Procesamiento Digital De Señales

Alumno: Pose, Fernando Ezequiel

Legajo: 143.791-4

Ciclo lectivo: 2° Cuatrimestre 2015

Profesor: Dr. Ing. Mariano Llamedo Soria

Trabajo práctico 1 - Ejercicio 1

1. Realizar las funciones necesarias para generar las siguientes señales:

- Senoidal. (Parámetros: fase(radianes))
- Cuadrada. (parámetros: ciclo de actividad (%))
- Triangular. (parámetros: punto de simetría (%))

Nota: Los parámetros comunes a todas serán:

- frecuencia de muestreo (Hz),
- frecuencia de la onda (Hz),
- amplitud (# samples)
- cantidad de muestras N.

Es decir que se podrá invocar la señal que genere la senoidal como:

signal = sinusoidal_func_name(1000, 100, 1, 1000, pi/2);

Siendo una senoidal muestreada a 1000 Hz, de 100 Hz, amplitud unitaria, de 1000 muestras y con una fase de pi/2 radianes.

2. Genere señales de ejemplo para corroborar el correcto funcionamiento
3. Probar dichas señales que están comprendidas en el rango de 0.1 fs hasta 1.1 fs de distintas amplitudes y fases. Al menos 4 señales de cada tipo. Compruebe los efectos del aliasing al aproximar y/o sobrepasar fs/2. Grafique las señales y discuta los resultados respecto a:

   i. Frecuencia esperada – frecuencia obtenida
   ii. Fase

Ejercicio 1.A - Banco de pruebas

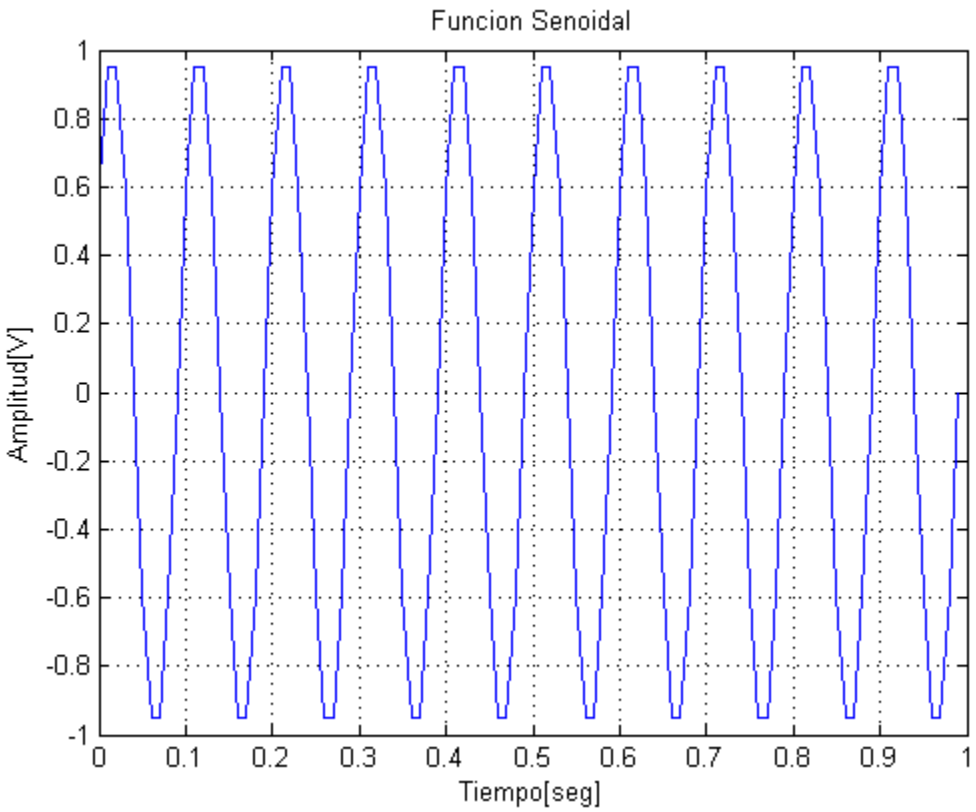
```
function [] = ejercicio1_a()
```

```
% Declaro los parámetros para realizar un juego de funciones, con el
% objetivo de probar el correcto funcionamiento de las funciones pedidas.

amplitud = 1;
frecuencia= 10;
offset = 0;
N = 100;
fs = 100;

phase = 0;
simet = 50;
duty  = 50;
```

Ejercicio 1.A - Señal Senoidal

Si bien la función senoidal retorna un vector con las muestras de la señal, objetivo del ejercicio, se incluye un vector tiempo el cual contiene los valores de tiempo a los cuales se fueron tomando dichas muestras. Este último se retorna con el objetivo de poder graficar la señal muestras en función del tiempo de las mismas.

```
[F_senoidal,t] = fsenoidal(amplitud, frecuencia, phase, offset, N, fs);

figure(1); set(gcf,'Name','Ejercicio 1a');
plot(t,F_senoidal);
title('Funcion Senoidal');
xlabel('Tiempo[seg]'); ylabel('Amplitud[V]'); grid;
```



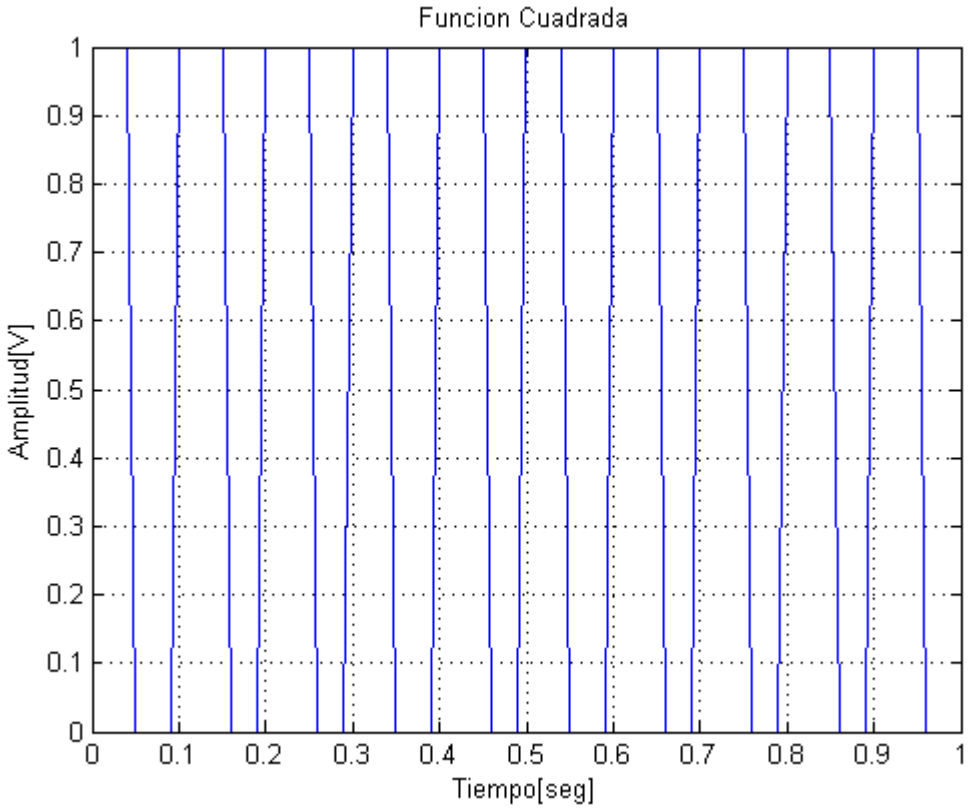
En la figura se puede observar una señal senoidal: amplitud = f(tiempo) Siendo los parámetros de la misma los que se detallan a continuación:

- Amplitud = 1
- Frecuencia de la señal = 10
- Offset = 0
- N = 100
- fs = 100
- phase = 0

Ejercicio 1.A - Señal Cuadrada

Nuevamente, al igual que en el caso de la señal senoidal, la función retorna dos parámetros, en el primero (F_cuadrada) se encontrarán las muestras de la señal cuadrada y en el segundo parámetro (t) se podrá encontrar los tiempos a los que se encuentran tomadas dichas muestras.

```
[F_cuadrada, t] = fcuadrada(amplitud, frecuencia, duty, offset, N, fs);

figure(2); set(gcf,'Name','Ejercicio 1a');
plot(t,F_cuadrada);
title('Funcion Cuadrada');
xlabel('Tiempo[seg]'); ylabel('Amplitud[V]'); grid;
```



En la figura se puede observar una señal cuadrada: amplitud = f(tiempo) Siendo los parámetros de la misma los que se detallan a continuación:

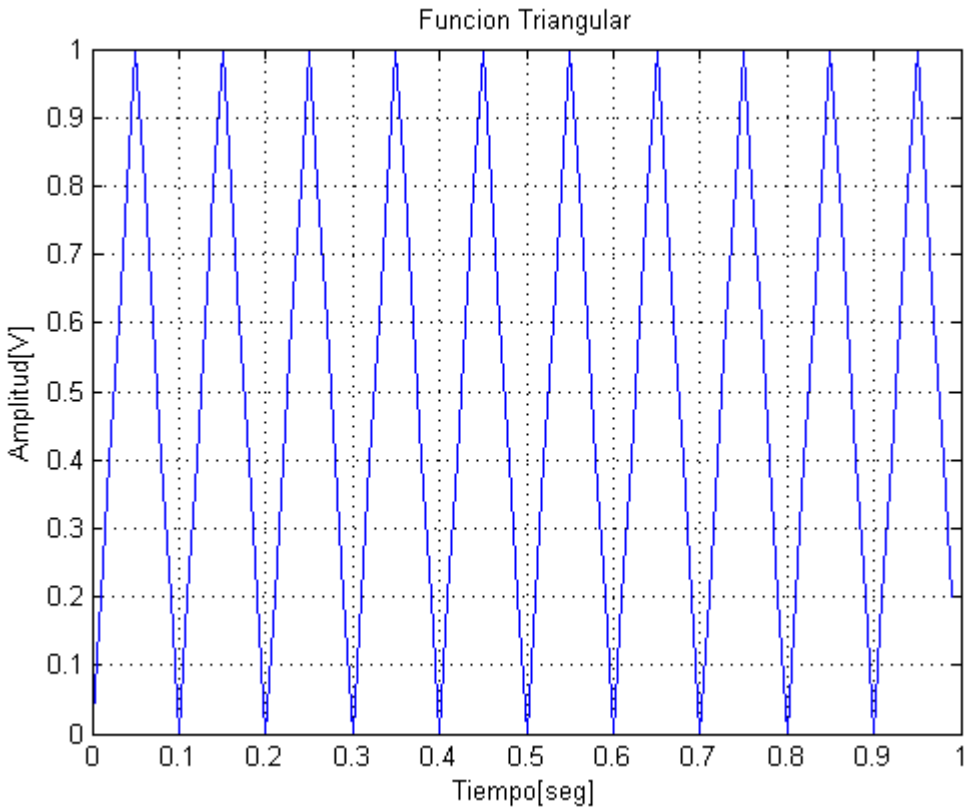
- Amplitud = 1
- Frecuencia de la señal = 10
- Offset = 0
- N = 100

- fs = 100
- duty = 50

Ejercicio 1.A - Señal Triangular

En ésta función también vale lo comentado tanto para la señal senoidal, como la señal cuadrada. En ésta el primer parámetro (F_trinagular) contendrá las muestras de la señal triangular y el segundo parámetro (t) contendrá nuevamente los tiempos a los cuales fueron tomadas dichas muestras.

```
[F_triangular, t] = ftriangular(amplitud, frecuencia, simet, offset, N, fs);

figure(3); set(gcf,'Name','Ejercicio 1a');
plot(t,F_triangular);
title('Funcion Triangular');
xlabel('Tiempo[seg]'); ylabel('Amplitud[V]'); grid;
```



En la figura se puede observar una señal triangular: amplitud = f(tiempo) Siendo los parámetros de la misma los que se detallan a continuación:

- Amplitud = 1
- Frecuencia de la señal = 10
- Offset = 0
- N = 100
- fs = 100
- Simetría = 50

```
end
```

Ejercicio 1.A – Códigos

A continuación se eximen los códigos generadores de las señales vistas en la parte A del ejercicio.

Ejercicio 1.A - Código: Función Cuadrada

```
%      \fn output = fcuadrada(amp, fo, ciclo, offset, N, fs)
%      \brief Genera una seÃ±al cuadrada
%      \details Los parametros son ingresados por el usuario
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%      \date 2015.08.19
%      \param Amplitud - Amplitud de la seÃ±al
%      \param fo       - Frecuencia de la seÃ±al
%      \param duty     - Ciclo de actividad
%      \param offset   - Offset de la seÃ±al
%      \param N        - Numero de muestras de la seÃ±al
%      \param fs       - Frecuencia de muestreo de la seÃ±al
%      \return output  - Vector con la seÃ±al cuadrada
%   \return x_temp  - Vector temporal:muestra

function [output, x_temp] = fcuadrada(amplitud, fo, duty, offset, N, fs)

    aux_y = zeros(N,1);

    x_temp = 0:1/fs:(N-1)*(1/fs);
    to = 1/fo;

    duty = duty/100;

    for j=1:1:N
        if  rem(x_temp(j),to) < (duty*to)
            aux_y(j) = 1;
```

```
        else
            aux_y(j) = 0;
        end
    end
    output = amplitud * aux_y + offset;
end
```

Ejercicio 1.A - Código: Función Senoidal

```
%       \fn [output,x_temp] = fsenoidal(amp, fo, phase, offset, N, fs)
%       \brief Genera una seÃ±al senoidal
%       \details Los parametros son ingresados por el usuario
%       \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%       \date 2015.08.17
%       \param amp      - Amplitud de la seÃ±al
%       \param fo       - Frecuencia de la seÃ±al
%       \param phase    - Fase inicial de la seÃ±al
%       \param offset   - Offset de la seÃ±al
%       \param N        - Numero de muestras de la seÃ±al
%       \param fs       - Frecuencia de muestreo de la seÃ±al
%       \return output  - Vector con la seÃ±al senoidal
%   \return x_temp  - Vector temporal:muestra

function [output,x_temp] = fsenoidal(amp, fo, phase, offset, N, fs)

    aux_y = zeros(N,1);
    x_temp = 0:1/fs:(N-1)*1/fs;

    for j =1:N
        aux_y(j,1) = offset + amp * sin(2*pi*fo*j/fs  + phase);
    end
    output = aux_y;
end
```

Ejercicio 1.A - Código: Función Triangular

```
%       \fn [output,x_temp] = ftriangular(amp, fo, ciclo, offset, N, fs)
%       \brief Genera una seÃ±al triangular
%       \details Los parametros son ingresados por el usuario
%       \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%       \date 2015.08.19
%       \param Amplitud - Amplitud de la seÃ±al
```

```
%       \param fo       - Frecuencia de la seÃ±al
%       \param simet    - Punto de simetrÃ-a
%       \param offset   - Offset de la seÃ±al
%       \param N        - Numero de muestras de la seÃ±al
%       \param fs       - Frecuencia de muestreo de la seÃ±al
%       \return output  - Vector con la seÃ±al cuadrada
%   \return x_temp  - Vector de tiempo:muestra
%**

function [output,x_temp] = ftriangular(amplitud, fo, simet, offset, N, fs)

    aux_y = zeros(N,1);

    x_temp = 0:1/fs:(N-1)*(1/fs);
    to = 1/fo;

    to = 1/fo;

    simet = simet/100;

    for j=1:1:N
        if  rem(x_temp(j),to) < (simet*to)
            aux_y(j) = (amplitud/(simet*to)) * rem(x_temp(j),to);
        else
            aux_y(j) = - amplitud / (to - (simet*to)) * (rem(x_temp(j),to) - (simet*to)) +
amplitud;
        end
    end
    output = aux_y + offset;
end
```

Ejercicio 1.B - Banco de pruebas: Ejercicio 1.B

Con los generadores de señales: senoidales, cuadradas y triangulares se desarrollo el script que se exime a continuación. Por razones de visualización, las conclusiones de este ejercicio, se podrán observar debajo de cada par de señales generadas de cada uno de los tipos pedidos. Es importante destacar, en éste caso, que todas las señales generadas fueron sampleadas con una fs de 100 Hz.

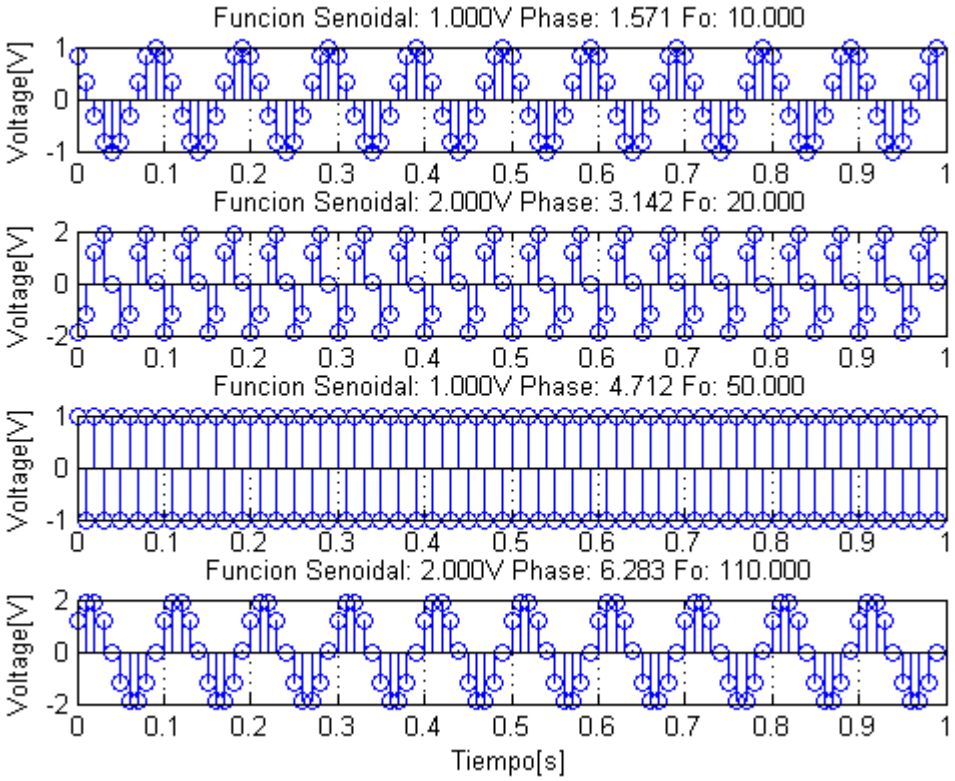
```
function [] = ejercicio1_b()
```

```
amplitud = 1;
fo= 10;
offset = 0;
N = 100;
fs = 100;  % Rango de variaciÃ³n de fs: 10 - 110

phase = pi/2;
simet = 10;
duty  = 10;
```

Ejercicio 1.B - Señal Senoidal

```
[F_senoidal,t] = fsenoidal(amplitud, fo, phase, offset, N, fs);
figure(1); set(gcf,'Name','Funcion senoidal');
subplot(4,1,1); stem(t,F_senoidal);
title(['Funcion Senoidal: ' sprintf('%3.3f',amplitud) 'V' ' Phase: ' sprintf('%3.3f',phase)
' Fo: ' sprintf('%3.3f',fo)]);
ylabel('Voltage[V]'); grid;


[F_senoidal,t] = fsenoidal(2*amplitud, 2*fo, 2*phase, offset, N, fs);
subplot(4,1,2); stem(t,F_senoidal);
title(['Funcion Senoidal: ' sprintf('%3.3f',2*amplitud) 'V' ' Phase: '
sprintf('%3.3f',2*phase) ' Fo: ' sprintf('%3.3f',2*fo)]);
ylabel('Voltage[V]'); grid;

[F_senoidal,t] = fsenoidal(amplitud, 5*fo, 3*phase, offset, N, fs);
subplot(4,1,3); stem(t,F_senoidal);
title(['Funcion Senoidal: ' sprintf('%3.3f',amplitud) 'V' ' Phase: '
sprintf('%3.3f',3*phase) ' Fo: ' sprintf('%3.3f',5*fo)]);
ylabel('Voltage[V]'); grid;

[F_senoidal,t] = fsenoidal(2*amplitud, 11*fo, 4*phase, offset, N, fs);
```

```
subplot(4,1,4); stem(t,F_senoidal);
title(['Funcion Senoidal: ' sprintf('%3.3f',2*amplitud) 'V' ' Phase: '
sprintf('%3.3f',4*phase) ' Fo: ' sprintf('%3.3f',11*fo)]);
xlabel('Tiempo[s]'); ylabel('Voltage[V]'); grid;
```



Conclusiones:

Señal 1: La primera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Fase : 1,751rad
- Fo : 10Hz

Para esta señal, la frecuencia de muestreo es superior al doble de la máxima frecuencia de la señal (en este caso se trata de un tono de 10Hz, señal acotada en banda) por tal motivo no se nota nada extraño. Bajo estas circunstancias se puede garantizar la reconstrucción de la señal original.

Señal 2: La segunda señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Fase : 3,142rad
- Fo : 20Hz

En esta segunda señal, nuevamente se repite lo expresado en las conclusiones antes dichas.

Señal 3: La tercera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Fase : 4,712rad
- Fo : 50Hz

En la tercera señal, se observa que la frecuencia de la misma es de 50Hz siendo la frecuencia de muestero 100Hz. Por tal motivo, la frecuencia de la señal coincide con la frecuencia de Nyquist, motivo por el cual se puede observar la señal formada por muestras de valor positivo y negativo de forma alternada.

