PROCESAMIENTO DIGITAL DE SEÑALES

TEMA 1: FUNDAMENTOS DE SISTEMAS DE TIEMPO DISCRETO

1. Señales y Sistemas de Tiempo Discreto

- Se introducirán conceptos de señales y sistemas de tiempo discreto.
- Para ello se detallarán un número importante de señales características de sistemas discretos y sus propiedades.
- Se enfocará el estudio a los sistemas lineales e invariantes en el tiempo, por la simplicidad de su análisis e implementación.
- Se definirán los conceptos de *linealidad*, *causalidad*, e *invariancia en el tiempo*
- Se pondrá especial énfasis a la representación de señales y sistemas.
- Para describir los sistemas de tiempo discreto se utilizarán diferentes métodos: Tablas de Valores, Ecuaciones de Diferencias, Transformada Z y Convolución.
- Por su importancia en DSP, se enfatizará la representación de las señales y sistemas por medio de ecuaciones de diferencias y transformada Z.

Señales de Tiempo Continuo y de Tiempo Discreto

Las señales se clasifican de una manera amplia en *señales analógicas* y *señales discretas*.

Una *señal analógica* será denotada por *f(t)*, una función del tiempo, en la cual la variable *t* puede representar una cantidad física, pero para nuestros fines supondremos que representa al tiempo en segundos.

Una *señal discreta* se denota por *x(k)*, en la cual la variable *k* es valuada entera y representa los instantes discretos del tiempo. En consecuencia es también llamada "señal discreta en el tiempo", la cual es una secuencia de números, y será denotada por alguna de las siguientes notaciones:

$$x(k) = \{x(k)\} = \{\dots, x(-1), \underset{\uparrow}{x(0)}, x(1), \dots\}$$

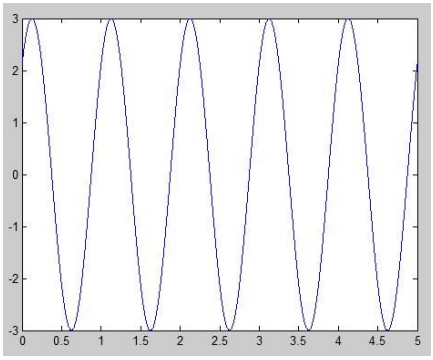
Donde la flecha hacia arriba indica el muestreo en *t=0*.

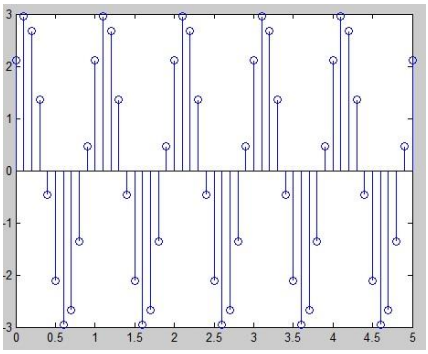
Diferentes maneras de denotar la función discreta
$f(kT), f(k), f_k, f(t_k)$ con $k = 0, 1, 2, 3, \dots$ número entero

Supongamos una señal de tiempo continuo *f(t)* y necesitamos discretizarla. Para ello debemos tomar muestras de la señal por ejemplo con una llave o CAD, con un período de muestreo T, que a nuestros fines será constante (no necesariamente es así, en casos particulares el periodo de muestreo T puede ser variable). La inversa del periodo T es la frecuencia de muestreo f_s :

Frecuencia de muestreo $f_s = 1/T$

$\omega_s = 2\pi f_s$





a) Función analógica (f(t) ∃ ∀ t>0) b) Función discreta (muestreo de f(t))

Para definir función en a), podemos tener:

1) f(t)
2) La gráfica

Para definir la función b), podemos tener:

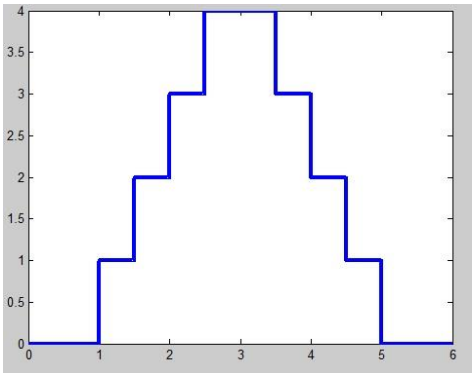
1) f(k)
2) La gráfica
3) la tabla de valores

Toda la información de la gráfica b) está contenida en la tabla de valores. La tabla nos permite tener un panorama general del comportamiento del sistema, y es permanente. Por ejemplo, una computadora es un STD, o más bien, un conjunto de STDs.

