INTRODUCCIÓN AL PROCESAMIENTO DE SEÑALES

Práctica 1:

Señales VIC y VID. Delta de Dirac. Periodicidad. Energía y Potencia.

1. **Impulsos continuos y discretos**

   a) Enuncie la propiedad de extracción de la delta de Dirac.

   b) Considerando la delta de Dirac como límite para $\epsilon \to 0$ de la función $\frac{1}{\epsilon} \sqcap ((t-\epsilon/2)/\epsilon)$, trate de interpretar la propiedad de extracción cuando se aplica a una función continua en $t=0$.

   c) Evalúe las siguientes integrales usando propiedades de la delta de Dirac

   I. $\displaystyle\int_{-\infty}^{t} \delta(\tau)d\tau$ II. $\displaystyle\int_{-2t}^{2t} \delta(\tau-3)d\tau$

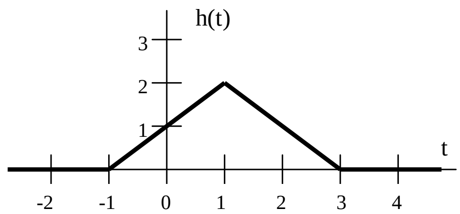
   III. $\displaystyle\int_{-\infty}^{\infty} u(\tau/2-6)\,\delta(\tau/3-5)d\tau$ IV. $\displaystyle\int_{-\infty}^{at+b} f(at-2-\tau/3)\,\delta((\tau-5)/2)d\tau$

   d) Enuncie una propiedad equivalente a la de extracción de la delta de Dirac para señales discretas.

   e) Escriba la señal $x[n]$ del ejercicio 3 en términos de impulsos (como suma de deltas de Kronecker).

2. **Manejo de Señales VIC**

   Dada la señal de variable independiente continua $h(t)$ de la figura, calcule y grafique las siguientes señales:



   a) $h(t+1)$ *b*) $h(2t-3)$

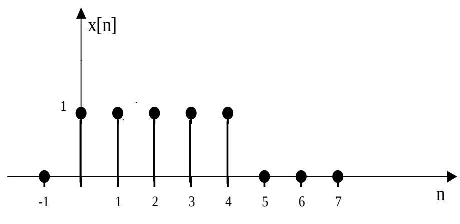
   c) $2\,h(-\frac{1}{2}(t+10))$ *d*) $h(\frac{t}{2})[u(t+2)-u(t-2)]$

   e) $h(t^2)$ *f*) $\mathrm{Impar}\{h(t)\}$

   g) $\mathrm{Par}\{h(t)\}$

3. **Manejo de Señales VID**

   Dada la señal de variable independiente discreta $x[n]$ de la figura, calcule y grafique las siguientes señales:



   a) $x[n-1]$ *b*) $x[2n]$

   c) $x[n^2]$ *d*) $x[-n-3]$

   e) $\mathrm{Impar}\{x[n]\}$ *f*) $\mathrm{Par}\{x[n]\}$

   g) $y[n] = \begin{cases} x[n/2] & n \text{ pares} \\ 0 & n \text{ impares} \end{cases}$

4. **Señales Periódicas y Aperiódicas**

   Determine si cada una de las siguientes señales es periódica o aperiódica. En el primer caso especifique su período fundamental.

   a) $x(t) = -2\,\mathrm{sen}(-0{,}2t + \frac{5\pi}{3})$ *b*) $x[n] = -2\,\mathrm{sen}(-0{,}2n + \frac{5\pi}{3})$ *c*) $x(t) = \frac{1}{2}[\cos(2t-\frac{\pi}{4})]^2$

   d) $x(t) = e^{j(\frac{\pi}{2}t-\pi)}$ *e*) $x[n] = e^{j(\frac{n}{2}-\pi)}$ *f*) $x(t) = 2\,\cos(2\pi t) + \mathrm{sen}(10t)$

   g) $x(t) = 2\,\cos(2\pi t)\,\mathrm{sen}(10t)$ *h*) $x[n] = \cos(2\pi n^2)$ *i*) $x(t) = \cos(2\pi t^2)$

   j) $x(t) = \displaystyle\sum_{k=-\infty}^{+\infty} c_k\, e^{\frac{j2\pi kt}{T_0}}$ con $c_1 \neq 0$

¿Cambiaría su respuesta de los incisos b) y e) pensando que el procesamiento de la señal se llevará a cabo en un sistema de cómputo que posee una representación de π con un número finito de decimales (por ej. 3,14; 3,1416; etc.)?