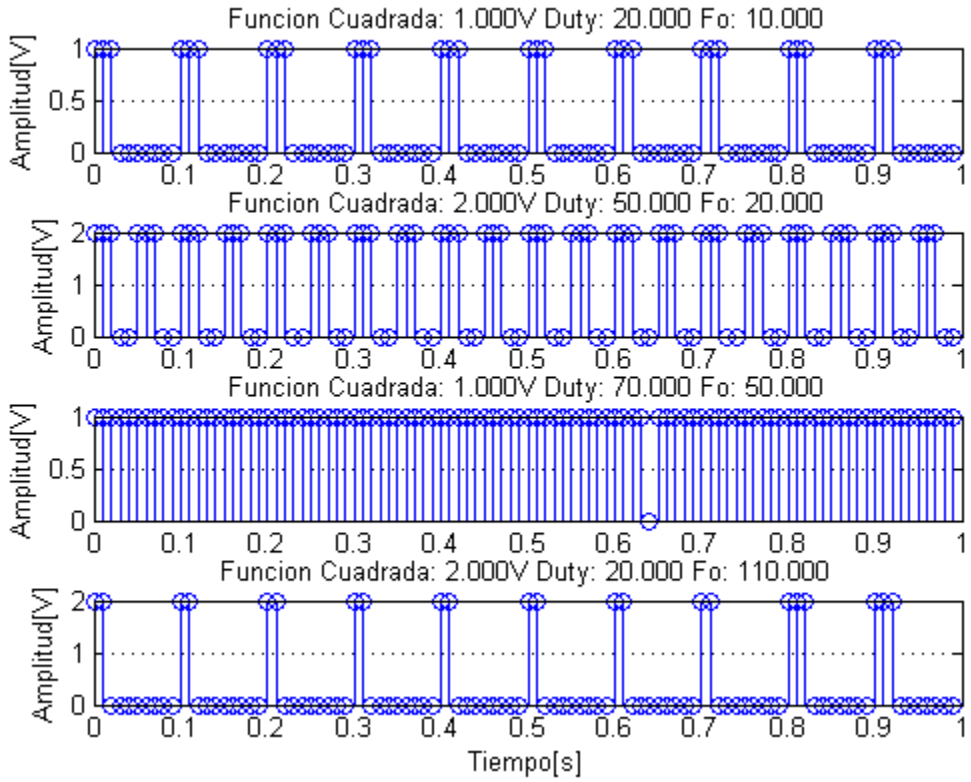
Señal 4: La cuarta señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Fase : 6,283rad
- Fo : 110Hz

Finalmente, en esta cuarta señal, podemos observar que la frecuencia de la señal es mayor a la frecuencia de Nyquist, por lo que se produce una distorsión conocida como aliasing; algunos autores la traducen como solapamiento. Este fenómeno, ocurre cuando los deltas con los cuales se muestrea están muy cerca, al convolucionar con el espectro de la señal se producirá un solapamiento. Este solapamiento deformara el espectro original impidiendo la reconstrucción de la señal original.

Ejercicio 1.b - Señal Cuadrada

```
[F_cuadrada,t] = fcuadrada(amplitud, fo, 2*duty, offset, N, fs);
figure(2); set(gcf,'Name','Funcion cuadrada');
subplot(4,1,1); stem(t,F_cuadrada);
title(['Funcion Cuadrada: ' sprintf('%3.3f',amplitud) 'V' ' Duty: ' sprintf('%3.3f',2*duty)
' Fo: ' sprintf('%3.3f',fo)]);
ylabel('Amplitud[V]'); grid;


[F_cuadrada,t] = fcuadrada(2*amplitud, 2*fo, 5*duty, offset, N, fs);
subplot(4,1,2); stem(t,F_cuadrada);
title(['Funcion Cuadrada: ' sprintf('%3.3f',2*amplitud) 'V' ' Duty: '
sprintf('%3.3f',5*duty) ' Fo: ' sprintf('%3.3f',2*fo)]);
ylabel('Amplitud[V]'); grid;

[F_cuadrada,t] = fcuadrada(amplitud, 5*fo, 7*duty, offset, N, fs);
subplot(4,1,3); stem(t,F_cuadrada);
title(['Funcion Cuadrada: ' sprintf('%3.3f',amplitud) 'V' ' Duty: ' sprintf('%3.3f',7*duty)
' Fo: ' sprintf('%3.3f',5*fo)]);
ylabel('Amplitud[V]'); grid;

[F_cuadrada,t] = fcuadrada(2*amplitud, 11*fo, 2*duty, offset, N, fs);
subplot(4,1,4); stem(t,F_cuadrada);
title(['Funcion Cuadrada: ' sprintf('%3.3f',2*amplitud) 'V' ' Duty: '
sprintf('%3.3f',2*duty) ' Fo: ' sprintf('%3.3f',11*fo)]);
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;
```



Conclusiones:

Señal 1: La primera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Duty : 20%
- Fo : 10Hz

Para esta señal, se respetó el teorema del muestreo, donde la frecuencia de muestreo es mayor al doble de la máxima frecuencia de la señal. Por este motivo, se podrá recuperar la señal sin reconocer ningún efecto en particular a ser mencionado.

Señal 2: La segunda señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Duty : 50%
- Fo : 20Hz

En esta señal, al igual que para el segundo caso de las señales senoidales, la frecuencia de muestreo supera a la frecuencia de Nyquist, motivo por el cual se podrá reconstruir la señal de forma adecuada.

Señal 3: La tercera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Duty : 70%
- Fo : 50Hz

Para esta tercera señal, la frecuencia es igual a la frecuencia de Nyquist (dos veces la máxima frecuencia de la señal) se puede observar del grafico que se tiene casi un espectro continuo, debido a que las muestras que se toman de la señal coinciden, casi siempre (al menos en este caso) con la parte de la señal en alto.

Señal 4: La cuarta señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Duty : 20%
- Fo : 110Hz

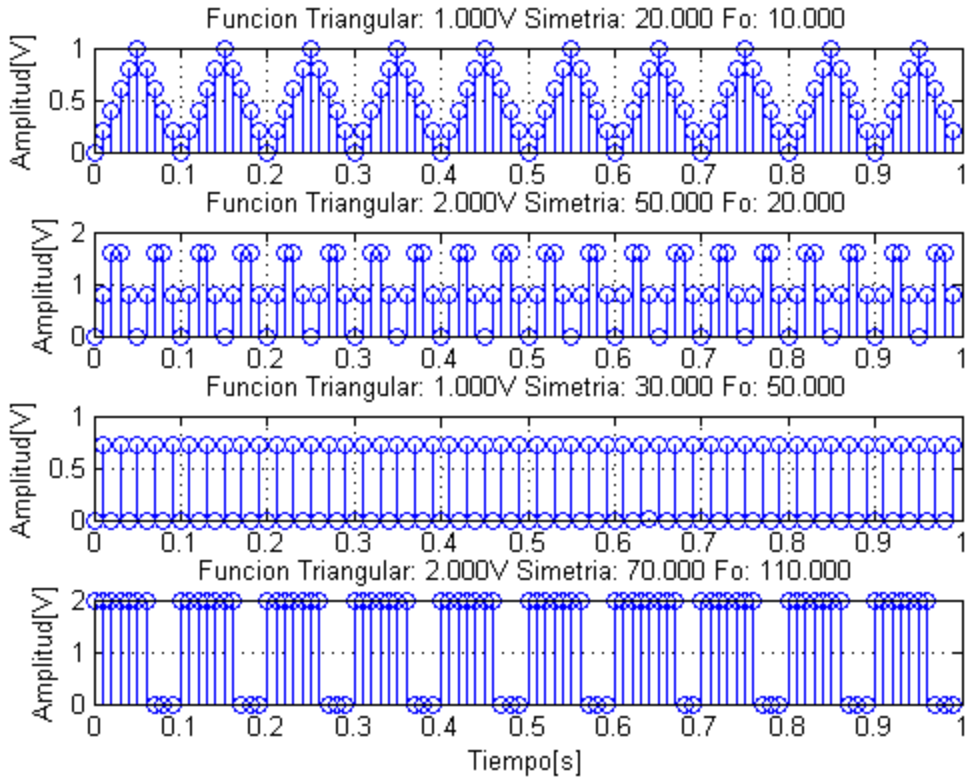
En la cuarta señal, se realiza un sobremuestreo de la señal, motivo por el cual se produce un efecto de aliasing, o solapamiento. Esta señal no se podrá reconstruir.

Como comentario final, se puede notar que en los cuatro casos vistos, la señal obtenida del muestreo no pierde la forma, pueda o no, ser reconstruida. Se debería realizar un estudio más completo para poder garantizar o rechazar la generalización de esta conclusión ante cualquier señal cuadrada.

Ejercicio 1.c - Señal Triangular

```
[F_triangular,t] = ftriangular(amplitud, fo, 5*simet, offset, N, fs);
figure(3); set(gcf,'Name','Funcion triangular');
subplot(4,1,1); stem(t,F_triangular);
title(['Funcion Triangular: ' sprintf('%3.3f',amplitud) 'V' ' Simetria: '
sprintf('%3.3f',2*simet) ' Fo: ' sprintf('%3.3f',fo)]);
ylabel('Amplitud[V]'); grid;


[F_triangular,t] = ftriangular(2*amplitud, 2*fo, 5*simet, offset, N, fs);
subplot(4,1,2); stem(t,F_triangular);
title(['Funcion Triangular: ' sprintf('%3.3f',2*amplitud) 'V' ' Simetria: '
sprintf('%3.3f',5*simet) ' Fo: ' sprintf('%3.3f',2*fo)]);
ylabel('Amplitud[V]'); grid;

[F_triangular,t] = ftriangular(amplitud, 5*fo, 3*simet, offset, N, fs);
subplot(4,1,3); stem(t,F_triangular);
```

```
title(['Funcion Triangular: ' sprintf('%3.3f',amplitud) 'V' ' Simetria: '
sprintf('%3.3f',3*simet) ' Fo: ' sprintf('%3.3f',5*fo)]);
ylabel('Amplitud[V]'); grid;

[F_triangular,t] = fcuadrada(2*amplitud, 11*fo, 7*simet, offset, N, fs);
subplot(4,1,4); stem(t,F_triangular);
title(['Funcion Triangular: ' sprintf('%3.3f',2*amplitud) 'V' ' Simetria: '
sprintf('%3.3f',7*simet) ' Fo: ' sprintf('%3.3f',11*fo)]);
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;

end
```



Conclusiones:

Señal 1: La primera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Simetría: 50%
- Fo : 10Hz

Señal 2: La segunda señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Simetría: 50%
- Fo : 20Hz

Señal 3: La tercera señal fue generada a partir de los siguientes parámetros:

- Amplitud: 1V
- Simetría: 30%
- Fo : 50Hz

Señal 4: La cuarta señal fue generada a partir de los siguientes parámetros:

- Amplitud: 2V
- Simetría: 70%
- Fo : 110Hz

Para las señales triangulares, se repiten las mismas conclusiones ya obtenidas. Se puede observar que cuando la frecuencia de muestreo es superior a dos veces la frecuencia máxima (frecuencia de Nyquist) la forma de onda se mantiene (ver las primeras dos señales). Luego en la frecuencia de Nyquist, se repite el caso de la señal senoidal a la frecuencia de Nyquist. Finalmente en la cuarta señal, cuando la frecuencia de sampleo es menor que el doble de la frecuencia máxima la señal pierde su forma, la misma ya no puede ser reconstruida nuevamente.

Trabajo práctico 1 - Ejercicio 2

Implemente un algoritmo que calcule la transformada discreta de Fourier (DFT).

Ejercicio 2 - Banco de pruebas.

```
function [] = ejercicio2()
```

Declaro los parámetros para realizar una función, senoidal en este caso y a partir de la misma probar el algoritmo desarrollado para calcular la Transformada Discreta de Fourier.

```
amplitud = 1;
frecuencia= 10;
offset = 0;
N = 100;
fs = 100;
phase = 0;

[F_senoidal,t] = fsenoidal(amplitud, frecuencia, phase, offset, N, fs);

f=0:fs/N:(N-1)*fs/N;
```

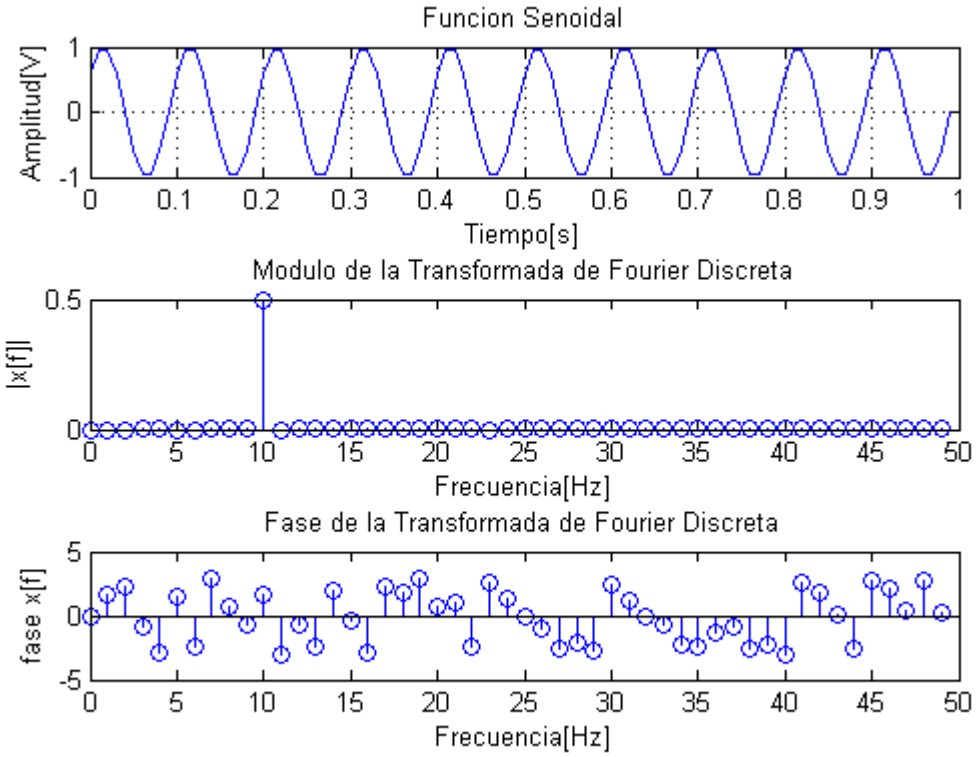
Es importante destacar bajo qué condiciones trabaja el algoritmo de la DFT para saber a qué tipo de señales se le puede aplicar esta transformada. La DFT requiere que la función de entrada sea una secuencia discreta y de duración finita. Dicha secuencia se suele generar a partir del muestreo de una función continua (ejercicio 1). Esta transformada únicamente evalúa las suficientes componentes frecuenciales para reconstruir un segmento finito que se analiza, en otras palabras, el segmento que se analiza es un único periodo de una señal periódica que se extiende de forma infinita.

Como resumen a lo dicho entonces, se dice que la DFT puede aplcarse siempre que la señal X[n] sea discreta, de duración finita y con X[n]=0 para No<n<0 donde N es el largo del array y N son la cantidad de puntos del espectro a calcular.

```
F_senoidal_t = my_dft(F_senoidal);
```

```
figure(1); set(gcf,'Name','Ejercicio 2');
subplot(3,1,1); plot(t,F_senoidal);
title('Funcion Senoidal');
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;

subplot(3,1,2);
stem(f(1:length(F_senoidal_t)/2),abs(F_senoidal_t(1:length(F_senoidal_t)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|');

subplot(3,1,3);
stem(f(1:length(F_senoidal_t)/2),angle(F_senoidal_t(1:length(F_senoidal_t)/2)));
title('Fase de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('fase x[f]');
```

```
end
```



Se puede observar que el espectro resultante, es un delta a la frecuencia de la señal. En este caso, la señal es un tono de 10Hz (señal senoidal). El caso de una señal cuadrada y triangular se estudiara en el ejercicio 3 del presente trabajo práctico.

Ejercicio 2 – Códigos

A continuación se exime el código generado para desarrollar la DFT de una señal.

Ejercicio 2 - Código: Transformada discreta de Fourier (DFT)

```
%       \fn output = my_dft(samples)
%       \brief Realiza la DFT de samples
%       \details Samples debe ser un vector columna
%       \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%       \date 2015.08.17
%       \param Samples - Vector con muestras de la seÃ±al
%       \return output - Vector columna con la DFT de la seÃ±al
%**

function output = my_dft(samples)

    N =length(samples);

    exponente = zeros(N,1);

    samples = samples';
    n = 1:1:N;

    exponente(n,1) = exp(-2* 1j * pi *n/N);

    for k=0:1:N-1
        output(k+1) = samples*(exponente.^k);
    end
    output = output * 1/N;
    output = output';
end
```

Trabajo práctico 1 - Ejercicio 3

1. Discutir si las señales del punto 1 son acotadas en banda(X(f) = 0 si f > Fs/2), ¿Cómo se podrán limitar en banda en caso que no están acotadas?
- Fundamente su respuesta en base a señales generadas con las funciones de 1 para los casos de estar o no acotadas en banda.
- Calcule y grafique los espectros con la DFT desarrollada en 2, de las señales generadas en 3.A para avalar sus conclusiones.

Ejercicio 3.A

Al tratarse la señal que se desea transformar de una señal discreta, el espectro será discreto y se encontrara "envuelto", según como fue visto en la teoría a partir de:

2. La transformada de una señal senoidal corresponde a dos pulsos en frecuencia en +/- fo (siendo fo la frecuencia de la señal a transformar)
3. La transformada de un pulso corresponde a una sinc()
4. La transformada de una señal triangular corresponde a una función sinc2().

Debido a esto las componentes se encontraran en dichas envolventes, siendo en el caso de la señal cuadrada y triangular una envolvente que en el espectro se encuentra entre -inf a inf, teniendo estas últimas señales transformadas componentes en todo el espectro. Es por esto que como se verá estas dos últimas señales comentadas no tendrán un espectro acotado en banda.

La forma de obtener una señal acotada en banda es a través de realizar un filtrado sobre la señal en cuestión. Esto se basa en realizar una convolución en tiempo discreto (lo que equivale a un producto en frecuencia). El filtrado nos permite eliminar las componentes en frecuencias que no deseamos. En nuestro ejercicio deseamos eliminar las componentes en frecuencia que superen la frecuencia de Nyquist, de modo tal de obtener la señal acotada en banda, para esto utilizaremos lo que denominaremos un filtro pasa bajos.

Ejercicio 3.B - Banco de pruebas

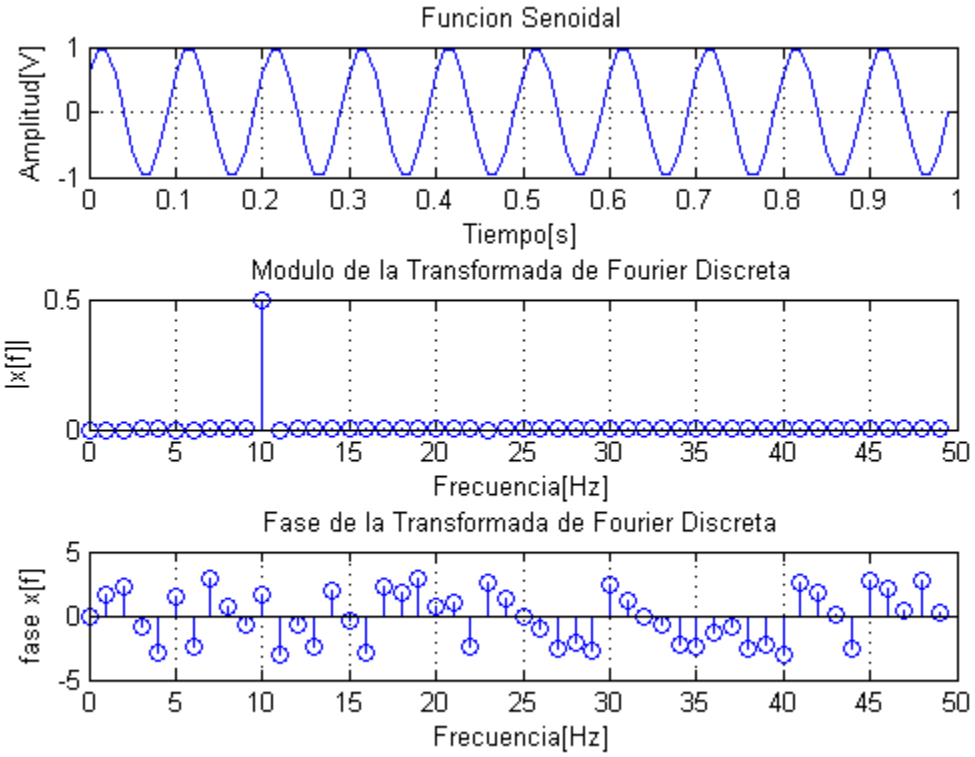
```
function [] = ejercicio3()
```

Declaro los parámetros para realizar un juego de funciones, de tal forma de poder realizar la formulación de las conclusiones pedidas.

```
amplitud = 1;
frecuencia= 10;
offset = 0;
N = 100;
fs = 100;

phase = 0;
simet = 50;
duty  = 50;
```

Ejercicio 3.A - Señal Senoidal

```
[F_senoidal,t] = fsenoidal(amplitud, frecuencia, phase, offset, N, fs);

figure(1); set(gcf,'Name','Ejercicio 3');
subplot(3,1,1); plot(t,F_senoidal);
title('Funcion Senoidal');
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;

f=0:fs/N:(N-1)*fs/N;
F_senoidal_t = my_dft(F_senoidal);

subplot(3,1,2);
stem(f(1:length(F_senoidal_t)/2),abs(F_senoidal_t(1:length(F_senoidal_t)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|'); grid;

subplot(3,1,3);
stem(f(1:length(F_senoidal_t)/2),angle(F_senoidal_t(1:length(F_senoidal_t)/2)));
title('Fase de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('fase x[f]'); grid;
```



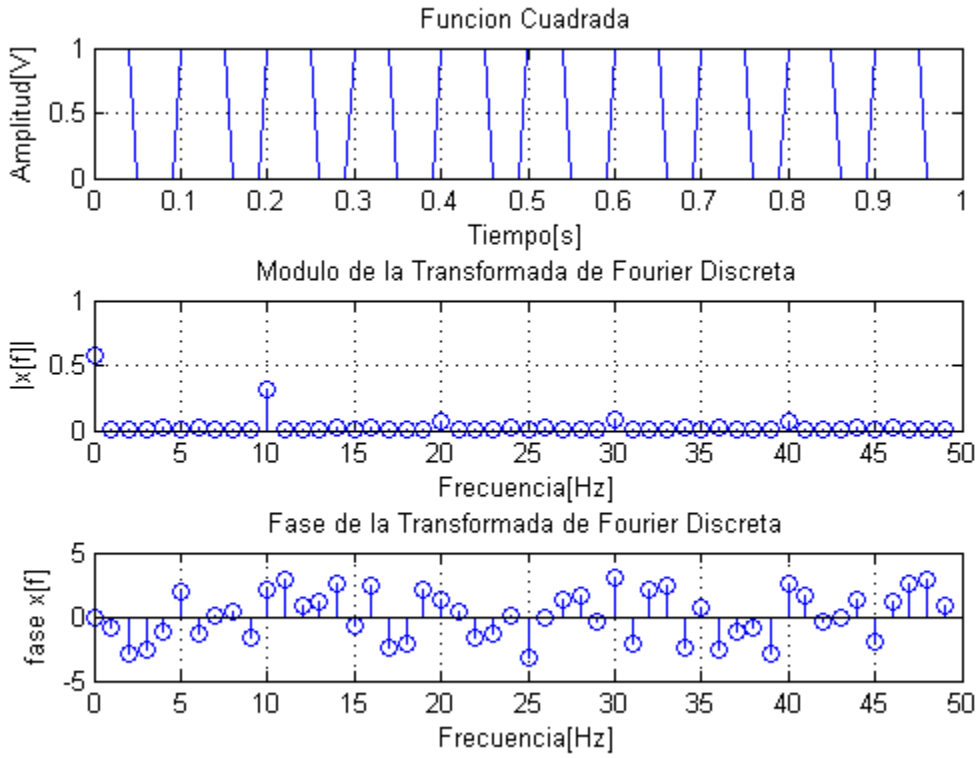
Señal 1: La señal senoidal fue generada, a partir de a partir de los parámetros que se exponen a continuación, utilizando los algoritmos generados en el ejercicio 1.A del trabajo práctico presente.

