Instituto Tecnológico Autónomo de México

Laboratorio de Redes Convergentes

**Práctica 1: DTMF y Potencia media**

Equipo: La Comunidad del Anillo

Integrantes

Francisco A. Calvillo López 167677

Lorena P. Barrera Rodríguez 164694

18 de agosto, 2020

***Laboratorio de Redes Convergentes***

# Práctica 1

## DTMF y Potencia media

***1. Objetivos***

*1.1. Generar funciones periódicas y graficarlas.*

*1.2. Capturar, reproducir y graficar señales de voz.*

*1.3. Calcular la Potencia media de una señal.*

***2. Trabajo Previo (ENTREGADO con anterioridad)***

*2.1. Investigue el concepto de DTMF (Dual Tone Multiple Frecuency) en telefonía.*

*2.2. Investigue el concepto de la Transformada Rápida de Fourier (FFT).*

*2.3. Investigue el concepto de Potencia media de una señal.*

*2.4. Investigue el producto matricial entre un vector y su transpuesto, alternando operandos.*

***3. Materiales y equipo***

*3.1. Una computadora con tarjeta de sonido, micrófono y bocinas,*

*3.2. Matlab.*

***4. Desarrollo***

Detalles generales. En las siguientes tres secciones (4.1, 4.2 y 4.3) deberá desarrollar, respectivamente, las tres siguientes funciones P1s4p1R1, P1s4p2R2 y P1s4p3R3, que cubren el total de lo requerido en cada sección. Dentro de cada una de estas funciones, al programarlas, deberá invocar a otras funciones, a) algunas propias de MATLAB, b) otras que ya vienen programadas en el Apéndice A de esta práctica, y c) las que usted deberá programar. Las funciones que usted programa deben estar estructuradas.

Para graficar las señales, además del vector que contiene las magnitudes de las amplitudes muestreadas (eje y) se requiere un vector que contenga los valores de tiempo correspondientes a las amplitudes muestreadas (eje x).

Para graficar el espectro de frecuencias de una señal, además del vector que contiene las magnitudes de las frecuencias (eje y) se requiere un vector que contenga los valores de frecuencias correspondientes a dichas magnitudes (eje x).

Como paso inicial, estudie cada una de las funciones del Apéndice A y dése cuenta cuando usar cada una. Después copie cada función hacia MATLAB, guardando cada función en un archivo (.m) respectivo que lleve el nombre de la función tal cual. Cuidado con la copia dado que este archivo es de Word y no de texto.

***4.1. Generación y trazado de señales periódicas.***

Antes de contestar nada, lea completa toda esta sección 4.1.

Objetivo. Realizar una función procDTMF(*Btn*), en Matlab, dentro de P1s4p1R1,que reciba un parámetro entero *Btn*, entre 0 y 15 (ver tabla más abajo) y dependiendo del valor de *Btn*, la función obtendrá las frecuencias baja y alta correspondiente, construyéndose dos señales senoidales con dichas frecuencias. También, se generará una nueva señal senoidal a partir de la suma de las dos señales senoidales anteriores. A cada señal (de las tres anteriores) periódica se le calculará su correspondiente Transformada Rápida de Fourier (FFT), debiendo graficarse todas ellas (las tres periódicas y sus transformadas). Además, de cada señal senoidal (tres) se deberá generar una reproducción sonora.

Observación. Los siguientes pasos, del 1 al 8, indican cómo desarrollar procDTMF(*Btn*). Algunas funciones de las que va a utilizar ya vienen programadas en el Apéndice A.

Paso 1. Obtenga las dos frecuencias correspondientes a *Btn* –baja *FsBaja* y alta *FsAlta* -. Para lograrlo, haga uso de la función [*FsBaja*, *FsAlta*]=funDTMF(*Btn*),que ya viene programada en el Apéndice A. Contempla el caso en que *Btn* está fuera de los límites 0-15, marcando el error con una impresión y terminando la operación (con *return*).

Tabla de frecuencias del teclado DTMF. Frecuencias bajas y altas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Btn*** | F5=1209 Hz | F6=1336 Hz | F7=1477 Hz | F8=1633 Hz |
| f1=697 Hz | 1 (**1**) | 2 (**2**) | 3 (**3**) | 12 (**A**) |
| f2=770 Hz | 4 (**4**) | 5 (**5**) | 6 (**6**) | 13 (**B**) |
| f3=852 Hz | 7 (**7**) | 8 (**8**) | 9 (**9**) | 14 (**C**) |
| f4=941 Hz | 10 (**\***) | 0 (**0**) | 11 (**#**) | 15 (**D**) |

Paso 2. Invoque a la función [*FsBaja*, *FsAlta*]=funDTMF(*Btn*), desde procDTMF(*Btn*), probándola con algún valor de *Btn*. Imprima, ambas frecuencias, con títulos adecuados, usando fprintf(‘...’, *FsBaja*, *FsAlta*).