5. **Energía, Potencia y Valor Medio**

   Para una señal (VIC o VID) x, que podría ser compleja, denotaremos con E_x a su energía, con P_x a su potencia y con $\bar{x}$ a su valor medio.

   a) Escriba las definiciones de E_x, P_x y $\bar{x}$ para señales VIC y para señales VID.

   b) Probar que si E_x es finita, entonces P_x es cero, y que si P_x es finita, entonces E_x es infinita.

   c) Calcular E_x, P_x y $\bar{x}$ para las siguientes señales.

   I. $x(t) = u(t)$

   II. $x(t) = A\text{sen}(2\pi f_0 t + \phi)$, con $f_0, \phi \in \mathbb{R}$

   III. $x(t) = 2e^{j6\pi t}$

   IV. $x[n] = (1/2)^n u[n]$

   V. $x[n] = \begin{cases} 3 & n = 3\,k, \quad k \in \mathbb{Z} \\ 0 & \text{c.c.} \end{cases}$

   d) Definiendo $x_P(t) = \text{Par}\{x(t)\}$ y $x_N(t) = \text{Impar}\{x(t)\}$, calcular E_x, E_{x_P} y E_{x_N} para la siguiente señal

   $x(t) = 3t \cdot [u(t) - u(t-2)]$

   e) Probar que, en general, para señales de energía $E_x = E_{x_P} + E_{x_N}$. Verificar que la señal anterior cumple con esto.

   f) Probar que, en general, para señales de potencia $P_x = P_{x_P} + P_{x_N}$, y que $\bar{x} = \bar{x}_P$.

   g) Definiendo $\tilde{x}[n] = x[n] - \bar{x}$, probar que $P_x = P_{\tilde{x}} + |\bar{x}|^2$.

6. **Usando Octave**

   Antes de comenzar este ejercicio quizás sea conveniente que lea el Apéndice sobre Octave.

   a) Genere la función $h(t)$ del ejercicio 2 creando un archivo de texto titulado '`tri1.m`'. Dicho archivo debe contener las siguientes sentencias:

```
function h = tri1(t)
    % TRI1 señal del ejercicio 2 de la Práctica 1
    h = (t+1).*(t >= -1 & t < 1) + (-t+3).*(t >= 1 & t < 3);
end
```

   En el ambiente de trabajo de Octave defina un vector para la variable independiente `t`, evalúe la señal y grafique ejecutando las siguientes instrucciones:

```
t=[-2:.001:8];
h=tri1(2*t-3);
plot(t,h);
```

   Repita los pasos anteriores y evalúe los restantes incisos (puede utilizar los comandos `grid`, `axis`, `title`, `xlabel` e `ylabel` para personalizar la presentación de los gráficos).
   Aclaración: en algunos casos puede parecer que existen diferencias con los resultados obtenidos analíticamente, ésto se resuelve definiendo adecuadamente el vector `t`.

   b) Genere una nueva función que permita definir la señal $x[n]$ del ejercicio 3, y verifique los resultados obtenidos. Deberá definir un nuevo vector `n`. En este caso puede utilizar el comando `stem` para graficar las secuencias.

c) Grafique las señales del ejercicio 4 (salvo inciso j) utilizando un rango de valores adecuados para que se note el tipo de señal que es. Para las señales complejas puede graficar parte real e imaginaria, o módulo y fase, o si acepta el desafío puede probar con un gráfico 3D (usando `plot3`).

d) Genere un *script* de Octave que permita obtener una aproximación para los cálculos del ejercicio 5*c)*, incisos II y IV.

7. **Comportamiento Aleatorio**

Existen señales que no pueden ser caracterizadas por un comportamiento determinístico (no podemos decir mediante una fórmula cómo se van a comportar en el tiempo). Estas señales se comportan de manera aleatoria. Para su caracterización se recurre al concepto de *proceso aleatorio* y quedan descriptas por medio de determinados parámetros estadísticos.