- Amplitud: 1V
- Fase : 0 rad
- Fo : 10 Hz
- Fs : 100 Hz

Se puede observar, que en el caso de la función senoidal, su transformada da un par de deltas en +/- la frecuencia de la señal, en este caso +/- 10Hz. Se observa que x(f) = 0 cuando f>fs/2 por lo tanto, como se esperaba, se trata de una señal acotada en banda.

Ejercicio 3.b - Señal Cuadrada

```
[F_cuadrada, t] = fcuadrada(amplitud, frecuencia, duty, offset, N, fs);

figure(2); set(gcf,'Name','Ejercicio 3');
subplot(3,1,1); plot(t,F_cuadrada);
title('Funcion Cuadrada');
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;

f=0:fs/N:(N-1)*fs/N;
F_cuadrada_t = my_dft(F_cuadrada);

subplot(3,1,2);
stem(f(1:length(F_cuadrada_t)/2),abs(F_cuadrada_t(1:length(F_cuadrada_t)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|'); grid;

subplot(3,1,3);
stem(f(1:length(F_cuadrada_t)/2),angle(F_cuadrada_t(1:length(F_cuadrada_t)/2)));
title('Fase de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('fase x[f]'); grid;
```



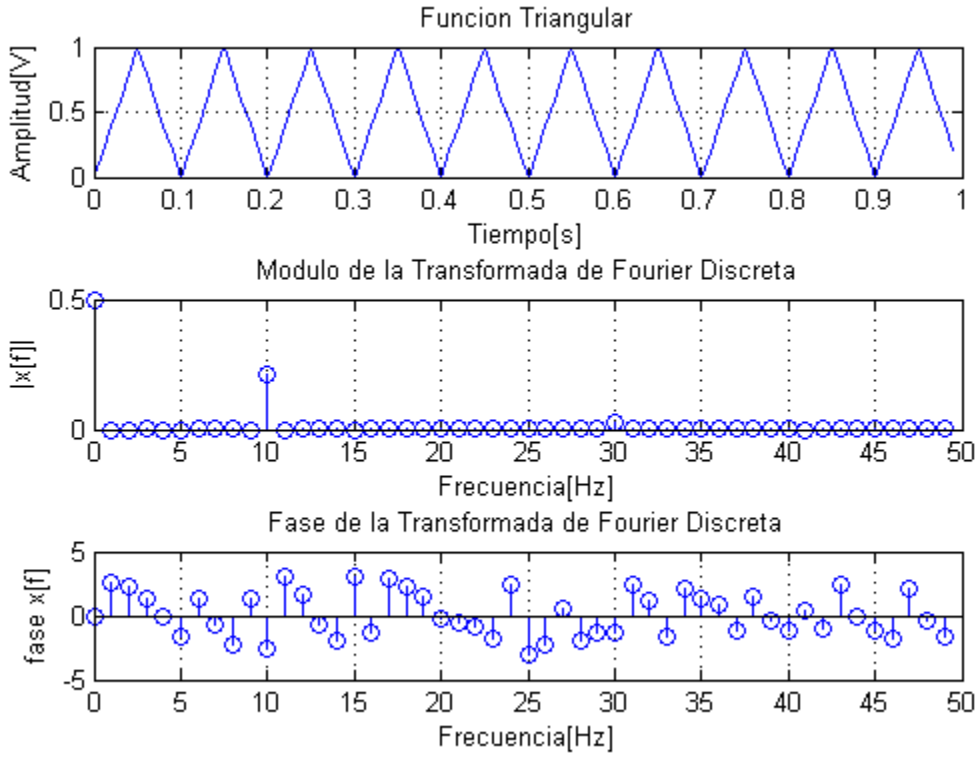
Señal 2: La señal cuadrada fue generada, a partir de a partir de los parámetros que se exponen a continuación, utilizando los algoritmos generados en el ejercicio 1.A del trabajo práctico presente.

- Amplitud: 1V
- Duty : 50 %
- Fo : 10 Hz
- Fs : 100 Hz

Del espectro obtenido queda verificado el análisis realizado en el primer punto del ejercicio. Podemos ver que esta señal no es acotada en banda x(f) ≠ 0 cuando f>fs/2

Ejercicio 3.c - Señal Triangular

```
[F_triangular, t] = ftriangular(amplitud, frecuencia, simet, offset, N, fs);

figure(3); set(gcf,'Name','Ejercicio 3');
subplot(3,1,1); plot(t,F_triangular);
title('Funcion Triangular');
xlabel('Tiempo[s]'); ylabel('Amplitud[V]'); grid;

f=0:fs/N:(N-1)*fs/N;
F_triangular_t = my_dft(F_triangular);

subplot(3,1,2);
stem(f(1:length(F_triangular_t)/2),abs(F_triangular_t(1:length(F_triangular_t)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|'); grid;

subplot(3,1,3);
stem(f(1:length(F_triangular_t)/2),angle(F_triangular_t(1:length(F_triangular_t)/2)));
title('Fase de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('fase x[f]'); grid;

end
```



Señal 3: La señal senoidal fue generada, a partir de a partir de los parámetros que se exponen a continuación, utilizando los algoritmos generados en el ejercicio 1.A del trabajo práctico presente.

- Amplitud: 1V
- Simetría: 50 %
- Fo : 10 Hz
- Fs : 100 Hz

Al igual que en el caso de la función cuadra, esta señal tampoco resulta acotada en banda, validando los conceptos dados en el primer punto del ejercicio.

Trabajo práctico 1 - Ejercicio 4

1. Verifique el efecto del leakage para una senoidal de f = fn *df siendo fn = round(N/4) y luego para fn = fn + di; siendo di = [0.01, 0.025, 0.5].
2. Verifique que ocurre si a la señal se le agregan ceros para prolongar su duración. Es decir si la señal tiene N muestras, agregue Mj ceros siendo Mj = [N/10, N, 10*N]
3. ¿Ha variado la resolución espectral en los casos de 3.B? ¿Cuál es el efecto que se produce en cada caso? Esta técnica se conoce como Zero Padding.

El efecto de leakage está presente siempre en la señales del mundo real, el mismo aparece cuando la frecuencia de la señal es distinta de k* fs/N siendo fs la frecuencia de sampleo, N la cantidad de muestras que se utilizan y k un numero entero entre 1 y N, es decir que no contamos con la resolución suficiente para resolver. Este efecto se puede apreciar al ver el espectro resultante donde la energía que tiene la señal original se distribuye alrededor de la componente principal, pero ya no coincide con esta. Este efecto trae asociado entre otras cosas, error en la medición de amplitud. A continuación se desarrolla un script que permite verificar el efecto.

Ejercicio 4 - Banco de prueba

```
function [] = ejercicio4()
```

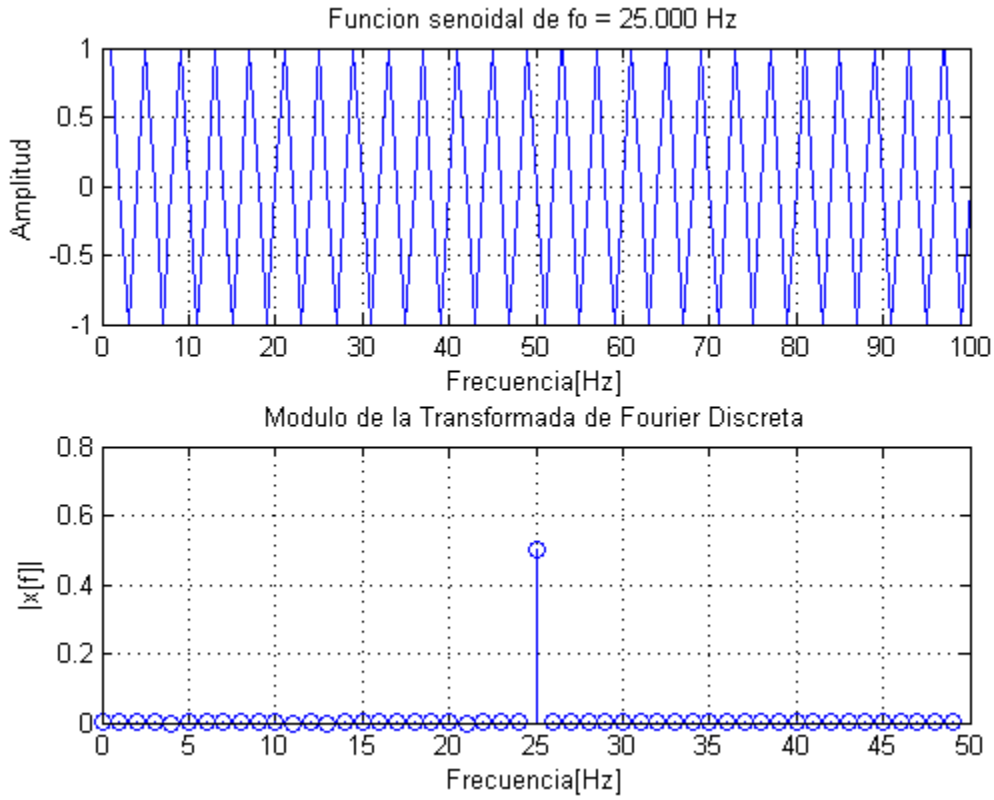
Declaro los parámetros para realizar un juego de funciones, de tal forma de poder realizar la formulaciÃ³n de las conclusiones pedidas.

```
amplitud = 1;
offset = 0;
N = 100;
fs = 100;
phase = 0;

df = fs/N;

fn = round(N/4) ;
fx = 1;
```

Ejercicio 4.A - Verificación del efecto del leakage

Señal fo = round(N/4) * df

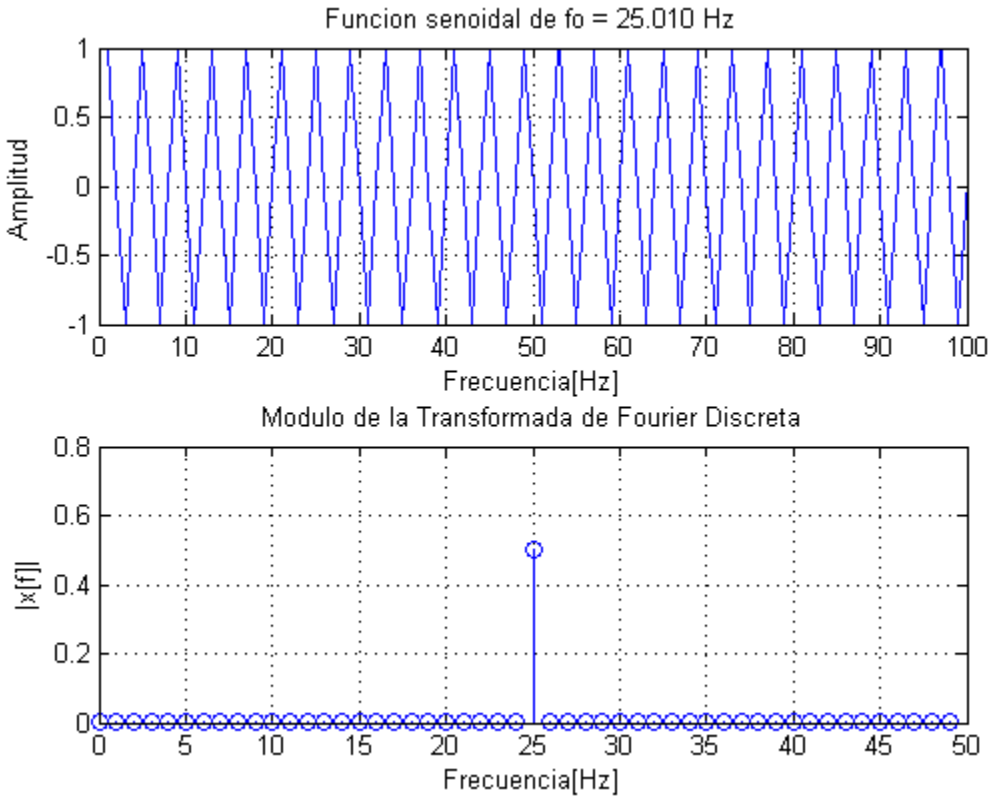
```
fo_1 = fn * df;
[F_senoidal,t] = fsenoidal(amplitud, fo_1, phase, offset, N, fs);
ploteo_ejercicio4(F_senoidal, fs, N, fo_1, fx);
fx = fx + 1;
```



En este primer ejemplo se puede ver que f es igual a k * fs/N cuando se tiene k = 25 que se encuentra dentro de los valores que puede tener K. Debido a esto no se tiene efecto de leakage o desparramo por lo que, el espectro obtenido se condice con la teoría.

Señal fo = (round(N/4)+di) * df siendo di = 0,01

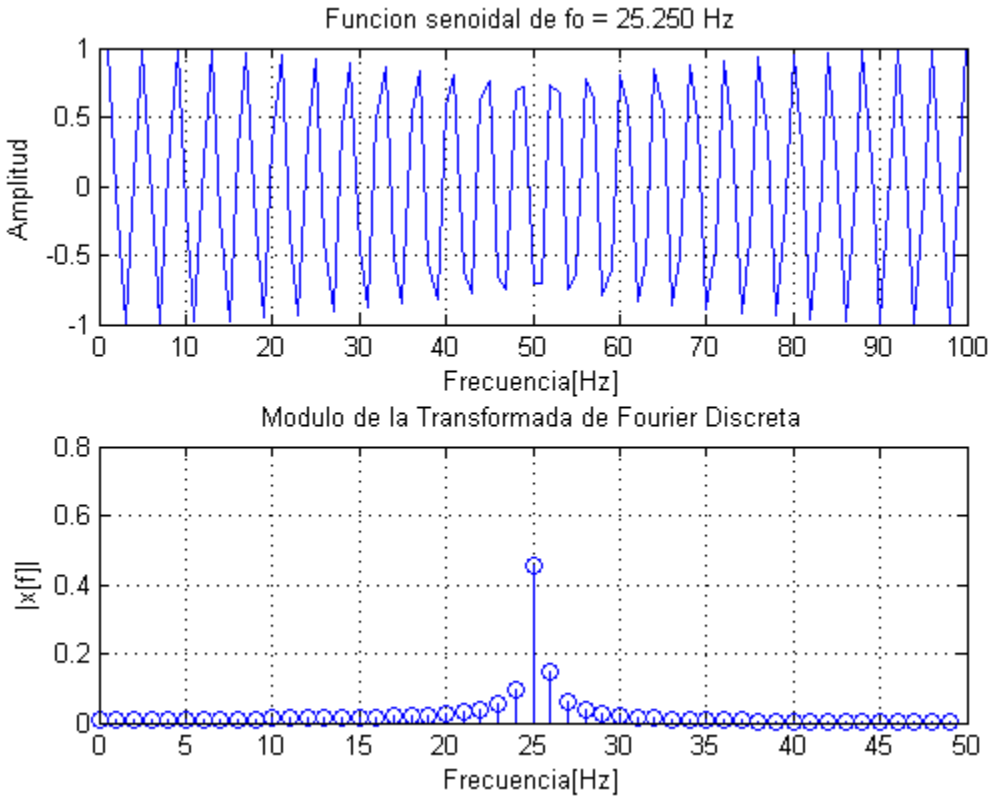
```
di = 0.01;
fn_aux = fn + di;
fo_2 = fn_aux * df;

[F_senoidal,t] = fsenoidal(amplitud, fo_2, phase, offset, N, fs);
ploteo_ejercicio4(F_senoidal, fs, N, fo_2, fx);
fx = fx + 1;
```



En este ejemplo, la frecuencia de la señal aumenta en una centésima, si bien aparece un efecto de leakage, el mismo no se visualiza tan fácilmente como si será el caso de las próximas señales.

Señal fo = (round(N/4)+di) * df siendo di = 0,25

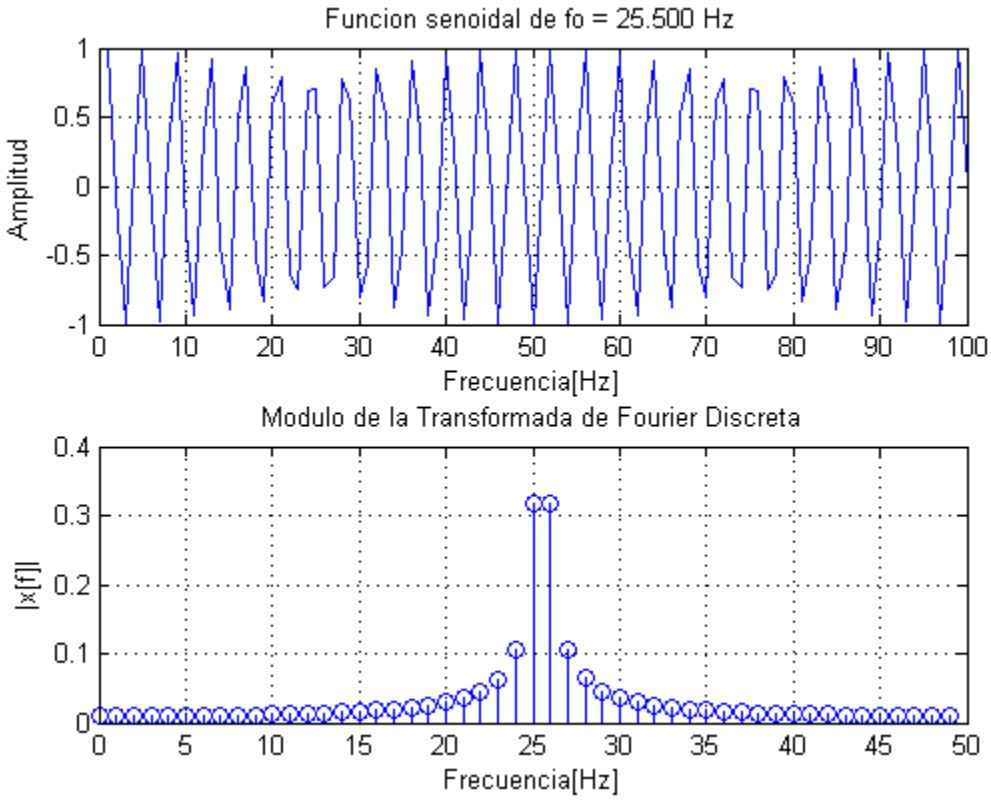
```
di = 0.25;
fn_aux = fn + di;
fo_3 = fn_aux * df;

[F_senoidal,t] = fsenoidal(amplitud, fo_3, phase, offset, N, fs);
ploteo_ejercicio4(F_senoidal, fs, N, fo_3, fx);
fx = fx + 1;
```



Se puede observar en la señal que se muestra que K = 25,250 este no es un numero entero entre los valores posibles que puede tomar K por lo que aparece el efecto que se está estudiando, por lo cual la energía comienza a repartirse por el espectro alrededor de la componente más energética.

Señal fo = (round(N/4)+di) * df siendo di = 0,5

```
di = 0.5;
fn_aux = fn + di;
fo_4 = fn_aux * df;

[F_senoidal,t] = fsenoidal(amplitud, fo_4, phase, offset, N, fs);
ploteo_ejercicio4(F_senoidal, fs, N, fo_4, fx);
fx = fx + 1;
```



Finalmente en este caso se repite nuevamente el mismo efecto mencionado en las anteriores dos señales, con la diferencia de que ahora las componentes de frecuencias alrededor de la componente principal contendrán mayor nivel energético.

Ejercicio 4.B - Técnica Zero Padding

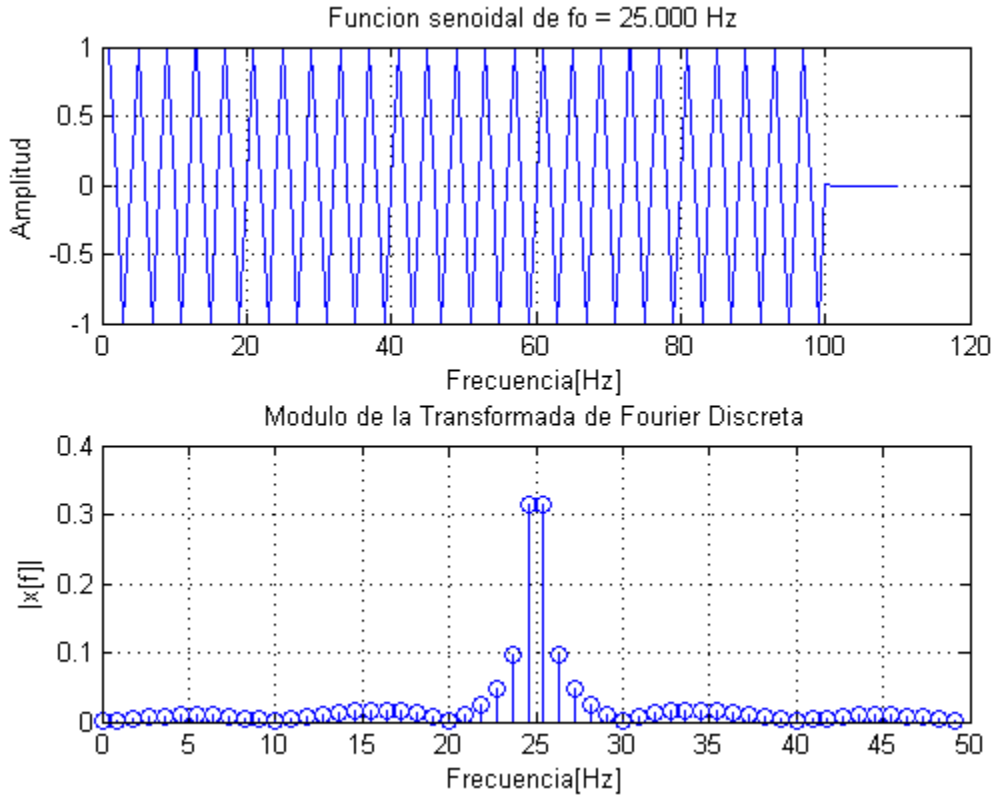
Para disminuir el efecto de leakage se puede aplicar la técnica conocida como Zero Padding a la señal. Esta técnica se basa en agregar ceros a la señal de modo tal de aumentar N y por lo tanto aumentar la resolución del espectro. Al realizar esta técnica y acercar los bins, esto trae como consecuencia inmediata (en el caso de que no se aumente el tiempo entre muestras) un aumento en la frecuencia de la señal.

