definition of (4.1) we have:

$$\begin{aligned} X(\omega) &= \sum_{0}^{3} \frac{1}{4} e^{-j\omega n} \\ &= \frac{1}{4}[1 + e^{-jw} + e^{-2jw} + e^{-3jw}] \\ &= \frac{1}{4} e^{-j\frac{3\omega}{2}} [e^{j\frac{3\omega}{2}} + e^{-j\frac{\omega}{2}} + e^{-j\frac{\omega}{2}} + e^{-j\frac{3\omega}{2}}] \\ &= \frac{1}{2} e^{-j\frac{3\omega}{2}} [\cos(\frac{3\omega}{2}) \cos(\frac{\omega}{2})] \end{aligned} \tag{4.3}$$

The plot of the magintude $X(\omega)$ is:

Listing 4.2: Plot of Magnitude

```
LIM = 2.3;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 0.5 * exp(-j * 1.5 * w) .* ( cos(w*1.5) + cos(w*0.5) );

subplot(2,1,1), plot(w,abs(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\|X(\omega)\|$','interpreter','latex');

subplot(2,1,2), plot(w,angle(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

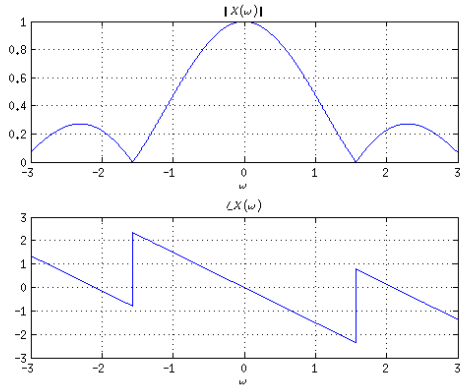


Figure 4.1: Magnitude $|X(\omega)|$

- $x[n] = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$

Solution

Aplying the definition of (4.1) we have:

$$
\begin{aligned}
X(\omega) &= \sum_{0}^{2} x[n]e^{-j\omega n} \\
&= [1 - 2e^{-jw} + e^{-2jw}] \\
&= e^{-jw}[e^{jw} - 2 + e^{-jw}] \\
&= e^{-jw})2[\cos(\omega) - 1] \\
&= 2e^{-jw}[\cos(\omega) - 1]
\end{aligned}
\tag{4.4}
$$

The plot of the magintude $X(\omega)$ is:

Listing 4.3: Plot of Magnitude

```
LIM = 0.7*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 2 * exp(-j*w) .* (cos(w)-1);

subplot(2,1,1), plot(w,abs(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\|X(\omega)\|$','interpreter','latex');

subplot(2,1,2), plot(w,angle(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

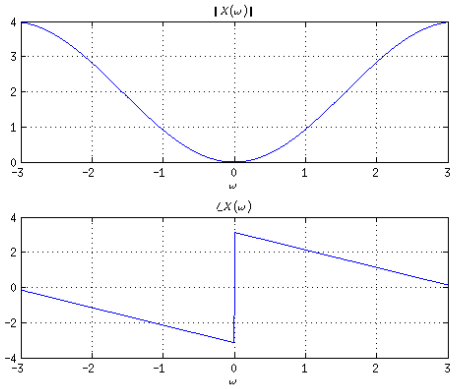


Figure 4.2: Magnitude $|X(\omega)|$

- $x[n] = 2\left(\frac{3}{4}\right)^n u[n]$

Solution

$$X(\omega) = \sum_{0}^{\infty} 2\left(\frac{3}{4}\right)^{n} e^{-j\omega n}$$

Recall that the geometric series solution is given by:

$$\sum_{0}^{\infty} ar^{n} = \frac{a}{1-r} \text{ if } |r| < 1 \tag{4.5}$$

Then, aplying (4.1) and (4.5) we have:

$$\begin{aligned} X(\omega) &= \sum_{0}^{\infty} 2\left(\frac{3}{4}e^{-j\omega}\right)^{n} \\ &= \frac{2}{1-\frac{3}{4}e^{-j\omega}} \end{aligned}$$

The plot of the magintude $X(\omega)$ is:

Listing 4.4: Plot of Magnitude

```
LIM = 0.7*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Xw = 2 ./(1 - 3/4*exp(-j*w) );

subplot(2,1,1), plot(w,abs(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\|X(\omega)\|$','interpreter','latex');

subplot(2,1,2), plot(w,angle(Xw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle X(\omega)$','interpreter','latex');
```

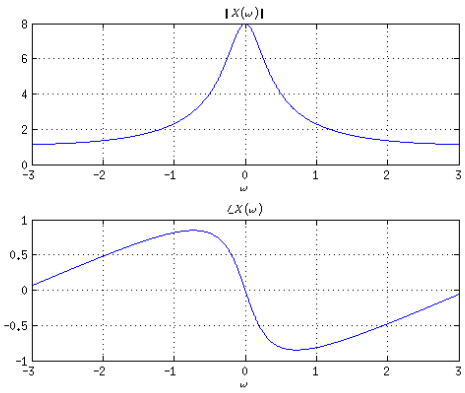


Figure 4.3: Magnitude $|X(\omega)|$

Problem 2

Compute by hand calculations the DFT of the signals given in Problem 1-a) and 1-b) and compare your answers to those found in Problem 1.

- $x[n] = \left[\begin{array}{cccc} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{array} \right]$

Solution

Using the function dft2 over the input vector we get:

$$X_k = \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array} \right)$$
$$X_0 = 1$$
$$X_1 = 0$$
$$X_2 = 0$$
$$X_3 = 0$$

To compare the results, we calculate the values from the analytic function obtained in 4.3 for each value of ω using:

$$\omega = \frac{2\pi}{4}k = \frac{\pi}{2}k$$

Listing 4.5: Comparison of results of DTFT vs DFT

```
dtft = @(w) 0.5 * exp(-j * 1.5 * w) .* ( cos(w*1.5) + cos(w
    *0.5) );

x = [1/4 1/4 1/4 1/4];
N = length(x);
k = [0:N-1];
w = 2*pi/N * k;

xdft = dft2(x, 0)
xdtft = dtft(w)
```

Getting the same result for both.

- $x[n] = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$

Solution

Using the function dft2 over the input vector we get:

$$\begin{aligned} X_k &= \begin{pmatrix} 0 & 1.5000 + 2.5981i & 1.5000 + 2.5981i \end{pmatrix} \\ X_0 &= 0 \\ X_1 &= 1.5000 + 2.5981i \\ X_2 &= 1.5000 - 2.5981i \end{aligned}$$

To compare the results, we calculate the values from the analytic function obtained in 4.4 for each value of ω using:

$$\omega = \frac{2\pi}{4}k = \frac{\pi}{2}k$$

Listing 4.6: Comparison of results of DTFT vs DFT

```
dtft = @(w) 2 * exp(-j*w) .* (cos(w)-1);

x = [1 -2 1];
N = length(x);
k = [0:N-1];
w = 2*pi/N * k;
```

```
7
8  xdft = dft2(x, 0)
9  xdtft = dtft(w)
```

And we get the same result for both.

Problem 3

Use MATLAB to compute the DFT of the signals in Problem 2. In order to increase your accuracy, pad the signals with zeros (for example. if you want to plot an additional 10 points, then add 10 zeros). For the signal in part c), compute the DFT for three cases: truncating the signal at N = 5, N = 10 and at N= 20. In each case, compare your answers to those found in Problem 1 by plotting the magnitude of both versus frequency.