Tiempo de muestreo: Es el tiempo que transcurre entre un evento y el siguiente (de 1 a 2, de 2 a 3, etc.)

Procesamiento en tiempo real: Es cuando el sistema puede realizar todo el proceso u operación, antes de que comience de nuevo, es decir hace todo durante el período de muestreo. Si no es así es un *sistema de tiempo diferido*.

Un *sistema digital* es un sistema de tiempo discreto, pero no todo STD es un sistema digital. Un sistema digital se llama también un *STD cuantizado*. En este caso, para cada período se define además una amplitud.



Señal de TD de energía finita: Es uma señal de tiempo discreto que verifica que

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

Supongamos uma señal que varíe según la ley conocida:

$x(k) = a^k$

$\{a^k\} = a^0, a^1, a^2, ..., a^k, ...$

Esta puede ser una señal de TD de energía finita

$$\left[\sum_{0}^{\infty} a^k = \lim_{n\to\infty} \frac{1-(a)^n}{1-a^2} = \frac{1}{1-a^2} \quad si \quad a^2 < 1 \right]$$

La condición es que sea $a < 1$

Señales especiales: Pulso y Escalón

$\{u(kT)\} = \{ 1 , 1 , 1 , 1 , 1 , 1 , 1 , ... \}$ Escalón

$\{\delta(kT)\} = \{ 1 , 0 , 0 , 0 , 0 , 0 , 0 , ... \}$ Impulso δ_0 (Delta Krönecker)

o bien $= \{\underbrace{0 , 0 , 0 , ...}_{\Gamma}, 1 , 0 , 0 , 0 , ...\}$ Impulso $\delta_\Gamma = \delta(k - \Gamma)$

SEÑALES CONTINUAS Y DISCRETAS

Las señales están clasificadas de manera amplia, en *señales continuas o analógicas* y *señales discretas.*

Señales Continuas: Una *señal analógica* será denotada por $x_a(t)$ en la cual la variable *t* puede representar una cantidad física; en particular para nosotros será el tiempo en segundos.

Señales Discretas: Una *señal discreta* será denotada por *x(n)*, en la cual la variable *n* es un valor entero y representa los instantes discretos en el tiempo. En consecuencia, es también llamada una señal discreta en el tiempo.

Representación de las señales de tiempo discreto:

En general representamos las secuencias discretas en la forma:

$$x(n) = \{x(n)\} = \{\ldots, x(-1), x(0), x(1), \ldots\}$$

La representación computacional requiere en general de una secuencia finita, con una referencia de tiempo.

- $x(n) = \{x(n)\} = \{\ldots, x(-1), \underset{\uparrow}{x(0)}, x(1), \ldots\}$

donde la flecha hacia arriba indica el muestreo en *t=0*.

- O bien: $x(n) = \{\ldots, x(-1), (x(0)), x(1), \ldots\}$ el valor entre paréntesis indica el muestreo en *t=0*.
- En Matlab podemos representar una *secuencia de duración finita* por un *vector fila* de valores apropiados. Sin embargo, tal vector no tendrá la información acerca del muestreo en la posición *n*.

Por lo tanto una representación correcta de *x(n)* requerirá dos vectores: uno para *x* y otro para *n.*

Por ejemplo, una secuencia: $x(n) = \{2, 1, -1, \underset{\uparrow}{0}, 1, 4, 3, 7\}$, ó $x(n) = \{2, 1, -1, (0), 1, 4, 3, 7\}$

puede ser representada en Matlab por: n = {-3, -2, -1, 0, 1, 2, 3, 4 } ; x ={2, 1, -1, 0, 1, 4, 3, 7}

En general usaremos la representación del vector *x* sólo cuando la información de la posición de muestreo no sea requerida, o cuando tal información sea trivial (esto es cuando la secuencia comience en n=0).

Una secuencia de duración infinita arbitraria no puede ser representada en Matlab debido a las limitaciones de memoria finita.

