lOMoARcPSD|95983

# Práctica 5. “Operaciones Básicas Sobre Señales En Matlab”

# OBJETIVO

*El alumno aplicara los conocimientos teóricos vistos en clase referente a las operaciones básicas sobre señales en una aplicación usando Matlab*

# EJEMPLOS GRÁFICOS

# Operaciones efectuadas sobre variables dependientes

* ***Escalamiento de amplitud.***

Imagen que contiene objeto, reloj

Descripción generada automáticamente

Operación de escalamiento de amplitud: Señal en tiempo continuo de amplitud y duración.

Versión escalada en amplitud de por un factor de.

* ***Suma.*** Diagrama

  Descripción generada automáticamente

Operación de suma: Las amplitudes de las señales y se suman en los instantes de tiempo correspondientes para formar la señal

.***Multiplicación.***

Gráfico, Gráfico de cajas y bigotes

Descripción generada automáticamente

Operación de multiplicación: Las amplitudes de las señales y se multiplican en los instantes de tiempo correspondientes para formar la señal.

* ***Diferenciación.***

Diagrama

Descripción generada automáticamente

Operación de diferenciación

* ***Integración.***

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

Operación de integración.

# Operaciones efectuadas sobre la variable independiente

* ***Escalamiento de tiempo.***

Diagrama

Descripción generada automáticamente

Operación de escalamiento de tiempo: Señal en tiempo continuo. Versión comprimida de por un factor de. Versión expandida de por un factor de.

Imagen en blanco y negro

Descripción generada automáticamente con confianza baja

Efecto de escalamiento de tiempo en una señal en tiempo discreto: Señal en tiempo discreto . Versión comprimida de por un factor de , con algunos valores de la original perdidos como resultado de la compresión.

* ***Reflexión.*** Diagrama

  Descripción generada automáticamente]

Operación de reflexión: Señal en tiempo continuo . Versión reflejada de alrededor del origen.

* ***Corrimiento en tiempo o desplazamiento.***

Diagrama

Descripción generada automáticamente

Operación de corrimiento en el tiempo: Señal en tiempo continuo, de amplitud y duración , simétrica en torno al origen. Versión recorrida o desplazada de en unidades de tiempo.

# Regla de la precedencia para el corrimiento en el tiempo y el escalamiento en el tiempo

Imagen que contiene antena, objeto, reloj

Descripción generada automáticamente

Regla de la precedencia aplicada correctamente: Señal en tiempo continuo, de amplitud y duración, simétrica en torno al origen. Señal intermedia, representando la versión recorrida en el tiempo de en unidades de tiempo. Señal deseada, resultado de la compresión de por un factor de.

Imagen que contiene objeto, reloj

Descripción generada automáticamenteRegla de la precedencia aplicada incorrectamente: Señal en tiempo continuo, de amplitud y duración, simétrica en torno al origen. Señal escalada en el tiempo. Señal obtenida recorriendo 3 unidades de tiempo.