Solution

Listing 4.7: Comparison of results of DTFT vs DFT

```
1  %function handlers declaraton
2  dtft_pa = @(w) 0.5 * exp(-j * 1.5 * w) .* ( cos(w*1.5) + cos(w
       *0.5) );
3  dtft_pb = @(w) 2 * exp(-j*w) .* (cos(w)-1);
4  dtft_pc = @(w) 2 ./(1 - 3/4*exp(-j*w) );
5
6
7  N = 32;
8  k = [0:N-1];
9  w = [0:0.001:2*pi];
10
11 figure(1);
12 %%%%%%%%%%%%%%%%%%%%%%% POINT (a) %%%%%%%%%%%%%%%%%%%%%%%%%
13 xn_pa = [1/4 1/4 1/4 1/4];
14
15 xk_pa = dft2(xn_pa, N);
16 xw_pa = dtft_pa(w);
17
18 %Plot blue continuous frecuency vs red dots discrete frec
19 subplot(2,1,1);
20 plot(2*pi*k/N, abs(xk_pa), 'ro', w, abs(xw_pa),'b');
21
22 %%%%%%%%%%%%%%%%%%%%%%% POINT (b) %%%%%%%%%%%%%%%%%%%%%%%%%
23 xn_pb = [1 -2 1];
24
25 xk_pb = dft2(xn_pb, N);
26 xw_pb = dtft_pb(w);
```

```
27
28  %Plot blue continuous frecuency vs red dots discrete frec
29  subplot(2,1,2);
30  plot(2*pi*k/N, abs(xk_pb), 'ro', w, abs(xw_pb),'b');
31
32  figure(2);
33  %%%%%%%%%%%%%%%%%%%%%%% POINT (c) %%%%%%%%%%%%%%%%%%%%%%%%
34  N = [5 10 20]
35  for i = 1:length(N)
36      k = [0 : N(i)-1];
37      xn_pc = 2*(3/4).^k;
38
39      xk_pc = dft2(xn_pc, N(i));
40      xw_pc = dtft_pc(w);
41
42      %Plot blue continuous frecuency vs red dots discrete frec
43      subplot(3,1,i);
44      plot(2*pi*k/N(i), abs(xk_pc), 'ro', w, abs(xw_pc),'b');
45  end
```

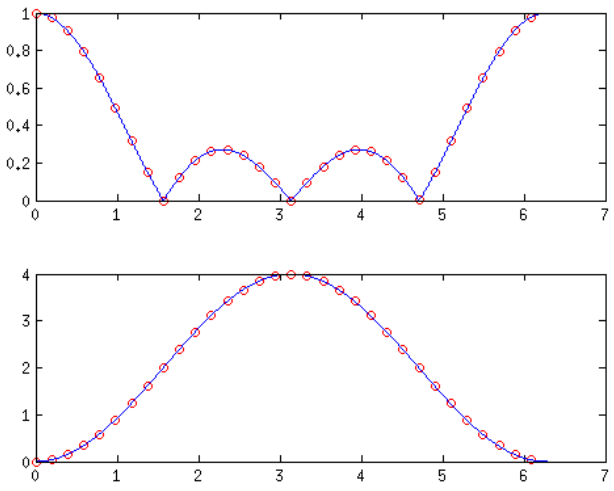


Figure 4.4: Plot of DFT vs FDTD for points (a) and (b)

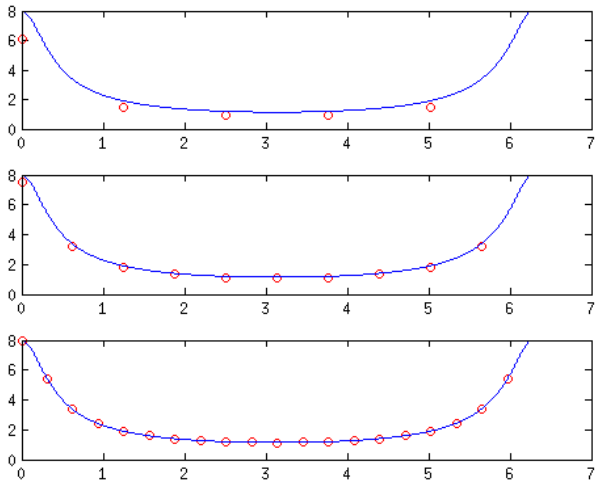


Figure 4.5: Plot of DFT vs FDTD for point (c) and N=[5,10,20]

Problem 4

Find the response of the following systems to the input below. Sketch the magnitude of each frequency response for $-\pi < \omega < \pi$ and determine the type of filter.

$x[n] = 2 + 2\cos(n\pi/4) + \cos(n2\pi/3 + \pi/2)$

- $H(\omega) = e^{-j\omega}\cos(\omega/2)$

Solution

$$X(\omega) = 4\pi\delta(\omega) + 2\pi[\delta(\omega - \frac{\pi}{4}) + \delta(\omega + \frac{\pi}{4})] + \pi[\delta(\omega - \frac{2\pi}{3}) + \delta(\omega + \frac{2\pi}{3})]$$

Listing 4.8: Magnitud y Fase de los espectros de las seales

```
fHw = @(w) cos(w/2) .* exp(-j * w);

LIM = 0.7*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
```

```
Hw = fHw(w);

subplot(3,2,1), plot(w,abs(Hw));
grid on;
xlim([-round(1.5*LIM) abs(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|H(\omega)|$','interpreter','latex');

subplot(3,2,2), plot(w,angle(Hw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle H(\omega)$','interpreter','latex');

fe = [-2*pi/3 -pi/4 0 pi/4 2*pi/3];
Ae = [pi 2*pi 4*pi 2*pi pi];
pe = [pi/2 0 0 0 -pi/2];

subplot(3,2,3), stem(fe,Ae);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('$|X(\omega)|$','interpreter','latex');

subplot(3,2,4), stem(fe,pe);
grid on;
xlabel('Frec [rad/s]');
ylabel('Phase');
title('$\angle X(\omega)$','interpreter','latex');


fe = [-2*pi/3 -pi/4 0 pi/4 2*pi/3];
Ae = [pi*abs(fHw(-2*pi/3)) 2*pi*abs(fHw(-pi/4)) 4*pi 2*pi*
    abs(fHw(pi/4)) pi*abs(fHw(2*pi/3))];
pe = [pi/2+2*pi/3 pi/4 0 -pi/4 -pi/2-2*pi/3];

subplot(3,2,5), stem(fe,Ae);
grid on;
xlabel('Frec [rad/s]');
ylabel('Amplitude');
title('$|Y(\omega)|$','interpreter','latex');

subplot(3,2,6), stem(fe,pe);
grid on;
xlabel('Frec [rad/s]');
ylabel('Phase');
title('$\angle Y(\omega)$','interpreter','latex');
```

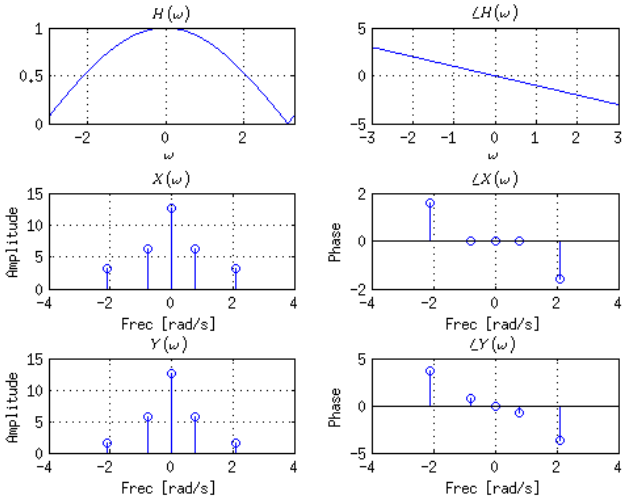


Figure 4.6: Plot for $H(\omega)\ X(\omega)\ Y(\omega)$

Output signal spectra is obtained by multiplying $X(\omega)$ with $H(\omega)$ as can be seen in figure 4.6. The values of the filter that multiplies the magnitudes of the input can be obtained by evaluating H in each of the frecuencies were the delta function appears.

