

Laboratorio de Tratamiento Digital de Señales

Curso 2020-2021. Sesión 1

1. El Laboratorio

La figura de la siguiente página muestra un esquema del laboratorio docente del Dpto. SSR. Como se puede observar todos los puestos de trabajo están conectados en una red local atendida por un servidor.

1.1 Acceso al puesto de trabajo (para los alumnos presenciales)

La identificación de usuario y la contraseña para los diferentes grupos de la asignatura LTDS son:

Identificación	Contraseña
TDS-G32	TDS-G32

donde TDS-G32 es el correspondiente turno de prácticas.

1.2 Acceso al servidor

Todos los puestos de trabajo tienen acceso al servidor del laboratorio, dispositivo Z:\, que se utiliza para el intercambio de información Profesor-Alumnos. La carpeta Z:\LTDS es la de la asignatura LTDS. La subcarpeta Z:\LTDS\COMUN contiene la información relevante a los tres grupos de clase. Las carpetas de los profesores de los diferentes grupos son:

Z:\TDS\JUAN SANTOS

Z:\TDS\JOSÉ PARERA

Z:\TDS\GUSTAVO CUEVAS

Z:\TDS\MIGUEL GARCIA

y se utilizan para el intercambio de información de cada profesor con sus alumnos. Los alumnos tienen derecho a leer documentos de estas carpetas, pero no a escribir.

En las carpetas de cada grupo se encuentra la carpeta ...ENTREGAS, con derecho a escribir pero no a leer por parte de los alumnos. Se utilizará esta carpeta para la entrega de trabajos personales al profesor correspondiente.

1.3 Carpeta de la asignatura

Los alumnos disponen de una carpeta personal en su puesto de trabajo, que será fijo y se asignará el primer día de asistencia al laboratorio:

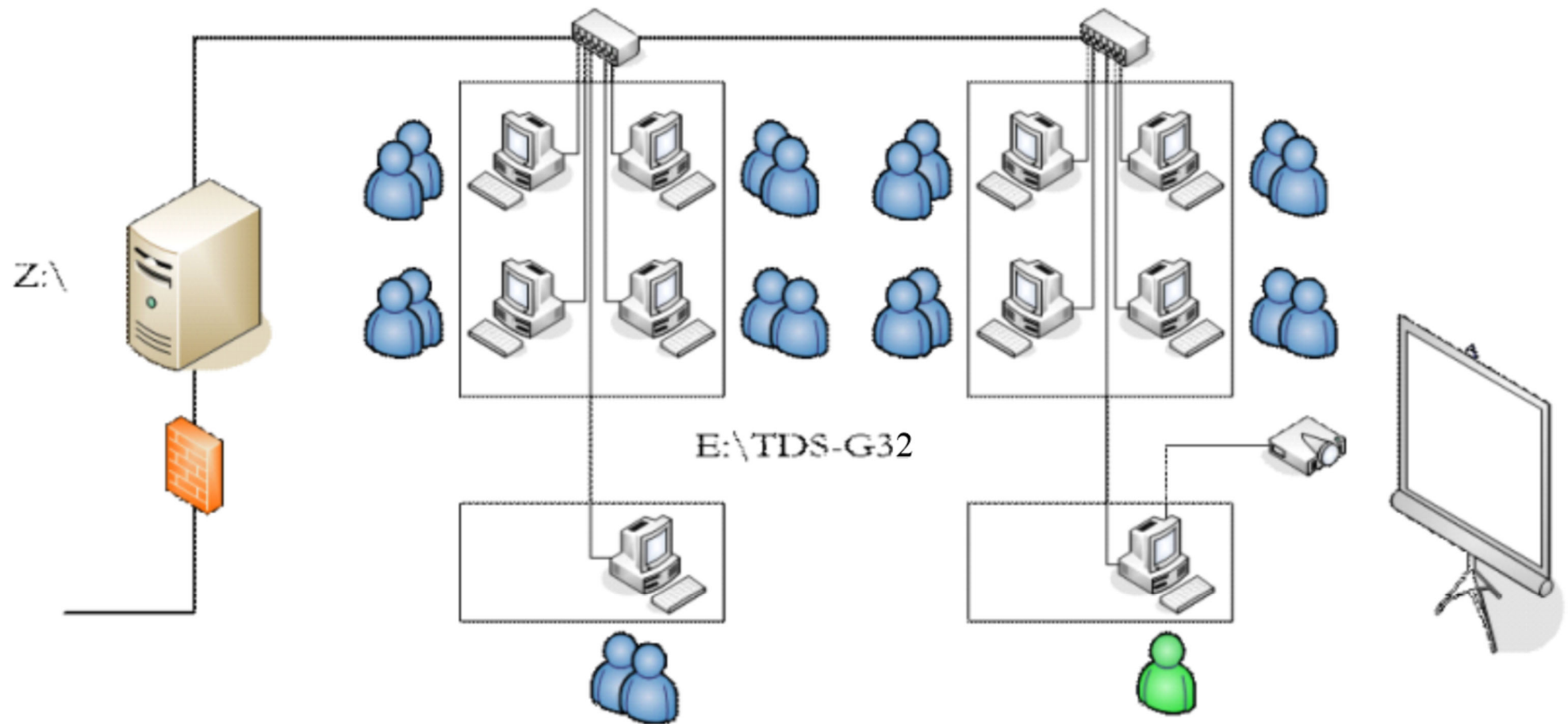
E:\TDS-G32

Se recomienda trabajar solamente en esta carpeta y además responsabilizarse de su contenido manteniendo los respaldos adecuados.

1.4 Entregas de los alumnos no presenciales.

Los alumnos que pertenezcan al grupo en-línea harán la entrega inmediatamente finalizada la práctica, mediante el entorno MOODLE.

Estructura del Laboratorio



2. El entorno de computación y visualización de datos MATLAB


MATLAB es el entorno de computación y visualización de datos que se ha convertido en el “estándar de facto” en universidades y empresas tecnológicas de todo el mundo. Sus elementos principales son:

- Una interfaz gráfica multi-ventana amigable
- Un amplio conjunto de funciones matemáticas que constituyen una impresionante calculadora científica
- Un lenguaje de programación
- Un editor/depurador adaptado a la sintaxis del lenguaje
- Un conjunto de “tool boxes” especializados (procesado de señales, imágenes, control, etc. etc.)

MATLAB ha sufrido el típico proceso de “gigantismo” del software, pero un uso básico de esta herramienta se puede aprender con facilidad. El objetivo último de las prácticas de la asignatura TDSÑ es que los alumnos aprendan procesamiento de señales básico con MATLAB, y para solventar la enorme extensión de MATLAB y su falta de conocimiento por parte del alumno, se irán introduciendo los elementos de la herramienta paulatinamente, en la medida que sean necesarios para resolver los ejercicios que se propongan.

Una vez arrancado el programa MATLAB, versión R201xy, se recomienda:

- Verificar que el directorio de trabajo es **E:\TDS-G32**. Se encuentra en **Current Folder** en la parte superior-central de la interfaz. En caso contrario se puede restituir de la forma

habitual en Windows con el icono  También se recomienda **no crear subcarpetas en la carpeta E:\TDS-G32** para evitar confusiones.

- Disponer las ventanas de la interfaz según la configuración por defecto mediante
HOME → Layout → Default

Maximizar la interfaz con el típico icono de las aplicaciones de Windows que se encuentra en la esquina superior derecha.

3. Manejo de señales con MATLAB

Una señal discreta de duración finita L (el tipo de señales que se pueden procesar mediante una máquina de cálculo) se define como:

$$x[n], n = N_1, N_1 + 1, \dots, N_2$$

Y por tanto se verifica que $x[n] = 0, n < N_1$ y $x[n] = 0, n > N_2$, mientras que la duración es

$$L = N_2 - N_1 + 1.$$

En el entorno de cálculo MATLAB una señal se corresponde con un vector \mathbf{x} , en general de números complejos en punto flotante y doble precisión (64 bits equivalentes a unos 15 dígitos decimales de precisión), y un vector \mathbf{n} de índices temporales ("eje temporal").

Hay que evitar confundir el índice temporal \mathbf{n} con los índices de los vectores de MATLAB. Estos últimos son siempre positivos en el rango 1 a longitud del vector.

4. Resolución de ejercicios de procesamiento de señales con MATLAB

La forma más directa de trabajar con MATLAB es introducir comandos en la ventana **Command Window** de forma interactiva. Por ejemplo:

```
>> cos(pi/3)
ans =
    0.5000
>>
```

El doble ángulo `>>` es el "prompt" de MATLAB. Se ha tecleado `cos(pi/3)` y el programa responde con `ans = 0.5000`

Este procedimiento es adecuado para ensayar ideas encaminadas a resolver un problema. Pero cuando tenemos un prototipo de solución se recomienda usar el editor. Para ello "picamos" en el icono de Nuevo Script (HOME → Layout → Default) que se encuentra en la esquina superior izquierda del interfaz, o bien damos CTRL*N. De inmediato se abre una página en blanco donde se pueden insertar los comandos que resuelven nuestro problema.