# AYUDA PARA EL DESARROLLO DE LA PRÁCTICA

|  |
| --- |
| **Operaciones efectuadas sobre variables dependientes** |
| **Escalamiento de amplitud** |
| Para realizar escalamiento de amplitud en MATLAB basta con multiplicar mediante el operador \* una señal dada por un escalar. Así, por ejemplo, dada una señal x(*t* ) que se desea escalar en una amplitud *A* , la sintaxis sería: A.\*x(t). El punto “.” antes del operador \* se hace necesario siempre y cuando la variable de tiempo de la que depende la señal este definida vectorialmente, en caso de que este definida simbólicamente se omite el punto.  Se aplican comentarios similares al caso de señales en tiempo discreto. |
| **Comandos** |
| %% Escalamiento de Amplitud en Tiempo Continuo y Discreto A = 5; %Factor de Escalamiento de amplitud  Fs = 10000; %Frecuencia de Muestreo f = 0.1; %Frecuencia Fundamental Hz  T = 3\*(1/f); %Número de Periodos a Graficar t = 0:1/Fs:T-1/Fs; %Tiempo de Simulación  xt = A.\*sin(2\*pi\*f.\*t); %Generación de Señal  subplot(2,1,1),plot(t,xt),grid on,ylim([-A A]) %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('x(t)') n = 0:T; %Vector de Tiempo Dicreto  xn = A.\*sin(2\*pi\*f.\*n); %Generación de Señal x[n] subplot(2,1,2),stem(n,xn),grid on,ylim([-A A]) %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('x[n]') |
| **Gráfica** |
| Gráfico, Histograma  Descripción generada automáticamente |
| **Suma** |
| **Descripción**  Para realizar la suma de dos señales en MATLAB® se requiere, además de utilizar el operador +, que las dos señales a sumar compartan la misma variable de tiempo; ya sea un vector (variable de tiempo numérica) o una variable simbólica. |
| **Comandos** |
| %% Suma de Señales en Tiempo Continuo y Discreto A = 5; %Amplitud  Fs = 10000; %Frecuencia de Muestreo  f1 = 0.1; %Frecuencia Fundamental de la Señal x1(t) Hz f2 = 0.2; %Frecuencia Fundamental de la Señal x2(t) Hz t = 0:1/Fs:30; %Tiempo de Simulación  x1t = A.\*sin(2\*pi\*f1.\*t); %Generación de Señal x1(t) x2t = A.\*cos(2\*pi\*f2.\*t); %Generación de Señal x2(t) yt = x1t + x2t; %Señal suma y(t)=x1(t)+x2(t) subplot(2,1,1),plot(t,yt),grid on %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y(t)=x1(t)+x2(t)') n = 0:30; %Vector de Tiempo Dicreto  x1n = A.\*sin(2\*pi\*f1.\*n); %Generación de Señal x1[n] x2n = A.\*cos(2\*pi\*f2.\*n); %Generación de Señal x2[n] yn = x1n + x2n; %Señal suma y[n]=x1[n] + x2[n] subplot(2,1,2),stem(n,yn),grid on %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('y[n]=x1[n]+x2[n]') |
| **Gráfica** |
| Gráfico, Gráfico de líneas  Descripción generada automáticamente |

|  |
| --- |
| **Multiplicación** |
| **Descripción**  Para este caso se aplica la misma descripción que para el “Escalamiento de Amplitud” teniendo en cuenta que el parámetro *A* ahora no será un escalar sino una señal. |
| **Comandos** |
| %% Multiplicación de Señales en Tiempo Continuo y Discreto A = 5; %Amplitud  Fs = 10000; %Frecuencia de Muestreo  f1 = 0.1; %Frecuencia Fundamental de la Señal x1(t) Hz f2 = 0.2; %Frecuencia Fundamental de la Señal x2(t) Hz t = 0:1/Fs:30; %Tiempo de Simulación  x1t = A.\*sin(2\*pi\*f1.\*t); %Generación de Señal x1(t) x2t = A.\*cos(2\*pi\*f2.\*t); %Generación de Señal x2(t) yt = x1t .\* x2t; %Señal suma y(t)=x1(t)\*x2(t) subplot(2,1,1),plot(t,yt),grid on %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y(t)=x1(t)\*x2(t)') n = 0:30; %Vector de Tiempo Dicreto  x1n = A.\*sin(2\*pi\*f1.\*n); %Generación de Señal x1[n] x2n = A.\*cos(2\*pi\*f2.\*n); %Generación de Señal x2[n] yn = x1n .\* x2n; %Señal suma y[n]=x1[n] \* x2[n] subplot(2,1,2),stem(n,yn),grid on %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('y[n]=x1[n]\*x2[n]') |
| **Gráfica** |
| Gráfico, Gráfico de líneas  Descripción generada automáticamente |

