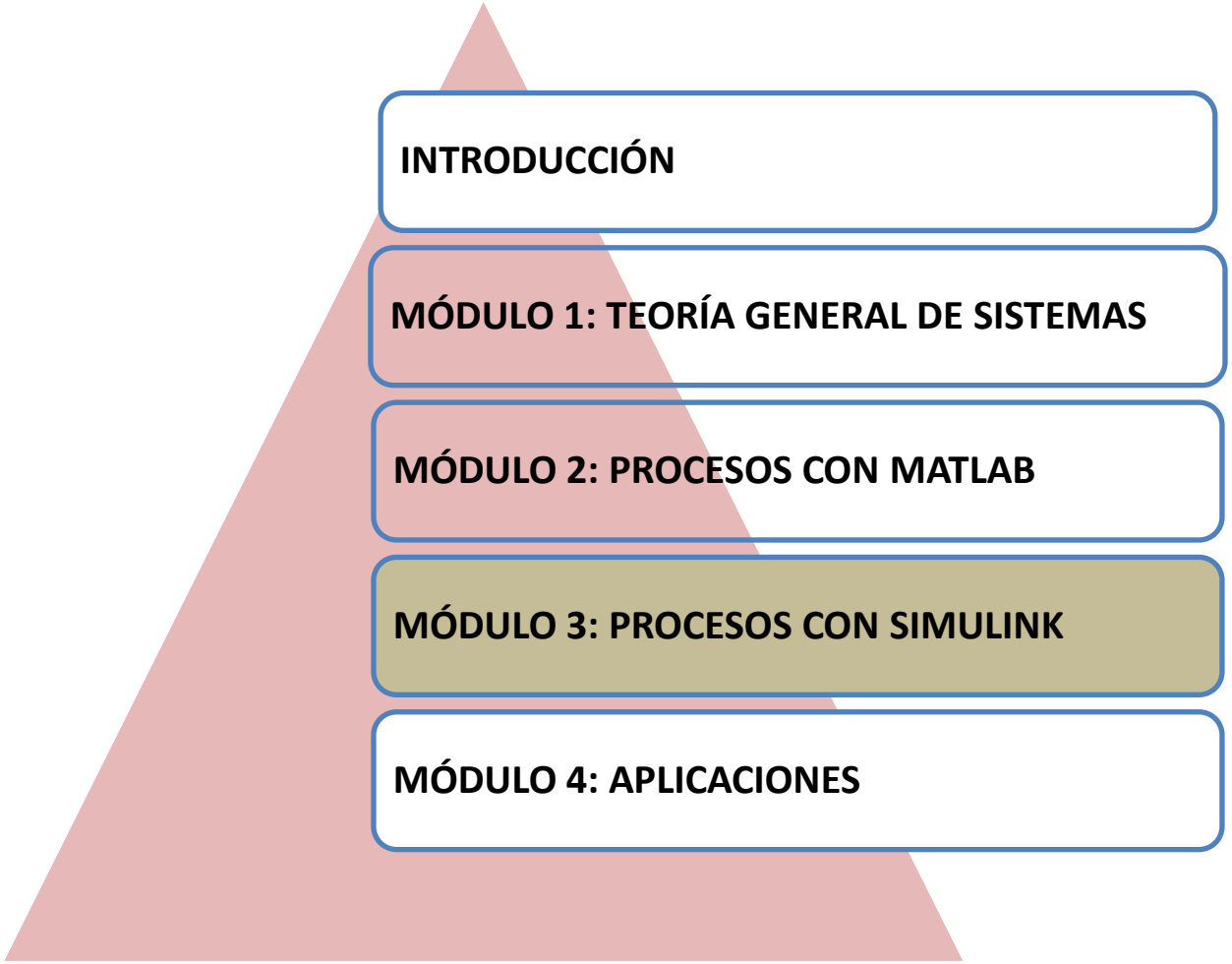
MAESTRÍA EN INGENIERÍA Y GESTIÓN AMBIENTAL

CURSO: SIMULACIÓN DE SISTEMAS AMBIENTALES

PROFESOR: M.I. Jorge Antonio Polanía P.

CONTENIDO


INTRODUCCIÓN
MÓDULO 1: TEORÍA GENERAL DE SISTEMAS
MÓDULO 2: PROCESOS CON MATLAB
MÓDULO 3: PROCESOS CON SIMULINK
MÓDULO 4: APLICACIONES