Also, H is a low-pass filter as can be seen in the spectra. The output function is given by:

$$\begin{aligned} y[n] &= (1)(2) + (0.9239)(2)\cos(n\pi/4 - \pi/4) + (0.8660)\cos(n2\pi/3 - \pi/6) \\ &= 2 + 1.85\cos(n\pi/4 - \pi/4) + 0.87\cos(n2\pi/3 - \pi/6) \end{aligned}$$

- $H(\omega) = e^{-j\omega/2}(1 - \cos(\omega/2))$

Solution

Listing 4.9: Magnitud y Fase de los espectros de las seales

```
1  fHw = @(w) (1 - cos(w/2)) .* exp(-j * w / 2);
2
3  LIM = 0.7*pi;
4  w = -1.5*LIM:LIM/1000:1.5*LIM;
5  Hw = fHw(w);
```

```
6
7  subplot(3,2,1), plot(w,abs(Hw));
8  grid on;
9  xlim([-round(1.5*LIM) abs(1.5*LIM)]);
10 xlabel('$\omega$','interpreter','latex');
11 title('$|H(\omega)|$','interpreter','latex');
12
13 subplot(3,2,2), plot(w,angle(Hw));
14 grid on;
15 xlim([-round(1.5*LIM) round(1.5*LIM)]);
16 xlabel('$\omega$','interpreter','latex');
17 title('$\angle H(\omega)$','interpreter','latex');
18
19 fe = [-2*pi/3 -pi/4 0 pi/4 2*pi/3];
20 Ae = [pi 2*pi 4*pi 2*pi pi];
21 pe = [pi/2 0 0 0 -pi/2];
22
23 subplot(3,2,3), stem(fe,Ae);
24 grid on;
25 xlabel('Frec [rad/s]');
26 ylabel('Amplitude');
27 title('$|X(\omega)|$','interpreter','latex');
28
29 subplot(3,2,4), stem(fe,pe);
30 grid on;
31 xlabel('Frec [rad/s]');
32 ylabel('Phase');
33 title('$\angle X(\omega)$','interpreter','latex');
34
35
36 fe = [-2*pi/3 -pi/4 0 pi/4 2*pi/3];
37 Ae = [pi*abs(fHw(-2*pi/3)) 2*pi*abs(fHw(-pi/4)) 4*pi 2*pi*
       abs(fHw(pi/4)) pi*abs(fHw(2*pi/3))];
38 pe = [pi/2+pi/3 pi/2 0 -pi/2 -pi/2-pi/3];
39
40 subplot(3,2,5), stem(fe,Ae);
41 grid on;
42 xlabel('Frec [rad/s]');
43 ylabel('Amplitude');
44 title('$|Y(\omega)|$','interpreter','latex');
45
46 subplot(3,2,6), stem(fe,pe);
47 grid on;
48 xlabel('Frec [rad/s]');
49 ylabel('Phase');
50 title('$\angle Y(\omega)$','interpreter','latex');
```

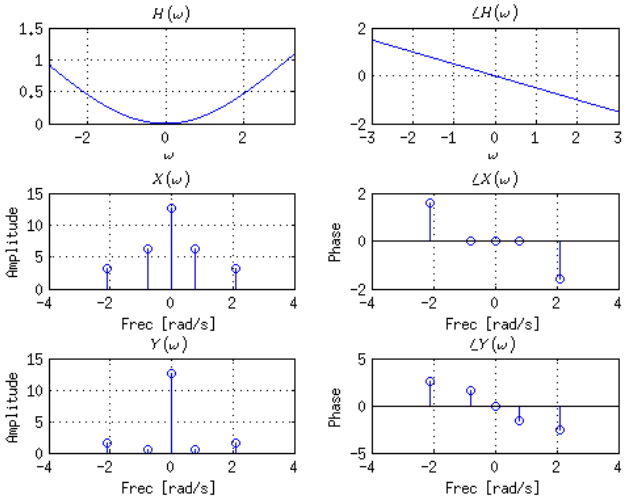


Figure 4.7: Plot for $H(\omega)\ X(\omega)\ Y(\omega)$

Output signal spectra is obtained by multiplying $X(\omega)$ with $H(\omega)$ as can be seen in figure 4.6. The values of the filter that multiplies the magnitudes of the input can be obtained by evaluating H in each of the frecuencies were the delta function appears.

Also, H is a high-pass filter as can be seen in the spectra so the constant frecuency will be filtered out. The output function is given by:

$$y[n] = 0.15\cos(n\pi/4 - \pi/8) + 0.5\cos(n2\pi/3 + \pi/6)$$

Problem 5

Find the DTFT and the DFT of $x[n] = [1{-}1]$. Sketch $X(\omega)$ for $- < \omega <$

Solution

- FDTD

$$
\begin{aligned}
X(\omega) &= e^{0} - e^{-j\omega} \\
&= e^{-j\omega/2}(e^{j\omega/2} - e^{-j\omega/2} \\
&= 2je^{-j\omega/2}\sin(\omega/2)
\end{aligned}
$$

Listing 4.10: Mlab for plotting $X(\omega)$

```
fHw = @(w) 2*j .* exp(-j * w / 2) .* sin(w/2) ;

LIM = 0.75*pi;
w = -1.5*LIM:LIM/1000:1.5*LIM;
Hw = fHw(w);

subplot(1,2,1), plot(w,abs(Hw));
grid on;
xlim([-round(1.5*LIM) abs(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$|H(\omega)|$','interpreter','latex');

subplot(1,2,2), plot(w,angle(Hw));
grid on;
xlim([-round(1.5*LIM) round(1.5*LIM)]);
xlabel('$\omega$','interpreter','latex');
title('$\angle H(\omega)$','interpreter','latex');
```

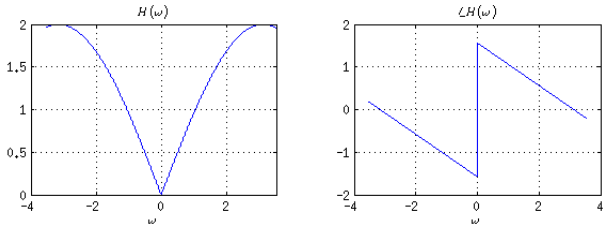


Figure 4.8: Plot for $X(\omega)$

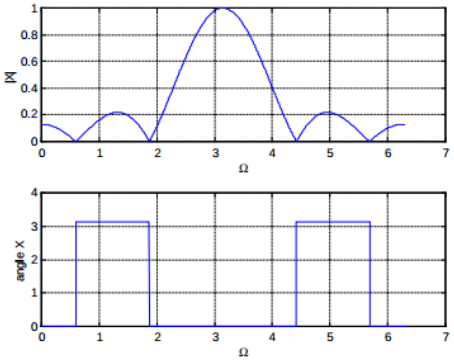
- DFT

$$
\begin{aligned}
X_k &= e^{0} - e^{\frac{2\pi k}{2}n} \\
&= 1 - e^{-j\pi k}
\end{aligned}
$$

$$
\begin{aligned}
X_0 &= 0 \\
X_1 &= 1 - e^{-j\pi} = 2
\end{aligned}
$$

Problem 6

The DTFT for an 6 point signal, $x[n]$, is shown below. Indicate on the plot the values of the DFT.