|  |
| --- |
| **Diferenciación** |
| **Descripción**  diff permite calcular la derivada de una expresión matemática, el argumento x es dicha expresión y puede ser tanto un vector(numérica) como una función simbólica. El argumento n es el grado de la derivada que se desea calcular, por ejemplo, para un n=2, diff calculará la segunda derivada de la expresión x. Finalmente, el argumento dim es la dimensión en la cual se desea calcular la derivada siempre y cuando x sea una matriz de tamaño superior al especificado en dim. Los argumentos n y dim son opcionales y para el caso de estudio que compete a esta práctica no será necesario definirlos, pues al omitir n en el comando diff, éste asume el cálculo de una derivada de grado 1. |
| **Comandos** |
| Para generar una señal que sea la derivada de otra señal en MATLAB® se tienen dos opciones: la primera es creando una señal simbólica que represente la señal original, luego se calcula la derivada con el comando diff y finalmente se hace una sustitución de la variable simbólica de tiempo por un vector de tiempo; la segunda opción es realizar la diferenciación de forma aproximada con la señal definida mediante el vector de tiempo directamente. A continuación, se presentan ejemplos de ambos casos.  %% Diferenciación en Tiempo Continuo y Discreto  %--------Diferenciación Método Aproximado-------------  Fs = 200; %Frecuencia de Muestreo  t = 0:1/Fs:2\*pi; %Tiempo de Simulación xt = sin(t); %Generación de Señal x(t)  yt = diff(xt).\*Fs; %Cálculo de la derivada y(t)=x'(t) subplot(2,2,1),plot(t(:,1:length(yt)),yt,'r',t,xt,'b'),legend('y(t)=Derivada de x(t)','Señal Original x(t)'),grid on %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('x(t) y su Diferenciación - Método Aproximado')  n = 0:7; %Vector de Tiempo Dicreto  xn = sin(n); %Generación de Señal x[n]  yn = diff(xn); %Cálculo de la derivada y[n]=x'[n] subplot(2,2,2),stem(n(:,1:length(yn)),yn,'Color','r'),hold on,  stem(n,xn,'Color','b'),legend('y[n]=Derivada de x[n]','Señal Original x[n]'),grid on %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('x[n] y su Diferenciación - Método Aproximado')  %--------Diferenciación Método Simbólico(Exácto)-------------  syms tiempo;  Fs = 200; %Frecuencia de Muestreo  t = 0:1/Fs:2\*pi; %Tiempo de Simulación  xt = sin(tiempo); %Generación de Señal x(t)  yt = diff(xt,tiempo); %Cálculo de la derivada y(t)=x'(t) xt = double(subs(xt,tiempo,t)); %Cambio de variable  yt = double(subs(yt,tiempo,t)); %Cambio de variable subplot(2,2,3),plot(t,yt,'r',t,xt,'b'),legend('y(t)=Derivada de x(t)','Señal Original x(t)'),grid on %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('x(t) y su Diferenciación - Método Simbólico(Exácto)')  n = 0:7; %Vector de Tiempo Dicreto  xn = sin(tiempo); %Generación de Señal x[n]  yn = diff(xn); %Cálculo de la derivada y[n]=x'[n] xn = double(subs(xn,tiempo,n)); %Cambio de varible yn = double(subs(yn,tiempo,n)); %Cambio de varible subplot(2,2,4),stem(n,yn,'Color','r'),hold on,  stem(n,xn,'Color','b'),legend('y[n]=Derivada de x[n]','Señal Original x[n]'),grid on %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('x[n] y su Diferenciación - Método  Simbólico(Exácto)') |

|  |
| --- |
| Imagen que contiene Gráfico de líneas  Descripción generada automáticamente |
|  |
| **Integración** |
| **Descripción**  El comando int que permite calcular tanto integrales definidas como indefinidas con respecto a una variable simbólica var. Especificar la variable simbólica var es opcional, si no se especifica int utiliza la variable predeterminada por symvar, es decir, la única variable simbólica de la expresión expr, en caso de que expr tenga más de una variable simbólica se deberá especificar con respecto a cuál se desea integrar. Los argumentos a y b permiten evaluar la integral en intervalos definidos desde a hasta b. Finalmente, los argumentos Name y Value son opcionales y permiten especificar funcionalidades adicionales sobre cómo se debe evaluar la integral, para esta práctica no será necesario profundizar en ello.  Es importante mencionar que el comando int no admite en su argumento expr funciones definidas con vectores de tiempo, por lo cual, es necesario realizar la operación de integración de forma simbólica y luego sustituir la variable de tiempo por su equivalente vectorial. Un ejemplo similar se realizó para la operación de diferenciación. |
| **Comandos** |
| %% Integración en Tiempo Continuo y Discreto syms tiempo;  Fs = 200; %Frecuencia de Muestreo  t = 0:1/Fs:2\*pi; %Tiempo de Simulación  xt = sin(tiempo); %Generación de Señal x(t)  yt = int(xt,tiempo); %Cálculo de la integral y(t)=integral de x(t) xt = double(subs(xt,tiempo,t)); %Cambio de variable  yt = double(subs(yt,tiempo,t)); %Cambio de variable subplot(2,1,1),plot(t,yt,'r',t,xt,'b'),legend('y(t)=Integral de x(t)','Señal Original x(t)'),grid on %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('x(t) y su Integración') n = 0:7; %Vector de Tiempo Dicreto  xn = sin(tiempo); %Generación de Señal x[n]  yn = int(xn); %Cálculo de la derivada y[n]=x'[n] xn = double(subs(xn,tiempo,n)); %Cambio de varible yn = double(subs(yn,tiempo,n)); %Cambio de varible subplot(2,1,2),stem(n,yn,'Color','r'),hold on,  stem(n,xn,'Color','b'),legend('y[n]=Integral de x[n]','Señal Original x[n]'),grid on %Grafica  xlabel('Tiempo (n)'),ylabel('Amplitud'), title('x[n] y su Integración') |
| **Gráfica** |
| Gráfico  Descripción generada automáticamente con confianza media |