Agrego N/10 Ceros

```
M = zeros(N/10,1);
```

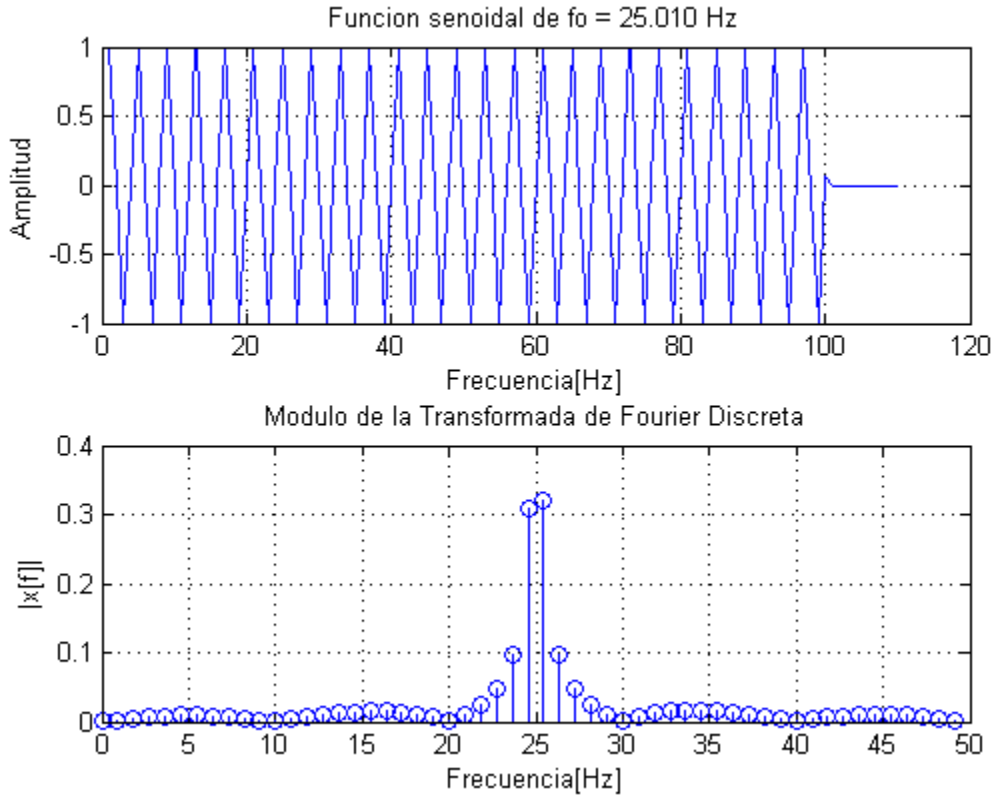
Señal fo = round(N/4) * df

```
[F_senoidal,t] = fsenoidal(amplitud, fo_1, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_1, fx);
fx = fx + 1;
```



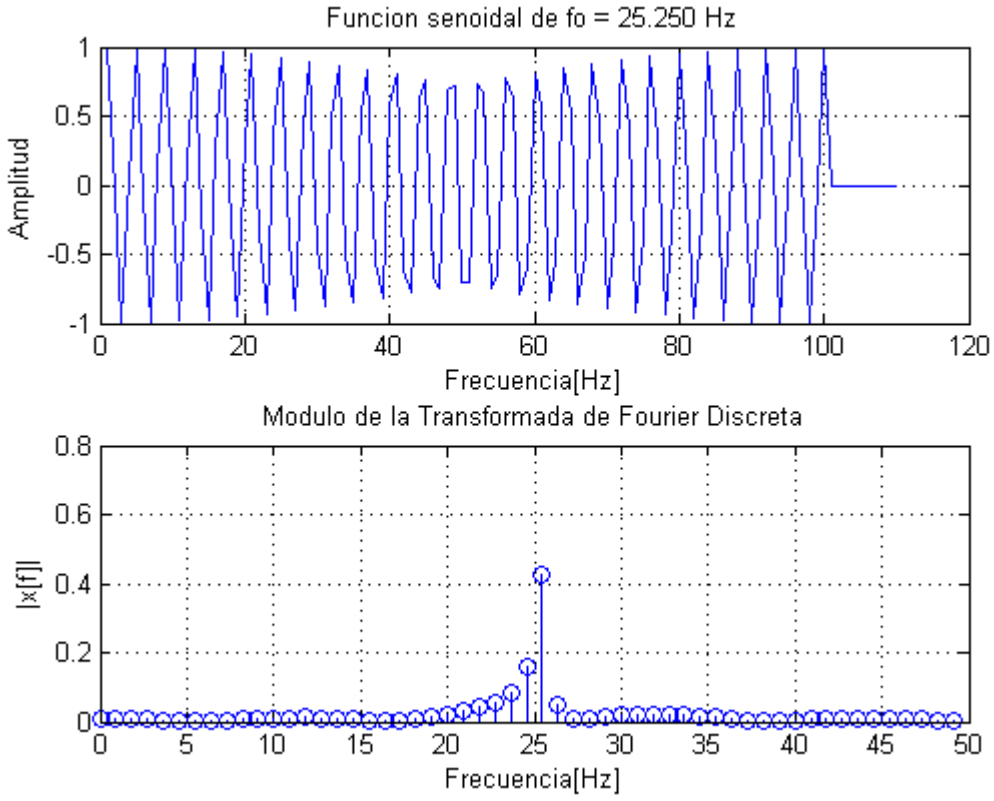
Señal fo = (round(N/4)+di) * df siendo di = 0,01

```
[F_senoidal,t] = fsenoidal(amplitud, fo_2, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_2, fx);
fx = fx + 1;
```



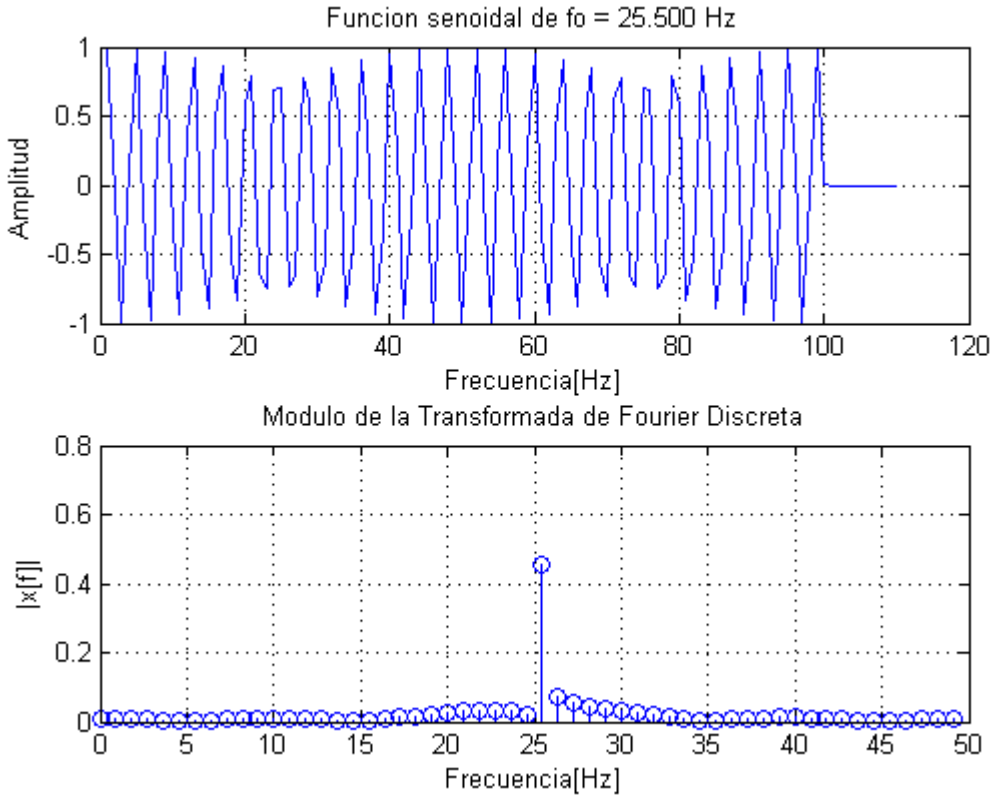
Señal fo = (round(N/4)+di) * df siendo di = 0,25

```
[F_senoidal,t] = fsenoidal(amplitud, fo_3, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_3, fx);
fx = fx + 1;
```



Señal fo = (round(N/4)+di) * df siendo di = 0,5

```
[F_senoidal,t] = fsenoidal(amplitud, fo_4, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_4, fx);
fx = fx + 1;
```

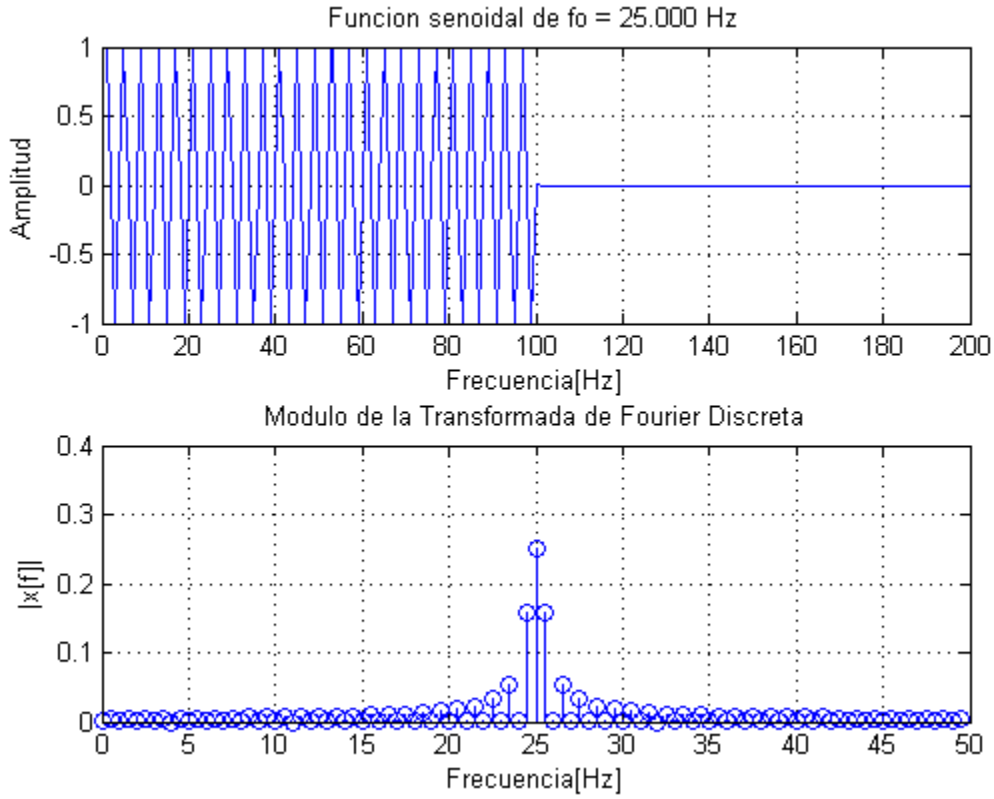


Agrego N Ceros

```
M = zeros(N,1);
```

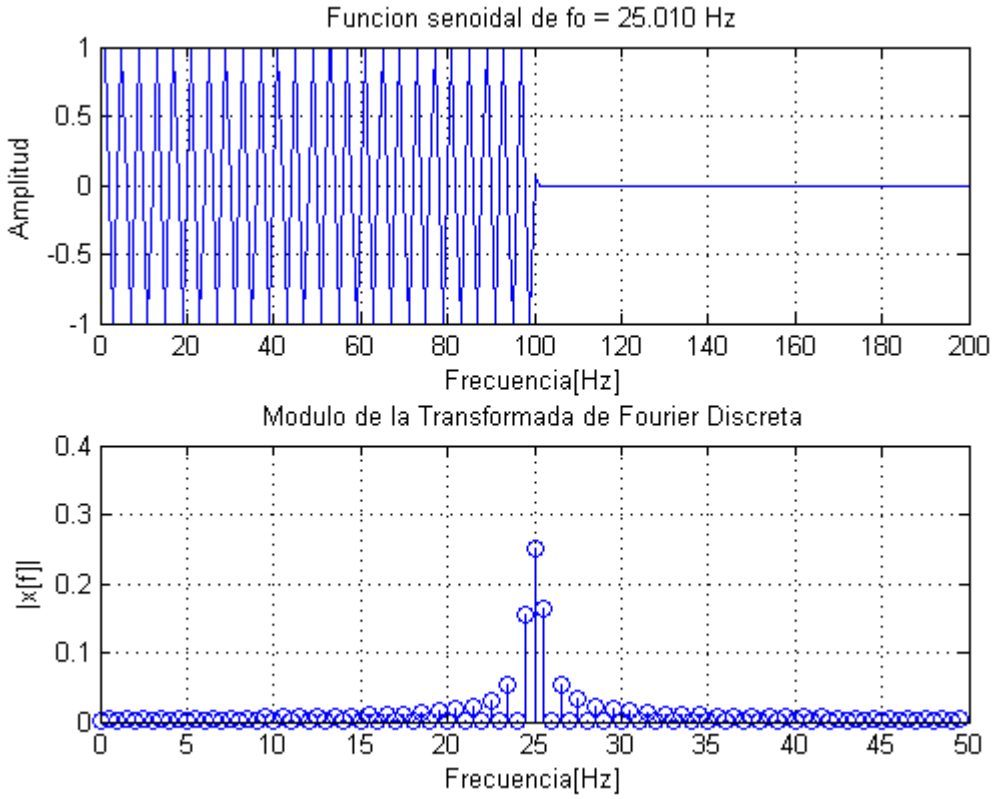
Señal fo = round(N/4) * df

```
[F_senoidal,t] = fsenoidal(amplitud, fo_1, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_1, fx);
fx = fx + 1;
```



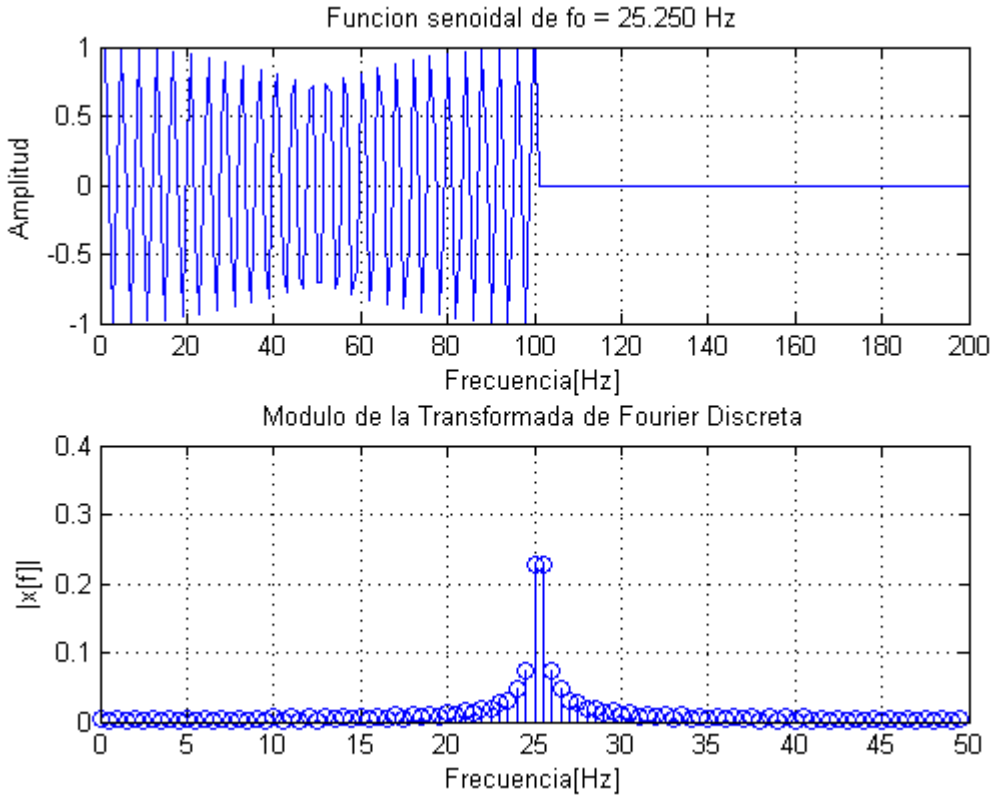
Señal fo = (round(N/4)+di) * df siendo di = 0,01

```
[F_senoidal,t] = fsenoidal(amplitud, fo_2, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_2, fx);
fx = fx + 1;
```



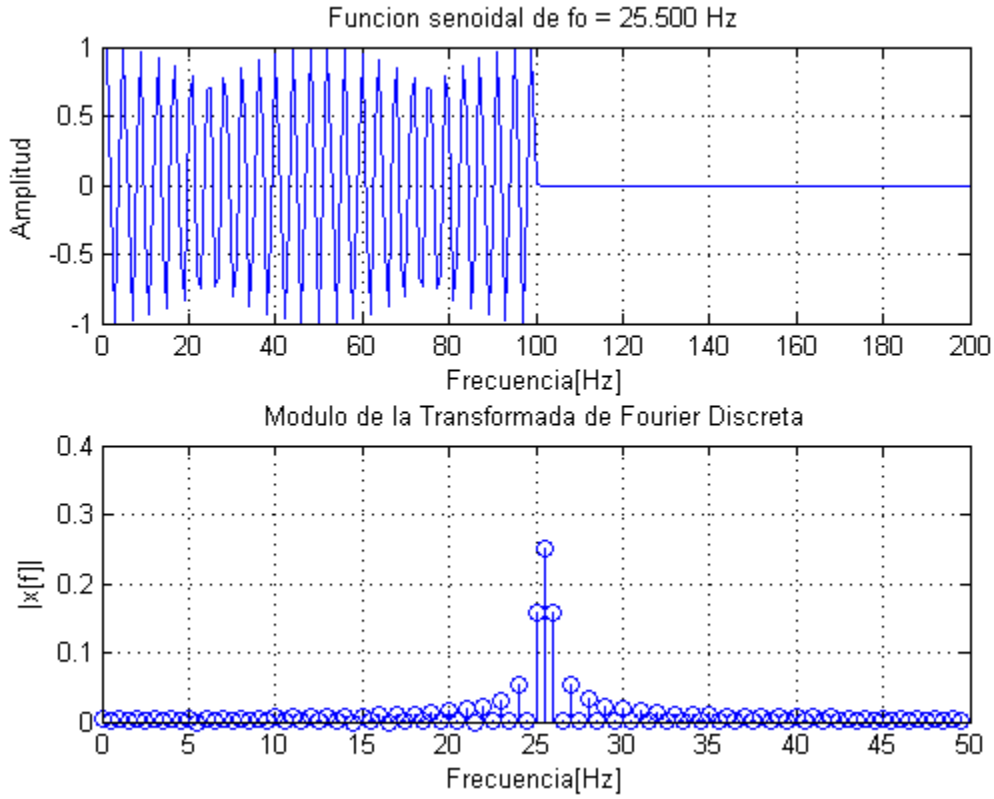
Señal fo = (round(N/4)+di) * df siendo di = 0,25

```
[F_senoidal,t] = fsenoidal(amplitud, fo_3, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_3, fx);
fx = fx + 1;
```



Señal fo = (round(N/4)+di) * df siendo di = 0,5

```
[F_senoidal,t] = fsenoidal(amplitud, fo_4, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_4, fx);
fx = fx + 1;
```

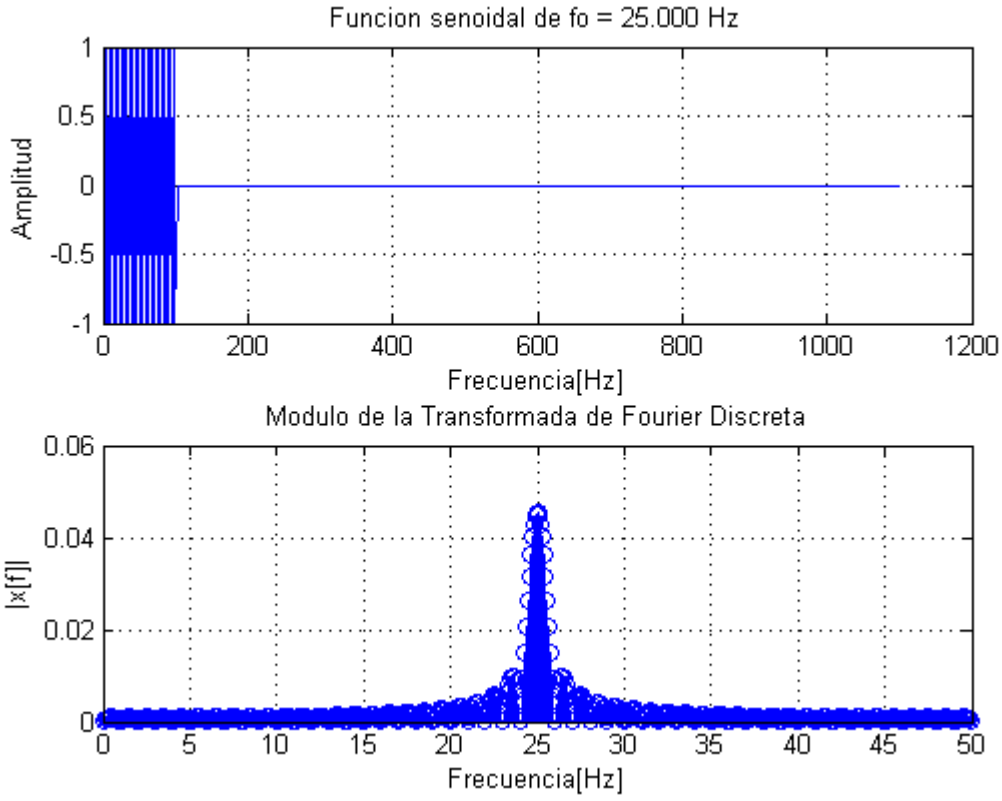


Agrego 10*N Ceros

```
M = zeros(10*N,1);
```

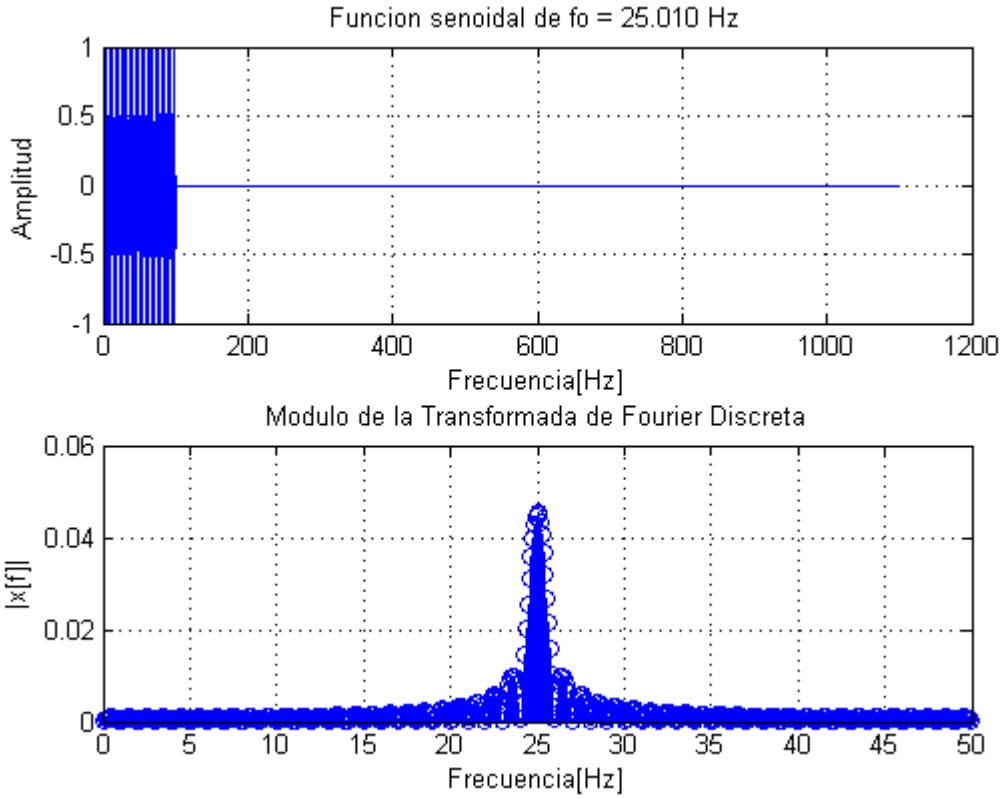
Señal fo = round(N/4) * df

```
[F_senoidal,t] = fsenoidal(amplitud, fo_1, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_1, fx);
fx = fx + 1;
```