Paso 3. Genere un vector *vxtm* horizontal de valores de tiempo correspondientes a intervalos de muestreo de 1/*Fm* para *Fm*=8000 Hz, durante 3 segundos (*Ttm*). Para ello emplee la función *vxtm*=geVcxTiem(*Ttm*,*Fm*) , que viene en el Apéndice A.

¿Cuántos valores se obtuvieron en *vxtm*? ¡Muestre como se hizo el cálculo!

Una vez ejecutada la función, nos dimos cuenta que entrega 24,000 valores debido a que se multiplica el tiempo (3 segundos) y la frecuencia de muestreo (8,000 Hz):

vxtm = Ttm \* Fm

Paso 4. Construya dos vectores de senoidales, con las dos frecuencias *Fs*, alta y baja, con la función: *fsmm*=funSe(*A*, *Fs*, *D*, *vxtm*), del Apéndice A, usando el mismo vector de tiempo *vxtm*. Los valores de la senoidal son: de amplitud *A*=1 y desfasamiento *D*=0.

Paso 5. Obtenga un nuevo vector mixto (señal mixta) *fsmmAB* como la suma de los dos vectores senoidales *fsmm*, calculados anteriormente con las dos frecuencias de *Btn*.

Paso 6. Calcule, para cada uno de los vectores senoidales *fsmm* (dos) y mixto *fsmmAB*, su respectivo vector magnitud de frecuencias *magDFT*, mediante Fourier discreto: *magDFT*=absFFT(*fsmm*), del Apéndice A.

Paso 7. Además, genere, un vector *vxfa* horizontal de valores de frecuencias, correspondiente al vector *magDFT*, con la función *vxfa*=geVcxFreFun(*magDFT*, *Fm*), del Apéndice A.

Paso 8. Grafique todos los vectores senoidales *fsmm* y mixto *fsmmAB* (contra *vxtm*) así como sus correspondiente vectores de magnitudes de frecuencias *magDFT* (contra *vxfa*). Utilice las funciones de Matlab: figure(), subplot(), plot() o stem(), title(), xlabel(), ylabel(). Observación: para el caso de las gráficas, con plot(), de los senoidales y mixto, sólo despliegue la 400ava. parte de los puntos; en el caso de las magnitudes de frecuencias, con stem(), sólo despliegue la mitad de los puntos.

¿Por qué en este último caso de graficación de frecuencias, sólo se deben incluir la mitad de los puntos?

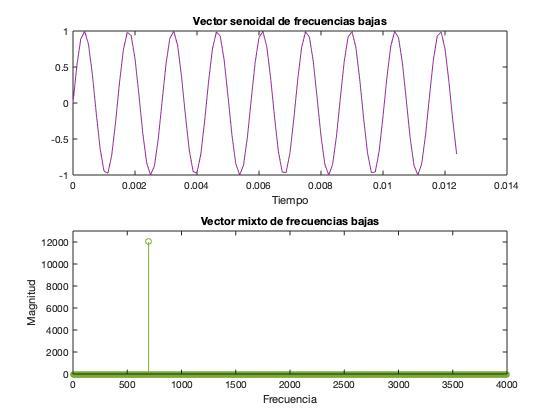
Las gráficas de magnitudes de frecuencias por medio de una DFT, al igual que la función, son *simétricas*, es por eso que puede mostrarse sólo la mitad de la gráfica.

Paso 9. Por cada vector senoidal (dos) y mixto, realice una reproducción sonora con la función de Matlab sound(*fsmm*, *Fm*). Agregue después del sonido una pausa con la función de Matlab pause(*4*).

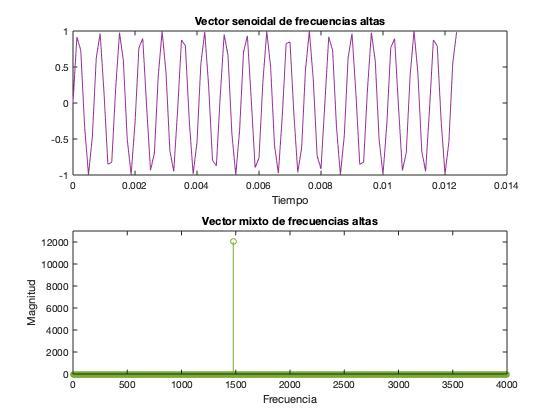
Paso 10. Desarrolle la función principal P1s4p1R1(), que contendrá las invocaciones procDTMF(*Btn*), a tres botones distintos, a saber: 3, 10 y 15.

