UNIVERSIDAD SERGIO ARBOLEDA

Análisis de señales

Laboratorio 06: Análisis de Fourier en Matlab

Escuela de Ciencias exactas e Ingeniería

Código: SA2020IIG02_LAB06

Profesor: Marco Teran

Name: _____ **Deadline:** 1 de diciembre de 2020

Abstract

Durante el desarrollo del presente laboratorio se pondrán a prueba los conceptos y propiedades del análisis de Fourier utilizando el paquete de desarrollo matemático Matlab. Se entenderá de forma práctica los conceptos análisis espectral. Se realizará análisis espectral de imágenes y filtrado en Matlab.

1 Desarrollo de la práctica de laboratorio

1. (5 points) **Responda brevemente en la sección de Marco Teórico de su plantilla de laboratorio las siguientes preguntas:**
   (a) Definir la transformada de Fourier discreta y sus propiedades, la DFT, tanto para vectores como para matrices de imágenes (2D-DFT).
   (b) ¿Qué es el espectro de una imagen? ¿Físicamente y matemáticamente que representa?
   (c) Investigue como añadir ruido Gaussiano a imágenes, busque ejemplos en Internet y libros de procesamiento digital de imágenes.
   (d) ¿Qué la función de dispersión de puntos (point-spread function) de una imagen?
   (e) ¿Qué es filtro de Wiener y para que sirve? Clases de filtro de Wiener.

2. (5 points) **Transformada de Fourier en Matlab:**

   Cualquier señal puede ser descrita como una suma de sinusoides con diferentes amplitudes y frecuencias. El paquete matemático Matlab para calcular la Transformada de Fourier y su inversa utiliza los siguientes comandos fft a ifft respectivamente. Compile mediante un ejemplo y comente las lineas de código de la siguiente función:

```
function [XX] = dft(x)
N = length (x);
XX = zeros (N, 1);
for k = 0:N - 1
      wk = 2 * pi * k/N;
      for n = 0:N - 1
            XX(k + 1) = XX(k + 1) + x(n + 1) * exp ( - j * n * wk);
      end
end
```

   ¿En que se diferencia el código anterior con el siguiente?

```
function [XX] = dftv(x)
x = x(:);
N = length (x);
XX = zeros (N, 1);
n = [0:N - 1] ' ;
for k = 0:N - 1
```

```
wk = 2 * pi * k/N;
XX(k + 1) = sum (x. * exp ( - j * n * wk));
end
```

3. (5 points) Compile, anexe las gráficas, defina cual es la función del siguiente código y comente cada linea de este:

```
dt = 1/100;
et = 4;
t = 0:dt:et;
y = 3*sin(4*2*pi*t) + 5*sin(2*2*pi*t);
subplot(2,1,1);
plot(t,y);
axis([0 et -8 8]);
xlabel('Time (s)');
ylabel('Amplitude');
Y = fft(y);
n = size(y,2)/2;
amp_spec = abs(Y)/n;
subplot(2,1,2);
freq = (0:79)/(2*n*dt);
plot(freq,amp_spec(1:80));
xlabel('Frequency (Hz)');
ylabel('Amplitude');
noise = randn(1,size(y,2));
ey = y + noise;
eY = fft(ey);
n = size(ey,2)/2;
amp_spec = abs(eY)/n;
figure
subplot(2,1,1);
plot(t,ey); grid on
axis([0 et -8 8]);
xlabel('Time (s)');
ylabel('Amplitude');
subplot(2,1,2);
freq = (0:79)/(2*n*dt);
plot(freq,amp_spec(1:80));
xlabel('Frequency (Hz)');
ylabel('Amplitude');
figure
plot(Y/n,'r+')
hold on
plot(eY/n,'bx')
fY = fix(eY/100)*100;
ifY = ifft(fY);
cy = real(ifY);
figure
plot(t,cy); grid on
axis([0 et -8 8]);
xlabel('Time (s)');
ylabel('Amplitude');
```

Código 1 – Código MATALB de la FFT

4. (5 points) Escriba un código nen MATLAB con los comandos necesarios para encontrar la magnitud

del espectro de la siguiente señal. Verifique que el eje de las frecuencias esté escalado y correcto. Defina una frecuencia de muestreo adecuada para el ejercicio (justifiquela con lo visto en clase):

$$f(n) = \sum_{k=10}^{20} (20 - k)\sin(2\pi k n)$$

1.1 Técnicas de realce de contraste

Para realizar el realce de contraste de una imagen, generalmente se utilizan las técnicas: imadjust, histeq y adapthisteq. A continuación se realizará la mejora de una imagen en escala de grises.

