

<https://www.halvorsen.blog>



Simulink

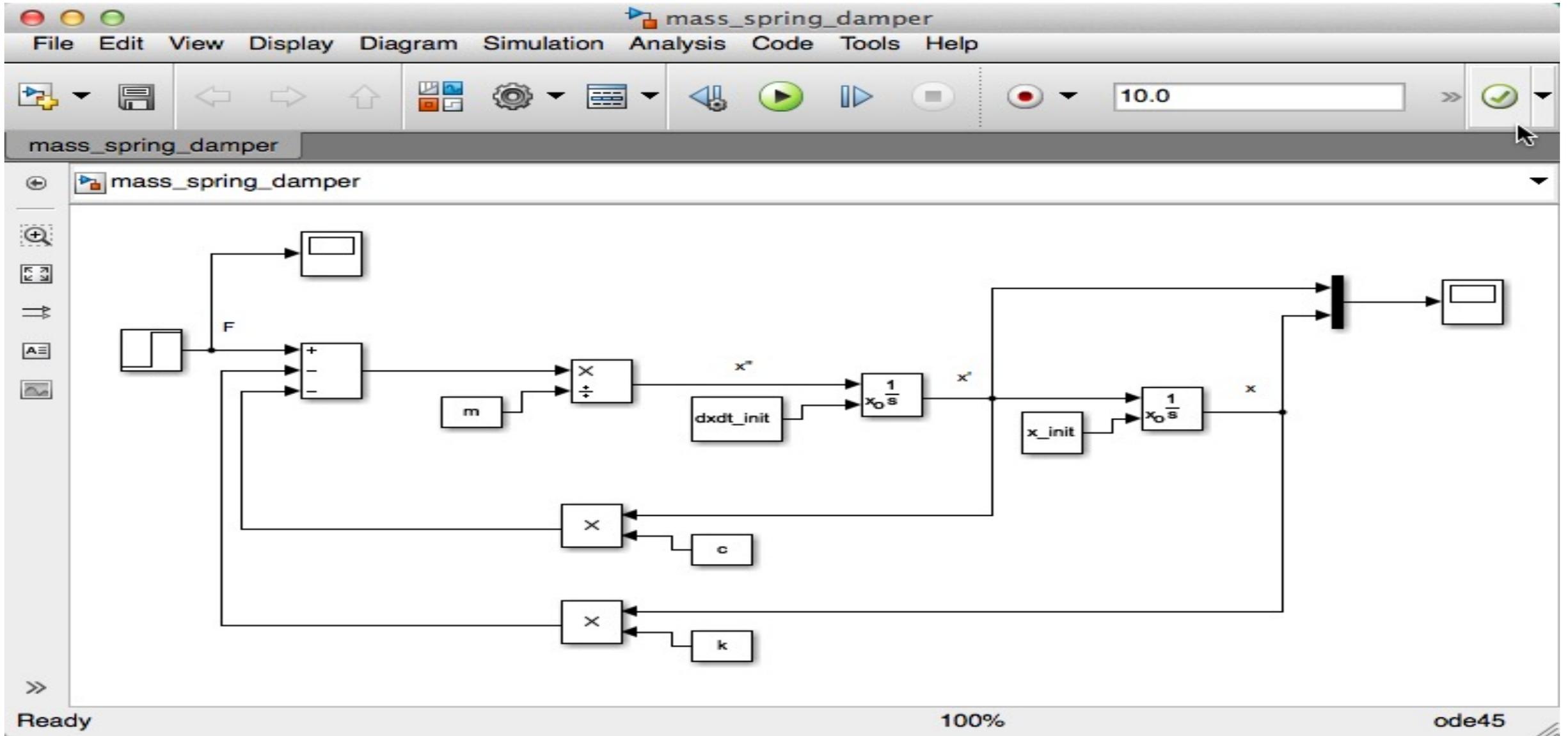
Graphical Programming and Simulation with MATLAB

Hans-Petter Halvorsen

What is Simulink?