Reporte 1. Reporte los programas Matlab realizados en sus respectivos archivos, liste los resultados obtenidos en la ejecución de P1s4p1R1(), junto con las gráficas obtenidas y las respuestas a las preguntas en este mismo documento. Adjunte estos archivos, de Matlab, al subir su documento, en un archivo zip.

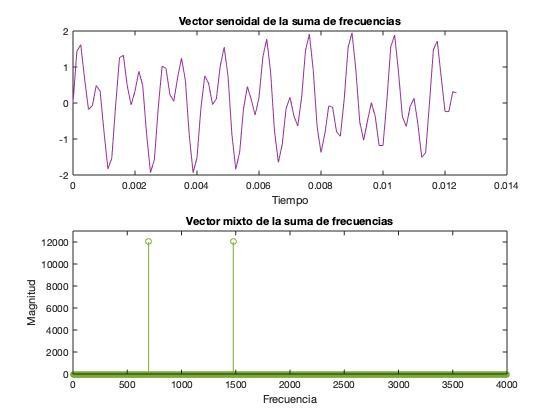
Para el **primer botón (3)** que se probó se obtuvieron sus frecuencias importantes. La frecuencia *más baja* es de 697 Hz y la *más alta* es de 1477 Hz. Los vectores de la frecuencia baja se ven:



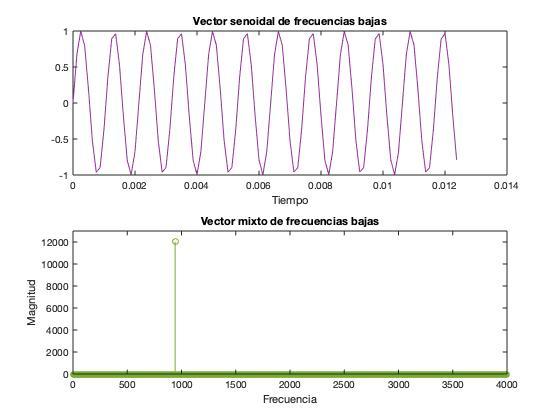
Los vectores de la frecuencia alta se ven así:



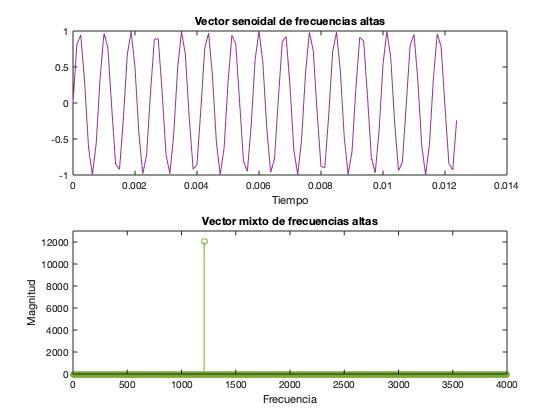
Por último, los vectores de la suma de frecuencias se ve así:



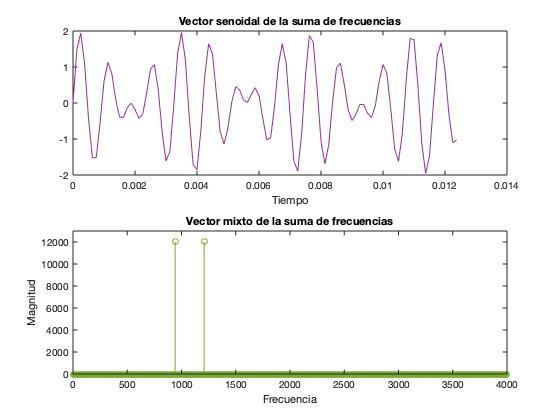
Por otro lado, el **segundo botón (10)** tiene una frecuencia *baja* de 941 Hz y una frecuencia *alta* de 1209 Hz. Los vectores de la frecuencia baja se ven así:



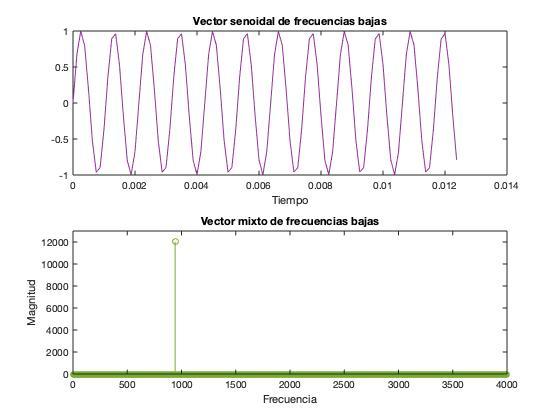
Los vectores de la frecuencia alta se ven así:



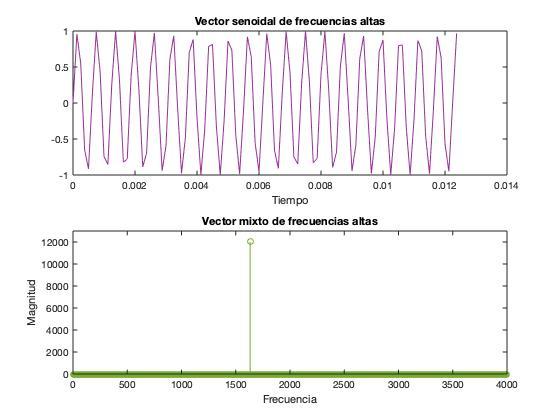
Por último, los vectores de la suma de frecuencias se ve así:



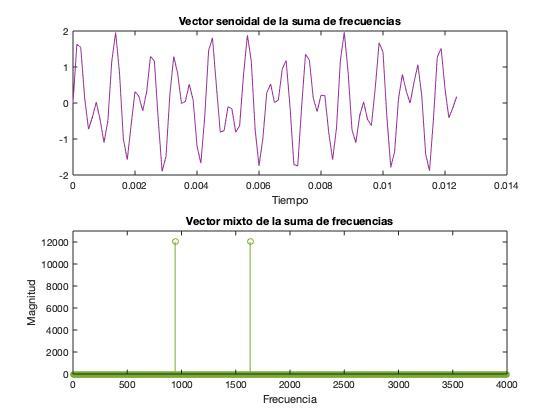
Finalmente, para el **tercer botón (15)** la frecuencia *baja* es de 941 y la frecuencia *alta* es de 1633 Hz. Los vectores de la frecuencia baja:



Los vectores de la frecuencia alta se ve así:



Por último, los vectores de la suma de frecuencias se ve así:



Para la función P1s4p1R1() se realizó la función procDTMF(Btn) usando las funciones proporcionadas en el anexo:

Código de Matlab:

function [] = procDTMF(Btn)

tiempo = 3;

frec = 8000;

[fBaja, fAlta] = funDTMF(Btn);

fprintf("la frecuencia baja es: " + fBaja + "\n");

fprintf("la frecuencia alta es: " + fAlta + "\n");

% Vector vxtm

vector = geVcxTiem(tiempo,frec);

% Vectores fsmm

vBaja = funSe(1,fBaja,0,vector);

vAlta = funSe(1,fAlta,0,vector);

% Vector fsmmAB

vSuma = vBaja + vAlta;

% Vectores magDFT

magBaja = absFFT(vBaja);

magAlta = absFFT(vAlta);

magSuma = absFFT(vSuma);

% Vectores vxfa

vecCoBaja = geVcxFreFun(magBaja, frec);

vecCoAlta = geVcxFreFun(magAlta, frec);

vecCoSuma = geVcxFreFun(magSuma, frec);

vRec = vector(1:100);

vBajaRec = vBaja(1:100);

vAltaRec = vAlta(1:100);

vSumaRec = vSuma(1:100);

figure

subplot(2,1,1)

plot(vRec, vBajaRec, 'color', [0.6, 0.2, 0.6])

title("Vector senoidal de frecuencias bajas")

xlabel("Tiempo")

subplot(2,1,2)

stem(vecCoBaja, magBaja, 'color', [0.4660, 0.6740, 0.1880])

title("Vector mixto de frecuencias bajas")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,13000])

figure

subplot(2,1,1)

plot(vRec, vAltaRec, 'color', [0.6, 0.2, 0.6])

title("Vector senoidal de frecuencias altas")

xlabel("Tiempo")

subplot(2,1,2)

stem(vecCoAlta, magAlta, 'color', [0.4660, 0.6740, 0.1880]);

title("Vector mixto de frecuencias altas")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,13000])

figure

subplot(2,1,1)

plot(vRec, vSumaRec, 'color', [0.6, 0.2, 0.6])

title("Vector senoidal de la suma de frecuencias")