Solution

$$
\begin{aligned}
N &= 6 \\
k &= \begin{pmatrix} 0 & 1 \cdots & N-1 \end{pmatrix} \\
\omega &= \frac{2\pi k}{N} \\
&= \begin{pmatrix} 0 & \frac{\pi}{3} & \frac{2\pi}{3} & \pi & \frac{4\pi}{3} & \frac{5\pi}{3} \end{pmatrix}
\end{aligned}
$$

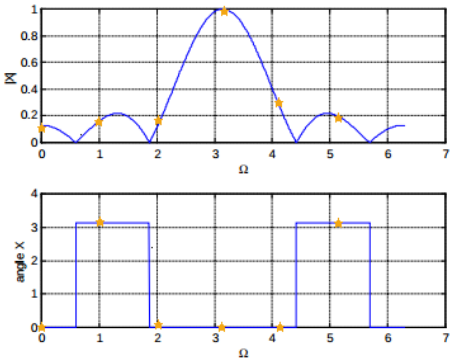


Figure 4.9: Values of discrete frecuency

Problem 7

Use the FFT to approximate the Fourier Transform of $x(t) = 4e^{-t}u(t)$. Consider the following cases:

1. Sampling period T = 1, N = 10.
2. Sampling period T = 1, N = 20.
3. Sampling period T = 0.5, N = 20.
4. Sampling period T = 0.1, N = 100

Solution

The fourier transform of $x(t)$ is:

$$X(\omega) = \frac{1}{1 + j\omega}$$

In order to approximate $X(\omega)$ using the FFT, the following equation must be used [1]:

$$X(k\Gamma) = \frac{1 - e^{-jk\Gamma T}}{jk\Gamma} X_k \tag{4.6}$$

Where $\Gamma = \frac{2\pi k}{NT}$ and X_k is the output from the FFT with $k = 0, 1, .., N - 1$.

Listing 4.11: Mlab for plotting $X(\omega)$ and its approximation using FFT

```
f = @(t) 4*exp(-t);
Fw = @(w) 4./(1+j*w);

%%%%%%%%%%% ANALYTICAL TRANSFORM %%%%%%%%%%%%
%Plot the analytical expression of the Fourier Transform
LIM = 20;
w = [0:0.1:LIM];
exactFw = Fw(w);

%%%%%%%%%%%%%%%%%% FFT %%%%%%%%%%%%%%%%%%%%%%
```

```
N = [10 20 20 100];
T = [1 1 0.5 0.1];

for i=1:length(N)
   t = [ 0: T(i) : T(i)*(N(i) -1) ]
   k = [ 0: N(i)-1]
   approxW = 2*pi*k/N(i)
   approxFw  = (1 - exp(-j*approxW))./(j*approxW/T(i)) .* fft(f
       (t));
   subplot(2,2,i), plot(w,abs(exactFw), 'b',approxW/T(i), abs(
       approxFw),'ro');
   ylabel(['N=', num2str(N(i))]);
   grid on;
end

title('$|X(\omega)|$','interpreter','latex');
```

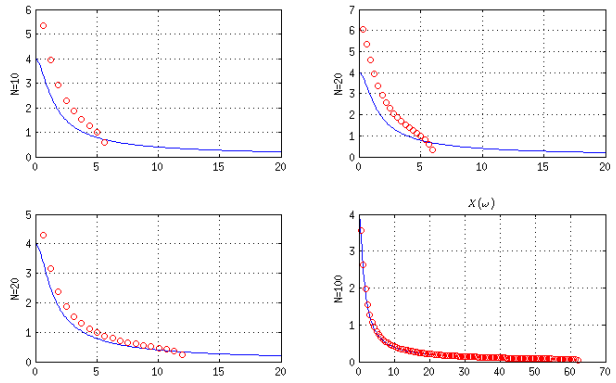


Figure 4.10: Plot of $X(\omega)$ and approximation using FFT for different T

Problem 8

Consider the continuous-time signal $x(t) = 2e{-}2tu(t)$. You wish to use the FFT to approximate X().

1. Determine the frequency ω_B such that $|X(\omega)| < 0.02|X(0)|$. The frequency content of $x(t)$ is negligible above this value

Solution

The analytical expression for the Fourier Transform of $x(t)$ is:

$$X(\omega) = \frac{2}{2+\omega j}$$
$$X(0) = \frac{2}{2} = 1$$

From the required condition, we have:

$$|X(\omega)| < 0.02|X(0)|$$
$$\frac{2}{\sqrt{4+\omega^2}} < 0.02$$
$$\sqrt{4+\omega^2} > \frac{2}{2*10^{-2}} = 100$$
$$\omega^2 < \sqrt{100^2 - 4^2}$$
$$\omega_B \approx 100$$

2. Determine an appropriate minimum value for the sampling period T from the information determined in the previous part.

Solution

From the Nyquist theorem we have:

$$\omega_s <= 2\omega_B$$
$$<= 200$$

Since $T_s = \frac{2\pi}{\omega_s}$:

$$Ts <= \frac{2\pi}{200}$$
$$<= \frac{\pi}{100}$$

3. Using the value of T determined in part 2), determine the number

of points N of $x(t)$ to be sampled so that the resolution of the approximation from the FFT is $\Gamma = 0.2rad/sec$.

Solution

From (4.6) we have:

$$\begin{aligned} \Gamma &= \frac{2\pi}{NT} \\ N &= \frac{2\pi}{\Gamma T} \\ &= 10^3 \end{aligned}$$

4. Use MATLAB to compute and plot the approximation along with the actual plot of $|X(\omega)|$.

Solution

Listing 4.12: Matlab code

```
f = @(t) 2*exp(-2*t);
Fw = @(w) 2./(2+j*w);

%%%%%%%%%%% ANALYTICAL TRANSFORM %%%%%%%%%%%%
%Plot the analytical expression of the Fourier Transform
LIM = 200;
w = [0:0.1:LIM];
exactFw = Fw(w);

%%%%%%%%%%%%%%%%%% FFT %%%%%%%%%%%%%%%%%%%%%%
N = 1000;
T = pi/100;

t = [ 0: T : T*(N -1) ]
k = [ 0: N-1]
approxW = 2*pi*k/N
approxFw  = (1 - exp(-j*approxW))./(j*approxW/T) .* fft(f(t));

subplot(211), plot(w,abs(exactFw), 'b')
grid on
title('$\|X(\omega)\|$','interpreter','latex');

subplot(212), plot(approxW/T, abs(approxFw),'r');
grid on;
title('$FFT Approximation$','interpreter','latex');
```

```
27  ylabel(['N=', num2str(N), ' T=', num2str(T)]);
```