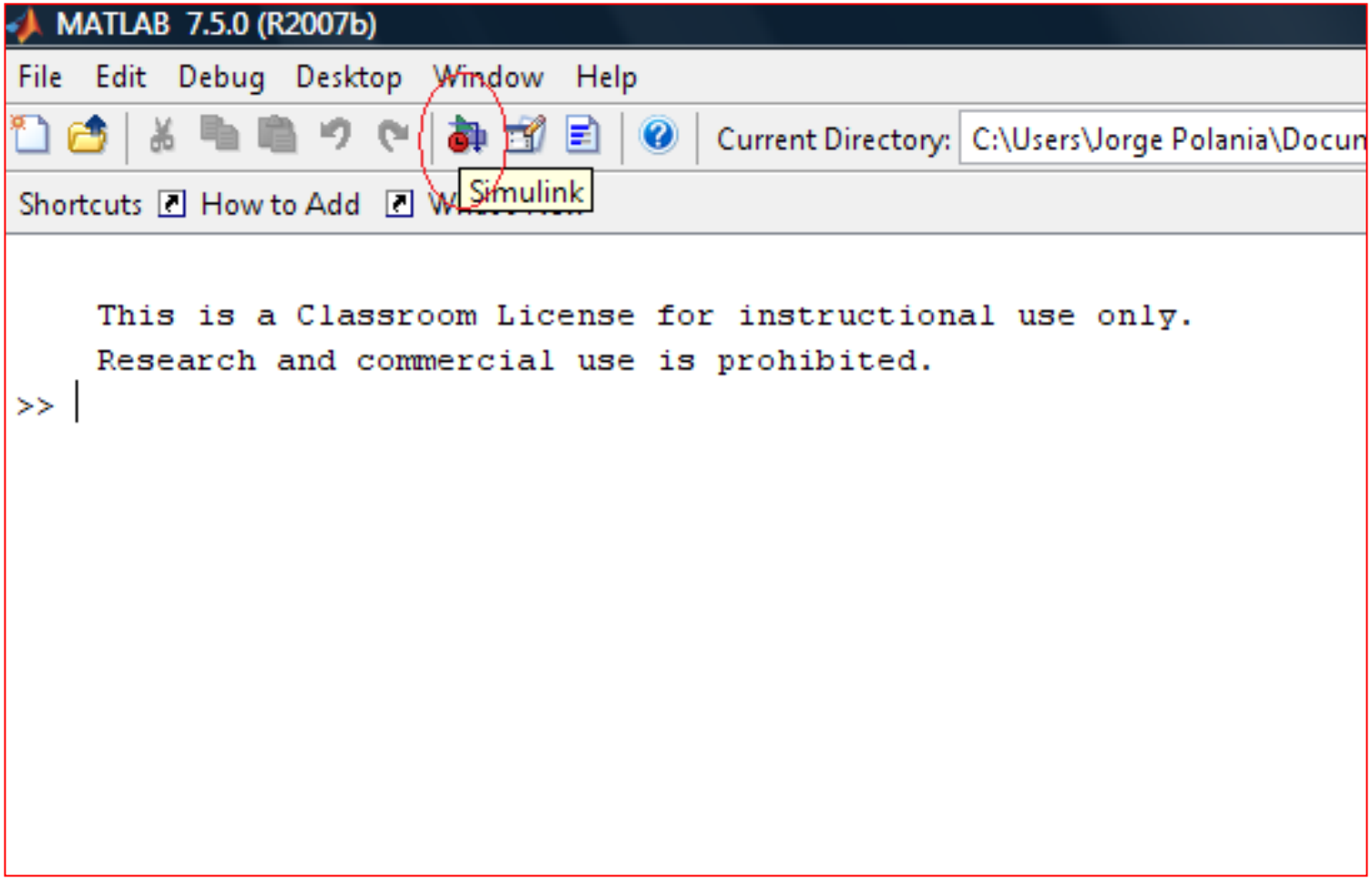
MÓDULO 3:
PROCESOS CON SIMULINK

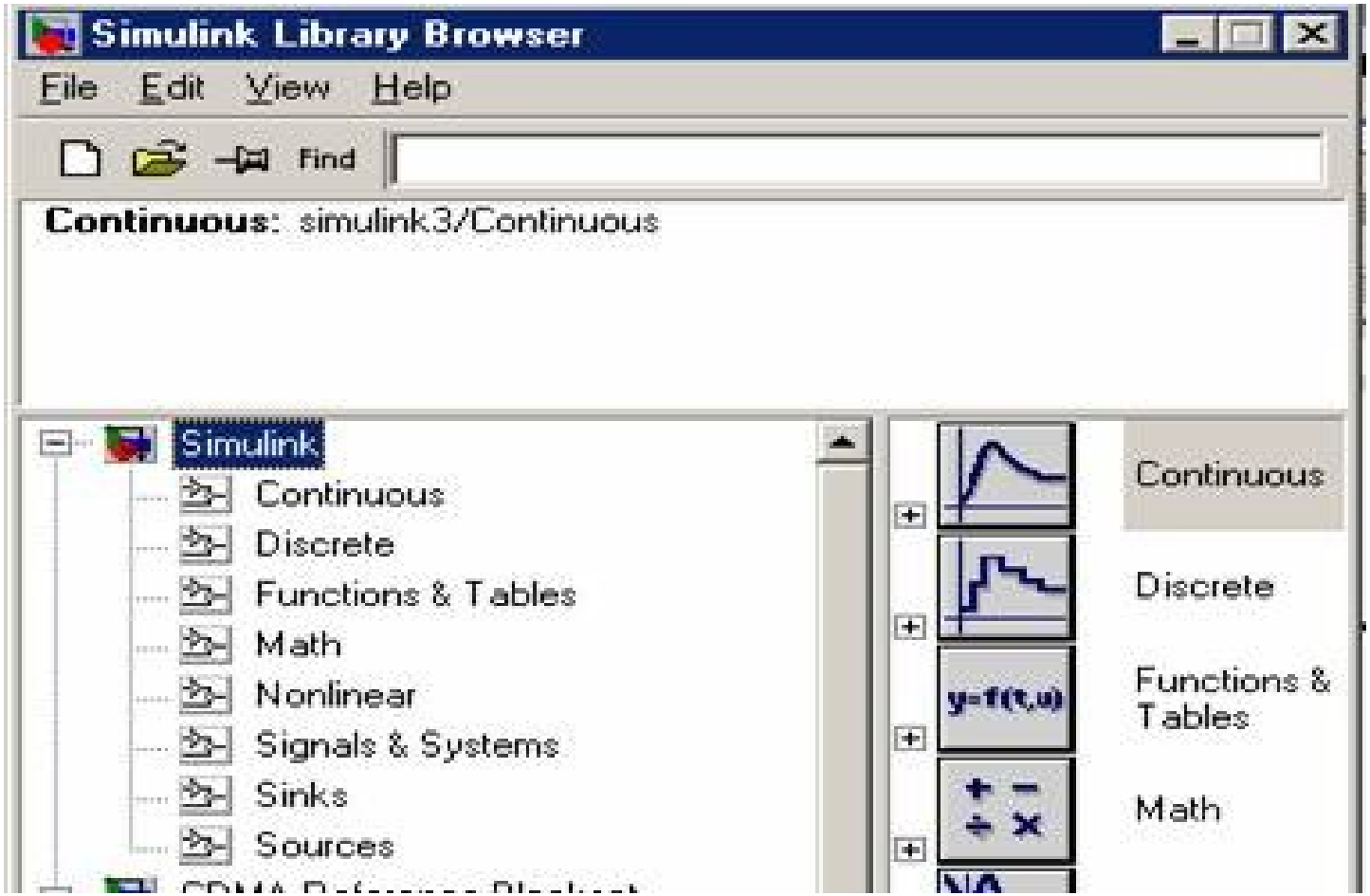
1. ELEMENTOS BÁSICOS
2. MODELADO DE SISTEMAS
3. FUNCIONES Y BUCLES
4. SIMULINK MATLAB EN PROCESOS

1. ELEMENTOS BÁSICOS

1.1 INTRODUCCIÓN

Simulink es una extensión de Matlab utilizado en el modelamiento y simulación de sistemas. Para arrancar Simulink se puede hacer desde el prompt de Matlab digitando el comando >>Simulink o utilizando el ícono  . Se abre la ventana Simulink Library Browser como se indica abajo y se puede diagramar un nuevo modelo activando el botón New Model , o sea el icono  o de File →New→Model

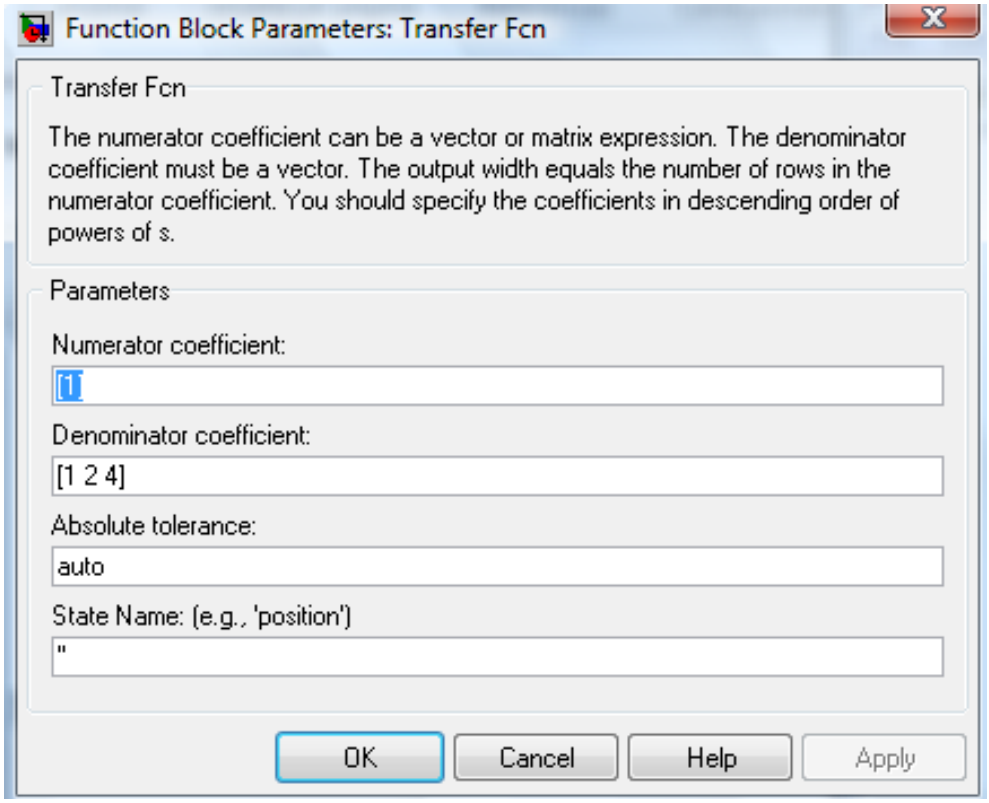

MATLAB 7.5.0 (R2007b)
File Edit Debug Desktop Window Help
Current Directory: C:\Users\Jorge Polania\Docun
Simulink
Shortcuts How to Add
This is a Classroom License for instructional use only.
Research and commercial use is prohibited.
>>

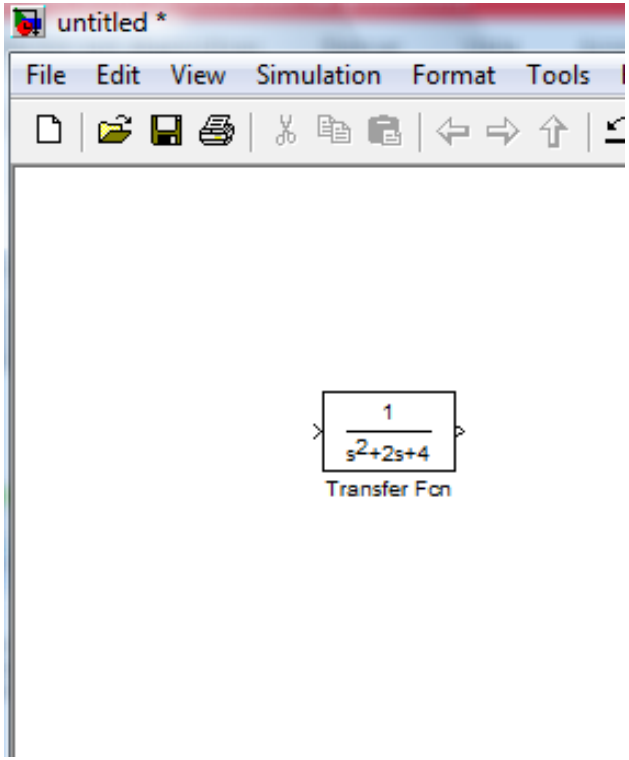


Un modelo es un conjunto de bloques que representa un sistema y como archivo tiene extensión *.mdl