Atributo de las señales de tiempo discreto

Las señales de tiempo discreto son sucesiones de números. Los sistemas de tiempo discreto trabajan sobre sucesiones de números. Es decir:

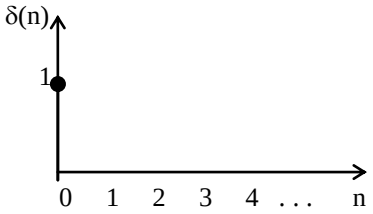
$$\{x_k\} = \{\underset{0\uparrow}{x_0}, \underset{T\uparrow}{x_1}, \underset{2T\uparrow}{x_2}, \ldots \underset{\kappa T\uparrow}{x_k}, \ldots\}$$

Tipos de secuencias:

En Procesamiento Digital de Señales utilizamos varias secuencias elementales para propósitos de análisis. A continuación se dan sus definiciones y representaciones en Matlab:

1. **Secuencia Impulso unitario (función Kröneker)**

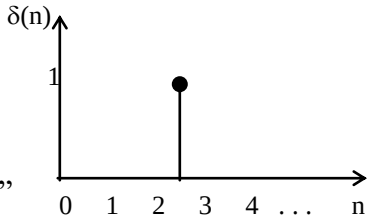
$$\delta(n) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} = \{\ldots 0, 0, (1), 0, 0, \ldots\}$$



En Matlab la función **zeros(1,N)** genera un vector fila de N ceros, el cual puede usarse para implementar δ(n) sobre un intervalo finito.

Para implementar:

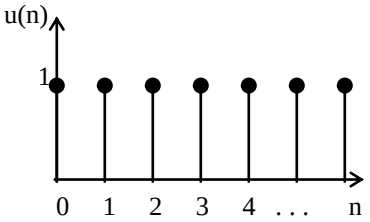
$$\delta(n-n_0) = \begin{cases} 1, & n = n_0 \\ 0, & n \neq n_0 \end{cases} = \{\ldots 0, 0, (1), 0, 0, \ldots\},$$



sobre el intervalo $n_1 \leq n_0 \leq n_2$, usamos la función de Matlab "impseq"

2. **Secuencia Escalón unitario**

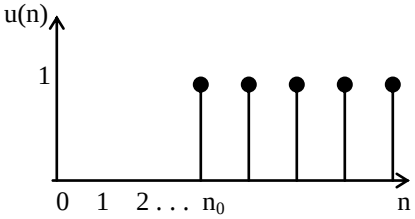
$$u(n) = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} = \{\ldots 0, 0, (1), 1, 1, \ldots\}$$



En Matlab la función **ones(1,N)** genera un vector fila de N unos, el cual puede usarse para implementar u(n) sobre un intervalo finito.

Para implementar

$$u(n-n_0) = \begin{cases} 1, & n \geq n_0 \\ 0, & n < n_0 \end{cases} = \{\ldots 0, 0, (1), 1, 1, \ldots\},$$



sobre el intervalo $n_1 \leq n_0 \leq n_2$, utilizamos la función de Matlab "stepseq".

3. **Secuencia exponencial valuada real**

$x(n) = a^n \quad \forall n\ ;\ a \in \mathbb{R}$

En Matlab se requiere un operador *arreglo* "**.**^" para implementar una secuencia exponencial real.

Por ejemplo, para generar: $x(n) = (0.9)^n$, con $0 \leq n \leq 10$,

necesitamos el siguiente script de Matlab:

>>n = [0 : 10]; x = (0.9).^n

4. **Secuencia exponencial valuada compleja**

$x(n) = e^{(\sigma + j\omega o)\, n}\ ;\ \forall n$

donde σ es llamado atenuación, y ω_o es al frecuencia en radianes.

En Matlab se usa una función "*exp*" para generar secuencias exponenciales.

Por ejemplo, para generar: x(n) = exp [(2 , j3) n] , con $0 \le n \le 10$,

Necesitamos el siguiente script de Matlab:

```
>> n = [0:10]; x = exp((2+3j)*n);
```

5. Secuencia sinusoidal

$x(n) = \cos(\omega_o n + \theta) ; \quad \forall n$
donde θ es la fase en radianes.
En Matlab se usa la función cos o sin para generar secuencias senoidales.
Por ejemplo, para generar

$x(n) = 3\cos(0.1\,\pi n + \pi/3) + 2\,sen(0.5\,\pi n)$, con $0 \le n \le 10$,

necesitamos un script de Matlab:

```
>> n = [0:10]; x =3*cos(0.1*pi*n+pi/3)+2*sin(0.5*p*n);
```

6. Secuencia random

Muchas secuencias prácticas no pueden ser descriptas por expresiones matemáticas como las anteriores. Este es el caso de las secuencias "random" o "estocásticas" y están caracterizadas por parámetros de las funciones densidad de probabilidad asociadas ó sus momentos estadísticos.