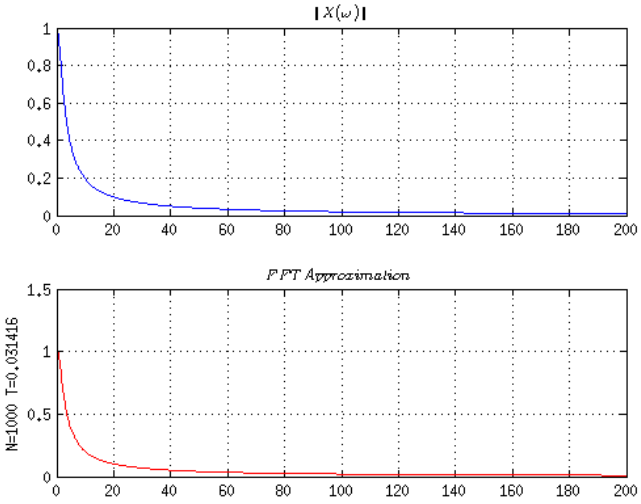


Figure 4.11: Analytical $|X(\omega)|$ vs FFT approximation

Chapter 5

Z-Transform

Problem 1

Let $T(n)$ be the temperature, in degrees Fahrenheit, of a cup of tea initially at a temperature of $180°$F that stays at a constant room temperature of $80°$F. After one minute, the tea temperature has decreased to $175°$F.

How hot is the tea after 30 minutes?

Hint: Use Newton's law of Cooling.

$$T(n-1) - T(n) = k[T(n) - 80] \tag{5.1}$$

Solution

From (5.1) we find K:

$$\begin{aligned} T(n-1) - T(n) &= k[T(n) - 80] \\ 175 - 180 &= k[180 - 80] \\ k &= \frac{-1}{20} \end{aligned}$$

Now we solve the difference equation using the Z-Transform:

$$T(0) = 180^\circ$$
$$T(1) = 175^\circ$$

$$\begin{aligned} zT(z) - zT(0) - T(z) &= kT(z) - 80\frac{z}{z-1} \\ T(z)[z-1-k] &= 180z - 80k\frac{z}{z-1} \\ T(z) &= z\frac{[180(z-1) - 80k]}{z-1} \\ \frac{T(z)}{z} &= \frac{180z - 180 - 80k}{(z-1)(z-1-k)} \\ &= \frac{A}{z-1} + \frac{B}{Z-1-k} \end{aligned}$$

Solving by partial fractions we find

$$A = \frac{80k}{k} = 90$$
$$B = \frac{180 + 180k - 180 - 80k}{1 + k - 1} = 100$$

Therefore:

$$\begin{aligned} T(z) &= 80\frac{z}{z-1} + 100\frac{z}{z-1-k} \\ T(n) &= 80u(n) + 100(1+k)^n \\ &= 80u(n) + 100(\frac{19}{20})^n \end{aligned}$$

When $n = 30$ we have:

$$\begin{aligned} T(30) &= 80 + 100(\frac{19}{20})^{30} \\ &\approx 101.464 \end{aligned}$$

Code implementation of the system:

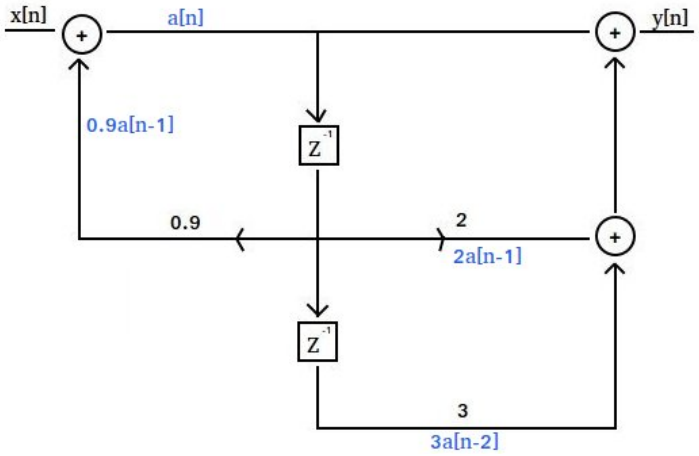
Listing 5.1: C++ Implementation

```cpp
/*
 * * ================================================
 * *
 * *       Filename:  main.cpp
 * *
 * *    Description:  Calculate Temperature using
 * *          Newton's Law of Cooling
 * *
 * *        Version:  1.0
 * *        Created:  01/19/09 20:26:43
 * *       Compiler:  g++
 * *
 * *         Author:  Sebasti n Gomez
 * *          Tatiana L pez G
 * *         Company: Universidad Tecnol gica de Pereira
 * *
 * * ================================================
 * */

#include <cstdio>
#include <cstdlib>
#include <algorithm>

#define rep(i,n) for(int i=0; i<=(n); i++)

using namespace std;

double temp[100];
double k = (-1.0/20.0);

void fill() {
    temp[0] = 180;
    temp[1] = 175;
    for (int i=1; i<50; i++) {
        temp[i+1] = temp[i] + k*(temp[i] - 80);
    }
}

int main() {
    fill();
    rep(i,30) {
        printf("T[%d] = %.5lf\n",i,temp[i]);
    }
}
```

Problem 2

For the following system:

1. Find the first 6 values for the impulse input $x[n] = \delta_0$
2. Find the first 6 values for the unitary step input $x[n] = u[n]$
3. Find the analytical expression for the system



Solution

From the follwing code (in C++) we get the first 6 values of the system for the 2 proposed inputs:

Listing 5.2: C++ Implementation

```
/*
 * * ================================================
 * *
 * *       Filename:  main.cpp
 * *
 * *    Description:  Calculate Temperature using
 * *          Newton's Law of Cooling
 * *
 * *        Version:  1.0
 * *        Created:  01/19/09 20:26:43
 * *       Compiler:  g++
 * *
 * *         Author:  Sebasti n Gomez
 * *          Tatiana L pez G
 * *         Company: Universidad Tecnol gica de Pereira
 * *
```

```
17   * * ================================================
18   * */
19
20  #include <cstdio>
21  #include <cstdlib>
22  #include <algorithm>
23
24  #define rep(i,n) for(int i=0; i<(n); i++)
25  #define repf(i,a,b) for (int i=(a); i<=(b); i++)
26  #define repb(i,a,b) for (int i=(a); i>=(b); i--)
27
28  using namespace std;
29
30  double x(int n) {
31    return (n>=0)?1.0:0.0; //unitary step
32    //return (n==0)?1.0:0.0; //impulse
33  }
34
35  double dpa[100];
36
37  double a(int n) {
38    if (n<0) return 0.0;
39    if (dpa[n] > 0) return dpa[n];
40    return dpa[n] = x(n) + 0.9*a(n-1);
41  }
42
43  double y(int n) {
44    return a(n) + 2.0*a(n-1) + 3.0*a(n-2);
45  }
46
47  int main() {
48    rep(i,100) dpa[i] = -1;
49    rep(i,10) {
50      printf("y[%d] = %.5lf\n",i,y(i));
51    }
52  }
```

---

1. First 6 values of the system for the impulse input:

```
y[0] = 1.00000
y[1] = 2.90000
y[2] = 5.61000
y[3] = 5.04900
y[4] = 4.54410
y[5] = 4.08969
```