1.2 BLOQUES BÁSICOS

Los elementos básicos son líneas y bloques. Los bloques están agrupados en: Sources, Links, Discrete, Continuos, Math, etc., tal como aparecen en la ventana anterior. Cada bloque tiene entradas y salida para realizar su interconexión. Por ejemplo, haga clic en continuos y luego clic en Transfer Fcn y arrastre el bloque a la ventana en blanco. Si quiere modificar la función de transferencia del bloque haga doble clic en él y digite los coeficientes del numerador y denominador en la nueva ventana que aparece. Para la función $1/(s^2 + 2s + 4)$ quedaría:

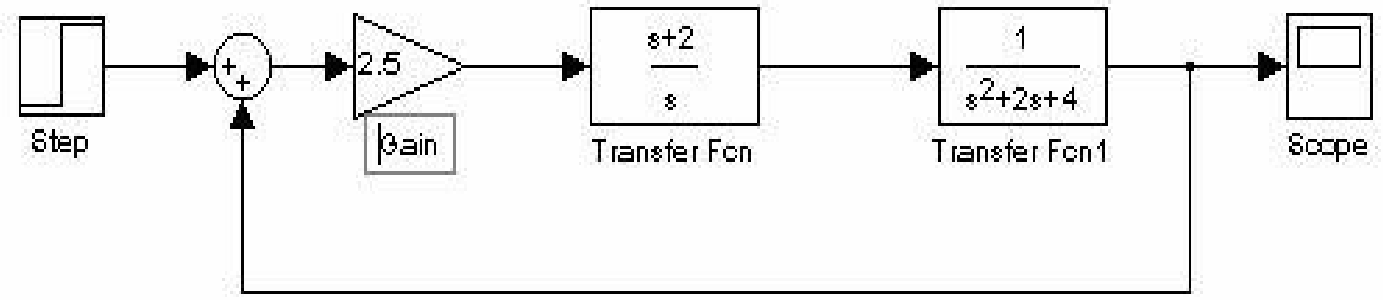

Function Block Parameters: Transfer Fcn
Transfer Fcn
The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.
Parameters
Numerator coefficient:
[1]
Denominator coefficient:
[1 2 4]
Absolute tolerance:
auto
State Name: (e.g., 'position')
''
OK
Cancel
Help
Apply


untitled *
File
Edit
View
Simulation
Format
Tools
1
s2+2s+4
Transfer Fcn

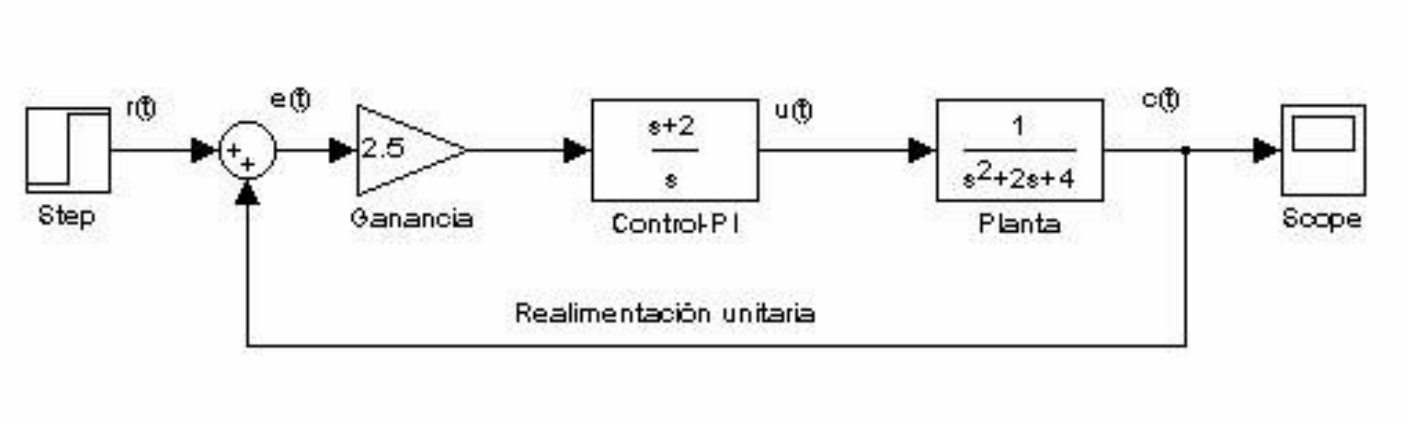
1.3 SISTEMA DE BLOQUES

Realizar el diagrama en bloques del siguiente sistema :

Lo primero es arrastrar los bloques a la página en blanco de forma que, Step es la función paso o escalón que se obtiene de Sources, Scope es el osciloscopio que se obtiene de Sinks, Transfer Fcn se obtiene de Continuos, Sum y Gain se obtienen de Math. Modifique los bloques dando doble clic sobre cada uno de ellos para cambiar sus parámetros o valores e interconéctelos.



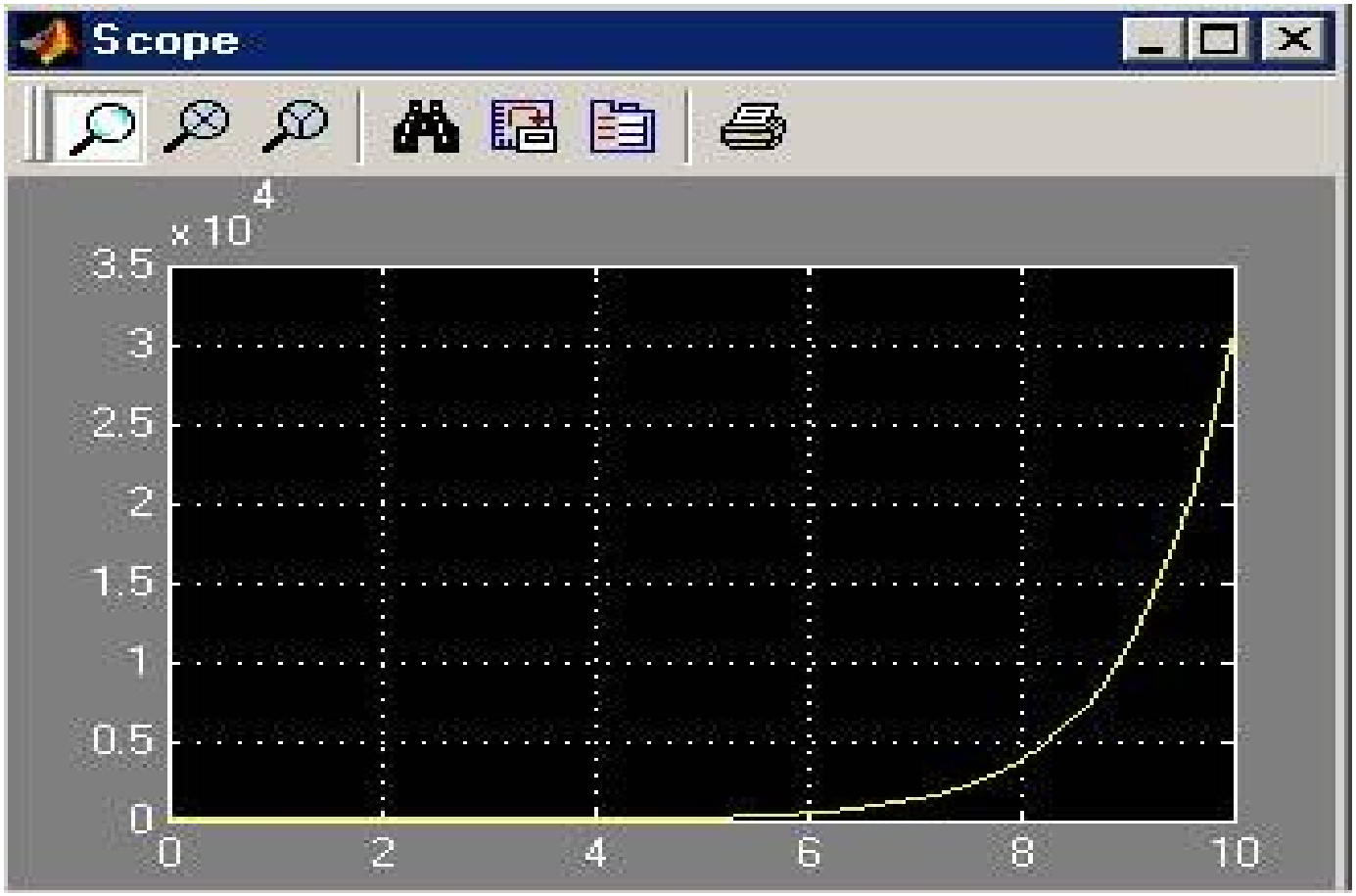
Lo segundo es cambiar los nombres a los bloques y asignar las variables o señales haciendo doble clic en el lugar en que se van a colocar y salvar el modelo especificándole un nombre, por ejemplo ejem1.mdl



Por último se debe simular el sistema. Para ello se configura la señal de entrada, en este caso la función paso (escalón). Dar doble clic y asignar los siguientes parámetros: Step time=0, Inicial value=0, Final value=1, Sample time=0.

