Diego Joel
Zuñiga Fragoso

317684

Sampling	PRACTICA 7 FECHA 31/05/2024

1. OBJETIVO

El propósito de esta práctica es examinar el proceso de muestreo de señales utilizando el software MATLAB y analizarlo tanto en el dominio del tiempo como en el de la frecuencia. Los objetivos de aprendizaje incluyen comprender el proceso de muestreo y su relación con la reconstrucción de señales, así como reconocer la importancia del teorema de muestreo de Nyquist y la tasa de Nyquist para prevenir la distorsión y el efecto de aliasing en las señales muestreadas, y así evitar una pérdida de información.

2. MARCO TEÓRICO

Muestreo de Señales

El muestreo es el proceso de convertir una señal continua en una secuencia de valores discretos. Este procedimiento es fundamental en el procesamiento digital de señales (DSP), ya que permite que las señales analógicas sean procesadas y analizadas utilizando sistemas digitales. El muestreo se lleva a cabo mediante la toma de muestras de la señal continua a intervalos regulares de tiempo.

Teorema de Muestreo de Nyquist

El teorema de muestreo de Nyquist, también conocido como el teorema de muestreo de Shannon-Nyquist, establece que una señal continua puede ser completamente reconstruida a partir de sus muestras si la tasa de muestreo es al menos el doble de la frecuencia máxima presente en la señal. Matemáticamente, si

$$f_s \geq 2f_{max}$$

Esta tasa mínima de muestreo se conoce como la tasa de Nyquist. Cumplir con esta condición evita la distorsión y el aliasing en la señal muestreada.

Aliasing

El aliasing es un fenómeno que ocurre cuando una señal es muestreada a una tasa inferior a la tasa de Nyquist. Como resultado, las frecuencias más altas se pliegan y aparecen como frecuencias más bajas en la señal muestreada. Este efecto puede causar distorsiones significativas y pérdida de información en la señal reconstruida.

Reconstrucción de Señales

La reconstrucción de una señal muestreada implica convertir las muestras discretas de vuelta a una señal continua. Esto se logra utilizando un filtro de reconstrucción, que suele ser un filtro pasa-

bajos ideal que elimina las frecuencias superiores a la mitad de la tasa de muestreo (la frecuencia de Nyquist). Si la señal ha sido muestreada adecuadamente según el teorema de Nyquist, la señal original puede ser recuperada sin pérdida de información.

Importancia de la Tasa de Muestreo

Seleccionar una tasa de muestreo adecuada es crucial para evitar el aliasing y asegurar la precisión en la representación de la señal original. En aplicaciones prácticas, a menudo se utiliza una tasa de muestreo ligeramente superior a la tasa de Nyquist para proporcionar un margen de seguridad adicional y compensar posibles imperfecciones en el sistema de muestreo.

Aplicaciones del Muestreo

El muestreo de señales tiene numerosas aplicaciones en la ingeniería y la tecnología. Algunos ejemplos incluyen:

- Telecomunicaciones: Transmisión de voz y datos digitales.
- Audio Digital: Grabación y reproducción de música en formatos digitales como MP3 y WAV.
- Procesamiento de Imágenes: Digitalización de imágenes y videos para su almacenamiento y procesamiento.
- Sistemas de Control: Implementación de controladores digitales en sistemas de automatización y robótica.

3. IMPLEMENTACIÓN EN MATLAB

Se anexa el código con explicaciones

Código

```
%% Señal Original
% Definimos variables
Fs = 5000;
Ts = 1/ Fs;
Tmax = 0.1;
t = 0: Ts : Tmax - Ts;

% Creamos funciones
xt = sin (200* pi * t )+ 1.7* sin (140* pi*t );
X = fft (xt);
L = length (xt);
P2 = abs (X / L);
P1 = P2 (1: L /2+1);
P1 (2: end -1) = 2* P1 (2: end -1);
f = Fs *(0:( L /2) ) /L;

% Graficamos en dominio del tiempo
figure;
plot (t ,xt), xlim ([ 0 0.05]), xticks (0:0.01:0.05), ylim ([ -3 3]), xlabel
('t'), ylabel ('Amplitude ');
grid on;

% Graficamos en dominio de la frecuencia
```