El editor es sensible a la sintaxis de MATLAB por lo que presta una ayuda inestimable para evitar los errores más simples, pero numerosos, de la programación.

Conviene anclar la ventana de edición a la interfaz mediante el correspondiente icono que se encuentra en la esquina superior derecha (flecha en la dirección esquina inferior derecha).

Las líneas que comienzan con el carácter `%` aparecerán en verde y se corresponden con comentarios (no ejecutables). Cada sesión de laboratorio constará de varios ejercicios y para la edición de su solución se propone usar las "cell" del editor. Una "cell" es un conjunto de líneas de código que comienzan con una línea de comentarios con dos caracteres `%` consecutivos. Esta forma de trabajo nos permitirá ejecutar cada ejercicio, ubicado en una celda, por separado, empleando los iconos de ejecución del editor adecuados.

5. Ayudas en línea de MATLAB

El comando `help item` de MATLAB proporciona información limitada acerca del uso de los comandos del lenguaje. Así, si queremos información sobre el uso de la función `cos(. . .)` teclearemos

```
help cos
```

El repositorio más extenso de información “online” sobre el uso de MATLAB es el **Help Browser**, al que se puede acceder mediante el icono **?** que se encuentra a la derecha del indicador del **Current Folder** (parte superior central de la interfaz).

También se puede acceder a la información extendida de un comando contenida en el **Help Browser** mediante el comando `doc ítem`, por ejemplo

```
doc cos
```

Ejercicios de la sesión 1 del laboratorio de TDSÑ

1. Cálculo y representación gráfica de las señales discretas más usuales

1.1 Calcule la señal impulso unidad

$$\delta[n], n = -10, -9, -8, \dots, 19, 20$$

y dibújela según el libro de texto de la asignatura.

Solución:

Empezamos creando el vector de índices ("eje temporal")

```
n=-10:20;
```

En este contexto el operador ":" crea un vector de enteros correlativos empezando por -10 terminando en 20. El ";" que termina el comando suprime la impresión en el terminal del resultado. Por tanto se recomienda terminar con ";" todos los comandos que calculan una cantidad, excepto que se quiera un eco en la pantalla del resultado.

A continuación creamos el vector señal impulso unidad

```
d=double(n==0);
```

Se ha utilizado el operador lógico "==", por lo que `n==0` produce un vector de datos lógicos (True/False), todos False (=0) menos para `n=0` en que el resultado es True (=1). El operador "double" convierte el vector de datos lógicos en un vector de datos reales en doble precisión. Se recomienda que como parte de un buen estilo de programación se conserven los convenios de notación típicos del procesado de señales (minúsculas para las señales, mayúsculas para las transformadas, *x* para la señal de entrada, *h* para la respuesta impulsiva, etc., etc.).

Para dibujar el impulso unidad creado emplearemos los siguientes comandos:

```
stem(n,d,'k','MarkerfaceColor','k'), title('Impulso unidad \delta[n]')
xlabel('n')
axis([-10 20 -0.1 1.1])
```

El comando `stem(n,d)` dibuja la señal según el estilo del libro de texto. El resto de los parámetros fijan el color de los círculos de la figura y el relleno de estos a negro (código k). El etiquetado de una figura es imprescindible y para ello MATLAB dispone de un amplio repertorio de comandos. `title('título de la figura')` escribe el título de la figura en la parte superior central del gráfico. En este contexto el carácter ascii "\" es un carácter de escape al intérprete LaTeX. Para profundizar en el manejo de símbolos matemáticos y letras griegas se recomienda buscar en el **Help Browser** de MATLAB la expresión "mathematical symbols". `xlabel('n')` rotula el eje horizontal, mientras que `axis(. . .)` establece los límites de la gráfica de tal modo que acomode el margen dinámico tanto de la señal como del eje temporal.

Con los comandos comentados hasta aquí hemos creado el eje temporal, el impulso unidad y una gráfica que consideramos satisfactoria. A continuación procederíamos a invocar el editor e introducir los comandos, quedando de esta forma (tómese a modo de ejemplo):

```
% Sesión 1 del laboratorio de TDSÑ

%% Apartado 1.1

% Eje "temporal" n
n=-10:20;

% Impulso unidad
d=double(n==0);

% Gráfica
stem(n,d,'k','MarkerfaceColor','k')
title('Impulso unidad \delta[n]')
xlabel('n')
axis([-10.5 20.5 -0.1 1.1])
```

A continuación procederíamos a guardar el contenido de la ventana de edición con el icono típico de Windows que se encuentra en la parte superior izquierda de la ventana de edición. En la ventana que se abre indicaríamos, por ejemplo, `sesion1`.