|  |
| --- |
| **Operaciones efectuadas sobre la variable independiente** |

|  |
| --- |
| **Escalamiento de tiempo** |
| **Descripción**  Para este caso se aplica la misma descripción que para el “Escalamiento de Amplitud” teniendo en cuenta la expresión que representa la operación de escalamiento de tiempo. En  términos matemáticos se podría decir que el escalamiento de tiempo es un cambio de variable en de *t* por *a*∗*t* en una función dada. A continuación, se presentan ejemplos para casos en tiempo continuo y discreto, con escalamientos mayores y menores que 1; debe observarse que para el caso de señales discretas con escalamiento de tiempo mayor a 1 hay pérdida de datos tal como se expresó en la fundamentación teórica de esta práctica. |
| **Comandos** |
| %% Escalamiento de Tiempo en Tiempo Continuo y Discreto A = 5; %Amplitud  Fs = 200; %Frecuencia de Muestreo f = 0.1; %Frecuencia fundamental Hz  T = 3\*(1/f); %Número de Periodos a Graficar duty = 50; %Ciclo útil  t = 0:1/Fs:T-1/Fs; %Variable t de la señal x(t)  xt = A.\*square(2\*pi\*f\*t,duty); %Generación de Señal  yt1 = A.\*square(2\*pi\*f\*2\*t,duty); %Señal escalada en el tiempo en un factor de 2 y1(t)=x(2t);  yt2 = A.\*square(2\*pi\*f\*0.5\*t,duty); %Señal escalada en el tiempo en un factor de 0.5 y2(t)=x(0.5t);  subplot(2,3,1),plot(t,xt),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (s)’),ylabel(‘Amplitud’), title(‘Señal Original x(t)’) subplot(2,3,2),plot(t,yt1),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (s)’),ylabel(‘Amplitud’), title(‘Señal Escalada en Tiempo y1(t)=x(2t)’)  subplot(2,3,3),plot(t,yt2),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (s)’),ylabel(‘Amplitud’), title(‘Señal Escalada en Tiempo y2(t)=x(2t)’)  n = 0:T; %Vector de Tiempo Dicreto  xn = A.\*square(2\*pi\*f\*n,duty); %Generación de Señal x[n]  yn1 = A.\*square(2\*pi\*f\*2\*n,duty); %Señal escalada en el tiempo en un factor de 2 y1[n]=x[2n]);  yn2 = A.\*square(2\*pi\*f\*0.5\*n,duty); %Señal escalada en el tiempo en un factor de 0.5 y2[n]=x[0.5n]);  subplot(2,3,4),stem(n,xn),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (n)’),ylabel(‘Amplitud’), title(‘y[n]=x[n]’) subplot(2,3,5),stem(n,yn1),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (n)’),ylabel(‘Amplitud’), title(‘y1[n]=x[2n]’) subplot(2,3,6),stem(n,yn2),grid on, ylim([-A-1 A+1]) %Grafica xlabel(‘Tiempo (n)’),ylabel(‘Amplitud’), title(‘y[n]=x[0.5n]’) |
| **Gráfica** |