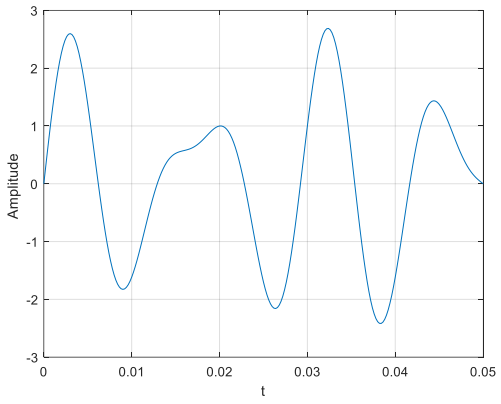
```
figure;
plot (f ,P1, 'r'), xlabel ('f(Hz)'), ylabel ('Magnitude '), xlim ([0 200]),
xticks (0:50:200), ylim ([0 2]), yticks (0:0.5:2) ;
grid on;

%% f_s = f_m
% Definimos variables
Fs = 100;
Ts = 1/ Fs;
Tmax = 0.1;
t = 0: Ts : Tmax - Ts ;

% Creamos funcione
x = sin (200* pi * t )+ 1.7* sin (140* pi*t ) ;
X = fft (x);
L = length ( x ) ;
P2 = abs ( X / L ) ;
P1 = P2 (1: L /2+1) ;
P1 (2: end -1) = 2* P1 (2: end -1) ;
f = Fs *(0:( L /2) ) /L ;

% Graficamos en dominio del tiempo
figure
stem (t ,x);
hold on;
plot (t ,x,':b'), xlim ([ 0 0.05]), xticks (0:0.01:0.05), ylim ([ -3 3]),
yticks ( -3:1:3), xlabel ('t'), ylabel ('Amplitude '), legend
('x_s1(nT)','x_s1(t)');
hold off;
grid on;

figure
plot (f , P1 ,'r'), xlabel ('f(Hz)'), ylabel ('Magnitude'), xlim ([0 200]),
xticks (0:50:200), ylim ([0 2]), yticks (0:0.5:2);
grid on;

%% f_s = 2f_m
% Definimos variables
Fs = 200;
Ts = 1/ Fs;
Tmax = 0.1;
t = 0: Ts : Tmax - Ts ;

% Creamos funcione
x = sin (200* pi * t )+ 1.7* sin (140* pi*t ) ;
X = fft (x);
L = length ( x ) ;
P2 = abs ( X / L ) ;
P1 = P2 (1: L /2+1) ;
P1 (2: end -1) = 2* P1 (2: end -1) ;
f = Fs *(0:( L /2) ) /L ;

% Graficamos en dominio del tiempo
figure
stem (t ,x);
hold on;
```