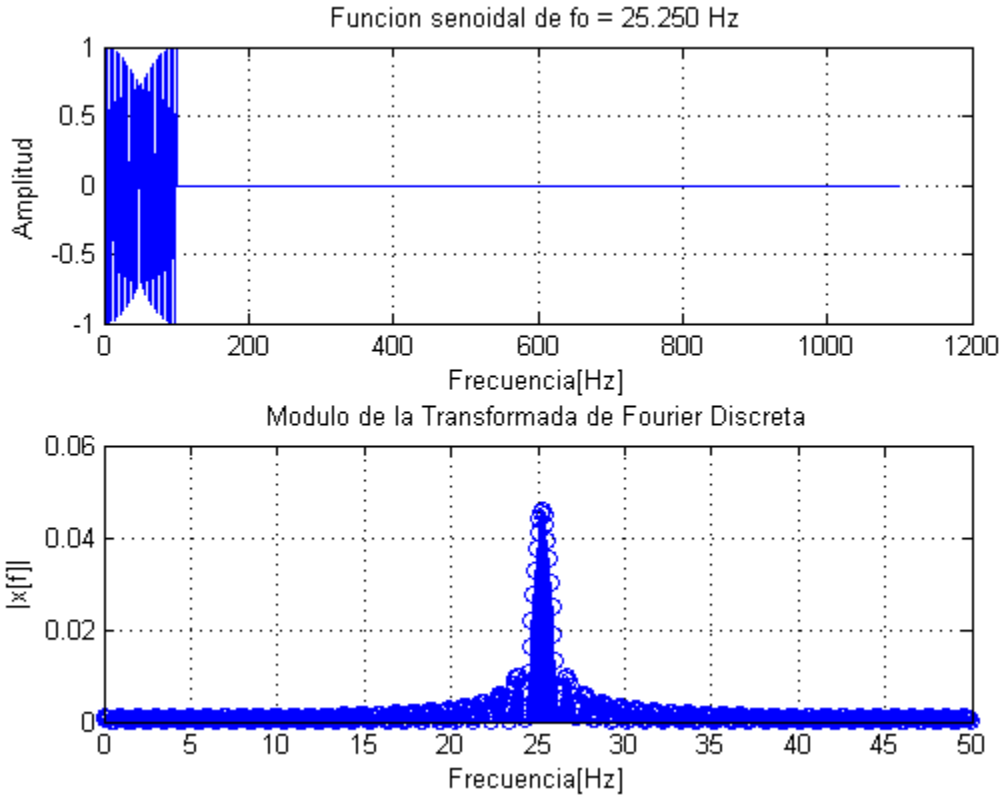
Señal fo = (round(N/4)+di) * df siendo di = 0,01

```
[F_senoidal,t] = fsenoidal(amplitud, fo_2, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_2, fx);
fx = fx + 1;
```



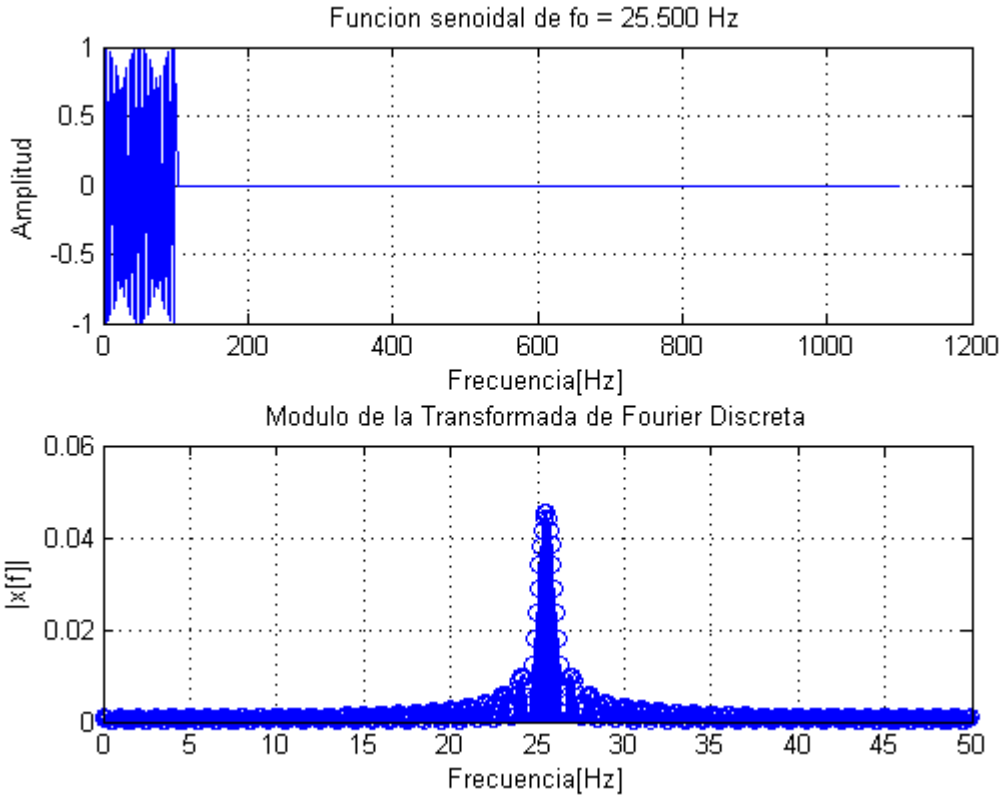
Señal fo = (round(N/4)+di) * df siendo di = 0,25

```
[F_senoidal,t] = fsenoidal(amplitud, fo_3, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_3, fx);
fx = fx + 1;
```



Señal fo = (round(N/4)+di) * df siendo di = 0,5

```
[F_senoidal,t] = fsenoidal(amplitud, fo_4, phase, offset, N, fs);
F_senoidal_b = [F_senoidal' M'];
ploteo_ejercicio4(F_senoidal_b', fs, (N+length(M)), fo_4, fx);
fx = fx + 1;

end
```



Como conclusión es interesante agregar a lo ya explicado que a medida que se aumenta la cantidad de muestras, motivo por el cual disminuye la resolución espectral, las amplitudes de las componentes en el espectro van cada vez siendo menores. Esto es debido a que se debe siempre conservar la energía de la señal (ver teorema de Parseval)

Ejercicio 4 - Código: Ploteo de espectros

```
%      \fn [] = ploteo_ejercicio4(F_senoidal, fs, N, fo, fx)
%      \brief Funcion que plotea los espectros correspondientes al ejercicio 4
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%      \date 2015.08.26
%   \param F_senoidal - Vector con la seÃ±al a plotear
%   \param fs         - Frecuencia de sampleo
%   \param N          - Numero de muestras de F_senoidal
%   \param fo         - Frecuencia de la seÃ±al
%   \param fx         - Numero de imagen

function [] = ploteo_ejercicio4(F_senoidal, fs, N, fo, fx)

f=0:fs/N:(N-1)*fs/N;

figure(fx);
subplot(2,1,1); plot(F_senoidal);
title(['Funcion senoidal de fo = ' sprintf('%3.3f',fo) ' Hz']);
xlabel('Frecuencia[Hz]'); ylabel('Amplitud'); grid;

F_senoidal_t = my_dft(F_senoidal);

subplot(2,1,2);
stem(f(1:length(F_senoidal_t)/2),abs(F_senoidal_t(1:length(F_senoidal_t)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|'); grid;

end
```

Trabajo práctico 1 - Ejercicio 5

Compare el tiempo de ejecución de la DFT implementada en 2 con la FFT en función del tamaño de la señal N. Puede utilizar las funciones tic y toc de Matlab. Grafique el tiempo de ejercuciÃ³n de ambas señales.

Algunos de los problemas que se pueden observar al realizar una DFY es la de un costo computacional elevado dado que se deben realizar N^2 multiplicaciones y (N-1)^2 sumatorias. El objetivo de este ejercicio es realizar una comparativa en tiempos entre realizar una DFT o una FFT sobre una señal.

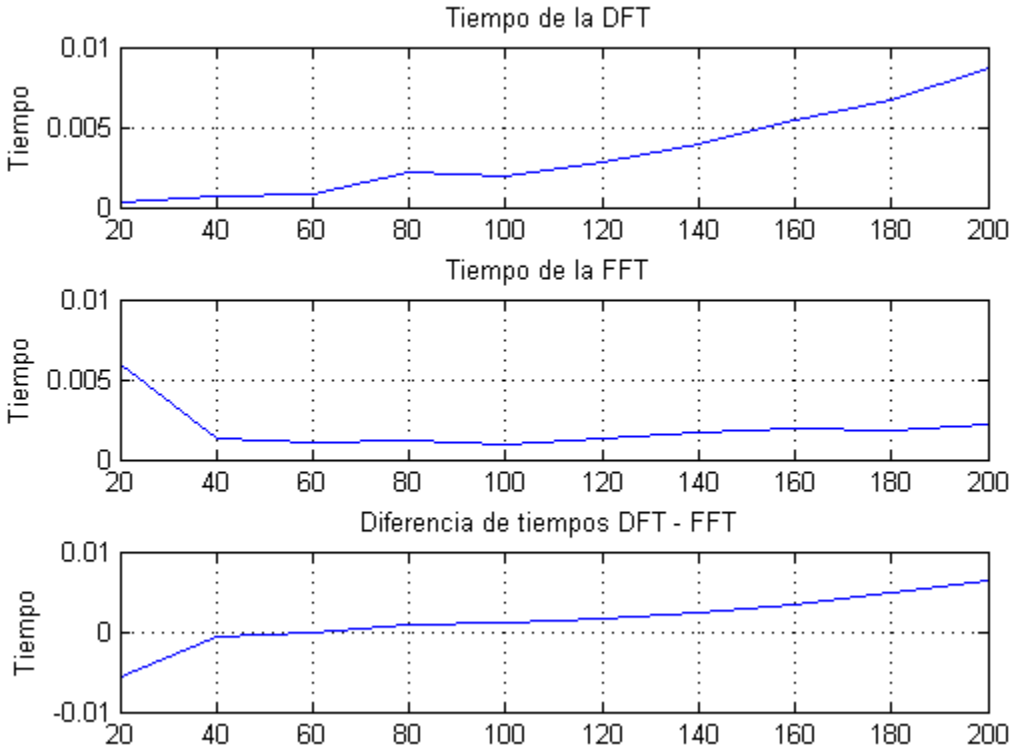
Para realizar la comparativa antes dicha se utilizara una señal senoidal generada con las funciones creadas en el ejercicio 1 del trabajo práctico cuyos parámetros se describen a continuación:

- Amplitud = 1V
- Frecuencia = 10Hz
- Offset = 0
- fs = 100
- phase = 0
- N = 20:20:200

Luego se desarrollaran las conclusiones.

Ejercicio 5 - Banco de pruebas

```
function [] = ejercicio5()

% Declaro los parÃ¡metros para realizar realizar una funciÃ³n senoidal que me
% permita verificar los tiempos necesarios para realizar una DFT y una FFT
% y asÃ- poder realizar una comparaciÃ³n entre las mismas.

amplitud = 1;
frecuencia= 10;
offset = 0;
fs = 100;
phase = 0;

N = 20:20:200;

    for i=1:length(N)

        [F_senoidal,t] = fsenoidal(amplitud, frecuencia, phase, offset, N(i), fs);

        % Tiempo DFT

        tic;
        F_senoidal_t = my_dft(F_senoidal);
        t_dft = toc;

        % Tiempo FFT

        tic;
        F_senoidal_t = fft(F_senoidal);
        t_fft = toc;

        time_dft(i) = t_dft;
        time_fft(i) = t_fft;
    end

    ejercicio5_ploteo(time_dft, time_fft, N);

end
```



De los resultados obtenidos se puede notar que a medida que la cantidad de muestras aumenta el tiempo en realizar la DFT sobre la señal estudiada aumenta de forma exponencial en comparación al tiempo en aplicar la FFT sobre la misma señal. Por lo que nos permite concluir en que realizar una FFT en una cantidad de muestras considerable (un numero mayo de 64 muestras suele considerarse) será más práctico y de menor costo computacional que realizar una DFT.

Finalmente se deja expuesto, a modo informativo, la cantidad de multiplicaciones que debe llevar a cabo una DFT por cada una multiplicación que realiza la FFT:

- N = 256 => DFT: 64 multiplicaciones / FFT: 1 multiplicación
- N = 1 048 576 => DFT: 205 multiplicaciones / FFT: 1 multiplicación
- N = 16 777 216 => DFT: 683 multiplicaciones / FFT: 1 multiplicación

Ejercicio 5 - Código: Ploteo de Gráficos

```
%       \fn [] = ejercicio5_ploteo(time_dft, time_fft, N)
%       \brief Plotea DFT, FFT y DFT - FFT
%       \details Plotea los tiempos en funciÃ³n de las muestras que tiene
%       \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%       \date 2015.08.29

function [] = ejercicio5_ploteo(time_dft, time_fft, N)
figure(1); set(gcf,'Name','Ejercicio 4');
subplot(3,1,1); plot(N,time_dft);
title('Tiempo de la DFT ');
ylabel('Tiempo'); grid;

subplot(3,1,2); plot(N,time_fft);
title('Tiempo de la FFT ');
ylabel('Tiempo'); grid;

subplot(3,1,3); plot(N,time_dft-time_fft);
title('Diferencia de tiempos DFT - FFT ');
ylabel('Tiempo'); grid;

end
```

Trabajo práctico 1 - Ejercicio 6

Para cada señal propuesta:

1. Sin realizar ninguna simulación, responda conceptualmente que contenido espectral deberá obtener tras evaluar su FFT.
2. Calcule su espectro en Matlab y discuta su predicción con los resultados obtenidos. Intente explicar dichos resultados.
3. Preste especial atención en su discusión a:

- La energía total de la señal
- La energía del tono en particular
- La localización del tono en el espectro

Siga las indicaciones para cada señal. (Ver señales en consignas PDF)

Luego de realizar la experimentación y observando los resultados obtenidos discuta si es fiable o no medir en el dominio de Fourier, por medio de la FFT los siguientes aspectos de una señal:

- Energía
- Contenido espectral
- Energía de un determinado ancho de banda o componente espectral
- Localización temporal de un determinado componente espectral

Ejercicio 6.A

Señal 1: En esta primera señal, señal senoidal periódica, en el espectro se debería poder observar un delta a la frecuencia de 9 df (resolución espectral). El resto del espectro debería tener componentes nulas.

Señal 2: Para esta segunda señal, señal senoidal de un solo ciclo, nuevamente deberíamos poder observar un delta en el espectro a 9 df (resolución espectral). A diferencia de la primera señal en este la amplitud de la componente debería disminuir de acuerdo a que en el tiempo se tiene menor cantidad de información. Además debido a que se completó con ceros parte de la señal podría observarse el efecto de leakage (desparramo) apareciendo componentes no nulas de frecuencia alrededor de la componente de mayor energía.

Señal 3: En esta señal, señal senoidal de un ciclo desplazada hacia el tercer periodo, en el espectro de módulo deberá ser igual a la señal 2. Las diferencias respecto de la señal 2 se observaran en el espectro de fase.

Señal 4: En la señal, dos ciclos de dos senoidales de dos frecuencias distintas, el espectro estará compuesto por dos deltas, debido a que las frecuencias de las mismas están muy cerca entre sí no se podrá identificar a las mismas. Además aparecerá el efecto de leakage debido a no contar con la suficiente resolución espectral.

Señal 5: Al igual que en el caso 2 y 3, en esta la señal, la seáal generada contendrá el mismo espectro en modulo que la señal 4 y los cambios se podrán observar respecto del espectro de fase debido a que ambas señales contienen la misma información.

Señal 6: Esta señal, está compuesta por tres ciclos de una señal senoidal de una única frecuencia, por lo tanto en el espectro aparecerá una delta (parecido al caso de la señal dos) con la diferencia que será de mayor amplitud debido a que la señal cuenta con mayor cantidad de información y por tal motivo mayor energía. Además aparecerá el efecto de leakage nuevamente debido a que puede no cumplirse f = K * fs/N siendo k = {1,2,N}

Señal 7: Esta señal, está compuesta por tres ciclos de tres funciones senoidales de misma frecuencia pero amplitudes diferentes. En el espectro debería ver el delta de mayor amplitud sobre la fo de la señal y luego a los alrededores del mismo, por el efecto de leakage otras componentes de menor frecuencia.

Señal 8: Esta señal está compuesta por tres ciclos de la señal 7 por lo tanto en el espectro se observara lo mismo, con la diferencia de al tener mayor cantidad de información, la amplitud de las componentes en frecuencia aumentara.

Señal 9: En este caso, se consta de dos ciclos de una senoidal defasados uno del otro en pi radianes (180°). Aplicando el principio de superposición, se deberá observar en el espectro de módulo a la frecuencia de la señal un valor nulo de componente.

Ejercicio 6.B.C.D - Banco de pruebas

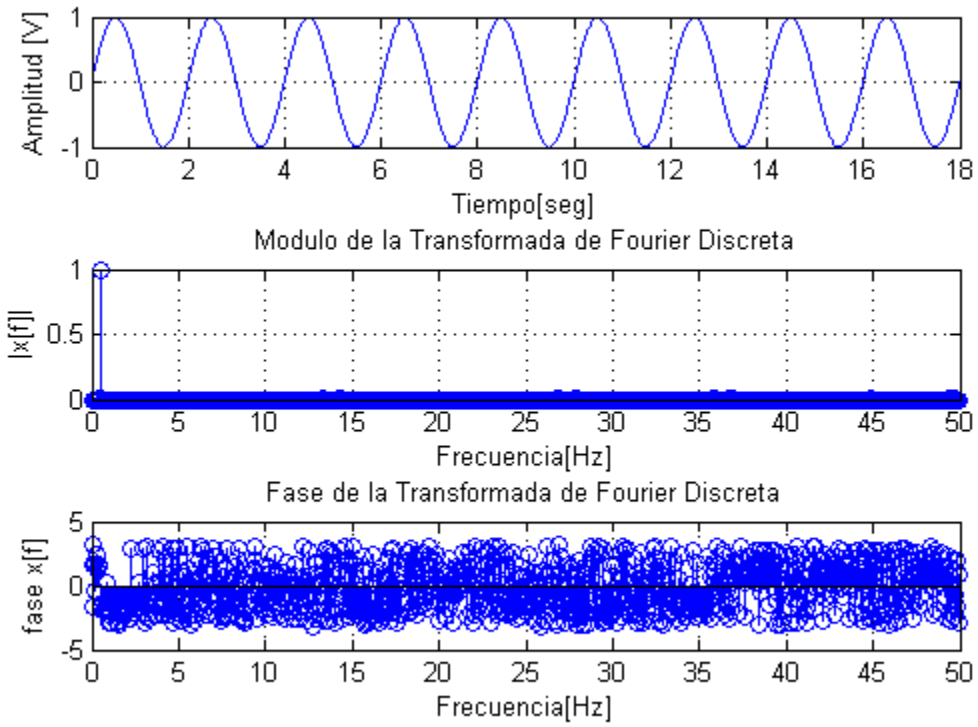
```
function [] = ejercicio6()
```

```
% Declaro los parÃ¡metros para realizar las funciones pedidas en el
% enunciado.

N = 1800;
fs = 100;
df = fs/N;

amp = 1;
fo= df;
offset = 0;
phase = 0;

fx = 1;
```

A continuación se pueden ver tanto el código como la señal generada (correspondientes a las pedidas en el enunciado del ejercicio) junto con el espectro correspondiente a cada señal, tanto el espectro de módulo como el de fase.