2. First 6 values of the system for the unitary step input:

```
y[0] = 1.00000
y[1] = 3.90000
y[2] = 9.51000
```

```
y[3] = 14.55900
y[4] = 19.10310
y[5] = 23.19279
```

3. Analytical expression:

$$\begin{aligned} y[n] &= a[n] + 2a[n-1] + 3a[n-2] \\ a[n] &= x[n] + 0.9a[n-1] \end{aligned}$$

Let $a[-1] = a[-2] = 0$, then:

$$\begin{aligned} a[z] + 2z^{-1}a[z] + 3z^{-2}a[z] &= y[z] \\ z^{-2}a[z](z^2 + 2z + 3] &= y[z] \end{aligned} \tag{5.2}$$

and

$$\begin{aligned} a[z] - 0.9z^{-1}a[z] &= x[z] \\ z^{-1}a[z](z - 0.9z) &= x[z] \\ a[z] &= \frac{x[z]}{z^{-1}(z - 0.9)} \end{aligned} \tag{5.3}$$

From (5.2) and (5.3) we have:

$$\begin{aligned} y[z] &= z^{-1}\frac{x[z]}{z - 0.9}(z^2 + 2z + 3) \\ \frac{y[z]}{z} &= \frac{(x^2 + 2z + 3)x[z]}{z^2(z - 0.9)} \\ &= \frac{A}{z} + \frac{B}{z^2} + \frac{C}{z - 0.9} \end{aligned}$$

Solving with partial fractions we have:

$$\begin{aligned} C &= 6.92 \\ B &= -3.33 \\ Az(z - 0.9) + B(z - 0.9) + cz^2 &= z^2 + 2z + 3 \\ (A + C) &= 1 \\ A &= -5.92 \end{aligned}$$

Therefore:

$$y[z] = -5.92 - \frac{3.33}{z} + 6.92\frac{z}{z-0.9}$$
$$y[n] = -5.92\delta_0[n] - 3.33\delta_0[n-1] + 6.92(0.9)^n$$

Problem 3

Find the output for the system in null state:

$$y[n] = \frac{1}{2}y[n-1] + 4x[n] + 3x[n-1]$$

To the input:

$$x[n] = e^{jn\pi/3}u[n]$$

Solution

Since the system is at null state we have:

$$y[-1] = 0$$
$$x[-1] = 0$$

The Z-Transform of the input $x[n]$ is:

$$x[z] = \frac{z}{z - e^{j\pi/3}}$$

Therefore,

$$
\begin{aligned}
y[z] &= \frac{1}{2}z^{-1}y[z] + 4x[z] + 3z^{-1}x[z] \\
\frac{y[z]}{z} &= \frac{x[z]z^{-1}(4z+3)}{z-\frac{1}{2}}) \\
&= \frac{4z+3}{z-0.5)z}\frac{z}{z-e^{j\pi/3}} \\
&= \frac{A}{z-0.5} + \frac{B}{z-e^{j\pi/3}}
\end{aligned}
$$

From partial fractions we have:

$$
\begin{aligned}
A &= \frac{5}{0.5-e^{j\pi/3}} = 5.77j \\
B &= \frac{4e^{j\pi/3}+3}{e^{j\pi/3}-0.5} = 4-5.77j
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
y[z] &= A\frac{z}{z-0.5} + B\frac{z}{z-e^{j\pi/3}} \\
y[n] &= A(0.5)^n + B(e^{j\pi/3})^n \\
&= 5.77j(0.5)^n + (4-5.77j)e^{jn\pi/3}
\end{aligned}
$$

Problem 4

Find the first 5 terms of the solution of:

$$y[n] = 0.9y[n-1] + x[n] + 2x[n-1] + 3x[n-2]$$

To the input:

$$x[n] = u[n]$$

1. Using the Z-Transform

Solution

Let $y[-1] = 0$,

$$
\begin{aligned}
y[z] &= 0.9[z^{-1}y[z]] + x[z] + 2z^{-1}x[z] + 3z^{-2}x[z] \\
\frac{y[z]}{z} &= \frac{x[z]z^{-2}(z^2 + 2z + 3)}{z - 0.9} \\
&= \frac{z^2 + 2z + 3}{z(z-1)(z-0.9)} \\
&= \frac{A}{z} + \frac{B}{z-1} + \frac{C}{z-0.9}
\end{aligned}
$$

We find the constants A, B, C with partial fractions:

$$
\begin{aligned}
A &= \frac{3}{0.9} = 3.33 \\
B &= \frac{6}{0.1} = 60 \\
C &= \frac{5.61}{-0.09} = -62.33
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
y[z] &= A + B\frac{z}{z-1} + C\frac{z}{z-0.9} \\
y[n] &= 3.33\delta_0[n] + 60u[n] - 62.33(0.9)^n
\end{aligned}
$$

From the follwing code (in C++) we get the first 6 values of the system:

Listing 5.3: C++ Implementation

```
/*
 * ================================================
 *
 *       Filename:  main.cpp
 *
 *    Description:  Calculate Temperature using
 *                  Newton's Law of Cooling
 *
 *        Version:  1.0
 *        Created:  01/19/09 20:26:43
 *       Compiler:  g++
 *
```

```
13   * *         Author:  Tatiana L pez G
14   * *         Company: Universidad Tecnol gica de Pereira
15   * *
16   * * ================================================
17   * */
18
19  #include <cstdio>
20  #include <cstdlib>
21  #include <algorithm>
22  #include <math.h>
23
24  #define rep(i,n) for(int i=0; i<(n); i++)
25
26  using namespace std;
27
28  double d(int n) {
29    return (n==0)?1.0:0.0; //impulse
30  }
31
32  double u(int n) {
33    return (n>=0)?1.0:0.0; //unitary step
34  }
35
36  double y(int n) {
37    return 3.33*d(n) + 60*u(n) -62.33*pow(0.9,n);
38  }
39
40  int main() {
41    rep(i,5) {
42      printf("y[%d] = %.5lf\n",i,y(i));
43    }
44  }
```

First 6 values of the system for the impulse input:

```
y[0] = 1.00000
y[1] = 3.90000
y[2] = 9.51000
y[3] = 14.55900
y[4] = 19.10310
```

2. Recursively

Solution

From the follwing code (in C++) we get the first 6 values of the system:

Listing 5.4: C++ Implementation

```cpp
/*
 * * ================================================
 * *
 * *       Filename:  main.cpp
 * *
 * *    Description:  Calculate Temperature using
 * *          Newton's Law of Cooling
 * *
 * *        Version:  1.0
 * *        Created:  01/19/09 20:26:43
 * *       Compiler:  g++
 * *
 * *         Author:  Tatiana L pez G
 * *         Company: Universidad Tecnol gica de Pereira
 * *
 * * ================================================
 * */

#include <cstdio>
#include <cstdlib>
#include <algorithm>
#include <math.h>

#define rep(i,n) for(int i=0; i<(n); i++)

using namespace std;

double x(int n) {
  return (n>=0)?1.0:0.0; //unitary step
}

double y(int n) {
  if(n<0) return 0;
  return 0.9*y(n-1) + x(n) + 2*x(n-1) + 3*x(n-2);
}

int main() {
  rep(i,5) {
    printf("y[%d] = %.5lf\n",i,y(i));
  }
}
```

First 6 values of the system for the impulse input:

```
y[0] = 1.00000
y[1] = 3.90000
y[2] = 9.51000
y[3] = 14.55900
y[4] = 19.10310
```

