|  |  |
| --- | --- |
|  | **Universidad Tecnológica Nacional**  **Facultad Regional Buenos Aires**  **Ingeniería en Sistemas de Información** |

**Matemática Superior**

Profesor: *María Inés Grand*

Ayudante: *Jonathan Castro*

Trabajo Práctico: *FILTROS*

Grupo: *Número de Grupo*

Curso: *Número de curso K*

Cuatrimestre: 2C / 2016

|  |  |  |
| --- | --- | --- |
| NOMBRE Y APELLIDO | LEGAJO N° | EMAIL CONTACTO |
| Pablo Hervida | 1473943 | elpabli09@gmail.com |
| Tomás Rodríguez Saavedra | 1472483 | tomas\_rs@hotmail.com |
|  |  |  |
|  |  |  |

Fecha de entrega: 18/11/2016

Calificación:

Observaciones:

***b. INTRODUCCIÓN:***

En este trabajo práctico se procedió a resolver un problema específico mediante herramientas de software matemático como el Matlab, utilizando conceptos aprendidos en la unidad de “Transformada Z” de esta asignatura, así como también conocimientos generales de programación de asignaturas anteriores de la carrera.

/\*Cambiar esto que viene abajo en la introducción por lo que realmente hicimos\*/

El objetivo era el de procesar una señal, en este caso una onda, de alguna forma para preservar las características deseadas y eliminar las no deseas (frecuencias altas por ejemplo). Durante el desarrollo se utilizó un archivo de sonido el cual está conformado por un tipo de onda específica la cual tiene características como por ejemplo: duración, altura, timbre e intensidad. Luego de realizar la aplicación del filtro y el procesamiento debido de la señal se obtuvo como resultado una señal compuesta de armónicos distintivos.  
***c. CONTENIDO:***

***i. Funciones y comandos***

**1. *Impz:*** devuelve la respuesta al impulso del filtro. Está definida para distintos parámetros:

- [h, t]=Impz(num, den, n, Fs) donde se especifica el número de muestras a graficar y la frecuencia de muestreo utilizada. Y se guarda los vectores de magnitud y eje de tiempo en h y t respectivamente.

- [h,t]=Impz(num, den) donde por defecto se considera Fs=1 Hz y grafica las diez primeras muestras.

**2. Conv:** devuelve la convolución de los vectores “u” y “v”, donde si “u” y “v” son vectores de coeficientes de polinomios, usar conv es equivalente a multiplicar los dos polinomios. La forma es:

- Conv(u,v) donde el resultado es un vector de longitud = longitud(u)+longitud(v)-1

Ejemplo: Sean u(x) = 2x + 3 y v(x) = x + 2

>>u = [2 3] u = 2 3

>>v = [1 2] v = 1 2

>>p = conv(u, v)

p = 2 7 6 2

Devuelve el polinomio p(x) = 2x2 + 7x + 6 y la longitud del vector resultado es 3 = 2 + 2 -1.

**3. Filter:** filtra los datos de entrada “x”, que puede ser un vector, matriz o un array multidimensional, usando una función de transferencia racional definida por los coeficientes de numerador y denominador “b” y “a”. El coeficiente “a” no puede ser cero. La forma es:

- y = filter(b,a,x)

**4. Soundsc / Sound:**

Sound: convierte la matriz de datos de la señal en sonido. Envía la señal de audio “y” al altavoz con la frecuencia de muestreo predeterminada de 8192Hertz. La forma es:

- sound(y)

- sound(y, Fs): lo mismo que sound(y) pero eligiendo la frecuencia de muestreo (Fs).

- sound(y, Fs, nBits): version extendida de sound(y, Fs) donde usa la cantidad de bits “nbits” por muestra.

Soundsc: Escala los datos y los reproduce como sonido. Ecala los valores de la señal de audio “y” para encajarlos en el rango de -1,0 a 1,0 y luego envía los datos al altavoz con la frecuencia predeterminada de 8192Hertz. Tiene la sintaxis parecida al sound:

- soundsc(y)

- soundsc(y, Fs)

- soundsc(y, Fs, nBits)

**5. Wavread / AudioRead:**

AudioRead: lee el archivo de audio pasado por parámetro, y devuelve la muestra de datos “y” una muestra de frecuencia “Fs” para esos datos. Tiene la forma:

- [y, Fs]=audioread(archivo)

Wavread: lee el archivo de audio de extension .wav. Es la versión obsoleta de audioread (en las últimas versiones de Matlab). Soporta datos en multicanal de hasta 32 bits por muestra. La sintaxis es muy parecida al audioread:

- y=wavread(archivo) donde los valores de la amplitud están en el rango de [-1,1]

**6. Plot:** Crea un plano en 2D con la información pasada en los parámetros de dominio “x” e imagen o función “y”. Si ambos parámetros son vectores entonces tienen que tener igual longitud, lo mismo si son matrices. Tiene la forma:

- plot(valoresDeDominio, valoresDeImagenOFuncionDeX)

**7. Syms:** Crea funciones y variables simbólicas. Tiene la forma:

- syms var1…varN donde crea variables simbólicas de 1 a N separadas por espacios.

- syms f(var1,…,varN) donde crea una función simbólica “f” y variables simbólicas “var1,…,varN” que representan los argumentos de entrada de “f”.

En caso de que no se le envíe ningún parámetro a “syms” entonces enumerará todas las variables simbólicas, funciones, vectores y matrices existentes del espacio de trabajo de Matlab.

***ii. Proceso de resolución:***  
Se deberá entregar un archivo .m que contenga el **código.**

***iii. ¿Qué diferencia encuentra entre CONV y FILTER en Matlab?***

La diferencia entre CONV y FILTER es que FILTER puede manejar filtros FIR y IIR, en cambio CONV toma dos parámetros y devuelve su convolución. Además, FILTER puede devolver los estados del filtro, de modo que pueda ser utilizado en posteriores llamadas. Un ejemplo en el que se parecen porque devuelven el mismo resultado es en: *conv(h,x)* y *filter(h,1,x)*, donde el parámetro “1” en filter indica que los coeficientes recursivos del filtro son solo [1]. Entonces por ejemplo, si se tiene un filtro IIR no se podrá usar *conv*.

***d. CONCLUSIÓN:***

***e. PROBLEMAS Y LECCIONES APRENDIDAS:***