Ejercicio 6.B.C.D - Primera Señal

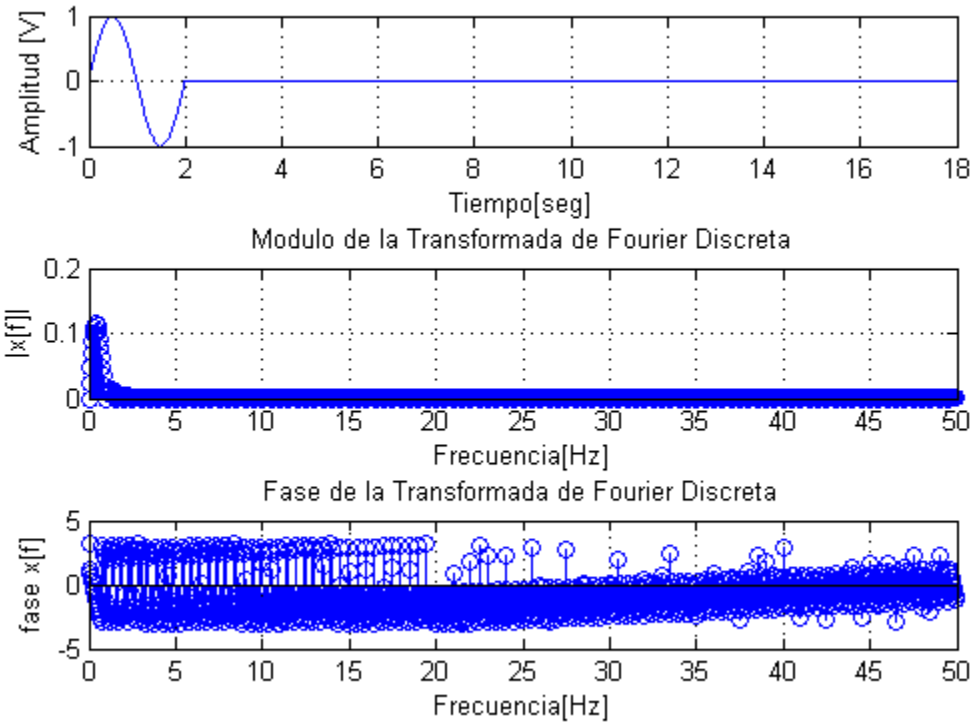
```
[output,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Segunda Señal

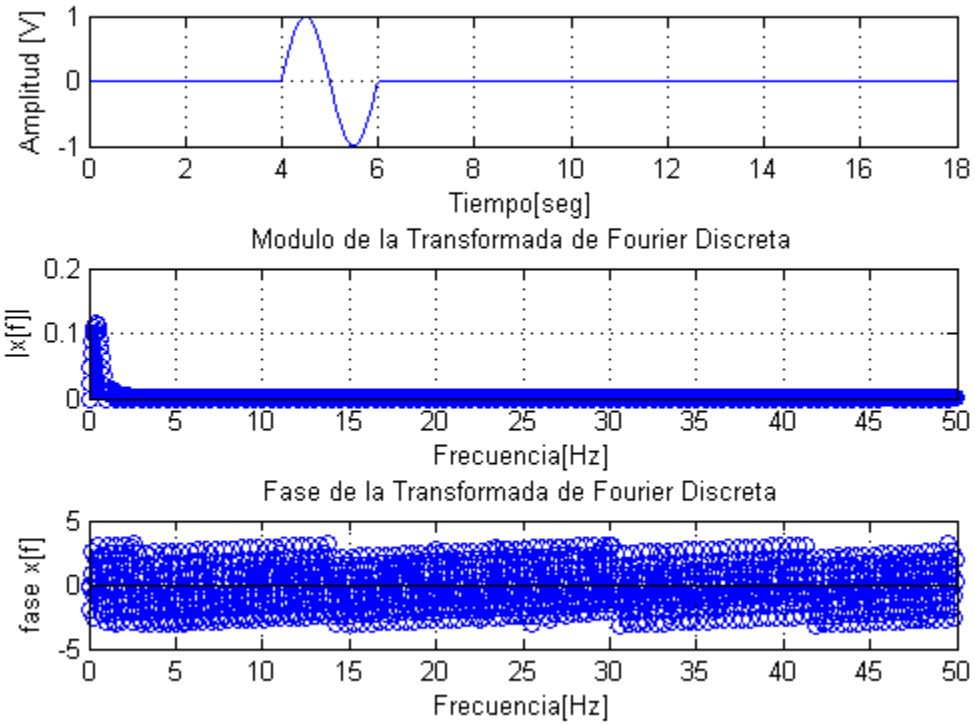
```
fx = fx + 1;
[output,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
j=N/9:length(output);
output(j) = 0;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Tercera Señal

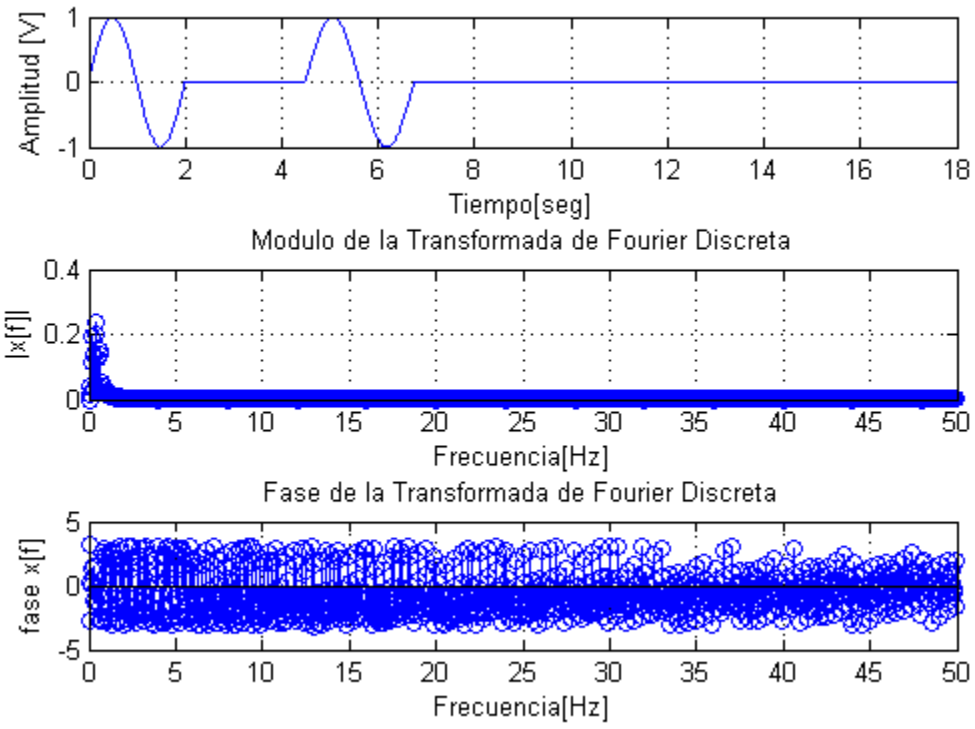
```
fx = fx + 1;
[output,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
j=1:2*N/9;
k = 3* N/9: length(output);
output(j) = 0; output(k) = 0;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Cuarta Señal

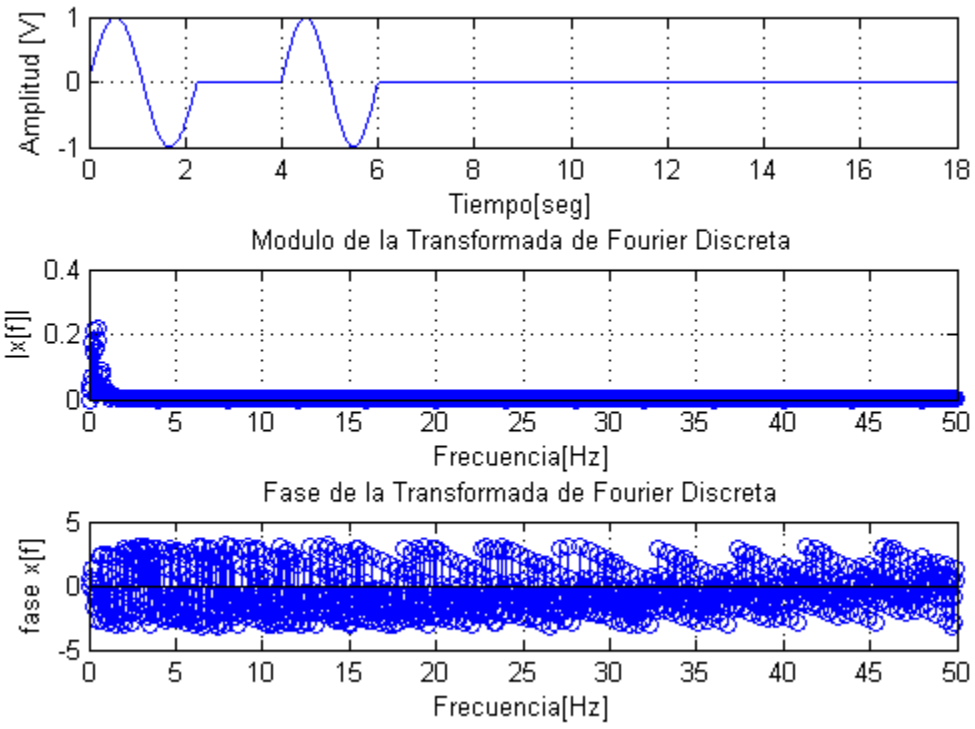
```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_2,x_temp] = fsenoidal(amp, 8*fo, phase, offset, N, fs);

j=N/9:length(output_1);
output_1(j) = 0;
f=1:2*N/8;
g = 3* N/8: length(output_2);
output_2(f) = 0; output_2(g) = 0;
output = output_1 + output_2;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Quinta Señal

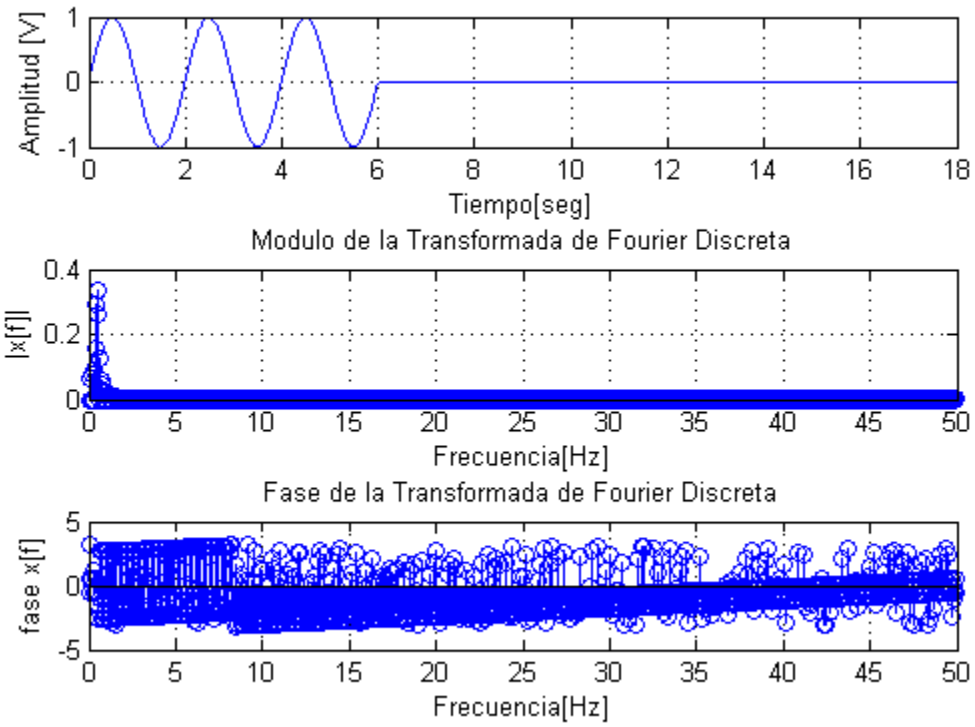
```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_2,x_temp] = fsenoidal(amp, 8*fo, phase, offset, N, fs);

j=N/8:length(output_2);
output_2(j) = 0;
f=1:2*N/9;
g = 3* N/9: length(output_1);
output_1(f) = 0; output_1(g) = 0;
output = output_1 + output_2;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Sexta Señal

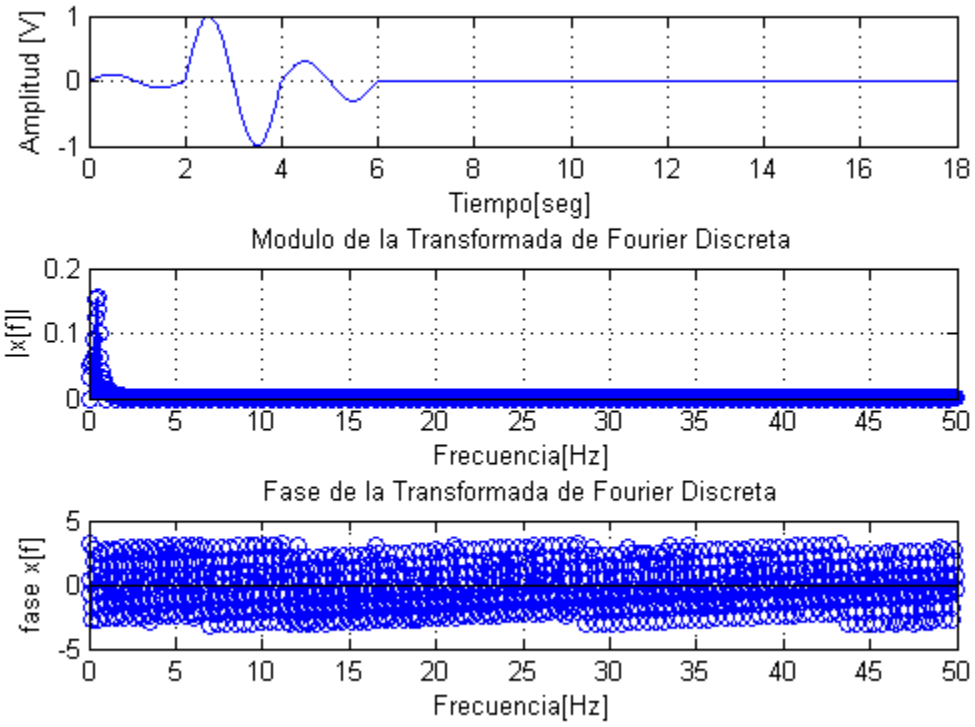
```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_2,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_3,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);

j=N/9:length(output_1);
output_1(j) = 0;
j=1:N/9; k = 2*N/9:length(output_2);
output_2(j)=0; output_2(k)=0;
j=1:2*N/9; k = 3*N/9:length(output_3);
output_3(j)=0; output_3(k)=0;
output = output_1 + output_2 + output_3;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Septima Señal

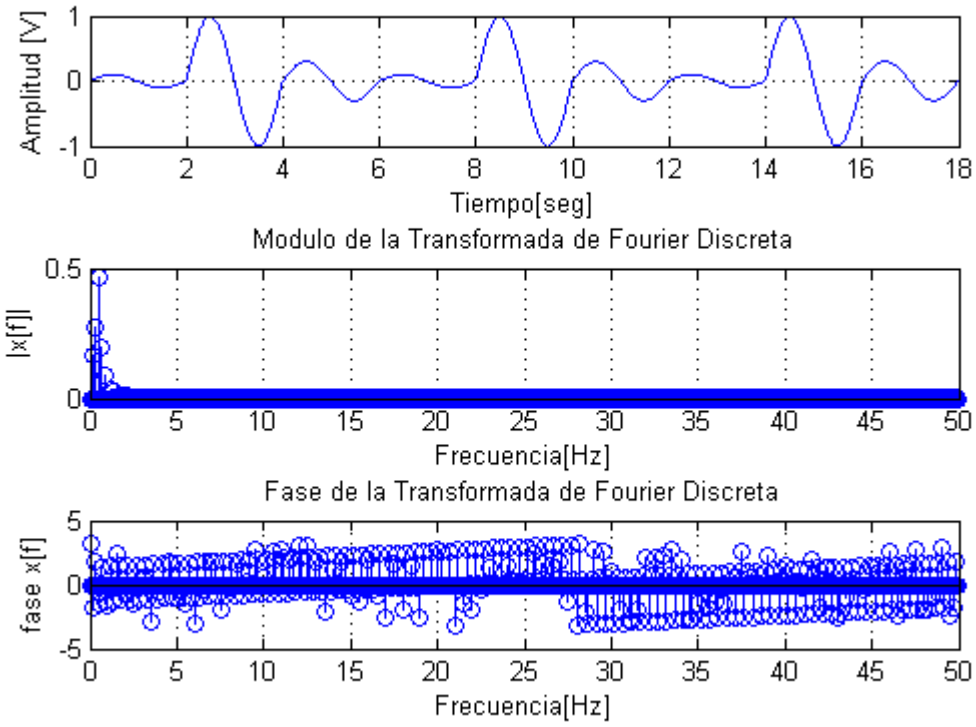
```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(0.1*amp, 9*fo, phase, offset, N, fs);
[output_2,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_3,x_temp] = fsenoidal(0.3*amp, 9*fo, phase, offset, N, fs);

j=N/9:length(output_1);
output_1(j) = 0;
j=1:N/9; k = 2*N/9:length(output_2);
output_2(j)=0; output_2(k)=0;
j=1:2*N/9; k = 3*N/9:length(output_3);
output_3(j)=0; output_3(k)=0;
output = output_1 + output_2 + output_3;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```



La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Octava Señal

```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(0.1*amp, 9*fo, phase, offset, N, fs);
[output_2,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
[output_3,x_temp] = fsenoidal(0.3*amp, 9*fo, phase, offset, N, fs);

j=N/9:length(output_1);
output_1(j) = 0;

j=1:N/9; k = 2*N/9:length(output_2);
output_2(j)=0; output_2(k)=0;

j=1:2*N/9; k = 3*N/9:length(output_3);
output_3(j)=0; output_3(k)=0;
output = output_1 + output_2 + output_3;
output = repmat(output(1:600),3);

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```

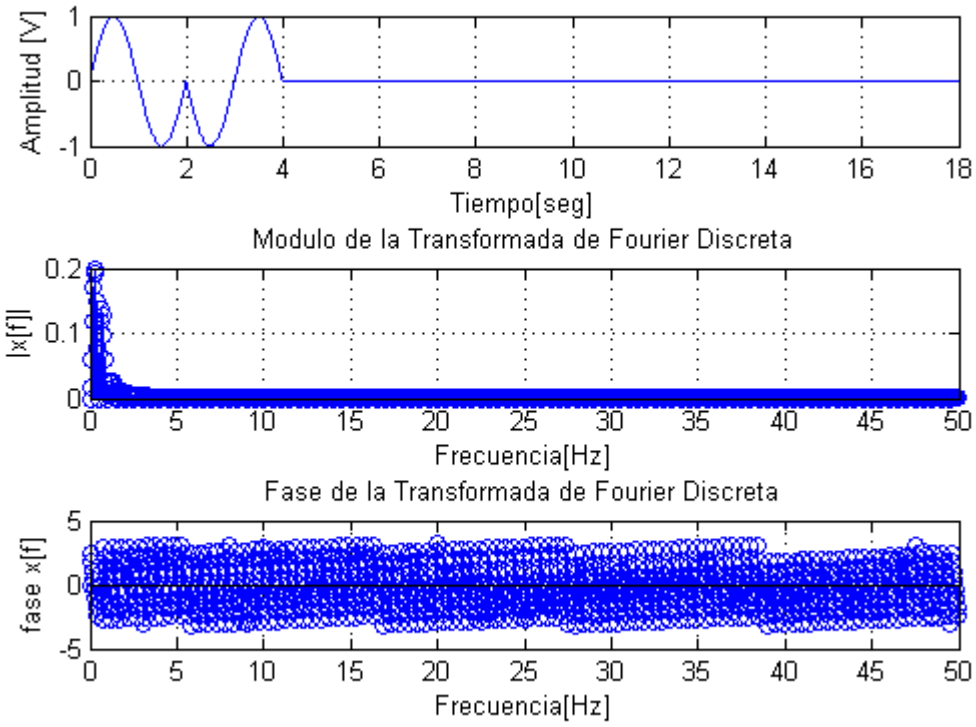


La simulación verifica lo esperado en 1.A

Ejercicio 6.B.C.D - Novena Señal

```
fx = fx + 1;
[output_1,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);
phase = pi;
[output_2,x_temp] = fsenoidal(amp, 9*fo, phase, offset, N, fs);

j=N/9:length(output_1);
output_1(j) = 0;
j=1:N/9;
f=2*N/9:length(output_2);
output_2(j) = 0; output_2(f) = 0;
output = output_1 + output_2;

f=0:fs/N:(N-1)*fs/N;

plot_ejercicio6(output,x_temp,f,N, fx);
```

```
end
```



La simulación verifica lo esperado en 1.A

Ejercicio 6 – Conclusiones

Respecto de la energía utilizar, medir en el domino de Fourier, por medio de la FFT, no presenta inconvenientes debido a que el cálculo de la energía tiene en cuenta todos los coeficientes (todas las muestras) y no depende ni de la ubicación de las componentes en el espectro ni de la fase, solo del cuadrado de la amplitud de las mismas. En lo que respecta a la localización temporal de un determinado componente espectral, este puede volverse dificultoso o no encontrar debido a la resolución espectral que depende de la frecuencia de muestreo "fs" y del número de muestras tomadas. Además para detectar el espectro de la señal respecto al desparramo producido por el efecto de leakage se debe tomar una cantidad de muestras suficientemente alta de moto tal de disminuir la resolución espectral. Se debe recordar que esta tiene su limitación, físicamente no se puede reducir la resolución tanto como uno quiera.

Ejercicio 6 - CÃ³digo: Ploteo de señales

```
function [] = plot_ejercicio6(output,x_temp,f,N, fx)

figure(fx); set(gcf,'Name','Ejercicio 6');
subplot(3,1,1); plot(x_temp,output,'b');
xlabel('Tiempo[seg]'); ylabel('Amplitud [V]'); grid;

output_fft = fft(output);
output_fft = output_fft*2/N;

subplot(3,1,2); stem(f(1:length(output)/2),abs(output_fft(1:length(output)/2)));
title('Modulo de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('|x[f]|'); grid;

subplot(3,1,3); stem(f(1:length(output)/2),angle(output_fft(1:length(output)/2)));
title('Fase de la Transformada de Fourier Discreta');
xlabel('Frecuencia[Hz]'); ylabel('fase x[f]'); grid;

end
```

Trabajo práctico 1 - Ejercicio 7

Simule el efecto de cuantizar una señal continua en el tiempo mediante un conversor analógico digital. Para ello analice señales determinísticas como las generadas en 1, y otras que varían de forma más aleatoria, por ejemplo añadiendo ruido mediante random. Muestre un ejemplo representativo de cada uno de estos casos. Puede simular una señal continua mediante una señal muestreada a una fs muy alta en comparación con las fs que quiere estudiar, y un tipo de dato de cuantización para cada tamaño de muestra mediante:

1. La propia señal temporal y su espectro
2. Una descripción de la señal, por medio de su energía, valor medio, RMS
3. Un histograma de los valores que toma la señal
4. Una descripción del histograma, por medio de estadísticos como la media, desvío estándar y varianza.

Nota (1): Se decidió separar el ejercicio en dos partes, una de señal determinística y una segunda parte para la señal no determinística. Dentro de cada parte se describe los valores característicos "b" y "c", los ploteos de la señal temporal y sus espectros "a" y finalmente los histogramas "d".

Nota (2): Para una correcta visualización, se incluye el código a medida que se muestra la resolución para un nivel de cuantización de bits. Luego para 8 y 16 bit solo se muestran las imágenes y conclusiones debido a que es el mismo código con la única variación de Nbit.