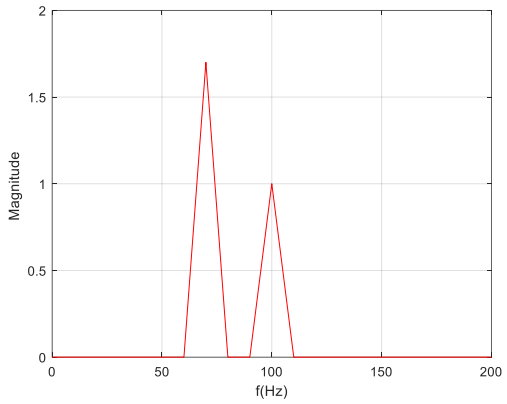
```
plot (t ,x,':b'), xlim ([ 0 0.05]), xticks (0:0.01:0.05), ylim ([ -3 3]),
yticks ( -3:1:3), xlabel ('t'), ylabel ('Amplitude '), legend
('x_s1(nT)','x_s1(t)');
hold off;
grid on;

figure
plot (f , P1 ,'r'), xlabel ('f(Hz)'), ylabel ('Magnitude'), xlim ([0 200]),
xticks (0:50:200), ylim ([0 2]), yticks (0:0.5:2);
grid on;

%% f_s = 3f_m
% Definimos variables
Fs = 300;
Ts = 1/ Fs;
Tmax = 0.1;
t = 0: Ts : Tmax - Ts ;

% Creamos funcione
x = sin (200* pi * t )+ 1.7* sin (140* pi*t ) ;
X = fft (x);
L = length ( x ) ;
P2 = abs ( X / L ) ;
P1 = P2 (1: L /2+1) ;
P1 (2: end -1) = 2* P1 (2: end -1) ;
f = Fs *(0:( L /2) ) /L ;

% Graficamos en dominio del tiempo
figure
stem (t ,x);
hold on;
plot (t ,x,':b'), xlim ([ 0 0.05]), xticks (0:0.01:0.05), ylim ([ -3 3]),
yticks ( -3:1:3), xlabel ('t'), ylabel ('Amplitude '), legend
('x_s1(nT)','x_s1(t)');
hold off;
grid on;

figure
plot (f , P1 ,'r'), xlabel ('f(Hz)'), ylabel ('Magnitude'), xlim ([0 200]),
xticks (0:50:200), ylim ([0 2]), yticks (0:0.5:2);
grid on;

%% f_s = 20f_m
% Definimos variables
Fs = 2000;
Ts = 1/ Fs;
Tmax = 0.1;
t = 0: Ts : Tmax - Ts ;

% Creamos funcione
x = sin (200* pi * t )+ 1.7* sin (140* pi*t ) ;
X = fft (x);
L = length ( x ) ;
P2 = abs ( X / L ) ;
P1 = P2 (1: L /2+1) ;
P1 (2: end -1) = 2* P1 (2: end -1) ;
```

```
f = Fs *(0:( L /2) ) /L ;

% Graficamos en dominio del tiempo
figure
stem (t ,x);
hold on;
plot (t ,x,':b'), xlim ([ 0 0.05]), xticks (0:0.01:0.05), ylim ([ -3 3]),
yticks ( -3:1:3), xlabel ('t'), ylabel ('Amplitude '), legend
('x_s1(nT)','x_s1(t)');
hold off;
grid on;

figure
plot (f , P1 ,'r'), xlabel ('f(Hz)'), ylabel ('Magnitude'), xlim ([0 200]),
xticks (0:50:200), ylim ([0 2]), yticks (0:0.5:2);
grid on;
```

