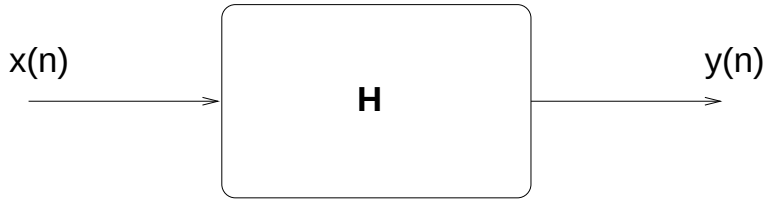
PRÁCTICA 5:

SISTEMAS DISCRETOS: RESPUESTA TEMPORAL.

1. Introducción: Ecuación en Diferencias

Un sistema discreto se define como aquel que responde con una señal discreta ante una excitación discreta. Se puede esquematizar de la siguiente manera:



donde la salida $y(n)$ es el resultado de aplicar el operador $\mathbf{H}$ sobre $x(n)$.

La forma más general de expresar la respuesta de un sistema discreto es el *producto de convolución*:

$$y(n) = \sum_{l=0}^{\infty} h(l)x(n-l) = h(n) * x(n) = x(n) * h(n) \tag{1}$$

donde $h(n)$ representa la respuesta impulsiva del sistema discreto y $x(n)$ la señal de entrada.

La expresión del producto de convolución anterior no es computacionalmente operativa, dado que la sumatoria se extiende desde $l = 0$ hasta $l = \infty$. Por ello, el cálculo de la respuesta del sistema suele hacerse a través de la *ecuación en diferencias*:

$$y(n) = -\frac{1}{a_0} \sum_{k=1}^{N} a_k y(n-k) + \sum_{l=0}^{M} b_l x(n-l) \tag{2}$$

donde se puede fijar $a_0 = 1$. A partir de esta expresión cabe definir dos tiposde sistemas discretos:

1. Sistemas de Respuesta Impulsiva Infinita (IIR): los vectores de parámetros $\mathbf{a}$ y $\mathbf{b}$ son no nulos, por lo que la respuesta impulsiva del sistema puede tener una duración infinita.
2. Sistema de Respuesta Impulsiva Finita (FIR): el vector $\mathbf{a}$ es nulo, por lo que se tiene que la respuesta impulsiva es $h(l) = b_l$, es decir, tiene una duración finita.

2. Filtros IIR

Se utilizará como ejemplo un sistema IIR con la siguiente ecuación en diferencias:

$$y(n) + 0{,}9y(n-2) = 0{,}3x(n) + 0{,}6x(n-1) + 0{,}3x(n-2) \tag{3}$$

1. Vamos a obtener las 30 primeras muestras de las respuesta impulsiva del sistema. La operación de filtrado se implementa en forma vectorial:

   $$y(n) = -\mathbf{a}\mathbf{y_n}^t + \mathbf{b}\mathbf{x_n}^t \tag{4}$$

   donde $\mathbf{y_n}$ y $\mathbf{x_n}$ son vectores que contienen la muestra actual en la primera dimensión ($\mathbf{y_n}(1) = 0$), la anterior en la segunda, etc ...

```
x=[1 zeros(1,29)];
a=[1. 0. 0.9];
b=[0.3 0.6 0.3];
xn=[0. 0. 0.];
yn=[0. 0. 0.];
for n=1:length(x)
        xn(3)=xn(2); xn(2)=xn(1); xn(1)=x(n);
        yn(3)=yn(2); yn(2)=yn(1); yn(1)=0.;
        y(n) = -a*yn' +b*xn';
        yn(1)=y(n);
end
```

Dibujar el resultado haciendo uso de la función *stem*.

2. Comprobar el resultado haciendo uso de la función *filter*.
3. Obtener la respuesta del sistema a un escalón unitario.
4. ¿Es estable el sistema anterior? La condición para comprobar la estabilidad del sistema es:

$$\sum_{n=0}^{\infty} |h(n)| < \infty \tag{5}$$

Esta condición puede comprobarse estudiando gráficamente la convergencia de la serie anterior.

5. Determinar la estabilidad de un sistema con la siguiente ecuación en diferencias:

$$y(n) - 2{,}5y(n-1) + y(n-2) = 4x(n) \tag{6}$$

Usar el método descrito anteriormente.