Los iconos para la ejecución de una celda se encuentran debajo del icono de apertura de un documento en blanco. Para la ejecución de todo el fichero se dispone del icono “triángulo verde” en la parte superior central de la pestaña de edición.

Si ancla la ventana gráfica a la interfaz del mismo modo como se ha hecho con la ventana del editor tendrá una interfaz más compacta y manejable. Así, el editor y los gráficos comparten la misma área de la pantalla y se puede seleccionar uno u otro “picando” en la correspondiente pestaña.

1.2 Repita el apartado 1.1 pero para la señal escalón unidad $u[n]$.

El único comando diferente es el que calcula la señal escalón, que pasa a ser

```
u=double(n>=0);
```

1.3 Repita el apartado 1.1 pero para la señal coseno

$$x[n] = \cos\left(\frac{2\pi}{11}n\right), n = -10, -9, \dots, 19, 20$$

El único comando diferente es el que calcula la señal coseno, que pasa a ser

```
x=cos(2*pi/11*n)
```

MATLAB evalúa el argumento del coseno de izquierda a derecha y respetando las precedencias de los operadores. Primero multiplica $2 \cdot \pi$, luego divide por 11 y por último multiplica el resultado por cada uno de los componentes del vector n . Hay que hacer notar que MATLAB es

una calculadora vectorial/matricial, así que calcula directamente el coseno de cada uno de los componentes del vector argumento $2\pi/11*n$.

También el comando para fijar el título cambia

```
title('cos(\omega_0n)')
```

El carácter `_` indica subíndice.

1.5 Repita el apartado anterior pero usando `plot` y estableciendo una rejilla en la gráfica.

Los comandos gráficos son ahora

```
plot(n,x,'k','MarkerfaceColor','k')
title('cos(\omega_0n)')
grid
```

Como se puede observar en la gráfica, MATLAB establece una interpolación lineal entre valores consecutivos de la señal. Este tipo de gráfico es más adecuado cuando la señal que se representa es de gran duración (por ejemplo, más de 100 puntos). La rejilla establecida con el comando `grid` permite leer los valores de la señal con más comodidad.

1.6 Calcule y dibuje (con `plot`) la señal coseno amortiguado

$$x[n] = e^{-\alpha n} \cos\left(\frac{2\pi}{11}n\right), n = 0, 1, \dots, 29 \quad (\alpha = 0.1)$$

En este caso la señal se calcula con el comando

```
x=cos(2*pi/11*n).*exp(-0.1*n);
```

Tanto el coseno como la exponencial son vectores de dimensión 30. Para multiplicar dos vectores punto-a-punto se emplea el operador `.*` (en inglés point-wise operation). Las operaciones punto-a-punto aplican a los operadores aritméticos `*`, `/` y `^` (exponenciación). El título de la gráfica se fija con el comando

```
title('e^{\-alphan}cos[\omega_0n]')
```

Nótese el empleo del carácter `^` para establecer superíndices y los caracteres `{}` para determinar el conjunto de caracteres a los que se les aplica la propiedad de superíndice (en este caso, $-\alpha n$).

1.6 Calcule y dibuje una señal aleatoria con distribución uniforme entre -1 y 2 y de duración $L=100$.

El comando para calcular la señal es

```
x=3*rand(L,1)-1;
```

1.7 Calcule y dibuje una señal aleatoria con distribución normal, con media 1 y desviación típica 2, y de duración $L=100$.

El comando para calcular la señal es

```
x=2*randn(L,1)+1;
```

Establezca los márgenes de la gráfica (comando `axis`) considerando que la señal no sobrepasa los valores $\pm 3\sigma$ en torno al valor medio.

2. Entrada/salida de datos en el entorno MATLAB

2.1 Cálculo de la energía cinética de un cuerpo de masa m kg que se desplaza con velocidad uniforme v m/s empleando el teclado y pantalla como dispositivos de entrada/salida.

Solución:

```
%% Apartado 2.1
% Usando el terminal
m=input('Indique la masa del objeto: ');
v=input('Indique la velocidad del objeto: ');
disp(['La energía cinética del objeto vale ',num2str(1/2*m*v^2),' julios'])
fprintf('La energia cinética del objeto vale %8.3f julios\n',1/2*m*v^2)
```

El comando `input` permite leer un dato desde el terminal y asignarle el valor a una variable. El texto en el argumento se le presenta al usuario para guiar la entrada. El comando `disp` presenta en la pantalla, con el adecuado formato, el resultado. El argumento del comando es una cadena de caracteres que en este caso se obtiene mediante la concatenación, con ayuda de los caracteres `[]`, de un texto definido entre los caracteres `"` y del resultado del comando `num2str` que convierte un valor numérico en una representación ascii apropiada.