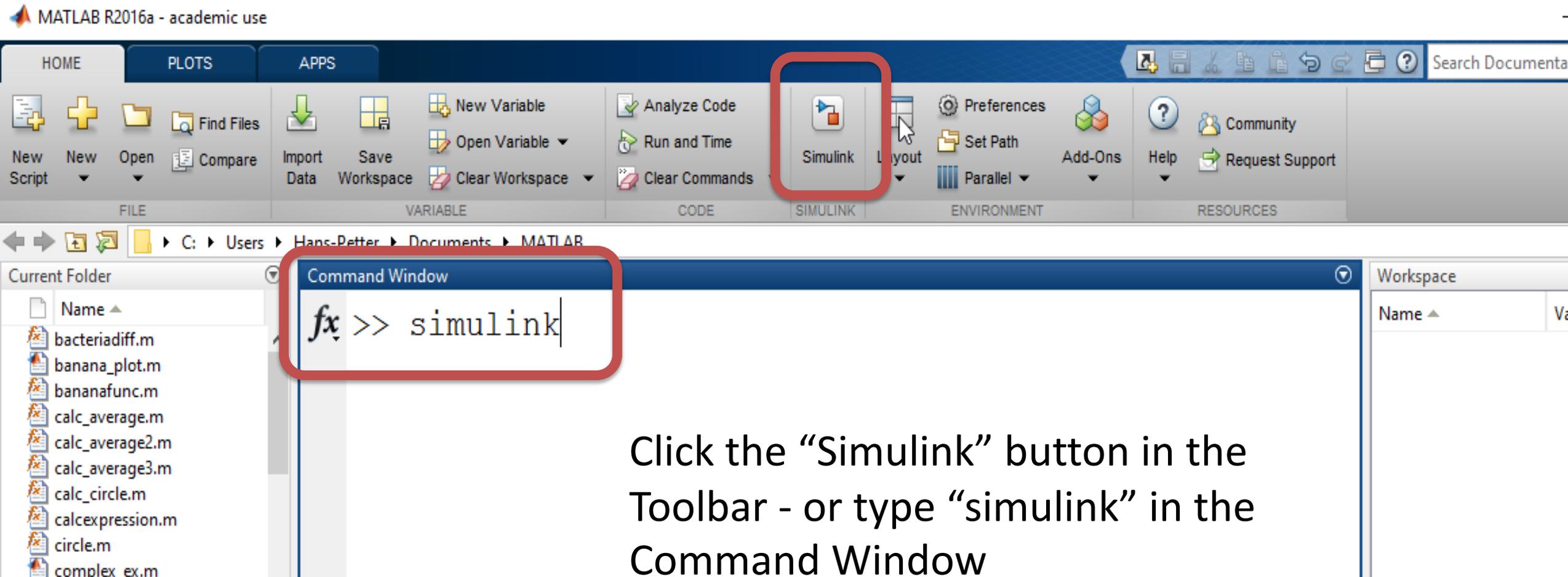
- Simulink is an “add-on” to MATLAB.
- You need to have MATLAB in order to use Simulink
- Simulink is used for Simulation of dynamic models
- In Simulink we create a Graphical Block Diagram for the system (based on the differential equations(s))

Simulink Model



In Simulink we create and configure graphical blocks and wire them together (based on the differential equations)

Start Simulink from MATLAB



The screenshot displays the MATLAB R2016a interface. The top toolbar is divided into sections: HOME, PLOTS, APPS, SIMULINK, ENVIRONMENT, and RESOURCES. The SIMULINK section contains a button labeled 'Simulink' which is highlighted with a red box. Below the toolbar, the Command Window is open and contains the text `fx >> simulink`, also highlighted with a red box. The left sidebar shows a file explorer with a list of MATLAB files, and the right sidebar shows the Workspace area.

Click the “Simulink” button in the Toolbar - or type “simulink” in the Command Window

Simulink Start Page

The screenshot shows the Simulink Start Page window. At the top, there is a red header with the "SIMULINK" logo. Below the header, the interface is divided into several sections. On the left, there is a sidebar with "Open...", "Recent", and "Projects" (Source Control..., Archive...). The main area has tabs for "New" and "Examples". Below the tabs is a search bar and a dropdown menu for "All Templates". The main content area is titled "My Templates" and contains a message: "You have not created any templates. Learn how to create templates." Below this, there is a section titled "Simulink" which contains a grid of six templates. The "Blank Model" template is highlighted with a red rounded rectangle. The other templates are "Blank Library", "Blank Project", "Code Generation", "Digital Filter", and "Feedback Controller".

Simulink Start Page

SIMULINK

Open...

Recent

Projects

Source Control...

Archive...

