Práctica No. 4
Generación de Señales Básicas
(MATLAB)

Objetivo:

- El estudiante escribirá los programas para MATLAB necesarios para obtener las gráficas de comportamiento de señales básicas.

Trabajo previo

- Leer la lectura 2: *Señales Discretas*
- Realizar las prácticas 3, 4 y 5 de la página de Introducción a MATLAB

Desarrollo

Señales continuas

En MATLAB programe el siguiente vector de tiempo:

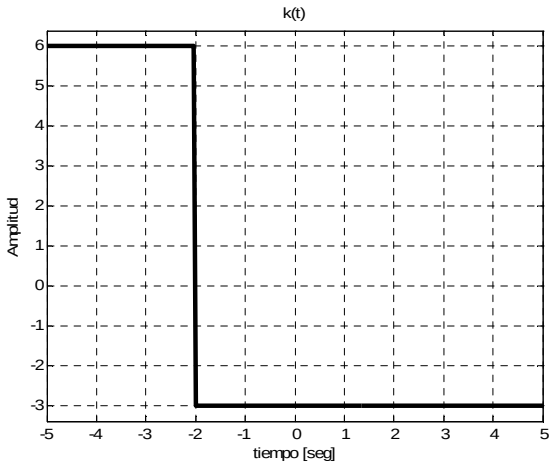
```
>> t=-5:.05:5;
```

1. Generación de una señal escalón general

```
function esc=escalon(A, B, ta, t)
% Programa que calcula una funcion escalon
% que se presenta en el instante ta
% A es la magnitud del escalon despues
de ta
% B es la magnitud del escalon antes de
ta
% t es el intervalo de tiempo en el cual
esta definida la funcion

y1=(t<ta);
y=(t>=ta);
esc=B.*y1+A.*y;
```

Utilice este programa para generar la señal que se muestra:



2. Generación de una señal escalón unitario cuya variable en el tiempo es transformable

```
function esc=escalonuni(t, A)
% Programa que calcula una funcion escalon unitario
% el cual puede someterse a distintas transformaciones
% de su argumento
%
% A es el argumento de la funcion:
%       A=k*(t+j)
% t es el intervalo de tiempo en el cual esta definida la funcion

y1=(A<0);
y=(A>=0);
esc=0*y1+1.*y;

plot(t, esc)
```

Utilice el programa para generar el escalón unitario $\boldsymbol{u_{e1}(t)}$

3. Generación de una señal exponencial cuya variable en el tiempo es transformable

```
function   [e,re,ie]=exponencial(A, r, t, B)
% Programa que calcula y grafica la funcion exponencial que puede ser
% sometida a distintas transformaciones en su variable independiente
% Sus caracteristicas son:
%
%       [e,re,ie]=exponencial(A, r, t, B)
%
%  donde las entradas son
%       A:   amplitud
%       r:   argumento
%       t:  intervalo de tiempo
%       B: Argumento de la funcion, el cual tiene la forma general
%
%                       B = k ( t + j )
%
%  Los datos de salida son
%
%    e:   señal exponencial completa (sinusoidal envuelta con componentes real e imaginaria)
%    re:  parte real de la señal exponencial
%    phi: parte imaginaria de la señal exponencial
%

%  Calculo de la señal exponencial

e=A*exp(r*B);

% calculo de la componente real del vector e

re=real(e);
```

```
% calculo de la componente iamginaria del vector e

ie=imag(e);

% Grafica de la funcion


a=size(ie,2);

if ie(1)==ie(a)
   plot(t,re),grid
else
   plot3(t,re,ie), grid
end
```

Utilice el programa para generar la señal exponencial $\boldsymbol{f(t) = 3e^{(-3+2j)t}}$

4. Generación de una señal senoidal cuya variable en el tiempo es transformable

```
function [y,fo,T]=seno(A,w,phi,t, B)
% Programa que calcula la funcion seno de forma tal que pueda ser sometida a
%transformaciones en su variable independiente. Sus caracteristicas son:
%
%        [y,fo,T]=seno(A,w,phi,t, B)
%
%  donde las salidas son
%        y:   señal seno
%        fo: frecuencia fundamental
%        T:  periodo fundamental
%
%  los datos de entrada son
%
%        A:   amplitud de la funcion
%        w:   frecuencia angular
%        phi:  defasamiento en radianes
%        t:    intervalo de tiempo
%        B:   Argumento que define la forma de transformar al tiempo, su
%             forma general es    B = k ( t + a )


fo=w/(2*pi);          %  Calculo de la frecuencia fundamental

T=1/fo;               %  Calculo del periodo fundamental

y=A*sin(w*B+phi);    % Declaración de la señal seno

plot(t,y)
```

Grafique la señal $\boldsymbol{c(t) = 2\cos(4t)}$

5. Generación de una señal rampa

```
function r=rampa(A, t, B)

% Programa que calcula una funcion escalon
% A es la pendiente de la rampa
% B es el argumento de la funcion, el cual es en general
%                B = k ( t + j )

r=A*B.*(B>=0);

plot(t,r)
```

Grafique la señal rampa unitaria

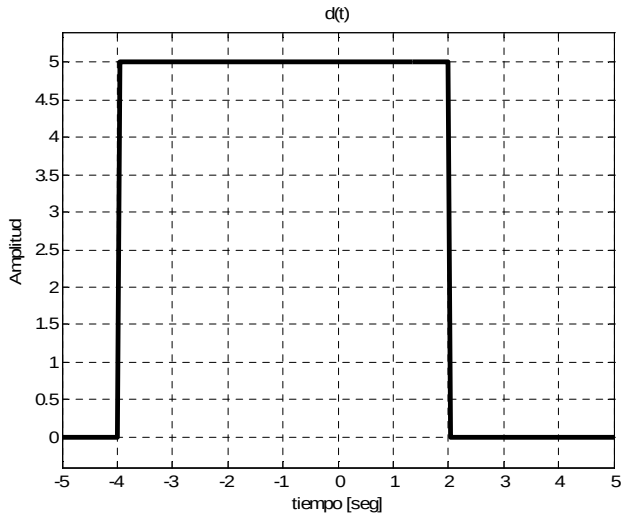
6. Generación de la señal rectángulo

```
function r=rectangulo(A, t, ta, tb)
% Programa que calcula y grafica un rectangulo simple
% A es la amplitud maxima del
rectangulo
% t es el intervalo de tiempo
% ta en el tiempo en que inicia el
rectangulo
% tb es el tiempo en que termina el
rectangulo

r=A*((t>ta)-(t>tb));

plot(t,r)
```

Grafique la siguiente señal rectángular:



7. Generación de una señal triángulo

```
function r=triangulo(t, ta)

% Programa que calcula un triangulo simple

r=rampa(1,t,t+ta)-2*rampa(1,t,t)+rampa(1,t,t-ta);
plot(t,r)
```

Señales discretas

Para generar las contrapartes discretas en el tiempo de las señales anteriores, simplemente modifique los programas cambiando el comando **plot** por **stem** y definiendo la variable del tiempo como

```
>> t=-5:1:5;
```

8. Genere la contraparte discreta en el tiempo de todas las señales anteriores.