- imadjust aumenta el contraste de la imagen mediante la asignación de nuevos valores de intensidad a la imagen de entrada. Por defecto se satura al 1% de los datos en las intensidades altas y bajas de los datos de entrada.
- histeq realiza una ecualización de histograma. Mejora el contraste de las imágenes mediante una transformación de los valores de intensidad de la imagen de entrada para que su nuevo histograma (la imagen de salida) coincida aproximadamente con un histograma de distribución especificado (una distribución uniforme por defecto).
- adapthisteq realiza una ecualización de histograma adaptativo con contraste limitado. A diferencia de histeq, esta opera en las regiones de datos pequeño (tiles) en lugar de toda la imagen. El contraste de cada *tile* se mejora para que el histograma de cada región de salida aproximadamente coincida con el histograma especificado (distribución uniforme por defecto). El realce del contraste puede ser limitado para evitar amplificar el ruido que puede estar presente en la imagen.

5. (5 points) Utilizando las configuraciones predeterminadas, comparar la efectividad de las tres técnicas siguientes. Leer una imagen en escala de grises en el espacio de trabajo. Mejorar la imagen utilizando las tres técnicas de ajuste de contraste.

```
pout = imread('pout.tif');
pout_imadjust = imadjust(pout);
pout_histeq = histeq(pout);
pout_adapthisteq = adapthisteq(pout);
```

Muestra la imagen original y las tres imágenes ajustadas en contraste utilizando subfiguras.

```
montage({pout,pout_imadjust,pout_histeq,pout_adapthisteq},'Size',[1 4])
title("Original Image and Enhanced Images using imadjust, histeq, and ...
    adapthisteq")
```

Realice el contraste de una imagen en escala de grises de su preferencia. Dibuje un mosaico de imágenes de los realces. Dibuje los histogramas de entrada y salidas obtenidos.

1.2 Diseño de filtros digitales

El objetivo del diseño de un filtro con ciertas características es el encontrar la respuesta al impulso de este $h[n]$. Los filtros con una respuesta al impulso de muestras se conocen como **FIR**. Mediante la implementación del comando fir1 se pueden obtener las muestras de la función característica de dicho impulso dependiendo del tipo de banda que desea rechazar. Implemente en Matlab: **help** fir1.

6. (2 points) Investigue:

1. ¿Que quiere decir que el comando fir1 permite obtener filtros de fase fase lineal?

2. ¿A que frecuencia respecto a la frecuencia de muestreo, f_s, se introducen las pulsaciones?
3. ¿Que es la ventana de Hamming? ¿Que otras funciones ventana puede utilizar Matlab (realice una tabla comparativa)?
4. ¿Para obtener una longitud de la respuesta al impulso de N muestras (orden del filtro) que parámetros de deben introducir en la función fir1?

Es posible realizar una comparación temporal de estas ventanas mediante la ejecución de las siguientes lineas de código:

```
N=75;
h_rect = boxcar(N);
h_hann = hanning(N);
h_hamm = hamming(N);
h_blac = blackman(N);
n=1:75;
plot(n,h_rect,'b',n,h_hann,'g',n,h_hamm,'r',n,h_blac,'m');
legend('Rectangular','Hanning','Hamming','Blackman');
axis([0 80 0 1.2])
```

Para realizar una comparación frecuencial ejecute continuamente:

```
clf
[H_rect,w] = freqz(h_rect);
[H_hann,w] = freqz(h_hann);
[H_hamm,w] = freqz(h_hamm);
[H_blac,w] = freqz(h_blac);
subplot(4,1,1)
semilogy(w,abs(H_rect))
axis([0 1 10e-5 10e2]); grid on
subplot(4,1,2)
semilogy(w,abs(H_hann));
axis([0 1 10e-5 10e2]); grid on
subplot(4,1,3)
semilogy(w,abs(H_hamm));
axis([0 1 10e-5 10e2]); grid on
subplot(4,1,4)
semilogy(w,abs(H_blac));
axis([0 1 10e-5 10e2]); grid on
```

7. (3 points) Realice comentarios a los resultados obtenidos tanto en el dominio temporal como en el dominio de la frecuencia.