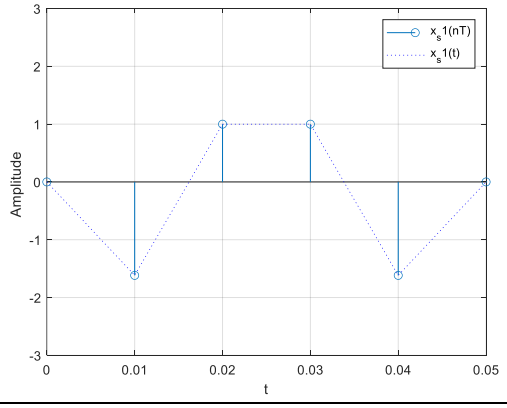
3. RESULTADOS

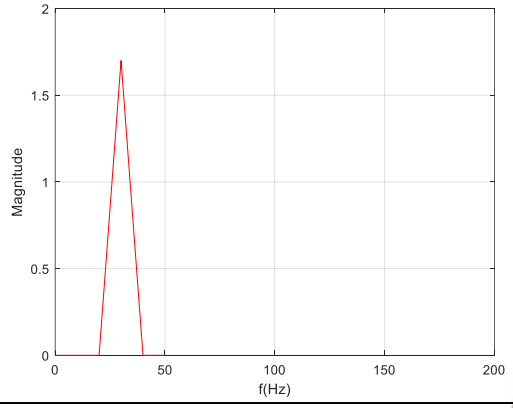
Señal original



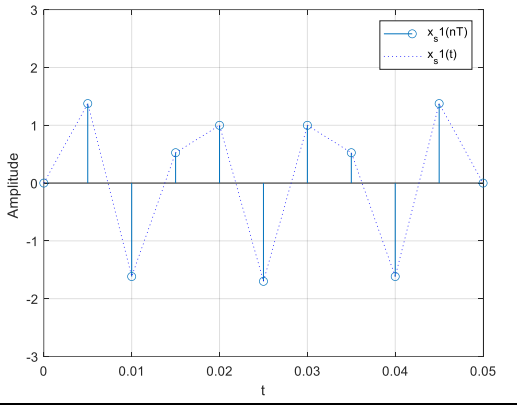


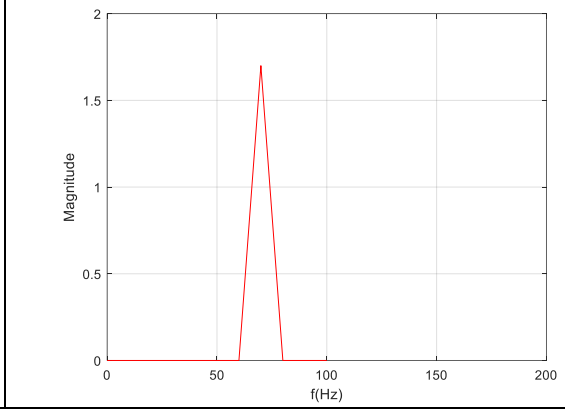
Fs = Fm



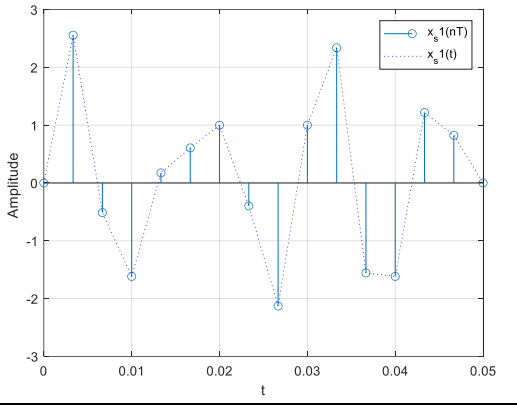


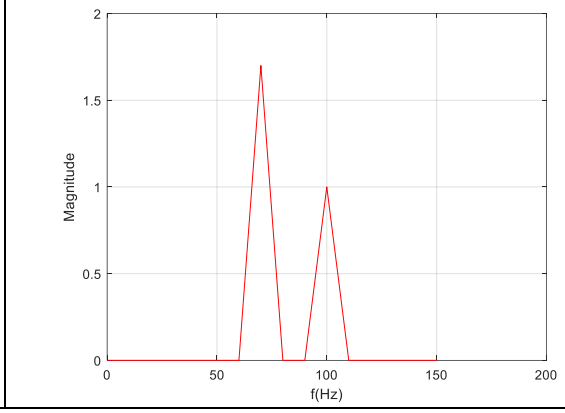
Fs = 2Fm



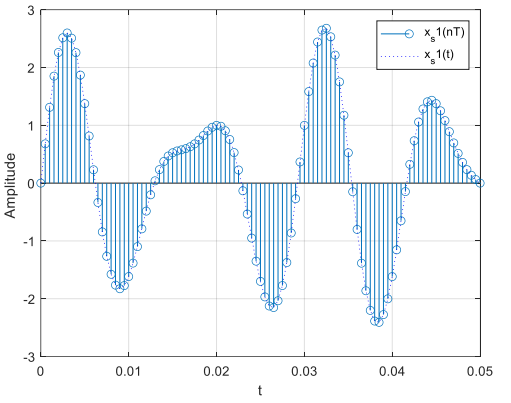


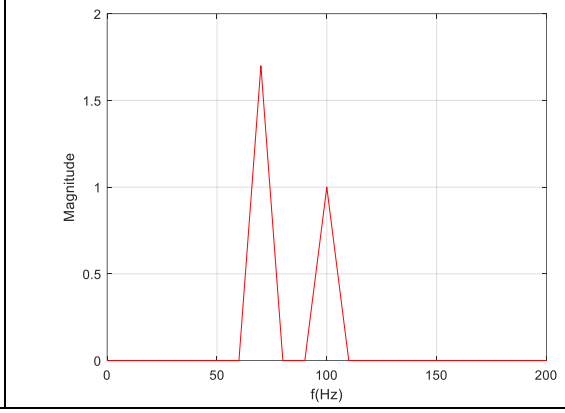
Fs = 3Fm





Fs = 20Fm





Como podemos observar, a partir de una frecuencia de muestreo 3 veces mayor que la frecuencia de la señal, aunque nuestro dominio del tiempo deja mucho que desear, nuestra transformada de Fourier representa muy bien las frecuencias que componen la señal.

4. CONCLUSIÓN

El estudio del muestreo de señales y su análisis en los dominios del tiempo y la frecuencia es esencial para comprender y aplicar técnicas de procesamiento digital. El teorema de muestreo de Nyquist y la tasa de Nyquist son conceptos fundamentales que garantizan la correcta muestreo y reconstrucción de señales, previniendo la distorsión y el aliasing. Estos principios son aplicables en una amplia variedad de campos, siendo esenciales a la hora de obtener y transferir información entre sistemas digitales y analógicos.