Para simular el sistema de control se escoge del menú Simulation -> Start o el icono  .y luego se hace doble clic en Scope para ver su respuesta o salida del sistema. Para observar además la entrada se puede colocar otro Scope a la salida de Step y se puede probar para varios pasos variando su amplitud, tiempo de inicio y tiempo de iniciación del paso.

Para observar mejor la respuesta se usa el botón Autoscale (binoculares ) de la ventana del Scope. Si quiere observar mejor la respuesta o parte de ella se pueden cambiar los parámetros de simulación, Simulation-> Simulation parameters. Por ejemplo cambiar el Start time y el Stop time y correr nuevamente la simulación.


Scope
x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10

1.4 MODELANDO UN SISTEMA

A) ECUACIONES DINÁMICAS

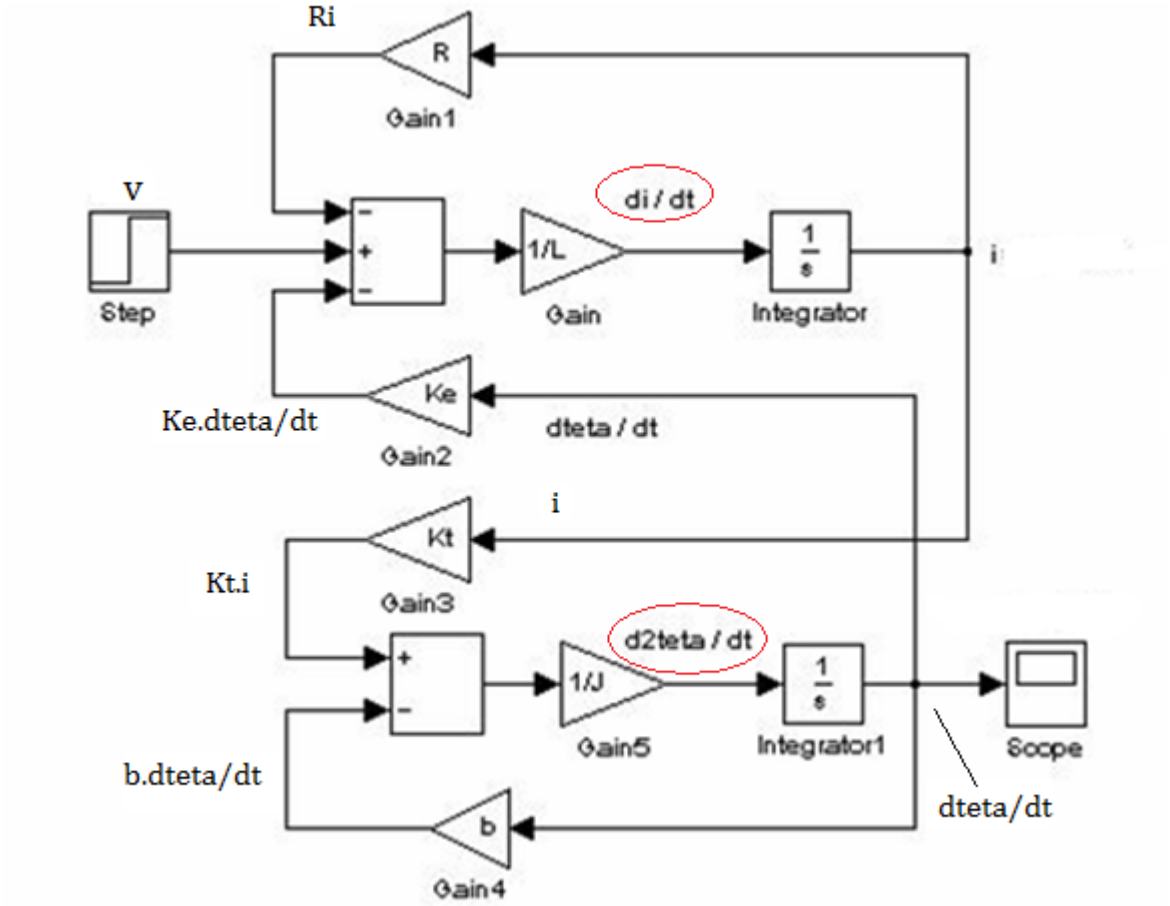
(1) $$J\frac{d^2\theta}{dt^2} = \frac{1}{J}\left(k_t i - b\frac{d\theta}{dt}\right)$$

(2) $$\frac{di}{dt} = \frac{1}{L}\left(-Ri + V - k_e\frac{d\theta}{dt}\right)$$

Los parámetros físicos
tienen los siguientes valores:

J = 0.01
b = 0.1
Ke = Kt = 0.01
R = 1
L = 0.5

B) MODELAMIENTO DEL SISTEMA



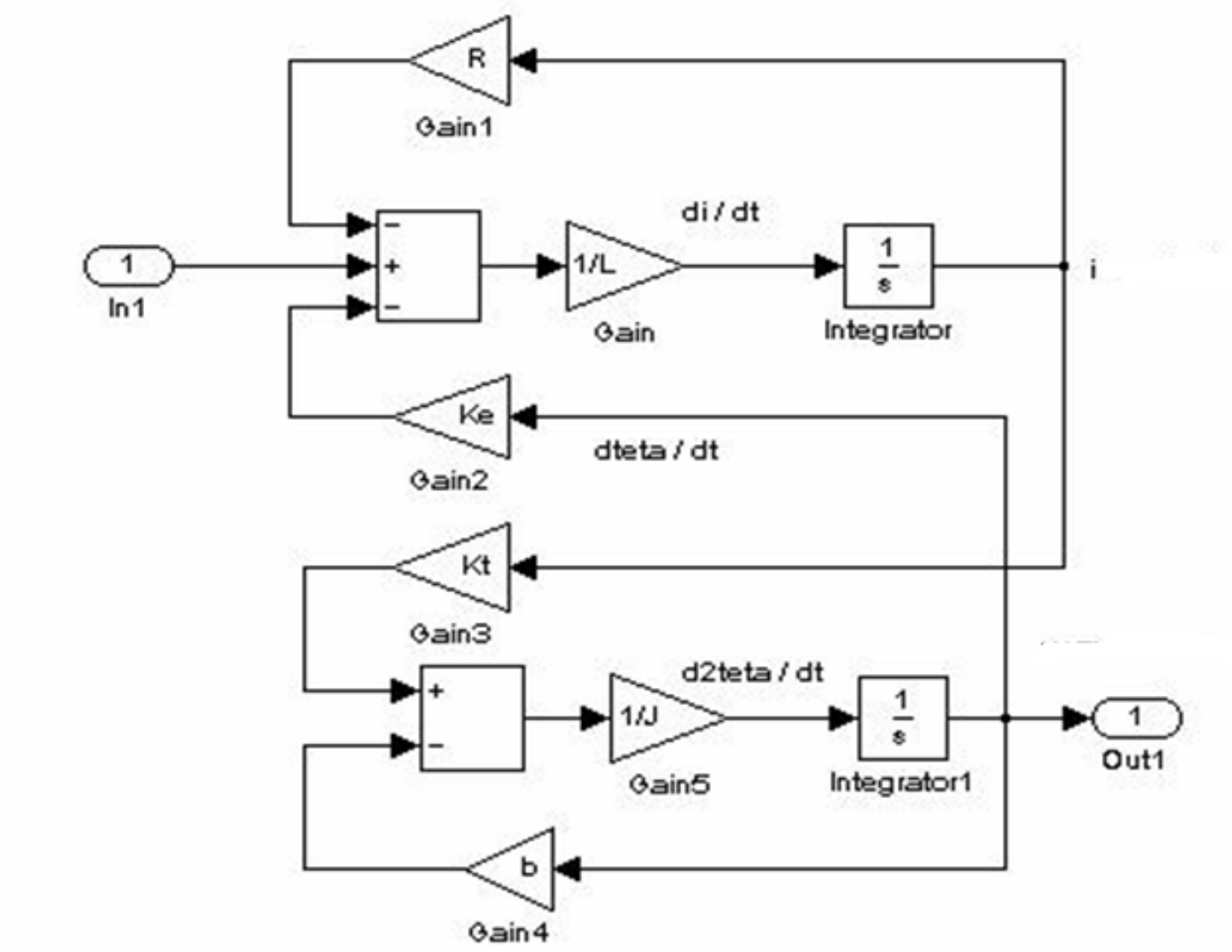
C) EXTRAER MODELO DEL SISTEMA

Para obtener la función de transferencia del sistema primero se trasladan los parámetros al modelo creando un archivo en Matlab (*.m) de la siguiente forma:

```
% valores de los parámetros del motor
J = 0.01;
b = 0.1;
Ke = 0.01;
Kt = 0.01;
R = 1;
L = 0.5;
```

Se ejecuta este archivo y se simula el modelo para una entrada de paso unitario de valor V = 0.01, con los siguientes parámetros de simulación: Stop time = 3 sg. Arranque la simulación y observe la salida.