Una forma alternativa de presentar los resultados es empleando el comando `fprintf`, similar al correspondiente comando del lenguaje C. Aquí hemos definido un formato de 8 caracteres ascii, con tres decimales. Para más información debe consultar la ayuda del comando `fprintf`.

2.2 Cargue la señal (forma de onda de una vocal /a/ muestreada con $F_s = 8000\text{Hz}$) contenida en el fichero `a.mat`, dibújela, calcule su energía instantánea y almacénela en un fichero que llamará `a2.mat`. Ejemplo de empleo del sistema de archivos como medio de entrada/salida.

Solución:

```
load a
```

```
n=length(a)-1;
plot(n,a), title('Forma de onda de /a/'), xlabel('n'), grid
a2=a.^2;
save a2 a2
```

Los ficheros *.mat almacenan datos en formato nativo MATLAB. Los comandos `load` y `save` leen y escriben en el sistema de archivos (en el **Current Folder** si no se especifica una ruta alternativa). Consultar la documentación en línea acerca de estos dos comandos. Obsérvese el uso de varios comandos MATLAB en la misma línea, pero deben estar separados por `,` o por `;`. Note también que el comando `length` calcula la longitud de un vector.

2.3 La forma de onda representada en el apartado anterior presenta unas oscilaciones de frecuencia aproximada al primer formante (o primera resonancia) de la vocal /a/. Mida el tiempo entre máximos de la forma de onda con el comando `ginput`, calcule el inverso de este tiempo y presente el valor del primer formante por pantalla.

Solución:

```
[Ind Amp]=ginput(2);
Fs=8000; Ts=1/Fs;
F1=1/((Ind(2)-Ind(1))*Ts);
fprintf('Primera resonancia aproximada %5.1f Hz\n',F1)
```

El comando `ginput` presenta una mira (cross-hair) sobre la gráfica. Pique con la mira sobre los dos primeros máximos y pulse la tecla ENTER después del segundo “click”.

Se obtienen las coordenadas x (Ind) e y (Amp) en dos vectores de longitud 2. Analice cómo se calcula el primer formante.

3. Ejercicios para resolver íntegramente por parte de los alumnos

3.1 Calcule y dibuje la señal

$$x[n] = \begin{cases} \frac{1}{\pi n} \sin\left(\frac{2\pi}{7}n\right), & -20 \leq n \leq 20, n \neq 0 \\ \frac{2}{7}, & n = 0 \end{cases}$$

Ayuda: Calcule $x[n]$ sin considerar el caso $n = 0$. A continuación, la siguiente línea de código MATLAB resuelve el caso $n = 0$:

```
x(n==0)=2/7;
```

3.2 Un sistema LTI causal y estable tiene una respuesta impulsiva

$$h[n] = 0.85^n u[n]$$

Calcule y dibuje $h[n]$ en el intervalo $-5 \leq n \leq 25$.

Sugerencia: calcule 0.85^n y $u[n]$ en el intervalo temporal especificado y a continuación multiplique punto-a-punto ambas señales para obtener $h[n]$.

3.3 Considere la señal de duración $L = 20$:

$$x[n] = \begin{cases} 0, & 0 \leq n \leq 4 \\ 1, & 5 \leq n \leq 10 \\ 0, & 11 \leq n \leq 19 \end{cases}$$

y la señal de duración $N = 30$:

$$h[n] = 0.85^n, \quad 0 \leq n \leq 29$$

Calcule, con el comando `conv(. . .)`, la convolución lineal $y[n] = x[n] * h[n]$, y dibújela con el comando `stem(. . .)`.

Recordatorio: la duración de la secuencia $y[n]$ es $L + N - 1$.

3.4 El tren de deltas discreto de período 4

$$p[n] = \sum_{k=-\infty}^{k=\infty} \delta[n - 4k]$$

tiene un desarrollo en serie de Fourier discreta

$$p[n] = \frac{1}{4} \sum_{k=0}^{k=3} e^{j\frac{\pi}{2}kn}$$

Se pide calcular la serie de Fourier en el período $0 \leq n \leq 3$ y dibujar el resultado con el comando `stem(. . .)`.