New Examples

Search

All Templates

My Templates [Learn More](#)

You have not created any templates. [Learn how to create templates.](#)

Simulink

Blank Model

Blank Library

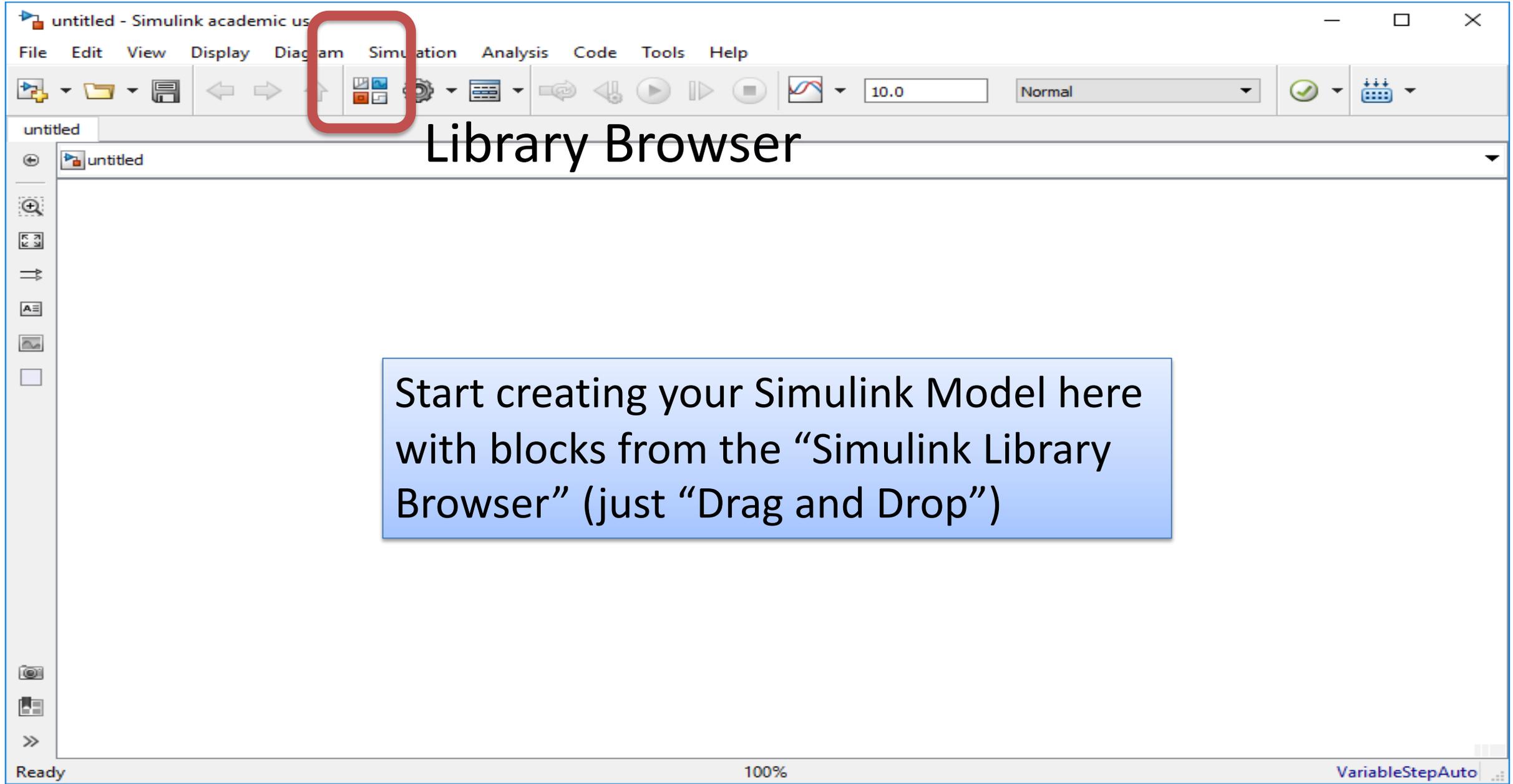
Blank Project

Code Generation

Digital Filter

Feedback Controller

Simulink Model Editor



Simulink Library Browser

The screenshot displays the Simulink Library Browser interface. The window title is "Simulink Library Browser". At the top, there is a search bar with the placeholder text "Enter search term" and several navigation icons. Below the search bar, the main area is titled "Simulink/ Commonly Used Blocks".