xlabel("Tiempo")

subplot(2,1,2)

stem(vecCoSuma, magSuma, 'color', [0.4660, 0.6740, 0.1880]);

title("Vector mixto de la suma de frecuencias")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,13000])

sound(vBaja, frec);

pause(4);

sound(vAlta, frec);

pause(4);

sound(vSuma, frec);

end

***4.2. Captura y reproducción de la señal de voz.***

Objetivo. Realizar una función P1s4p2R2() en Matlab, que lleve a cabo un análisis similar al del inciso 4.1, pero ahora, con tres señales de audio (voz) grabadas previamente en archivos. Estas señales de audio serían las equivalentes a las señales mixtas (vectores mixtos), del inciso 4.1. Las grabaciones de voz y el guardado en archivo (.*wav*), se llevarán a cabo con el programa OcenAudio; también pueden llevar a cabo la grabación con cualquier otra aplicación similar.

Paso previo al uso de Matlab. Utilice el programa de captura OcenAudio para capturar tres señales de audio de voz de formato WAV (tres archivos *.wav*). Al abrir una nueva grabación se le deberá indicar que será monoaural, la frecuencia de muestreo (*Fm*) a 8000 Hz y duración (*Ttm*) de 3 s. Al salvar la grabación en un archivo de audio (.*wav*) se indicará la característica de 16 bits por muestra.

Paso 1. Ulilice la función de MatLab [*funVoz*, *Fm*]=audioread(*‘archivo.wav’*) para leer de cada archivo la señal grabada, cargándola en un vector *funVoz* de Matlab. La función audioread(), siempre genera un vector vertical.

Paso 2. Genere, para cada *funVoz* un vector *vxtm* de valores de tiempo correspondientes a intervalos de muestreo de 1/Fm con el *Fm* obtenido con audioread(). Para ello emplee la función *vxtm*=geVcxTiemFun(*funVoz*, *Fm*), que viene en el Apéndice A.

Paso 3. Calcule, para cada uno de los vectores *funVoz* (tres), su respectivo vector *magDFT* con las magnitudes del contenido de frecuencias, con la función: *magDFT*=absFFT(*funVoz*), del Apéndice A.

Paso 4. Además, genere, un vector *vxfa* de valores de frecuencias, correspondiente al vector *magDFT*, con la función *vxfa*=geVcxFreFun(*magDFT*,*Fm*), del Apéndice A.

Paso 5. Grafique todos los vectores (tres) *funVoz* (contra *vxtm*) así como sus correspondiente vectores de magnitudes de frecuencias *magDFT* (contra *vxfa*). Utilice las funciones de Matlab: figure(), subplot(), plot() o stem(), title(), xlabel(), ylabel().

Observación: para el caso de las gráficas, con plot(), de los vectores *funVoz*, sólo despliegue la 10ma. parte de los puntos; en el caso de las magnitudes de frecuencias, con stem(), sólo despliegue la mitad de los puntos.

En las tres gráficas de magnitudes de frecuencias vs frecuencias, en que rangos de frecuencias se despliega la mayor potencia de las frecuencias:

Aproximadamente, la mayor potencia en todos los audios es de los 0 Hz a los 700 Hz.

Paso 6. Reproduzca cada señal capturada (tres) utilizando la función sound(*funVoz*,*Fm*), escuchando la señal obtenida. Use después una pausa.

Paso 7. Al final, dentro de P1s4p2R2() mande ejecutar el comando whos de Matlab. Este comando muestra las características de todas las variables creadas durante la ejecución de P1s4p2R2().

De las variables vectores *funVoz*, *magDFT*, *vxtm*, y *vxfa*, diga cuales son vectores verticales y cuáles son vectores horizontales:

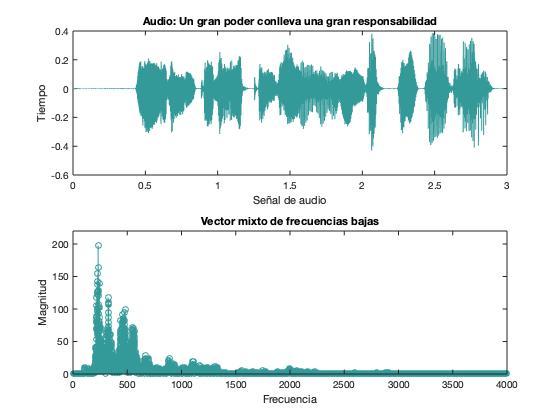
Los tres vectores de *funVoz* y *magDFT*, respectivamente, son vectores **verticales** de 23424, 23406 y 23824 elementos. Por otro lado, los tres vectores de *vxtm* y *vxfa*, respectivamente, son vectores **horizontales** de 23424, 23406 y 23824 elementos.