Como segundo paso se debe obtener el modelo del sistema. Para esto, borre el bloque Scope y cámbielo por Out obtenido de la librería de Signals&Systems. Haga lo mismo para Step cambiándolo por In de esta misma librería. Los bloques In y Out definen la entrada y salida del sistema que le gustaría extraer. Salve este modelo. El sistema quedará así:


R
Gain1
di / dt
1
In1
1/L
Gain
1
s
Integrator
i
Ke
dteta / dt
Gain2
Kt
Gain3
d2teta / dt
1/J
Gain5
1
s
Integrator1
1
Out1
b
Gain4

Como tercero y último paso, después de desarrollado el modelo y salvarlo por ejemplo con el nombre sistema.mdl se ejecutan los siguientes comandos:

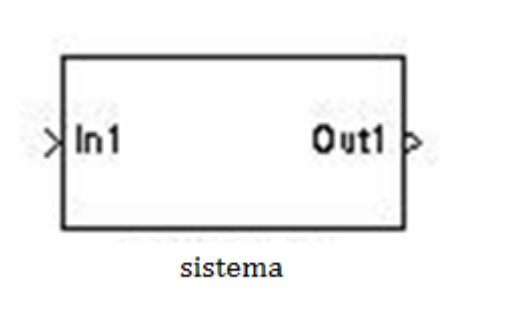
```
% obtener el modelo del sistema
[num, den] = linmod('sistema')
Gps = tf(num, den)
```

$$Gp(s) = \frac{2}{s^2 + 12s + 20.02}$$

FUNCIÓN DE TRANSFERENCIA DEL SISTEMA

D) CREANDO UN SUBSISTEMA

Abra una nueva ventana y arrastre de la librería Signals&Systems el bloque SubSystem , haga doble clic en este bloque, abra el modelo sistema.mdl (el que tiene In y Out como terminales) cópielo y péguelo en la nueva ventana de subsistema anterior. Cierre ventanas y aparece una nueva con el bloque con los terminales del subsistema creado. Déle el nombre sistema. Si a este bloque de subsistema se le da doble clic aparece el modelo completo diseñado anteriormente. Otra forma es señalar los bloques de interés, ir a menú Edit --> create Subsytem



3. FUNCIONES Y BUCLES

Matlab permite la ejecución de conjuntos de comandos escritos secuencialmente en la ventana de edición y que son almacenados en un archivo ***nombre.m***. Para ejecutar un archivo basta con teclear su nombre (sin extensión) en modo interactivo en la ventana de comando y pulsar ***enter***. En el archivo .m se pueden introducir textos explicativos comenzando la línea con el símbolo **%**.

3.1 FUNCTION

El comando ***function*** permite la definición de funciones con la siguiente sintaxis:

```
function  parámetros_salida = nombre_función(parámetros_entrada)
                    cuerpo de la función
```

Una vez definida la función se guarda en un archivo *nombre_función.m* para su posterior utilización. Cuando los parámetros de salida son más que uno se sitúan entre corchetes separados por comas. Si los parámetros de entrada son más que uno se separan por comas.

Ejemplo:

Solución de una ecuación de segundo grado: $ax^2+bx+c=0$

```
function [x1,x2] = cuadratica(a,b,c)
%Esta funcion calcula las raices de una ecuacion cuadratica
radical = sqrt(b^2-4*a*c);
x1= (-b+radical) / (2*a);
x2= (-b-radical) / (2*a);
```

Se guarda el archivo como: cuadratica.m

Calcular las raíces de: $x^2+2x+3=0$

```
>> [x1,x2]=cuadratica(1,2,3)

x1 =
   -1.0000 + 1.4142i
 x2 =
   -1.0000 - 1.4142i
```

3.2 GLOBAL

Normalmente cada función de Matlab definida como un archivo–M contiene sus variables como variables locales, esto es, su efecto es al interior de este archivo independientemente de otros archivos. Es posible definir otras variables que tenga efecto en otros archivos-M con variables globales usando el comando ***global*** con la siguiente sintaxis:

global x y z ... define las variables x, y, z,....como globales

3.3 FOR

Permite ejecutar de forma repetitiva un comando o grupo de comandos varias veces. Tiene la siguiente sintaxis:

caso1:

```
for  i= 1:n
      comandos
end
```

caso2:

```
for i=n:-0.2:1
            comandos
end
```

caso 3:

```
for i=1:m
            for i=n
            comandos
            end
end
```

Ejemplo:

Sumar los enteros impares de 1 a 100

```
suma=0;
for i=1:2:100
   suma=suma+i;
end
disp('El resultado es:  ')
display(suma)
```

```
El resultado es:

suma =

      2500
```

Ejemplo:

Calcular la suma de los elementos de una matriz

```
M=[1 2 -2 4; 0 -3 1 0; 2 -1 4 3];
% numero de filas
m=length(M(:,1));
% numero de columnas
n=length(M(1,:));
suma=0;
for i=1:m
   for j=1:n
      suma=M(i,j)+suma;
   end
end
display(suma)
```

3.4 WHILE

Mientras se ejecuta una condición se ejecutan los comandos o sentencias.

```
while    condición
         sentencias
end
```

Ejemplo:

Calcular los volúmenes de las esferas para radio igual a: 1,2,3,4,5

```
r=0;
while r<5
    r=r+1;
    vol=(4/3)*pi*r^3;
    display(r)
    display(vol);
end
```

Ejemplo:

Genere una tabla que suministre los inversos, cuadrados y raíces cuadradas del 1 al 5

```
i=0;
while i<5
    i=i+1;
    A(i)=i;
    B(i)=1/i;
    C(i)=i^2;
    D(i)=sqrt(i);
end
E=[A',B',C',D']
```

E =

1.0000	1.0000	1.0000	1.0000
2.0000	0.5000	4.0000	1.4142
3.0000	0.3333	9.0000	1.7321
4.0000	0.2500	16.0000	2.0000
5.0000	0.2000	25.0000	2.2361

3.5 IF

caso 1:

```
if condición
        sentencias
end
```

caso 2:

```
if condición1
        bloque1
elseif condición2
        bloque2
else % sino cumple condiciones anteriores
        bloque4
end
```

Operadores de relación:

>	Mayor que
>=	Mayor o igual que
<	Menor que
<=	Menor o igual que
==	Igual
~=	No es igual a

Operadores lógicos:

&	AND
|	OR
~	NOT
xor	XOR

Ejemplo:

```
calif = input('Dame la calificacion:  ');
if  calif >= 3.0
    disp(' ')
    disp('Aprobado')
end
if calif < 3.0
    disp(' ')
    disp('Desaprobado')
end
```

Ejemplo:

El precio del vino está condicionado a la cantidad requerida. Hasta 5 botellas el precio unitario es de $10.000, de 6 a 12 botellas el precio es de $12.000, y a partir de 13 a $15.000. Elaborar un programa, que pegunta cuántas botellas se desean, indique el precio unitario y el total del gasto.

```
c=input('¿Cuántas botellas quiere? ');
if c<5
    Pu=10000;
    Pt=Pu*c;
elseif c<=12
    Pu=12000;
    Pt=Pu*c;
else
    Pu=15000;
    Pt=Pu*c;
end
disp('Precio unitario: ')
Pu
disp('Precio total: ')
Pt
```

3.6 SWITCH

Se evalúa una expresión y se compara con las expresiones en *case*. Se ejecuta el bloque que corresponda con ese resultado. Si ninguno es igual se ejecuta el bloque de *otherwise.*

```
switch expresión
      case expresión1
            bloque1
            case expresión2
                  bloque2
            .....................
otherwise
            bloque3
```

3.7 INPUT

Permite imprimir un mensaje y recuperar como valor el resultado de una expresión tecleada por el usuario.

n = input (‘Teclee el polinomio’)

9.6.8 DISP

Permite imprimir un mensaje en pantalla.

disp(‘Universidad de Colombia’)

EJEMPLO: CONTAMINACIÓN DE UN RÍO

Un río ha sido contaminado aguas arriba. La concentración (cantidad de contaminante / volumen) decaerá y se dispersará corriente abajo. El problema es predecir la concentración del contaminante en cualquier punto teniendo en cuenta el tiempo y el espacio.