Problem 5

Solve the following difference equation using z-transforms:

$$\begin{aligned} y[n] + 3y[n-1] + 2y[n-2] &= 2x[n] - x[n-1] \\ y[-1] &= 0 \\ y[-2] &= 1 \end{aligned}$$

To the input:

$$x[n] = u[n]$$

Solution

$$\begin{aligned} y[z] + 3z^{-1}y[z] + 2(z^{-2}y[z] + 1) &= 2\frac{z}{z-1} - \frac{1}{z-1} \\ y[z](z^2 + 3z + 2) + 2z^2 &= 2\frac{z^3}{z-1} - \frac{z^2}{z-1} \\ y[z] &= \frac{2z^3 - z^2 - 2z^3 + 2z^2}{(z-1)(z+1)(z+2)} \\ &= \frac{z^2}{(z-1)(z+1)(z+2)} \\ \frac{y[z]}{z} &= \frac{z}{(z-1)(z+1)(z+2)} \\ &= \frac{A}{z-1} + \frac{B}{z+1} + \frac{C}{z+2} \end{aligned}$$

We find the values of A, B, C using partial fractions:

$$\begin{aligned} A &= \frac{1}{6} \\ B &= \frac{1}{2} \\ C &= \frac{-2}{3} \end{aligned}$$

Therefore,

$$\begin{aligned} y[z] &= \frac{1}{6}\frac{z}{z-1} + \frac{1}{2}\frac{z}{z+1} - \frac{2}{3}\frac{z}{z+2} \\ y[n] &= \frac{1}{6}u[n] + \frac{1}{2}(-1)^n - \frac{2}{3}(-2)^n \end{aligned}$$

Chapter 6

Haar Base

Problem 1

Create a GUI that plots the approximation of the function $x(t) = t^2$ using Haar Base. The values for

$$\phi(2^j - t) = \begin{cases} 1, & \text{if } \frac{k}{2^j} <= t <= \frac{k+1}{2^j} \\ 0, & \text{in any other case} \end{cases}$$

With

$$C_k = 2^j \int_{\frac{k}{2^j}}^{\frac{k+1}{2^j}} f(t)\phi(t-k)\, dt$$

Solution

For j=0 we have:

$$\begin{aligned} C_k &= \int_k^{k+1} t^2 \phi(t-k)\, dt \\ &= \frac{1}{3} t^3 \Big|_k^{k+1} \\ &= \frac{1}{3}[3k^2 + 3k + 1] \text{ with } \; k = -3, -2, -1, 0, 1, 2 \\ f(t) &= \sum_{k=-3}^{2} \frac{1}{3}[3k^2 + 3k + 1]\phi(t-k) \end{aligned}$$

For j¿=0 we use the following code to plot the GUI in matlab:

Listing 6.1: Approximation Function

```
function [ tk, fs ] = approximate_t2( j, t )
%APPROXIMATE Aproximacin de la funcin t^2 mediante steps

j2=2^j;
k=[j2*(-t) : j2*(t)-1]
fs = 1/(j2^2) * (k.^2 + k + 1/3)
tk = k/j2
%plot(tk,tk.^2,'r')
%hold on
%stairs(tk,fs,'b')

end
```

Listing 6.2: Plot of the Approximation Function

```
function [ output_args ] = plotAprox(tk,f,fs)
%PLOTAPROX Summary of this function goes here
%   Detailed explanation goes here

x = linspace(tk(1),-tk(1),int32(20));
plot(x,f(x),'r')
hold on
stairs(tk,fs,'b')

end
```