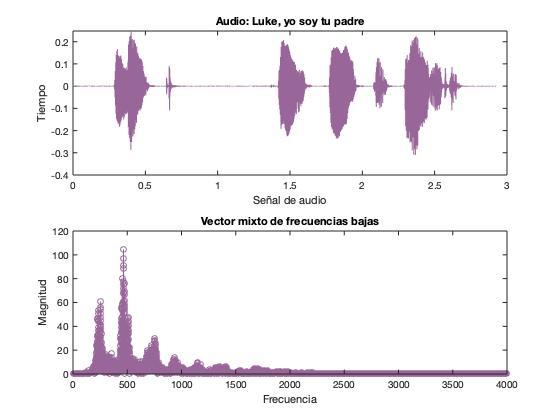
¿Por qué algunos son vectores verticales y por qué otros son vectores horizontales?

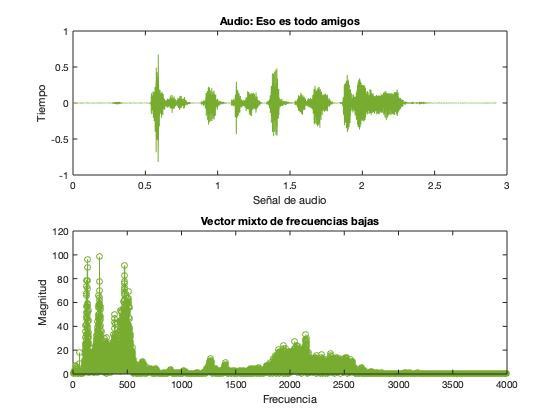
Estos cambios de “orientación” de los vectores se debe a las operaciones matriciales que se realizan con ellos.

Reporte 2. Reporte los programas Matlab realizados en sus respectivos archivos, liste los resultados obtenidos en la ejecución de P1s4p2R2(), junto con las gráficas obtenidas y las respuestas a las preguntas.

Se realizaron tres grabaciones para probar esta función y sus gráficas son las siguientes:







La información de todas las variables es:

**Name Size Bytes Class**

audio1 23424x1 187392 double

audio2 23406x1 187248 double

audio3 23824x1 190592 double

frec1 1x1 8 double

frec2 1x1 8 double

frec3 1x1 8 double

mag1 23424x1 187392 double

mag2 23406x1 187248 double

mag3 23824x1 190592 double

v1 1x23424 187392 double

v2 1x23406 187248 double

v3 1x23824 190592 double

vFrec1 1x23424 187392 double

vFrec2 1x23406 187248 double

vFrec3 1x23824 190592 double

Código de Matlab:

function [] = P1s4p2R2()

% Matriz [funVoz, Fm]

[audio1, frec1] = audioread("porky.wav");

[audio2, frec2] = audioread("luke.wav");

[audio3, frec3] = audioread("spidey.wav");

% Vectores vxtm

v1 = geVcxTiemFun(audio1,frec1);

v2 = geVcxTiemFun(audio2,frec2);

v3 = geVcxTiemFun(audio3,frec3);

% Vectores magDFT

mag1 = absFFT(audio1);

mag2 = absFFT(audio2);

mag3 = absFFT(audio3);

% Vectores vxfa

vFrec1 = geVcxFreFun(mag1, frec1);

vFrec2 = geVcxFreFun(mag2, frec2);

vFrec3 = geVcxFreFun(mag3, frec3);

figure

subplot(2,1,1)

plot(v1, audio1, 'color', [0.4660, 0.6740, 0.1880])

title("Audio: Eso es todo amigos")

xlabel("Señal de audio")

ylabel("Tiempo")

subplot(2,1,2)

stem(vFrec1, mag1, 'color', [0.4660, 0.6740, 0.1880])

title("Vector mixto de frecuencias bajas")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,120])

figure

subplot(2,1,1)

plot(v2, audio2, 'color', [0.6, 0.4, 0.6])

title("Audio: Luke, yo soy tu padre")

xlabel("Señal de audio")

ylabel("Tiempo")

subplot(2,1,2)

stem(vFrec2, mag2, 'color', [0.6, 0.4, 0.6])

title("Vector mixto de frecuencias bajas")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,120])

figure

subplot(2,1,1)

plot(v3, audio3, 'color', [0.2, 0.6, 0.6])