El modelo más simple para un contaminante químico se basa en descomposición química. Si d es la velocidad de descomposición química:

MODELO DINÁMICO

El modelo en función del tiempo es,

$$u^{k+1} = u^k - \Delta t\, d\, u^k, \quad \Delta t = T/Kmax,$$

y para la estabilidad se requiere $1 - \Delta t\, d > 0$.

El modelo debe tener en cuenta además la posición del contaminante porque está en una corriente. Se asume entonces que la concentración dependerá tanto del espacio como del tiempo.

Discretizando tanto el espacio y el tiempo e igualando la concentración a:

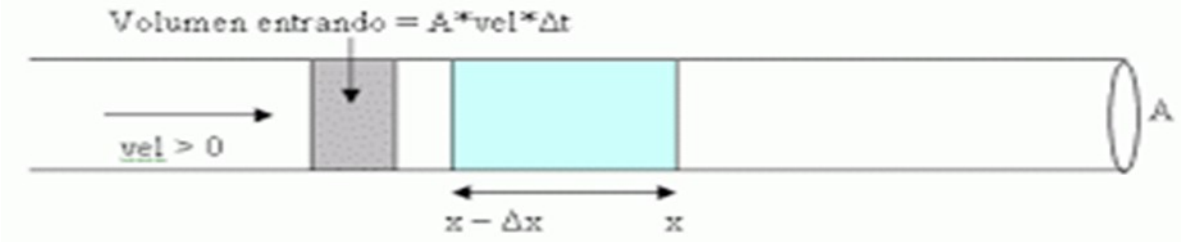
$u(i\Delta x, k\Delta t)$

donde $\Delta t = T/kmax$, $\Delta x = L/n$, L es la longitud de la corriente, n es el número de puntos

El modelo tendrá la forma general:

cambio en cantidad ≈ (cantidad entrando río arriba) - (cantidad saliendo rió abajo) - (cantidad descomponiéndose en un intervalo de tiempo).

Esto es bosquejado en la figura debajo dónde el paso de tiempo no ha sido indicado



Haciendo A como el área de sección transversal de la corriente. La cantidad entrando en el lado izquierdo del volumen A Δx es,

$A\,\Delta t\ vel\ u_{i-1}^{k}$

La cantidad saliendo en el lado derecho del volumen A Δx es,

$-A\,\Delta t\ vel\ u_{i}^{k}$

Luego, el cambio en la cantidad a partir de la velocidad de la corriente es,

$A\,\Delta t\ vel\ u_{i-1}^{k} - A\,\Delta t\ vel\ u_{i}^{k}$

La cantidad de contaminante en el volumen AΔx en el tiempo kΔt es
$A\,\Delta x\ u_{i}^{k}$

La cantidad de contaminante que se está descomponiendo, es

$A\,\Delta x\,\Delta t\,dec\,u_i^{k,}$ (dec es la velocidad de descomposición)

El cambio durante el intervalo de tiempo en la cantidad de contaminante en el pequeño volumen $A\,\Delta x$ es:

$$A\,\Delta x\,u_i^{k+1} - A\,\Delta x\,u_i^{k} = A\,\Delta t\,vel\,u_{i-1}^{k} - A\,\Delta t\,vel\,u_i^{k} - A\,\Delta x\,\Delta t\,dec\,u_i^{k}$$

Ahora, dividiendo por $A\,\Delta x$ y resolviendo explícitamente para u_i^{k+1}

$$\mathbf{u_i^{k+1} = vel\,(\Delta t/\Delta x)u_{i-1}^{k} + (1 - vel\,(\Delta t/\Delta x) - \Delta t\,dec)\,u_i^{k}} \qquad (1)$$

donde:
$i = 1, \ldots, n-1$
$k = 0, \ldots, kmax - 1$

u_i^0 = dado para i = 1, …, n-1 (2)

u_i^k = dado para k= 1, …, kmax (3)

La ecuación (2) es la concentración inicial, y (3) es la concentración lejos río arriba.

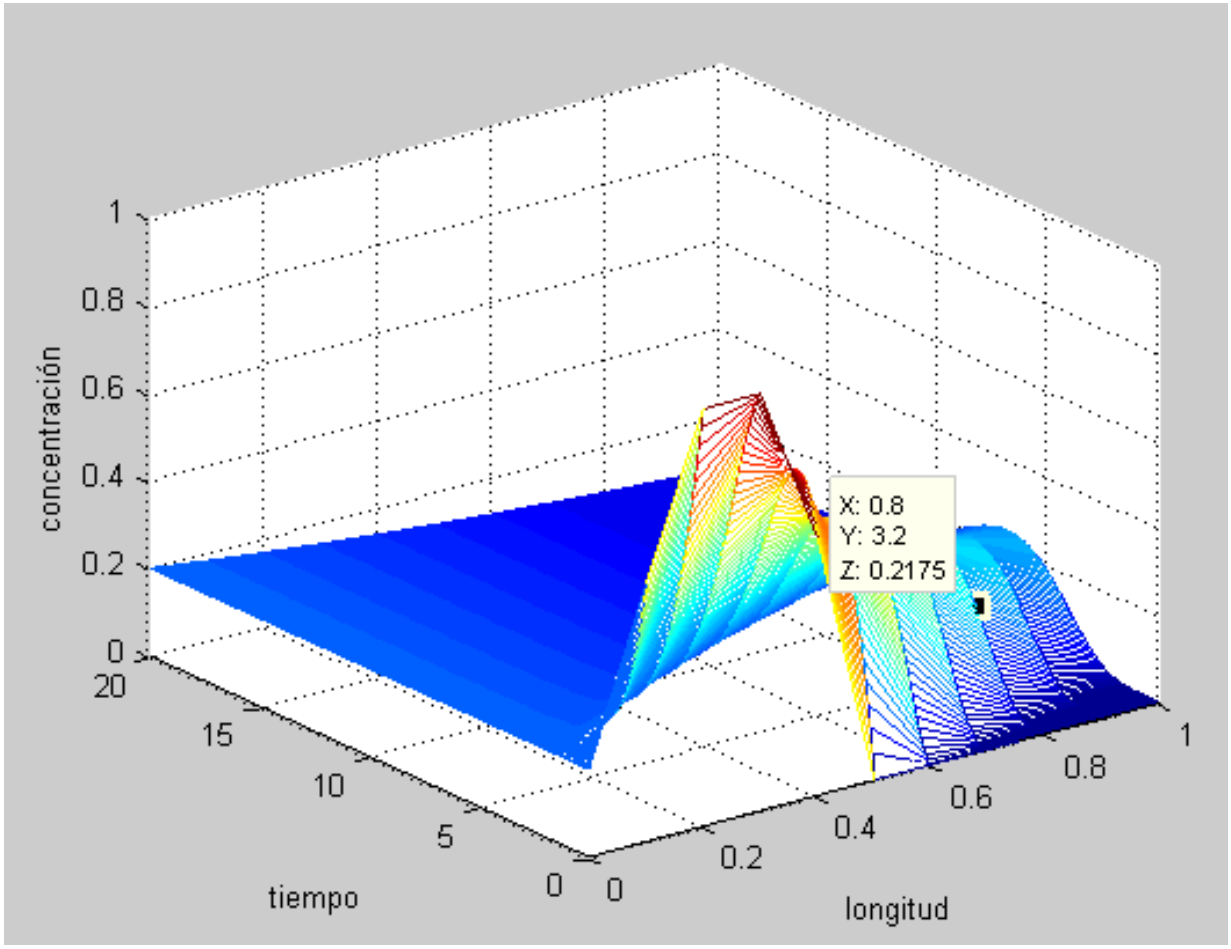
Para calcular todos los u_i^{k+1},lo cual de ahora en adelante se denotara por u(i,k+1), se debe usar un lazo anidado donde el lazo i (espacio) está adentro y el lazo k (tiempo) es el lazo exterior. En este modelo de flujo u(i,k+1) depende directamente de dos valores previamente calculados u(i-1,k) y u(i,k).