|  |
| --- |
| Diagrama  Descripción generada automáticamente |
| **Reflexión** |
| **Descripción**  No existe algún comando especial para realizar esta operación; y podría seguirse un análisis similar al de “Escalamiento de tiempo” tomando la reflexión como un cambio de variable de  *t* por −*t* en una función dada. |
| **Comandos** |
| %% Reflexión en Tiempo Continuo y Discreto Fs = 200; %Frecuencia de muestreo  t = -5:1/Fs:5; %Variable de tiempo para x(t) n = -5:5; %Variable de tiempo para x[n]  xt = ustep(t); %Generación de Señal Tiempo Continuo yt = ustep(-t); %Reflexión de la señal y(t)=x(-t) xn = ustep(n); %Generación de Señal Tiempo Discreto yn = ustep(-n); %Reflexión de la señal y[n]=x[-n]  subplot(2,2,1),plot(t,xt),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('Señal Original x(t)') subplot(2,2,2),plot(t,yt),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y(t)=x(-t)') subplot(2,2,3),stem(n,xn),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('Señal Original x[n]') subplot(2,2,4),stem(n,yn),grid on,ylim([0 1.1]) %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y[n]=x[-n]') |
| **Gráfica** |

|  |
| --- |
| Gráfico, Gráfico de líneas, Gráfico de dispersión  Descripción generada automáticamente |
| **Corrimiento en tiempo** |
| **Descripción**  No existe algún comando especial para realizar esta operación; y podría seguirse un análisis similar al de “Escalamiento de tiempo” tomando el corrimiento de tiempo como un cambio de  variable de *t* por *t ± t*0 en una función dada. |
| **Comandos** |
| %% Corrimiento en Tiempo o Desplazamiento en Tiempo Continuo y Discreto Fs = 200; %Frecuencia de muestreo  t = -5:1/Fs:5; %Variable de tiempo para x(t) n = -5:5; %Variable de tiempo para x[n]  xt = ustep(t); %Generación de Señal Tiempo Continuo  yt1 = ustep(t+2); %Desplazamiento de la señal y1(t)=x(t+2) yt2 = ustep(t-2); %Desplazamiento de la señal y2(t)=x(t-2) xn = ustep(n); %Generación de Señal Tiempo Discreto  yn1 = ustep(n+2); %Desplazamiento de la señal y1[n]=x[n+2] yn2 = ustep(n-2); %Desplazamiento de la señal y2[n]=x[n-2] subplot(2,3,1),plot(t,xt),grid on,ylim([0 1.1]) %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('Señal Original x(t)') subplot(2,3,2),plot(t,yt1),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y(t)=x(t+2)') subplot(2,3,3),plot(t,yt2),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y(t)=x(t-2)')  subplot(2,3,4),stem(n,xn),grid on,ylim([0 1.1]) %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('Señal Original x[n]') subplot(2,3,5),stem(n,yn1),grid on,ylim([0 1.1]) %Grafica xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y[n]=x[n+2]') subplot(2,3,6),stem(n,yn2),grid on,ylim([0 1.1]) %Grafica  xlabel('Tiempo (s)'),ylabel('Amplitud'), title('y[n]=x[n-2]') |
| **Gráfica**  Diagrama  Descripción generada automáticamente con confianza media |

# PROCEDIMIENTOS

# De la aplicación generada en la Práctica No.2, el alumno agregará nuevos componentes a su GUI, es decir a su generador de señales, para que el usuario pueda seleccionar si desea realizar operaciones sobre la variable dependiente o independiente sobre alguna(s) señal(es) generada(s).

# Si el alumno selecciona la opción de operación sobre la variable independiente, se desplegará un componente (p.e. menú), donde se puedan realizar las operaciones de escalamiento de tiempo, reflexión y corrimiento en tiempo, si el usuario selecciona alguna de estas tres operaciones se efectuará la operación sobre la señal generada y se podrá visualizar gráficamente el resultado de realizar la operación sobre la señal en cuestión.

# Si el usuario seleccionara que requiere realizar operaciones sobre la variable dependiente, el alumno implementará los procedimientos necesarios para que se puedan implementar las 5 operaciones que se pueden realizar según la selección del usuario: Escalamiento de amplitud, suma, multiplicación, diferenciación e integración. Ejemplo: Si se requiere realizar la suma de dos señales, la aplicación deber requerir la generación de dos señales con las características que el usuario indique, posteriormente realizar la operación y mostrar el resultado en un gráfico.

# El alumno también deberá implementar los componentes necesarios para que el usuario pueda realizar operaciones simultaneas sobre la variable independiente.

# El alumno reportará su metodología de implementación, así como las bases teóricas necesarias para su desarrollo, los resultados obtenidos, así como sus conclusiones y discusiones del trabajo realizado.