title("Audio: Un gran poder conlleva una gran responsabilidad")

xlabel("Señal de audio")

ylabel("Tiempo")

subplot(2,1,2)

stem(vFrec3, mag3, 'color', [0.2, 0.6, 0.6])

title("Vector mixto de frecuencias bajas")

xlabel("Frecuencia")

ylabel("Magnitud")

axis ([0, 4000, 0,220])

sound(audio1);

pause(3);

sound(audio2);

pause(3);

sound(audio3);

pause(3);

whos;

end

***4.3. Cálculo de la Potencia Media de una señal.***

Objetivo. Realizar una función P1s4p3R3() en Matlab, que lleve a cabo el cálculo de la potencia media de las tres señales de audio generadas y analizadas en 4.2.

Paso 1. Utilice la función de MatLab [*funVoz*, *Fm*]=audioread(*‘archivo.wav’*) para leer de cada archivo la señal grabada, cargándola en el vector *funVoz*.

Paso 2. Verifique las características de sus variables utilizando el comando whos.

Paso 3. Realice la función *pm*=funPoMe(*funVoz*) para calcular la potencia media de una señal *funVoz* de acuerdo con la fórmula:

Potencia Media = ∑ |xi |2 / N, siendo xi el i-ésimo elemento de *funVoz*.

donde N es el número total de muestras de *funVoz*. Recuerde que puede utilizar el producto matricial entre un vector y su transpuesto.

Dentro de esta función, deberá determinar con [*filas*, *columnas*]=size(*funVoz*) si *funVoz* es un vector horizontal o vertical, para así conocer el orden de los operandos del producto matricial.

El vector resultante de los audios es vertical.

Paso 4. Invoque la función *pm*=funPoMe(*funVoz*) para cada uno de los vectores de voz *funVoz*.

Paso 5. Imprima la potencia media de las tres señales de voz *funVoz* capturadas.

Reporte 3. Reporte los programas Matlab realizados en sus respectivos archivos, liste los resultados obtenidos en la ejecución de P1s4p3R3(), indicando las unidades de la potencia obtenida, considerando que las señales de prueba están en milivolts.

En esta función se obtiene la potencia media para cada audio. Para el **primer** audio es de 0.0037345 Watts, para el **segundo** audio es de 0.0020802 Watts y para el **tercero** es de 0.0065463 Watts.

Código de Matlab:

function [] = P1s4p3R3()

% Matriz [funVoz, Fm]

[audio1, frec1] = audioread("porky.wav");

[audio2, frec2] = audioread("luke.wav");

[audio3, frec3] = audioread("spidey.wav");

whos;

potMedia1 = funPoMe(audio1);

potMedia2 = funPoMe(audio2);

potMedia3 = funPoMe(audio3);

fprintf("La potencia media del primer audio es: " + potMedia1 + "\n");

fprintf("La potencia media del segundo audio es: " + potMedia2 + "\n");

fprintf("La potencia media del tercer audio es: " + potMedia3 + "\n");

end

***5. Trabajo complementario***

*5.1. Investigue algunas aplicaciones de la codificación DTMF en telefonía (Dual Tone Múltiple*

*Frecuency)*

Los tonos DTMF (multifrecuencia de doble tono) son utilizados en los teléfonos para la marcación por tonos, que suenan cuando se presionan las teclas de número. Esta opción es útil para introducir una contraseña o el número de una cuenta cuando se llama a un sistema automatizado, como un servicio bancario.

De igual forma, se usa ampliamente en el campo de la comunicación tradicional y también desempeña un papel importante en VoIP. La transmisión de comunicación en la red IP adopta la conmutación de paquetes en lugar de la conmutación de circuitos tradicional y las características de la red IP son inestables. Como resultado, a diferencia de la solución DTMF en los campos tradicionales, las soluciones DTMF en VoIP son diversas.

Las soluciones comunes son las siguientes:

* Utilice el método INFO de señalización SIP para transportar señales DTMF.
* Lleve las señales DTMF durante la transmisión de medios RTP

***6. Referencias***

[1] Samsung (mayo 12,2018), ¿Qué es DTMF? [Online], Disponible: <https://www.samsung.com/latin/support/mobile-devices/what-is-dtmf/>

[2] L. Jiménez (mayo 30,2019), Aplicación de DTMF en VoIP [Online], Disponible: [https://forum.huawei.com/enterprise/es/aplicación-de-dtmf-en-voip/thread/532147-100257](https://forum.huawei.com/enterprise/es/aplicaci%C3%B3n-de-dtmf-en-voip/thread/532147-100257)

Apéndice A

----------------------------------------------------------------------

function [FBaja,FAlta]=**funDTMF**(Btn)

% Esta funcion entrega las dos frecuencias correspondientes al boton Btn.