Para comenzar a tomar noción de cómo se "ven" estas señales, recurriremos a las herramientas de generación de señales aleatorias que posee Octave. Vale la pena aclarar que la generación de estas señales se lleva a cabo de forma determinística, mediante métodos que permiten obtener señales que *lucen como aleatorias*, y que por ello se denominan pseudo-aleatorias.

a) Mediante las siguientes instrucciones puede generar una serie de valores (denominado vector en la nomencaltura de Octave, y que interpretaremos como los valores que toma nuestra señal) de una señal aleatoria con distribución uniforme.

```
N = 1000; x = rand(1,N); figure, plot(x); figure, hist(x);
y = 10*(rand(1,N)-0.5); figure, plot(y); figure, hist(y);
```

Vea qué ocurre al ejecutar nuevamente las instrucciones. Compare las figuras. ¿En qué se parecen y en qué difieren? Vea qué sucede al modificar el valor de `N`.

b) Ejecute las sentencias siguientes y compare con los resultados anteriores (en este caso la distribución es de tipo Normal o Gaussiana)

```
N = 1000; z = randn(1,N); figure, plot(z); figure, hist(z);
```

Vea qué sucede al modificar el valor de `N`. Puede modificar la cantidad de intervalos utilizados por la gráfica del histograma, utilizando la sentencia `hist(z,K)` con `K` el número de intervalos a utilizar (por ejemplo 20, 50, etc.).

c) Para analizar qué sucede cuando se suman señales aleatorias con distribución uniforme, ejecute las sentencias siguientes

```
N = 1000; M = 2; x = rand(M,N)-0.5; y = sum(x,1);
figure, hist(x'); figure, hist(y);
```

Vea cómo resulta la señal `y` (su histograma). ¿Qué sucede si se cambia el valor de `M` por los valores 5, 10, 20? ¿A qué se parece el histograma de estas señales? Vea cuáles son los valores máximos que puede tomar la señal en cada caso.

Algunos resultados

1. *b*) I. $u(t)$ II. $u(2t-3)-u(-2t-3)$ III. 3 IV. $2f(at-11/3)u(at+b-5)$

4. *a*) $T=10\pi$ *b*) Ap. *c*) $T=\pi/2$ *d*) $T=4$ *e*) Ap. *f*) Ap.
g) Ap. *h*) $N=1$ *i*) Ap. *j*) $T=T_0$

5. *c*)

	E_x	P_x	$\bar{x}$
I	∞	1/2	1/2
II	∞	$A^2/2$	0
III	∞	4	0
IV	4/3	0	0
V	∞	3	1

d) $E_x=24$, $E_{x_P}=12$, $E_{x_N}=12$

Apéndice sobre Octave

A continuación se dejan algunas sugerencias útiles para el uso del software Octave a lo largo de la cursada. Es recomendable incorporar algunos de estos consejos para poder realizar las tareas con el utilitario de cada práctica y especialmente, aplicarlos en los laboratorios.

1. **Trabajar de manera ordenada:** Para que sea sencillo ordenar el trabajo realizado en Octave se sugiere la siguiente jerarquía de directorios:

   a) Comenzar con la carpeta ***Documents*** o ***Mis Documentos*** (o ***home*** si se trabaja en Linux) y crear la carpeta ***AnSyS***.

   b) Dentro de la carpeta ***AnSyS*** crear una carpeta llamada ***Octave***.

   c) Dentro se sugiere crear una carpeta para funciones útiles, ***funciones*** y luego se puede crear una carpeta en el mismo nivel por cada práctica: ***P1***, ***P2***, ***P3***, etc. La idea es que en la carpeta ***funciones*** se encuentren todos los scripts (archivos ***.m***) que sean utilizados en varios de los trabajos prácticos a lo largo de la cursada.

   d) Dentro de la carpeta correspondiente a cada práctica puede crearse una nueva carpeta por cada ejercicio, o destinar un script a cada ejercicio, se deja esto último a gusto del estudiante.