3. Filtros FIR

Considerar el filtro FIR que proporciona como salida la media de las 10 últimas muestras de entrada (incluyendo la muestra de entrada actual).

1. Obtener y dibujar la respuesta impulsiva del sistema (30 muestras).
2. Haciendo uso de la función *filter*, obtener y dibujar la respuesta a un escalón unitario (30 muestras). ¿Cuál es el retardo introducido por el filtro? Realizar también el filtrado con la función de convolución *conv* y señalar la diferencia en el resultado respecto a *filter*.

4. Convolución Circular

La operación de filtrado puede implementarse como producto de DTFTs. Sin embargo, y como ya se ha visto anteriormente, la DTFT no es computacionalmente operativa, por lo que se usa un muestreo de la misma, la DFT. Por tanto, sería deseable que el filtrado sea implementable como producto de DFTs. Desafortunadamente, el producto de DFTs no se corresponde con la operación de convolución lineal en el dominio temporal, sino con otro tipo de convolución conocida como *convolución circular*, definida como:

$$x(n) \otimes h(n) = \textstyle\sum_{l=0}^{N-1} h(l)x((n-l) \bmod N) \tag{7}$$

$$x((n-l) \bmod N) = \begin{cases} x(n-l) & n = l, l+1, \ldots, N-1 \quad (n \geq l) \\ x(n+N-l) & n = 0, 1, \ldots, l-1 \quad (n < l) \end{cases} \tag{8}$$

donde N es la longitud de la convolución.

La convolución circular de por sí no tiene un gran interés, ya que la operación de filtrado corresponde a una convolución lineal. Sin embargo, su cómputo es eficiente si se realiza a través de algoritmos FFT, por lo que es útil como medio para obtener la convolución lineal.

1. Escribir una función MatLab que implemente la convolución circular de longitud N de dos señales haciendo uso de la expresión (7) (Nota: si las señal es de longitud inferior a N, se rellena con ceros, mientras que en caso contrario se recorta la señal hasta N). Usar el filtro de media y la señal escalón unitario (de longitud 30 muestras) del apartado anterior. Usar $N = 100$.

2. Comprobar el funcionamiento de la función anterior construyendo otra función que haga la misma convolución, pero haciendo uso de la FFT. Comparar la velocidad de ejecución de ambas funciones MatLab.

3. La convolución lineal de dos señales $x(n)$ y $h(n)$, de longitudes N_x y N_h, respectivamente, da como resultado una señal de longitud $N_x + N_h - 1$. Comprobar que la convolucion circular de longitud $N_x + N_h - 1$ de las dos señales (filtro de media y escalón unitario anteriores) coincide con su convolución lineal. Esta propiedad permite realizar eficientemente la operacion de filtrado mediante convolución circular usando FFT (esto se analiza mas detenidamente en el siguiente subapartado).

4.1. Filtrado mediante convolución circular (Opcional)

Al realizar una operación de filtrado de una señal en tiempo real, normalmente se desconoce a priori la longitud de la señal a filtrar, es decir, su longitud es indefinida. Para poder realizar su filtrado, es necesario dividir la señal de entrada al filtro en bloques cortos $x_k(n)$ de longitud L, sobre los que se va realizando sucesivamente el filtrado, como se resume en las siguientes expresiones:

$$x_k(n) = \begin{cases} x(n) & kL \leq n < (k+1)L \\ 0 & c.c. \end{cases}$$

$$x(n) = \sum_{k=0}^{\infty} x_k(n)$$

$$y(n) = x(n) * h(n) = \sum_{k=0}^{\infty} x_k(n) * h(n)$$

Cada término parcial $x_k(n) * h(n)$ es de longitud $L + N_h - 1$, por lo que es computado mediante una convolución circular de longitud $L + N_h - 1$. El resultado final $x * h$ se obtiene sumando los resultados parciales. Obsérvese que cada resultado parcial queda solapado en $N_h - 1$ muestras con los resultados parciales anterior y posterior. En cada zona de superposición, la señal de salida se obtiene como suma de las dos superpuestas. Este método se denomina de *solapamiento-suma*.

Implementar este método como una función MatLab. Probar con una señal de voz y el filtro de media anterior, para varios valores de L.