PROGRAMA EN MATLAB

```
%CONTAMINACIÓN DE UN RÍO
% Flujo en una corriente
clear;
% Longitud de la corriente
L = 1.0;
% Duración de Tiempo
T = 20;
K = 200;
dt = T/K;
n = 10;
dx = L/n;
vel = 0.1;     %velocidad del rio
decay = 0.1;   %velocidad de descomposic
```

```
% Concentración Inicial
for i = 1:n+1
x(i) =(i-1)*dx;
u(i,1) =(i<=(n/2+1))*sin(pi*x(i)*2)+(i>(n/2+1))*0;
end
% Concentración Río arriba
for k=1:K+1
time(k) = (k-1)*dt;
u(1,k) = -sin(pi*vel*0)+.2;
end
```

```
% Lazo de Tiempo
for k=1:K
% Lazo de Espacio
for i=2:n+1;
u(i,k+1) =(1 - vel*dt/dx -decay*dt)*u(i,k) + vel*dt/dx*u(i-1,k);
end
end
mesh(x,time,u')
xlabel('longitud')
ylabel('tiempo')
zlabel('concentración')
```

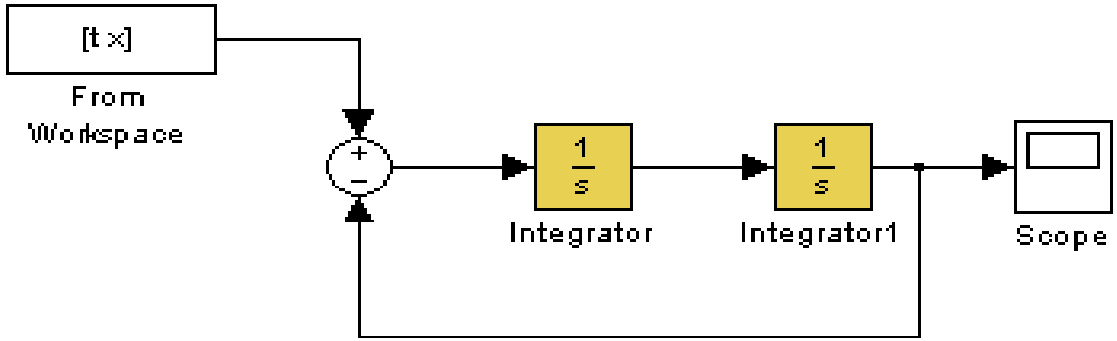

concentración
1
0.8
0.6
0.4
0.2
0
20
15
10
5
0
tiempo
0
0.2
0.4
0.6
0.8
1
longitud
X: 0.8
Y: 3.2
Z: 0.2175

4. SIMULINK - MATLAB

4.1 DE MATLAB A SIMULINK

Para utilizar señales de Matlab a Simulink de la librería *Sources* se utiliza el bloque *From Workspace*.

Ejemplo: Resolver la ecuación $y'' + y = e^{t}$, $y'(0) = 0$, $y(0) = 3$



En workspace
t=[0:0.001:0.999]
t=t'
x=exp(t)

El vector [t x] se ejecuta en Matlab en el workspace de la siguiente forma:

```
>> t = 0:0.001:0.999;
>> t = t';
>> x = exp(t)
```

Al ejecutarse Simulink toma los datos entregados por Matlab. No olvidar colocar condición inicial y(0) = 3 en el *integrador.*

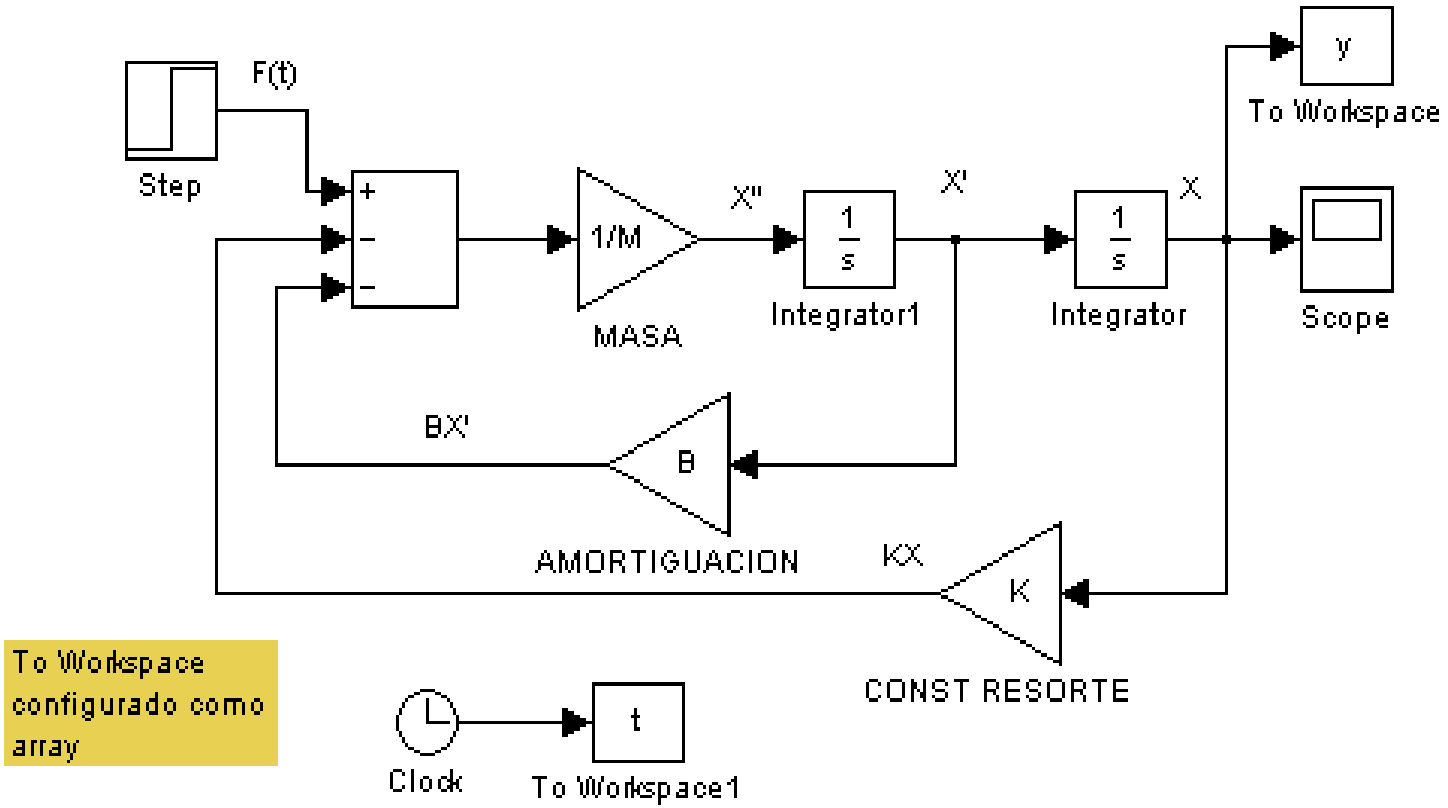
DE SIMULINK A MATLAB

Para enviar datos de Simulink a Matlab se utiliza de la librería Sinks el bloque To Workspace.

EJEMPLO:

Resolver la ecuación: $f(t) = Mx'' + Bx' + Kx$, M=1, B=1, K= 10, F(t) = 5

x”=1/M[f(t)-Bx’-kx]



En Matlab:

>> plot(t,y)

4.2 SISTEMA HIDRÁULICO

Para el siguiente problema hallar la variación de h si el caudal normal Q es de 10 lit/min y en t=5 seg se aplica una perturbación de 2 lit/min. El valor de K=10, A= 2 m^2.