2. **Uso de funciones que se encuentren en otra carpeta:** Es deseable que todas las funciones que se utilicen recurrentemente en las resoluciones con Octave se encuentren en una misma carpeta, en nuestro caso, en la carpeta ***funciones***. Para ello se debe indicar a Octave el *path* de la carpeta donde se encuentran los scripts que se desean utilizar. Por ejemplo, si se tiene un script **Ej1a.m** localizado en **\AnSyS\Octave\P1** y se desea usar el script **cajon.m** localizado en **\AnSyS\Octave\funciones** deberá agregarse lo siguiente al comienzo del script **Ej1a.m** :

```
addpath('../funciones')   % Para incluir los .m de la carpeta funciones.
```

3. **Para graficar como se debe:** Muchas veces no se tiene en cuenta que al realizar un gráfico deben considerarse varias cuestiones: escala de cada eje, tamaño de la letra de los ejes y la leyenda (si la hay), grosor de las líneas o puntos graficados, etc. Por lo tanto, como agregar todas estas características a cada gráfico puede consumir tiempo y líneas de código, y es una operación que se repetirá muchas veces a lo largo de la materia, se recomienda construir una función que automatice lo anterior.

   A modo de ejemplo se propone la siguiente función, **plotCompleto.m**, que puede utilizarse libremente o modificarse a gusto de cada estudiante:

```
function plotCompleto(lim_ejes,eje_x,eje_y,titulo,tam_letra,col,grosor,t,x)

    % plotCompleto(lim_ejes,eje_x,eje_y,titulo,tam_letra,col,grosor,t,x)
    %
    % Realiza un gráfico de una SVIC de manera adecuada.
    % Ejemplo de uso:
    %    t = -40:0.01:40;
    %    xa = -2*sin(-0.2*t + 5/3*pi);
    %    plotCompleto([t(1) t(end) -2 2],'t','f(t)','Señal f(t)',20,'r*-',1.5,t,xa)

    figure('units','normalized','outerposition',[0 0 1 1]); % Creo y maximizo figura.
    plot(t,x,col,'Linewidth',grosor);  % Grafico. Color (y marcador) y grosor.
    axis(lim_ejes); grid on;   % Límites de los ejes. Grilla.
    set(gca,'FontSize', tam_letra);   % Tamaño de letra para la leyenda y ejes.
    xlabel(eje_x);   % Nombro el eje x.
    ylabel(eje_y);   % Nombro el eje y.
```

```
    title(titulo);   % Coloco título para el gráfico.

end
```

4. **Ejemplo:** gráfico de la SVIC $x(t) = \sqcap(t)$.

   Se define el script **cajon.m**[1] y se lo guarda dentro de la carpeta ***funciones***. Luego se crea un script, en este caso guardado en la carpeta ***P1*** con el siguiente contenido:

```
clear all;close all; clc;   % Al comenzar un script...
addpath('../funciones')  % Para incluir los .m de la carpeta funciones.

dt = 1e-2;      % paso temporal
t = -1:dt:1;    % vector tiempo
x = cajon(t);   % función cajón definida en "cajon.m", dentro
                % de la carpeta "funciones"
% gráfico
plotCompleto([-1 1 -0.5 1.5],'t','Amplitud','Gráfico de la SVIC',25,'m*-',1.5,t,x);
```

5. **Últimas consideraciones:**

   - Tenga en cuenta que al usar *addpath* el símbolo **../** simboliza subir una carpeta en jerarquía relativo al lugar donde se encuentre el script que se está ejecutando. En caso de usar un número de carpetas diferente al propuesto, debe considerarse la cantidad correcta de **../** para que el comando sea exitoso.
   - Al crear funciones o scripts es útil comentar las primeras líneas con información sobre dicho script. Dicha información puede ser accedida al ingresar *help función*. Por ejemplo intente: `help plot`.
   - ★ Se deja para investigación personal el sistema de control de versiones **git**[2] que también podría utilizarse como agregado a la gestión de scripts de Octave. Esto es interesante ya que en el mundo profesional **git** es muy utilizado y sería bueno que el que esté interesadx, pueda informarse al respecto.

[1] Similar a Ej 6, pero en vez de una función triangular se debe crear una función que valga 1 si $|t| < 1/2$.
[2] Sitio web: https://git-scm.com/