Listing 6.3: GUI

```
function varargout = haar(varargin)
% HAAR MATLAB code for haar.fig
```

```
3  %      HAAR, by itself, creates a new HAAR or raises the
       existing
4  %      singleton*.
5  %
6  %      H = HAAR returns the handle to a new HAAR or the handle
       to
7  %      the existing singleton*.
8  %
9  %      HAAR('CALLBACK',hObject,eventData,handles,...) calls the
        local
10 %      function named CALLBACK in HAAR.M with the given input
       arguments.
11 %
12 %      HAAR('Property','Value',...) creates a new HAAR or
       raises the
13 %      existing singleton*.  Starting from the left, property
       value pairs are
14 %      applied to the GUI before haar_OpeningFcn gets called.
       An
15 %      unrecognized property name or invalid value makes
       property application
16 %      stop.  All inputs are passed to haar_OpeningFcn via
       varargin.
17 %
18 %      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
       allows only one
19 %      instance to run (singleton)".
20 %
21 % See also: GUIDE, GUIDATA, GUIHANDLES
22
23 % Edit the above text to modify the response to help haar
24
25 % Last Modified by GUIDE v2.5 04-Apr-2013 15:42:43
26
27 % Begin initialization code - DO NOT EDIT
28 gui_Singleton = 1;
29 gui_State = struct('gui_Name',       mfilename, ...
30                    'gui_Singleton',  gui_Singleton, ...
31                    'gui_OpeningFcn', @haar_OpeningFcn, ...
32                    'gui_OutputFcn',  @haar_OutputFcn, ...
33                    'gui_LayoutFcn',  [] , ...
34                    'gui_Callback',   []);
35 if nargin && ischar(varargin{1})
36     gui_State.gui_Callback = str2func(varargin{1});
37 end
38
39 if nargout
40     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin
           {:});
41 else
42     gui_mainfcn(gui_State, varargin{:});
43 end
44 % End initialization code - DO NOT EDIT
45
46
47 function calcAndPlot(handles)
48 cla
```

```
49  tmax = str2num(get(handles.tmax,'String'));
50  j = str2num(get(handles.j,'String'));
51  [tk,fs]=approximate_t2(j,tmax);
52  plotAprox(tk,handles.ft2,fs);
53
54  % --- Executes just before haar is made visible.
55  function haar_OpeningFcn(hObject, eventdata, handles, varargin)
56  % This function has no output args, see OutputFcn.
57  % hObject    handle to figure
58  % eventdata  reserved - to be defined in a future version of
        MATLAB
59  % handles    structure with handles and user data (see GUIDATA)
60  % varargin   command line arguments to haar (see VARARGIN)
61
62  handles.ft2 = @(t) t.^2;
63
64  calcAndPlot(handles)
65
66
67  % Set the current data value.
68  %handles.current_data = handles.peaks;
69  %plot(handles.current_data)
70
71  % Choose default command line output for haar
72  handles.output = hObject;
73
74  % Update handles structure
75  guidata(hObject, handles);
76
77  % UIWAIT makes haar wait for user response (see UIRESUME)
78  % uiwait(handles.figure1);
79
80
81  % --- Outputs from this function are returned to the command
        line.
82  function varargout = haar_OutputFcn(hObject, eventdata, handles
        )
83  % varargout  cell array for returning output args (see
        VARARGOUT);
84  % hObject    handle to figure
85  % eventdata  reserved - to be defined in a future version of
        MATLAB
86  % handles    structure with handles and user data (see GUIDATA)
87
88  % Get default command line output from handles structure
89  varargout{1} = handles.output;
90
91
92  % --- Executes on button press in pbCalcular.
93  function pbCalcular_Callback(hObject, eventdata, handles)
94  % hObject    handle to pbCalcular (see GCBO)
95  % eventdata  reserved - to be defined in a future version of
        MATLAB
96  % handles    structure with handles and user data (see GUIDATA)
97  calcAndPlot(handles)
98
99  % --- Executes on selection change in popupmenu2.
```

```
100 function popupmenu2_Callback(hObject, eventdata, handles)
101 % hObject    handle to popupmenu2 (see GCBO)
102 % eventdata  reserved - to be defined in a future version of
        MATLAB
103 % handles    structure with handles and user data (see GUIDATA)
104
105 % Hints: contents = cellstr(get(hObject,'String')) returns
        popupmenu2 contents as cell array
106 %        contents{get(hObject,'Value')} returns selected item
        from popupmenu2
107
108
109 % --- Executes during object creation, after setting all
        properties.
110 function popupmenu2_CreateFcn(hObject, eventdata, handles)
111 % hObject    handle to popupmenu2 (see GCBO)
112 % eventdata  reserved - to be defined in a future version of
        MATLAB
113 % handles    empty - handles not created until after all
        CreateFcns called
114
115 % Hint: popupmenu controls usually have a white background on
        Windows.
116 %       See ISPC and COMPUTER.
117 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
        defaultUicontrolBackgroundColor'))
118     set(hObject,'BackgroundColor','white');
119 end
120
121
122 % --- Executes on slider movement.
123 function slider1_Callback(hObject, eventdata, handles)
124 % hObject    handle to slider1 (see GCBO)
125 % eventdata  reserved - to be defined in a future version of
        MATLAB
126 % handles    structure with handles and user data (see GUIDATA)
127
128 % Hints: get(hObject,'Value') returns position of slider
129 %        get(hObject,'Min') and get(hObject,'Max') to determine
         range of slider
130
131
132 % --- Executes during object creation, after setting all
        properties.
133 function slider1_CreateFcn(hObject, eventdata, handles)
134 % hObject    handle to slider1 (see GCBO)
135 % eventdata  reserved - to be defined in a future version of
        MATLAB
136 % handles    empty - handles not created until after all
        CreateFcns called
137
138 % Hint: slider controls usually have a light gray background.
139 if isequal(get(hObject,'BackgroundColor'), get(0,'
        defaultUicontrolBackgroundColor'))
140     set(hObject,'BackgroundColor',[.9 .9 .9]);
141 end
142
```

```
143
144
145 function tmax_Callback(hObject, eventdata, handles)
146 % hObject    handle to tmax (see GCBO)
147 % eventdata  reserved - to be defined in a future version of
        MATLAB
148 % handles    structure with handles and user data (see GUIDATA)
149
150 % Hints: get(hObject,'String') returns contents of tmax as text
151 %        str2double(get(hObject,'String')) returns contents of
        tmax as a double
152
153
154 % --- Executes during object creation, after setting all
        properties.
155 function tmax_CreateFcn(hObject, eventdata, handles)
156 % hObject    handle to tmax (see GCBO)
157 % eventdata  reserved - to be defined in a future version of
        MATLAB
158 % handles    empty - handles not created until after all
        CreateFcns called
159
160 % Hint: edit controls usually have a white background on
        Windows.
161 %       See ISPC and COMPUTER.
162 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
        defaultUicontrolBackgroundColor'))
163     set(hObject,'BackgroundColor','white');
164 end
165
166
167
168 function j_Callback(hObject, eventdata, handles)
169 % hObject    handle to j (see GCBO)
170 % eventdata  reserved - to be defined in a future version of
        MATLAB
171 % handles    structure with handles and user data (see GUIDATA)
172
173 % Hints: get(hObject,'String') returns contents of j as text
174 %        str2double(get(hObject,'String')) returns contents of
        j as a double
175
176
177 % --- Executes during object creation, after setting all
        properties.
178 function j_CreateFcn(hObject, eventdata, handles)
179 % hObject    handle to j (see GCBO)
180 % eventdata  reserved - to be defined in a future version of
        MATLAB
181 % handles    empty - handles not created until after all
        CreateFcns called
182
183 % Hint: edit controls usually have a white background on
        Windows.
184 %       See ISPC and COMPUTER.
185 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'
        defaultUicontrolBackgroundColor'))
```

```
186         set(hObject,'BackgroundColor','white');
187     end
```

Chapter 7

Haar Transform

Algorithm

Implement the haar transform for discrete values

Solution

The implementation was made in c.

Listing 7.1: Haar Implementation

```
/*
 * =====================================================================
 *
 *       Filename:  haar.cpp
 *
 *    Description:  Discrete Haar Transform Implementation
 *
 *        Version:  1.0
 *        Created:  30/05/13 06:14:25
 *       Revision:  none
 *       Compiler:  gcc
 *
 *         Author:  Tatiana L pez G (ZePoLiTaT),
    tatiana@sirius.utp.edu.co
 *        Company:  Grupo de Investigaci n Sirius
 *
 * =====================================================================
 */
```

```
18 #include <cstdio>
19 #include <cstdlib>
20 #include <cmath>
21 #include <cstring>
22 #include <ctime>
23 #include <algorithm>
24 #include <map>
25
26 void printm(float *m)
27 {
28   int s = sizeof(m);
29
30   printf("\n>> [");
31   for(int i=0; i<s; i++)
32     printf("%.4f ",m[i]);
33   printf("]");
34 }
35
36 void haar(float *donees, float *sdetail)
37 {
38   float moyenne, difference;
39
40   int N = sizeof(donees);
41   int NHalf = N>>1;
42   int ti;
43
44   N = NHalf;
45   while(N>0)
46   {
47     for (int i=0; i<N; i++)
48     {
49       ti = i<<1;
50       moyenne = (donees[ti] + donees[ti+1]) / 2.0f;
51       difference = donees[ti] - moyenne;
52
53       sdetail[N + i] = difference;
54       donees[i] = moyenne;
55     }
56     N = N>>1; //shift right by 1 bit is the same as divide by 2
57   }
58
59   sdetail[0] = moyenne;
60 }
61
62 int main ( int argc, char *argv[] )
63 {
64   int N;
65   float *donees;
66   float *sdetail;
67
68   //Read N from standard input
69   scanf("%d", &N);
70
71   //Create arrays of size N
72   donees = (float *)malloc(N* sizeof(*donees));
73   sdetail = (float *)malloc(N* sizeof(*sdetail));
74
```

```
75    //Fill the values of initial array from stdin
76    for(int i=0; i<N; i++)
77      scanf("%f", &donees[i]);
78
79    printm(donees);
80    haar(donees, sdetail);
81    printm(sdetail);
82
83    delete donees;
84    delete sdetail;
85
86    return EXIT_SUCCESS;
87  }
```

This program receives the data input from the standard input. Therefore, it can be executed from the command line as:

```
./haar < input.in
```

Where input.in is a file containing the integer N in the first line and N points of data in the second:

```
8
7 1 6 6 3 -5 4 2
```

For this input, the algorithm prints the following output:

```
>> [7.0000 1.0000 6.0000 6.0000 3.0000 -5.0000 4.0000 2.0000 ]
>> [3.0000 2.0000 -1.0000 -2.0000 3.0000 0.0000 4.0000 1.0000 ]
```

Bibliography

[1] Edward Kamen and Bonnie Heck. *Fundamentals of Signals and Systems: With MATLAB Examples*. Prentice Hall PTR, 2000.

[2] Georgia Tech School of Electrical and Computer Engineering. Worked Problems, Chapter 4. `http://users.ece.gatech.edu/~bonnie/book/worked_problems/Chap4_Fseries_sol.pdf`. [Online; accessed Apr-2013].