1.2.1 Diseño de filtros digitales con fir1

8. (10 points) Encuentre los coeficientes de la función característica (respuesta al impulso) $h[n]$ de un filtro digital FIR con las siguientes especificaciones:

- Banda de paso: $150 - 250\ Hz$
- Ancho de banda de transición $f_{tr} = 50\ Hz$
- Características de atenuación: $A_p \leqslant 0.1\ dB,\ A_s \geqslant 60\ dB$
- Implemente ventana tipo *Kaiser* con una frecuencia de muestreo $f_s = 1\ kHz$.

Represente gráficamente el filtro diseñado. Visualice el diagrama de Bode (magnitud y fase de la respuesta en frecuencia del filtro diseñado) mediante freqz(h,1). Las frecuencias se deben normalizar a $\frac{f_s}{2}$.

Diseño de filtro paso-bajo, LPF Encuentre los coeficientes de la función característica (respuesta al impulso) $h[n]$ de un filtro digital FIR con las siguientes especificaciones:

- Frecuencia de corte de la banda de paso: 1.5 kHz
- Ancho de banda de transición $f_{tr} = 0.5\ kHz$
- Atenuación en la banda atenuada: $> 50\ dB$
- Frecuencia de muestreo $f_s = 8\ kHz$.

Represente gráficamente el filtro diseñado. Visualice el diagrama de Bode (magnitud y fase de la respuesta en frecuencia del filtro diseñado) mediante freqz(h,1). Las frecuencias se deben normalizar a $\frac{f_s}{2}$.

Para obtener la representación gráfica de $h[n]$ se recomienda implementar el código:

```
n=-(N-1)/2:(N-1)/2;
stem(n,h); grid on
```

1.2.2 Enfoque de imágenes usando el filtro Wiener

9. (10 points) La deconvolución de Wiener (filtrado), deconvwnr, puede ser útil cuando son conocidos o estimados la función de dispersión de puntos (point-spread function) y el nivel de ruido. Leemos la imagen,

```
I = im2double(imread('cameraman.tif'));
imshow(I);
title('Imagen original (cortesia de MIT)');
```

Simule una imagen borrosa que se podría obtener con el movimiento de la cámara. Cree una función de dispersión de puntos, PSF, correspondiente al movimiento lineal en 21 píxeles (LEN=21), en un ángulo de 11 grados (THETA=11). Para simular el desenfoque, convolucione el filtro con la imagen usando imfilter.

```
LEN = 21;
THETA = 11;
PSF = fspecial('motion', LEN, THETA);
blurred = imfilter(I, PSF, 'conv', 'circular');
imshow(blurred);
title('Imagen borrosa');
```

Para restaurar la imagen borrosa, utilice el siguiente algoritmo. La sintaxis más simple para deconvwnr es deconvwnr(A, PSF, NSR), donde A es la imagen borrosa, PSF es la función de dispersión de puntos y NSR es la relación ruido-potencia-señal-potencia. La imagen borrosa formada en el paso anterior no tiene ruido, así que usaremos 0 para NSR.

```
wnr1 = deconvwnr(blurred, PSF, 0);
imshow(wnr1);
title('Restored Image');
```

Realice el procedimiento anterior, pero para una imagen a la que se agregó ruido,

```
noise_mean = 0;
noise_var = 0.0001;
blurred_noisy = imnoise(blurred, 'gaussian', ...
noise_mean, noise_var);
imshow(blurred_noisy)
title('Simulate Blur and Noise')
```