$$A\frac{dh}{dt} = q(t) - K\sqrt{h}$$

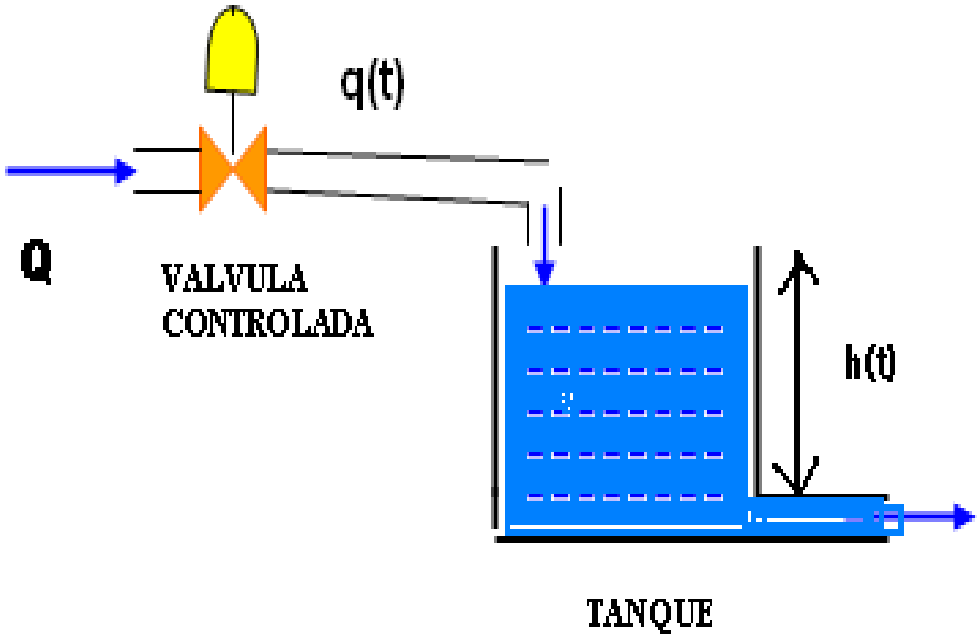
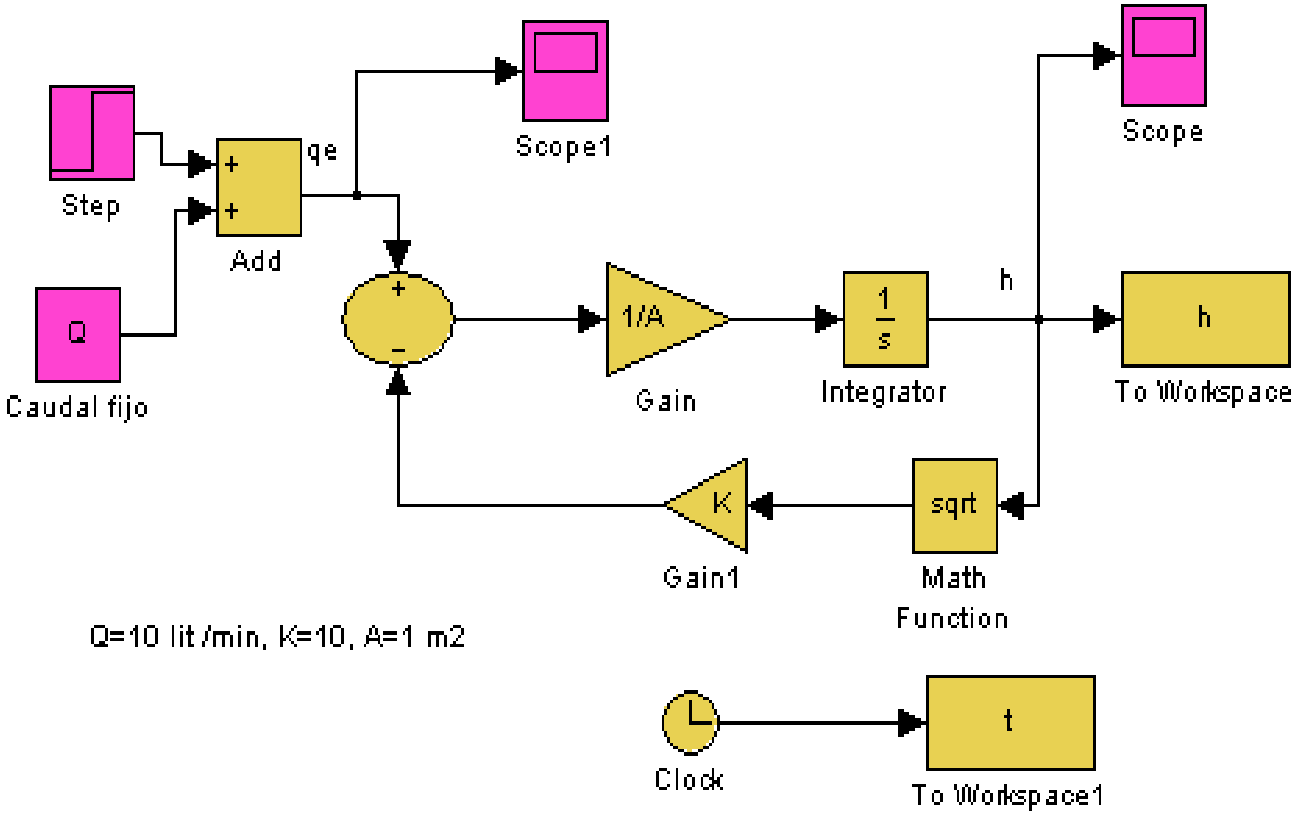
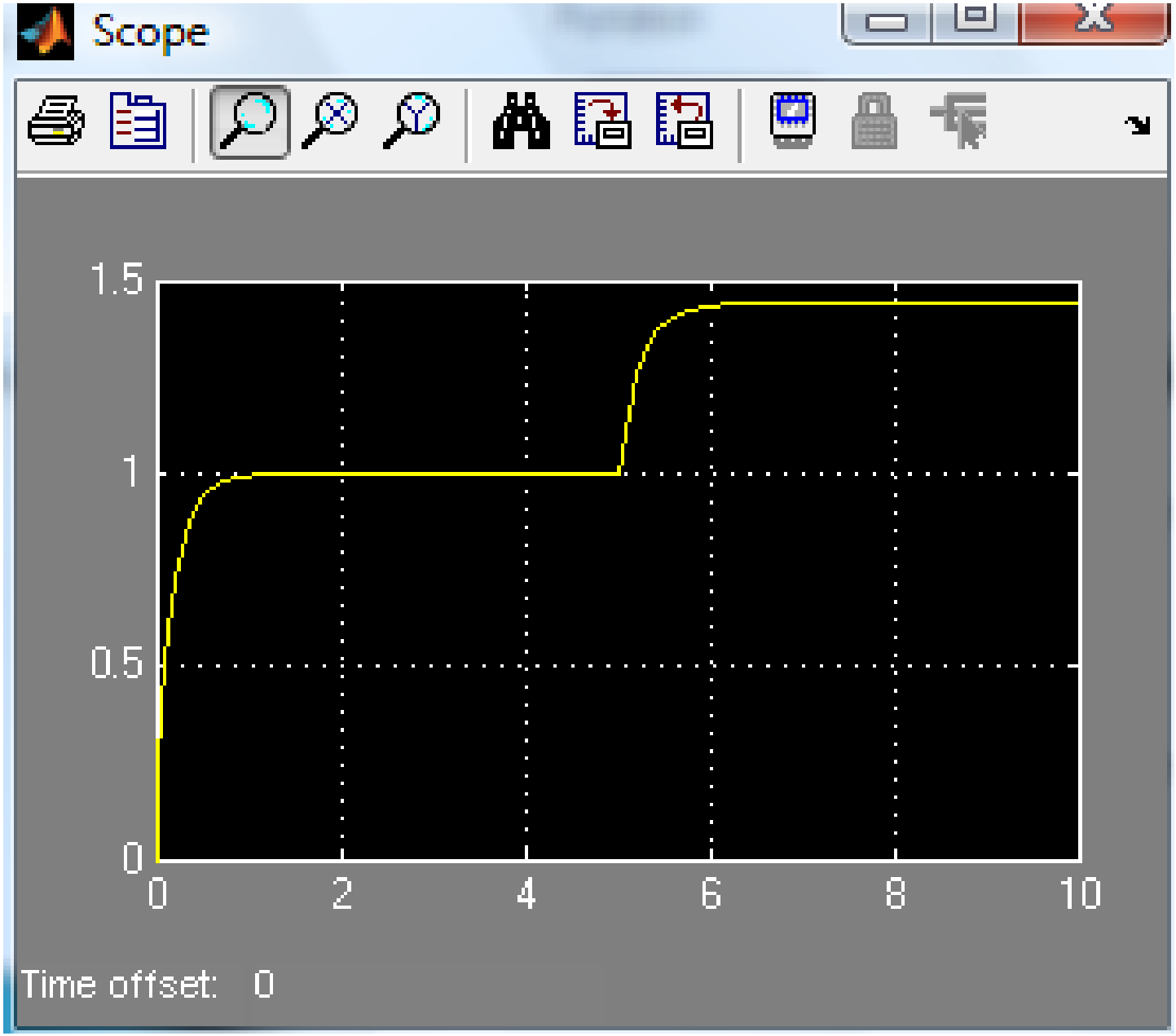


Diagrama Simulink:




Scope
1.5
1
0.5
0
0
2
4
6
8
10
Time offset: 0