Ejercicio 7 - Banco de prueba

```
function [] = ejercicio7 ()
```

```
% Parametros de prueba:
amplitud = 1;
fo = 1000;
fase = 0;
offset = 0;
N = 1000;
fs = 44000;

fx = 1;
```

Análisis con un nivel de cuantización de 4 bits

A continuación se simula el efecto de cuantizar una señal con un nivel de cuntización de 4 bit. Se realiza además un análisis sobre el ruido de cuantización para dicho nivel.

```
Nbit = 4;
```

```
fx = func_det(amplitud, fo, fase, offset, N, fs, Nbit, fx);
```

```
fx = func_no_det(amplitud, fo, fase, offset, N, fs, Nbit, fx);
```

```
end
```

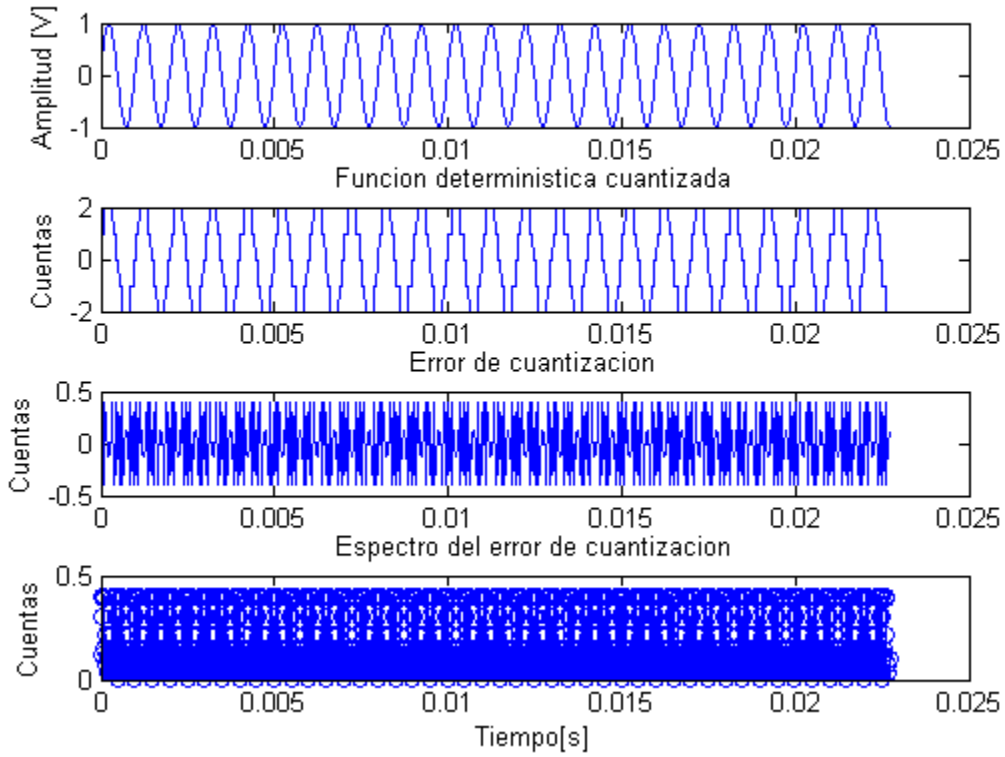
Ejercicio 7 - Código: Función determinística

```
%      \fn [] = [fx] = func_det(amplitud, fo, fase, offset, N, fs, Nbit, fx)
%      \brief  Realiza el ejercicio a partir de una seÃ±al determinÃ-stica
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param  amplitud _ Amplitud de la seÃ±al
%   \param  fo       - Frecuencia de la seÃ±al
%   \param  fase     - Fase inicial de la seÃ±al
%   \param  offset   - Offset de seÃ±al
%   \param  N        - Numero de muestras de la seÃ±al
%   \param  fs       - Frecuencia de sampleo de seÃ±al
%   \param  Nbit     - NÃºmero de cuantizaciÃ³n de ADC
%   \param  fx       - Figura actual
%   \return fx       - PrÃ³xima figura
%      \date 2015.08.19

function [fx] = func_det(amplitud, fo, fase, offset, N, fs, Nbit, fx)
```

```
% ParÃ¡metros para el ADC:
K = 1;
Vref = 3.3;

% ParÃ¡metros para generar el ruido:
mu = 0;

% Preparo el ADC:
escalado_sup = (2^(Nbit-1))-1;
Resolucion = Vref/escalado_sup;

%Obtengo la seÃ±al deterministica - Ejercicio 1:
[y,t] = fsenoidal (amplitud, fo, fase, offset, N, fs);

Cuentas = y/Resolucion;
Sq_1 = round(Cuentas);
Sn_1 = Sq_1 - Cuentas;
```

Ploteo de gráficos

```
plot_ejercicio7(y,t,Sq_1,Sn_1,fx);
fx = fx + 1;
```



Análisis de la señal: energía, valor medio y valor eficaz de la señal determinística.

```
disp('SeÃ±al deterministica: ');
```

```
SeÃ±al deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

```
[Energia V_medio V_eficaz] = datos_EVV(y);
```

Energía:

```
disp(['El valor de energia es: ' num2str(Energia) ' Joule']);
```

```
El valor de energia es: 500 Joule
```

Valor Medio:

```
disp(['El valor medio es: ' num2str(V_medio) ' V']);
```

```
El valor medio es: 0.0074909 V
```

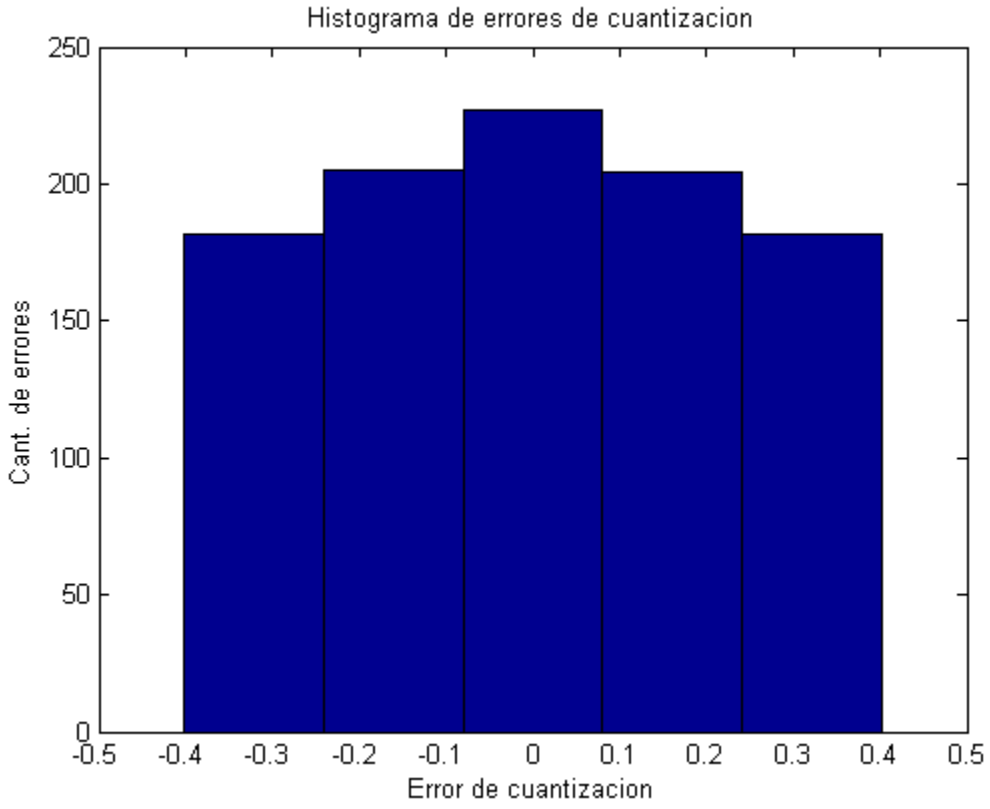
Valor Eficaz:

```
disp(['El valor eficaz es: ' num2str(V_eficaz) ' V']);
```

```
El valor eficaz es: 0.70711 V
```

Histograma

```
plot_ejercicio7_hist(Sn_1,fx);
fx = fx + 1;
```



Análisis del histograma: Media, desvío estándar y varianza.

```
disp('AnÃ¡lisis de Sn: ');
```

```
AnÃ¡lisis de Sn:
```

```
[Energia V_medio V_eficaz Var Desvio] = datos_EVVVD(Sn_1);
```

Energía:

```
disp(['El valor de energia es: ' num2str(Energia) ' Joule']);
```

```
El valor de energia es: 60.7418 Joule
```

Valor Medio:

```
disp(['El valor medio es: ' num2str(V_medio) ' V']);
```

```
El valor medio es: 0.00011018 V
```

Valor Eficaz:

```
disp(['El valor eficaz es: ' num2str(V_eficaz) ' V']);
```

```
El valor eficaz es: 0.24646 V
```

Varianza:

```
disp(['La varianza es: ' num2str(Var)]);
```

```
La varianza es: 60.7418
```

Desvío estándar:

```
disp(['El desvio es: ' num2str(Desvio)]);
```

```
El desvio es: 0.24646
```

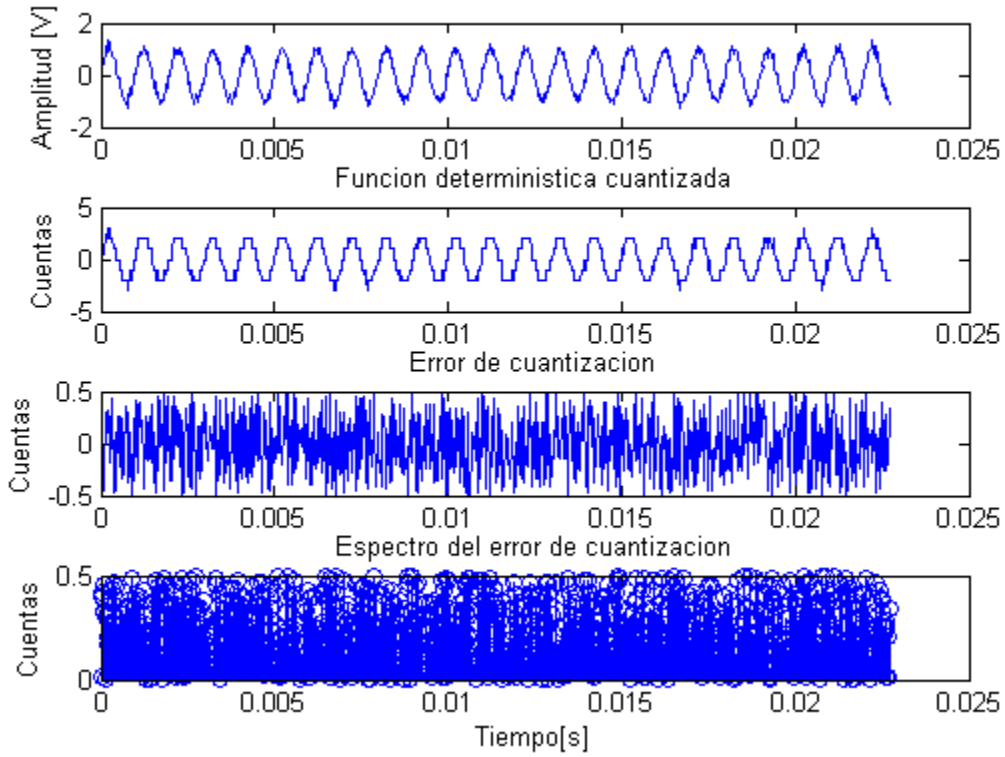
```
end
```

Ejercicio 7 - Código: Función no determinística

```
%      \fn [] = [fx] = func_no_det(amplitud, fo, fase, offset, N, fs, Nbit, fx)
%      \brief  Realiza el ejercicio a partir de una seÃ±al no determinÃ-stica
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param  amplitud _ Amplitud de la seÃ±al
%   \param  fo       - Frecuencia de la seÃ±al
%   \param  fase     - Fase inicial de la seÃ±al
%   \param  offset   - Offset de seÃ±al
%   \param  N        - Numero de muestras de la seÃ±al
%   \param  fs       - Frecuencia de sampleo de seÃ±al
%   \param  Nbit     - NÃºmero de cuantizaciÃ³n de ADC
%   \param  fx       - Figura actual
%   \return fx       - PrÃ³xima figura
%      \date 2015.08.19

function [fx] = func_no_det(amplitud, fo, fase, offset, N, fs, Nbit, fx)
```

```
% ParÃ¡metros para el ADC:
K = 1;
Vref = 3.3;

% ParÃ¡metros para generar el ruido:
mu = 0;

% Preparo el ADC:
escalado_sup = (2^(Nbit-1))-1;
Resolucion = Vref/escalado_sup;

% Armo el ruido que le sumo a la senoidal para hacer una seÃ±al no
% deterministica.

[y,t] = fsenoidal (amplitud, fo, fase, offset, N, fs);

x = randn (N,1);
y = y' + x' * amplitud/10 + mu;
Cuentas = y/Resolucion;
Sq_2 = round(Cuentas);
Sn_2 = Sq_2 - Cuentas;

% Ploteo de grÃ¡ficos
```

```
plot_ejercicio7(y,t,Sq_2,Sn_2,fx);
fx = fx + 1;
```



Análisis de la señal: energía, valor medio y valor eficaz de la señal no determinística.

```
disp('SeÃ±al no deterministica: ');
```

```
SeÃ±al no deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

```
[Energia V_medio V_eficaz] = datos_EVV(y);
```

Energía:

```
disp(['El valor de energia es: ' num2str(Energia) ' Joule']);
```

```
El valor de energia es: 512.0105 Joule
```

Valor Medio:

```
disp(['El valor medio es: ' num2str(V_medio) ' V']);
```

```
El valor medio es: 0.0042277 V
```

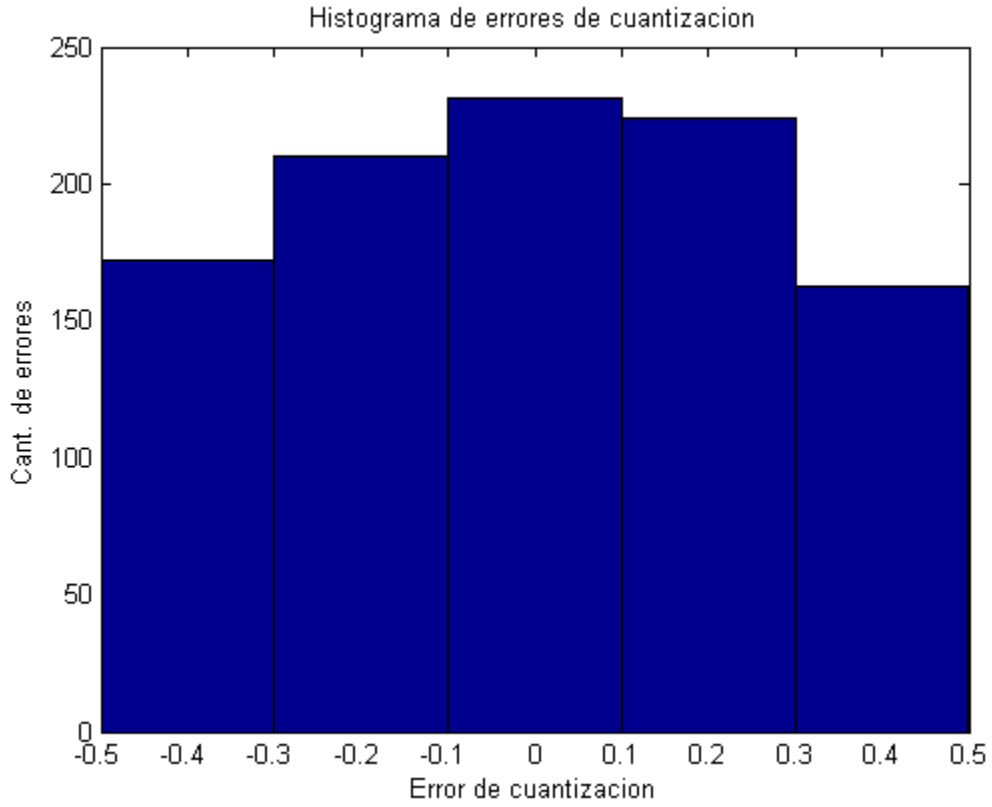
Valor Eficaz:

```
disp(['El valor eficaz es: ' num2str(V_eficaz) ' V']);
```

```
El valor eficaz es: 0.71555 V
```

Histograma

```
plot_ejercicio7_hist(Sn_2,fx);
fx = fx + 1;
```



Análisis del histograma: Media, desvío estándar y varianza.

```
disp('AnÃ¡lisis de Sn: ');
[Energia V_medio V_eficaz Var Desvio] = datos_EVVVD(Sn_2);
```

```
AnÃ¡lisis de Sn:
```

Energía:

```
disp(['El valor de energia es: ' num2str(Energia) ' Joule']);
```

```
El valor de energia es: 73.6953 Joule
```

Valor Medio:

```
disp(['El valor medio es: ' num2str(V_medio) ' V']);
```

```
El valor medio es: -0.0019679 V
```

Valor Eficaz:

```
disp(['El valor eficaz es: ' num2str(V_eficaz) ' V']);
```

```
El valor eficaz es: 0.27147 V
```

Varianza:

```
disp(['La varianza es: ' num2str(Var)]);
```

```
La varianza es: 73.6915
```

Desvío estándar:

```
disp(['El desvio es: ' num2str(Desvio)]);
```

```
El desvio es: 0.27146
```

```
end
```

Ejercicio 7 - Código: Ploteo de señales.

```
%      \fn [] = plot_ejercicio7(y,t,Sq,Sn,fx)
%      \brief Plotea la seÃ±al y el espectro.
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param y  - SeÃ±al
%   \param t  - Vector tiempo:muestra
%   \param Sq - SeÃ±al cuantificada
%   \param Sn - SeÃ±al error de cuantizaciÃ³n
%   \param fx - NÃºmero de figura
%      \date 2015.08.19


function [] = plot_ejercicio7(y,t,Sq,Sn,fx)

figure(fx); set(gcf,'Name','Ejercicio 7');
subplot(4,1,1); plot(t,y);
ylabel('Amplitud [V]');

subplot(4,1,2); plot(t,Sq);
title('Funcion deterministica cuantizada');
ylabel('Cuentas');

subplot(4,1,3); plot(t,Sn);
title('Error de cuantizacion');
ylabel('Cuentas');

subplot(4,1,4); stem(t,abs(Sn));
title('Espectro del error de cuantizacion');
xlabel ('Tiempo[s]'); ylabel('Cuentas');

end
```

Ejercicio 7 - Código: Ploteo de histograma.

```
%      \fn [] = plot_ejercicio7_hist(Sn,fx)
%      \brief Plotea el histogrÃ¡ma de la seÃ±al: Cant. de errores en funciÃ³n
%          de errores de cuantizaciÃ³n.
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param Sn - SeÃ±al
%   \param fx - NÃºmero de figura
%      \date 2015.08.19


function [] = plot_ejercicio7_hist(Sn,fx)

figure(fx); set(gcf,'Name','Ejercicio 7');
hist(Sn,5);
title('Histograma de errores de cuantizacion');
xlabel('Error de cuantizacion'); ylabel('Cant. de errores');

end
```

Ejercicio 7 - Código: Cálculo de energía, valor medio y valor eficaz.

```
%      \fn [] = datos_EVV(y)
%      \brief Informa los valores estudiados de la seÃ±al.
%      \details Informa: EnergÃ-a, valor medio, valor eficaz
%      \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param  y        - SeÃ±al a estudiar
%   \return Energia  - EnergÃ-a de la seÃ±al
%   \return V_medio  - Valor medio de la seÃ±al
%   \return V_eficaz - Valor eficaz de la seÃ±al
%      \date 2015.08.19

function [Energia V_medio V_eficaz] = datos_EVV(y)

% Energia:
Energia = fenergia(y);

% Valor Medio:
V_medio = fv_medio(y);

% Valor Eficaz:
V_eficaz = feficaz(y);

end
```

Ejercicio 7 - Código: Calculo de energía, valor medio, valor eficaz, varianza y desvío

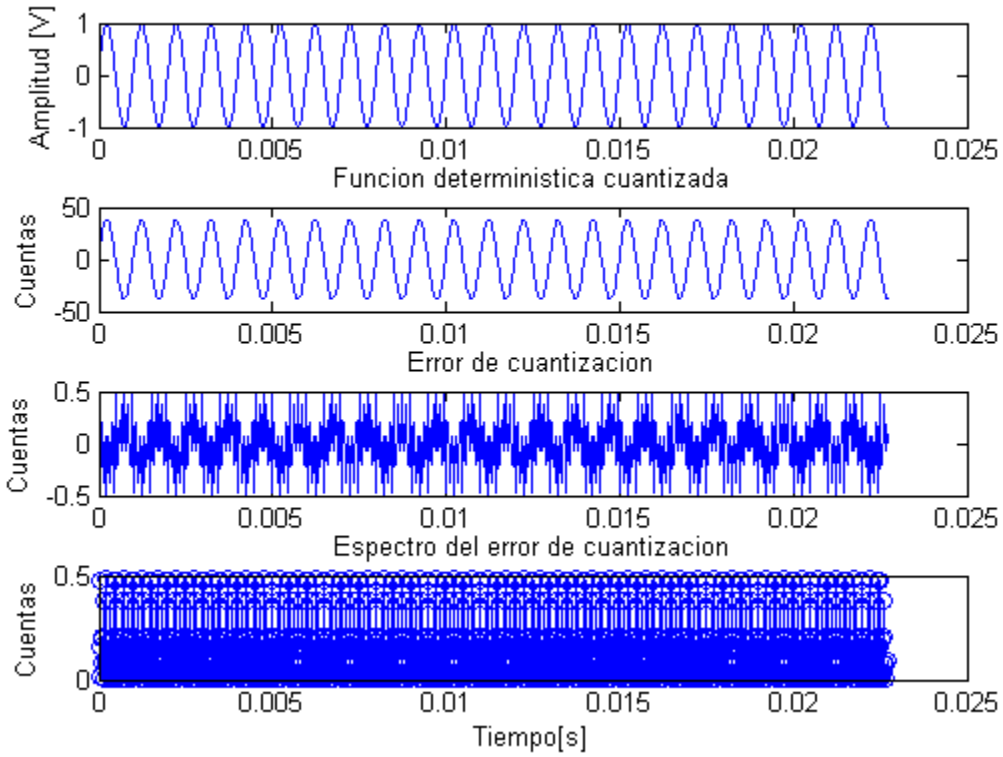
La función informa los valores de energía, valor medio, valor eficaz, varianza y desvío estándar de la señal.

```
function [Energia Medio Eficaz Var Desv] = datos_EVVVD(Sn)
%
%       \fn [] = datos_EVVVD(Sn)
%       \brief Devuelve los valores acerca de la seÃ±al que se pasa como parÃ¡metro.
%       \details Informa: EnergÃ-a, valor medio, valor eficaz, varianza y desvio
%       \author Pose, Fernando Ezequiel (fernandoepose@gmail.com)
%   \param  Sn      - SeÃ±al a estudiar
%   \return Energia - EnergÃ-a de la seÃ±al
%   \return Medio   - Valor medio de la seÃ±al
%   \return Eficaz  - Valor eficaz de la seÃ±al
%   \return Var     - Varianza de la seÃ±al
%   \return Desv    - Desvio estÃ¡ndar de la seÃ±al
%       \date 2015.08.19

% Energia:
Energia = fenergia(Sn);

% Valor Medio:
Medio = fv_medio(Sn);

% Valor Eficaz:
Eficaz = feficaz(Sn);

% Varianza:
Var = fvar(Sn);

% Desvio estÃ¡ndar:
Desv = fdesvio(Sn);

end
```

Análisis con un nivel de cuantización de 8 bits

A continuación se simula el efecto de cuantizar una señal con un nivel de cuntización de 8 bit. Se realiza además un análisis sobre el ruido de cuantización para dicho nivel.