% Btn: Boton del 0 al 15 segun la tabla de DTMF.

% FBaja: Frecuencia Baja

% FAlta: Frecuencia Alta

if Btn<0 | Btn>15

fprintf('ERROR(funDTMF): boton inexistente entre [0-15]= %7.0f\n', Btn);

FBaja=-1; FAlta=-1;

return

end

%Frecuencias bajas del DTMF (Hz)

f1=697; f2=770; f3=852; f4=941;

fb=[f4,f1,f1,f1,f2,f2,f2,f3,f3,f3,f4,f4,f1,f2,f3,f4];

%Frecuencias altas del DTMF (Hz)

f5=1209; f6=1336; f7=1477; f8=1633;

fa=[f6,f5,f6,f7,f5,f6,f7,f5,f6,f7,f5,f7,f8,f8,f8,f8];

FBaja=fb(Btn+1);

FAlta=fa(Btn+1);

fprintf('Combinacion de frecuencias= %12.0f\n', Btn);

fprintf('Frecuencia baja= %15.2f Hz\n', FBaja);

fprintf('Frecuencia alta= %15.2f Hz\n', FAlta);

fprintf('Saliendo de funDTMF\n\n');

return;

----------------------------------------------------------------------

function magDFT=**absFFT**(fsmm)

% absFFT: Calcula el vector magnitud de la DFT a partir de

% fsmm : Vector función magnitud de señal

% magDFT: Vector magnitud de la DFT (Discrete Fourier T.)

funDFT = fft(fsmm); % Calculo de la DFT

magDFT = abs(funDFT); % Magnitud de la DFT

fprintf('Saliendo de absFFT\n\n');

return;

----------------------------------------------------------------------

function fsmm=**funSe**(A,F,D,vxtm)

% funSe: Genera un vector funcion senoidal a partir de

% A : Amplitud maxima (v)

% F : Frecuencia de la onda (Hz)

% D : Defasamiento (grados-radianes)

% vxtm : Vector con los X tiempos de muestreo (seg). Escala.

% fsmm : Vector funcion senoidal modulada y muestreada (vector)

fsmm=A\*sin(2\*pi\*F\*vxtm);

fprintf('Saliendo de funSe\n\n');

return;

----------------------------------------------------------------------

function vxfa=**geVcxFreFun**(magDFT,Fm)

% geVcxFreFun: Genera un vector de coordenadas X a partir de

% magDFT: Vector magnitud de la DFT

% Fm: Frecuencia de muestreo (Hz)

% vxfa: vector de coordenadas X correspondientes a las armónicas. Escala.

Tm=1/Fm; % Periodo del muestreo base (seg)

N=length(magDFT); % Total de puntos de frecuencias

vp=0:N-1; % Vector de puntos

FAmin=1/(N\*Tm); % Armonica mas pequeña

vxfa=vp\*FAmin; % Coordenadas de las Armonicas

fprintf('Saliendo de geVcxFreFun\n\n');

return;

----------------------------------------------------------------------

function vxtm=**geVcxTiem**(Ttm,Fm)

%geVcxTiem: Generacion de un vector de coordenadas X de tiempo a partir de

% Ttm : Tiempo total de muestreo base (seg)

% Fm : Frecuencia de muestreo base (Hz)

% vxtm: Vector coordenadas X con los tiempos de muestreo. Escala.

Tm=1/Fm; % Periodo del muestreo base (seg)

N=fix(Ttm/Tm); % Total de puntos a muestrear

vpm=0:N-1; % Vector con los puntos de muestreo

vxtm=vpm\*Tm;

fprintf('Saliendo de geVcxTiem\n\n');

return;

----------------------------------------------------------------------

function vxtm=**geVcxTiemFun**(funVoz,Fm)

%geVcxTiemFun: Generacion de un vector de coordenadas X de tiempo a partir de

% funVoz: Vector señal de voz muestreada (vector)

% Fm: Frecuencia de muestreo base (Hz)

% vxtm: Vector con los momentos de tiempo de muestreo.Escala.

Tm=1/Fm; % Periodo del muestreo base (seg)

N=length(funVoz); % Total de puntos a muestrear

vp=0:N-1; % Vector con los puntos de muestreo

vxtm=vp\*Tm;

fprintf('Saliendo de geVcxTiemFun\n\n');

return;

----------------------------------------------------------------------