En Matlab se disponibles de dos tipos de secuencias pseudo-random:

1) La función rand (1,N) genera una secuencia random de longitud N cuyos elementos están uniformemente distribuidos en el intervalo [0,1]
2) La función randn(1,N) genera una secuencia random gaussiana de longitud N con media 0 y varianza 1.

Otras secuencias random pueden ser generadas usando transformaciones de las funciones anteriores.

GUIA DE FUNCIONES PARA TRABAJOS PRÁCTICOS CON Matlab

Resumen de algunas secuencias útiles y operaciones y su correspondiente función en Matlab

1. Impulso unitario (función Kröneker) **Función en Matlab**

$$\delta(n) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} = \{\ldots 0, 0, (1), 0, 0, \ldots\}$$ impseq

O, en general

$$\delta(n-n_0) = \begin{cases} 1, & n = n_0 \\ 0, & n \neq n_0 \end{cases} = \{\ldots 0, 0, (1), 0, 0, \ldots\}$$ En un intervalo $n_1 \leq n_0 \leq n_2$

Lo que computacionalmente es más sencillo representar

2. Escalón unitario

$$u(n) = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} = \{\ldots 0, 0, (1), 1, 1, \ldots\}$$ stepseq

Y computacionalmente

$$u(n-n_0) = \begin{cases} 1, & n \geq n_0 \\ 0, & n < n_0 \end{cases} = \{\ldots 0, 0, (1), 1, 1, \ldots\}$$

En un intervalo $n_1 \leq n_0 \leq n_2$.

3. Exponencial real

$x(n) = a^n, \quad \forall \quad n;\ a \in \mathcal{R}$

4. Exponencial compleja

$x(n) = e^{(\sigma n + j\omega o)n}, \quad \forall \quad n$

donde σ es el parámetro de atenuación y ω_o es la frecuencia en radianes.

5. Sinusoidal

$x(n) = cos\ (\omega_o n + \theta), \quad \forall \quad n$

donde θ es la fase en radianes.

6. Aleatoria

- En general no pueden ser expresadas directamente por una expresión matemática cerrada.
- Se describen por su función densidad de probabilidades o momentos estadísticos.

- Computacionalmente dos tipos de secuencias aleatorias son útiles: a) de distribución uniforme y Gaussiana.
- En general están caracterizadas por su media y varianza.

7. Periódica

$$x(n) = x(n + N)$$

el menor *N* que satisface la anterior se denomina período fundamental.

Usaremos $\hat{x}(n)$ para denotar una secuencia periódica. Una secuencia de este tipo se ilustra en el ejemplo 2.1.

Operaciones sobre secuencias **Función en Matlab**

1. Suma de señales sigadd

$$\{x_1(n)\} + \{x_2(n)\} = \{x_1(n) + x_2(n)\}$$

En particular, la única restricción para sumarlas como vectores, es que tengan igual longitud.

(Ver ejemplo 2.2)

2. Multiplicación de señales sigmult

$$\{x_1(n)\}.\{x_2(n)\} = \{x_1(n).\ x_2(n)\}$$

En particular, la única restricción para el producto muestra a muestra, como vectores, es que tengan igual longitud.

(Ver ejemplo 2.2)

3. Escalamiento

$$\alpha\{x(n)\} = \{\alpha\, x(n)\}$$

donde α es un escalar

4. Desplazamiento sigshift

$$y(n) = \{x(n - k)\}$$

Haciendo m = n – k , entonces n = m + k , de donde

$$y(m + k) = \{x(m)\}$$

por lo que esta operación no produce ningún efecto sobre el vector *x*

(Ver ejemplo 2.2)

5. Reversión (Holding) sigfold

$$y(n) = \{x(- n)\}$$

Cada muestra es reflejada alrededor de n = 0

6. Sumatoria de muestras

$$y(n) = \sum_{n=n_1}^{n_2} x(n) = x(n_1) + ... + x(n_2)$$

7. Producto de muestras

$$y(n) = \prod_{n=n_1}^{n_2} x(n) = x(n_1) \times ... \times x(n_2)$$

8. Señal energía

$$\varepsilon_x = \sum_{n=-\infty}^{\infty} x(n)x^*(n) = \sum_{n=-\infty}^{\infty} |x(n)|^2$$

donde el símbolo * denota el complejo conjugado

9. Señal potencia

$$\mathcal{P}_x = \frac{1}{N} \sum_{n=0}^{N-1} |x(k)|^2$$