Función determinística



Análisis de la señal: energía, valor medio y valor eficaz de la señal determinística

```
SeÃ±al deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

Energía:

```
El valor de energia es: 500 Joule
```

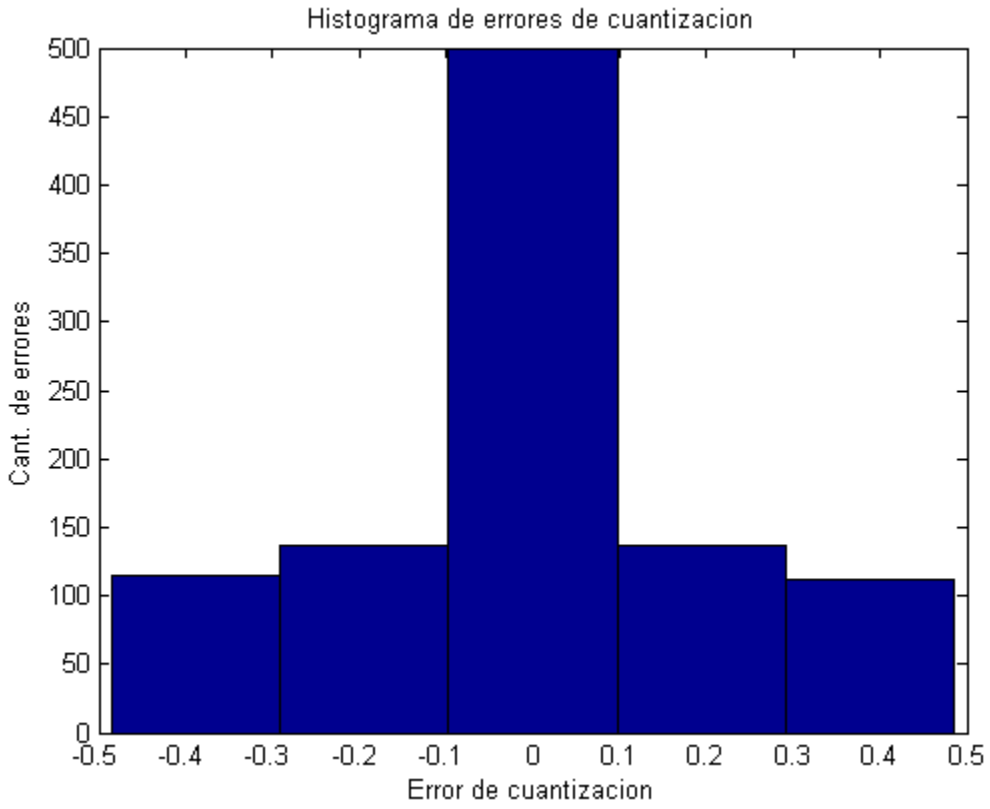
Valor Medio:

```
El valor medio es: 0.0074909 V
```

Valor Eficaz:

```
El valor eficaz es: 0.70711 V
```

Histograma



Análisis del histograma: Media, desvío estándar y varianza.

```
AnÃ¡lisis de Sn:
```

Energía:

```
El valor de energia es: 55.4648 Joule
```

Valor Medio:

```
El valor medio es: -0.0012867 V
```

Valor Eficaz:

```
El valor eficaz es: 0.23551 V
```

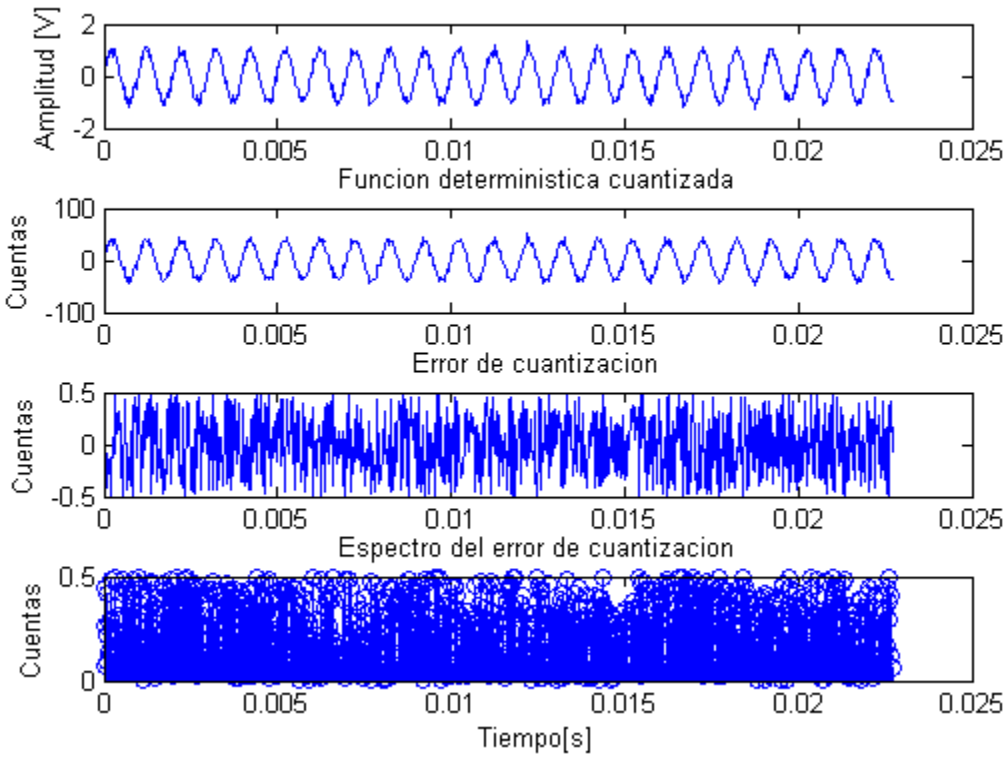
Varianza:

```
La varianza es: 55.4631
```

Desvío estándar:

```
El desvio es: 0.23551
```

Función no determinística



Análisis de la señal: energía, valor medio y valor eficaz de la señal no determinística

```
SeÃ±al no deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

Energía:

```
El valor de energia es: 508.3207 Joule
```

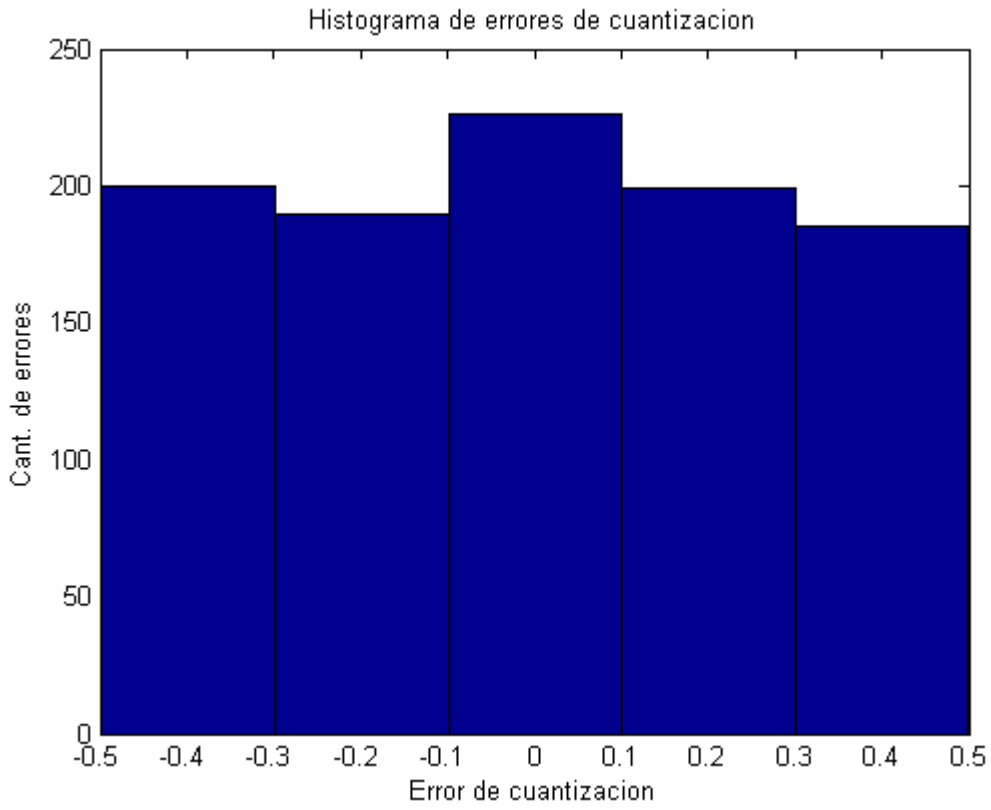
Valor Medio:

```
El valor medio es: 0.011181 V
```

Valor Eficaz:

```
El valor eficaz es: 0.71297 V
```

Histograma



Análisis del histograma: Media, desvío estándar y varianza.

```
AnÃ¡lisis de Sn:
```

Energía:

```
El valor de energia es: 80.0902 Joule
```

Valor Medio:

```
El valor medio es: -0.0032898 V
```

Valor Eficaz:

```
El valor eficaz es: 0.283 V
```

Varianza:

```
La varianza es: 80.0794
```

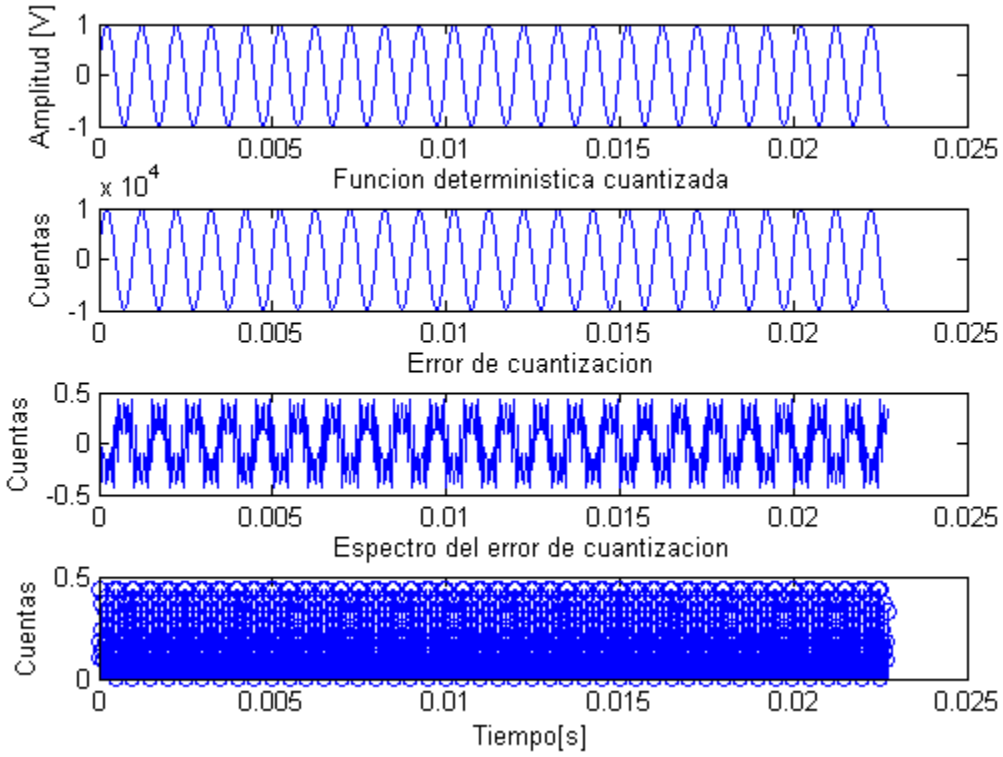
Desvío estándar:

```
El desvio es: 0.28298
```

Análisis con un nivel de cuantización de 16 bits

A continuación se simula el efecto de cuantizar una señal con un nivel de cuantización de 16 bit. Se realiza además un análisis sobre el ruido de cuantización para dicho nivel.

Función determinística



Análisis de la señal: energía, valor medio y valor eficaz de la señal determinística.

```
SeÃ±al deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

Energía:

```
El valor de energia es: 500 Joule
```

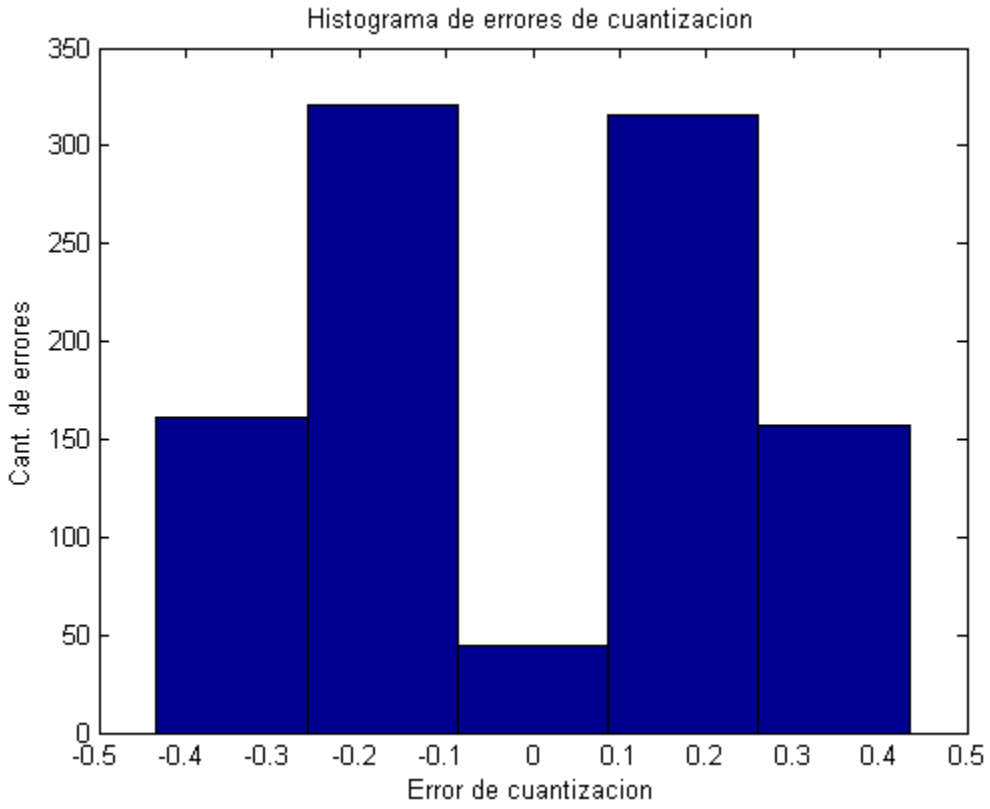
Valor Medio:

```
El valor medio es: 0.0074909 V
```

Valor Eficaz:

```
El valor eficaz es: 0.70711 V
```

Histograma



Análisis del histograma: Media, desvío estandár y varianza.

```
AnÃ¡lisis de Sn:
```

Energía:

```
El valor de energia es: 62.5011 Joule
```

Valor Medio:

```
El valor medio es: -0.0022309 V
```

Valor Eficaz:

```
El valor eficaz es: 0.25 V
```

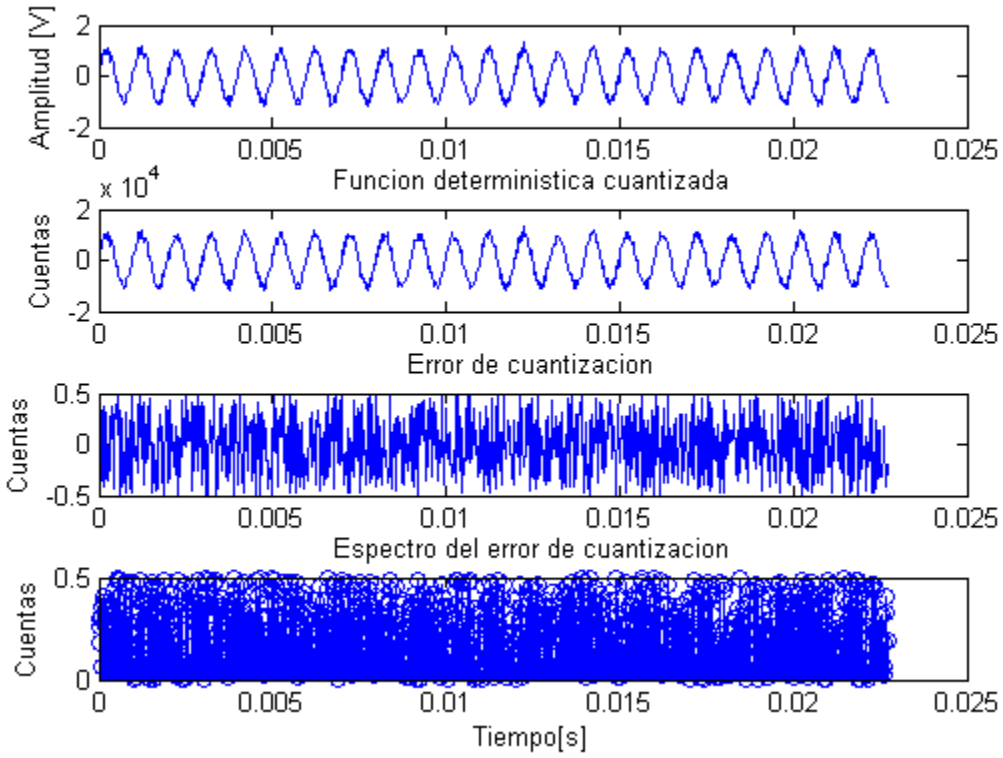
Varianza:

```
La varianza es: 62.4961
```

Desvío estándar:

```
El desvio es: 0.24999
```

Función no determinística



Análisis de la señal: energía, valor medio y valor eficaz de la señal no determinística.

```
SeÃ±al no deterministica:
```

Datos de: Energía, valor medio y valor eficaz.

Energía:

```
El valor de energia es: 510.2322 Joule
```

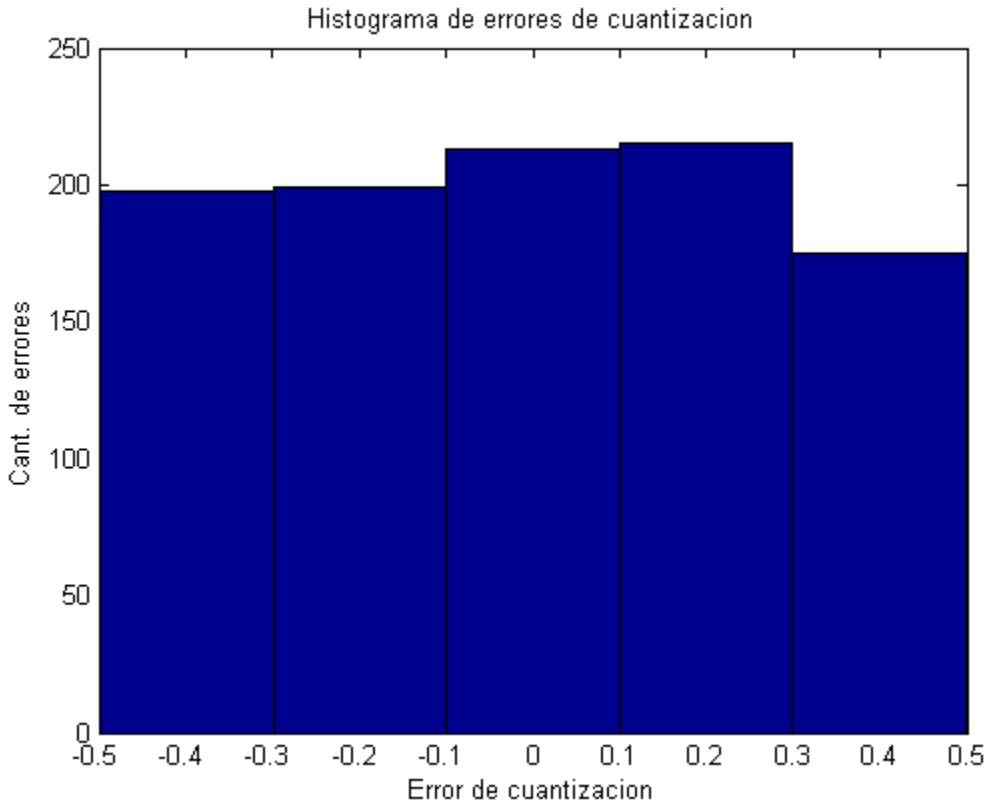
Valor Medio:

```
El valor medio es: 0.011318 V
```

Valor Eficaz:

```
El valor eficaz es: 0.71431 V
```

Histograma



Análisis del histograma: Media, desvío estándar y varianza.

```
AnÃ¡lisis de Sn:
```

Energía:

```
El valor de energia es: 78.4056 Joule
```

Valor Medio:

```
El valor medio es: -0.0038953 V
```

Valor Eficaz:

```
El valor eficaz es: 0.28001 V
```

Varianza:

```
La varianza es: 78.3904
```

Desvío estándar:

```
El desvio es: 0.27998
```

Conclusiones

De la resolución del ejercicio se pueden obtener las siguientes conclusiones:

1. El ruido de cuantificación depende de la resolución, a mayor cantidad de bits, menor LSB y por lo tanto menor piso de error de cuantización.
2. A medida que se cuantiza con un mayor número de bits la señal cuantizada cada vez se aproxima más a la señal a cuantizar.
3. La varianza de la cantidad de errores respecto a los errores de cuantización (ver histogramas) es igual a la energía de la señal error de cuantización (Sn)
4. Cuando la señal de entrada es muy determinística (primer caso) no se puede comprobar la hipótesis de uniformidad. Si es la señal determinística entonces el ruido se lo ve periódico.