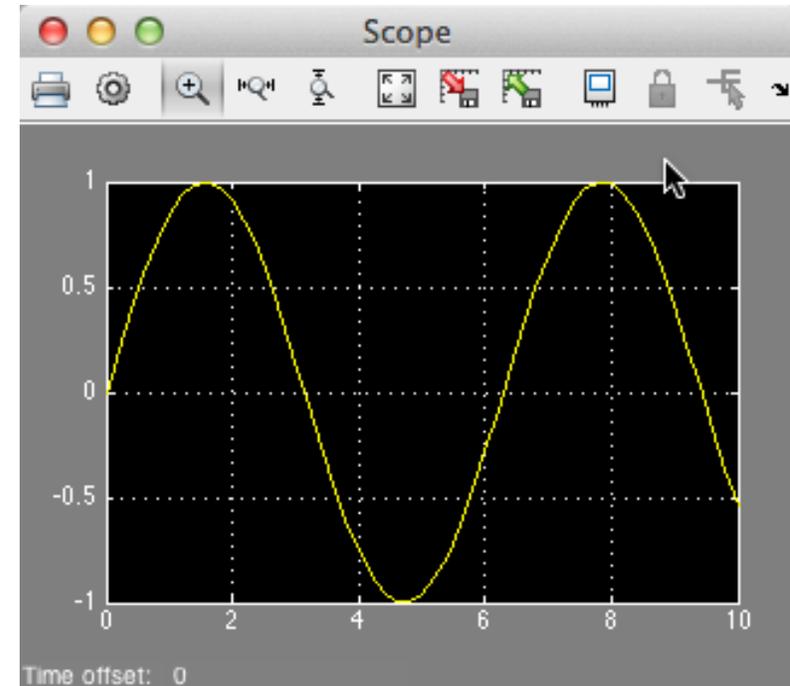
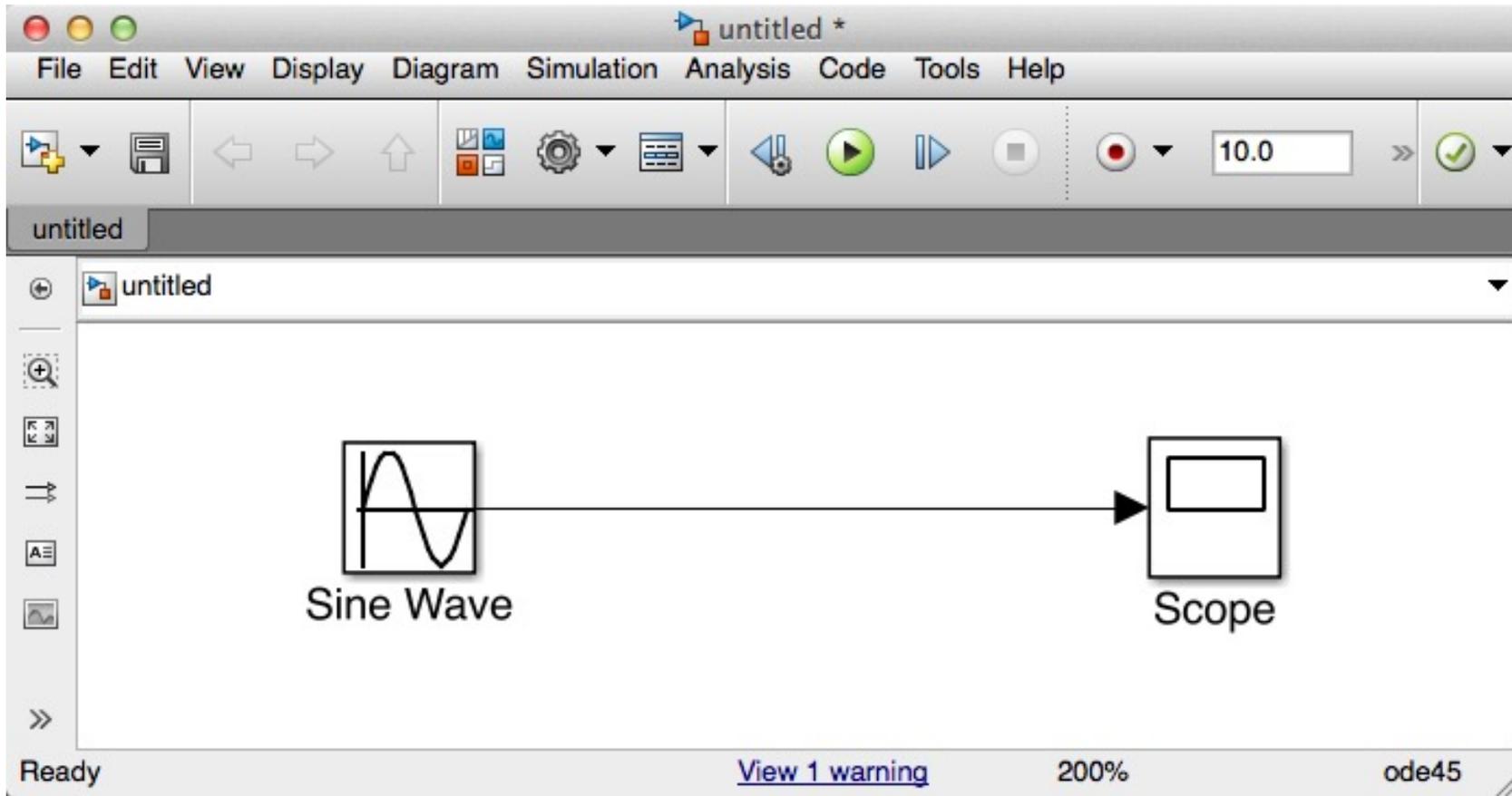
On the left side, there is a tree view showing the following categories:

- Simulink
 - Commonly Used Blocks (highlighted)
 - Continuous
 - Dashboard
 - Discontinuities
 - Discrete
 - Logic and Bit Operations
 - Lookup Tables
 - Math Operations
 - Model Verification
 - Model-Wide Utilities
 - Ports & Subsystems
 - Signal Attributes
 - Signal Routing
 - Sinks
 - Sources
 - User-Defined Functions
 - > Additional Math & Discrete
 - > Communications System Toolbox
 - > Communications System Toolbox HDL Support
 - > Computer Vision System Toolbox
 - > Control System Toolbox
 - > DSP System Toolbox
 - > DSP System Toolbox HDL Support
 - > Embedded Coder
 - > Fuzzy Logic Toolbox
 - > HDL Coder
 - > Image Acquisition Toolbox
 - > Instrument Control Toolbox
 - > Model Predictive Control Toolbox
 - > Neural Network Toolbox
 - > OPC Toolbox

On the right side, a grid of block icons is displayed, each with a label below it:

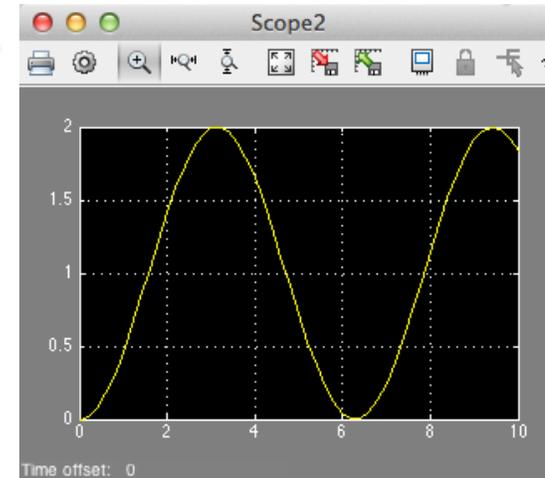
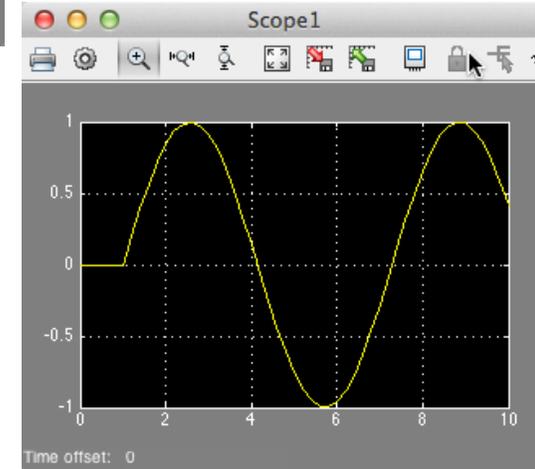
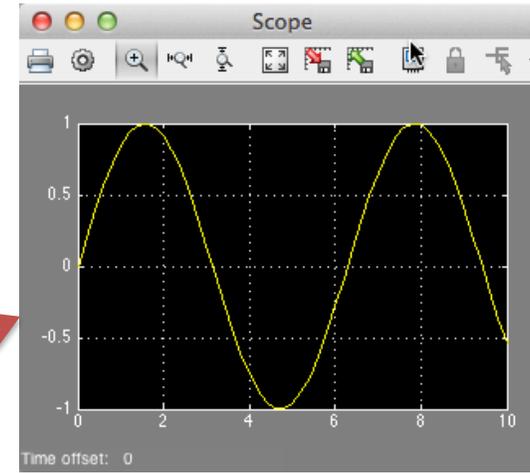
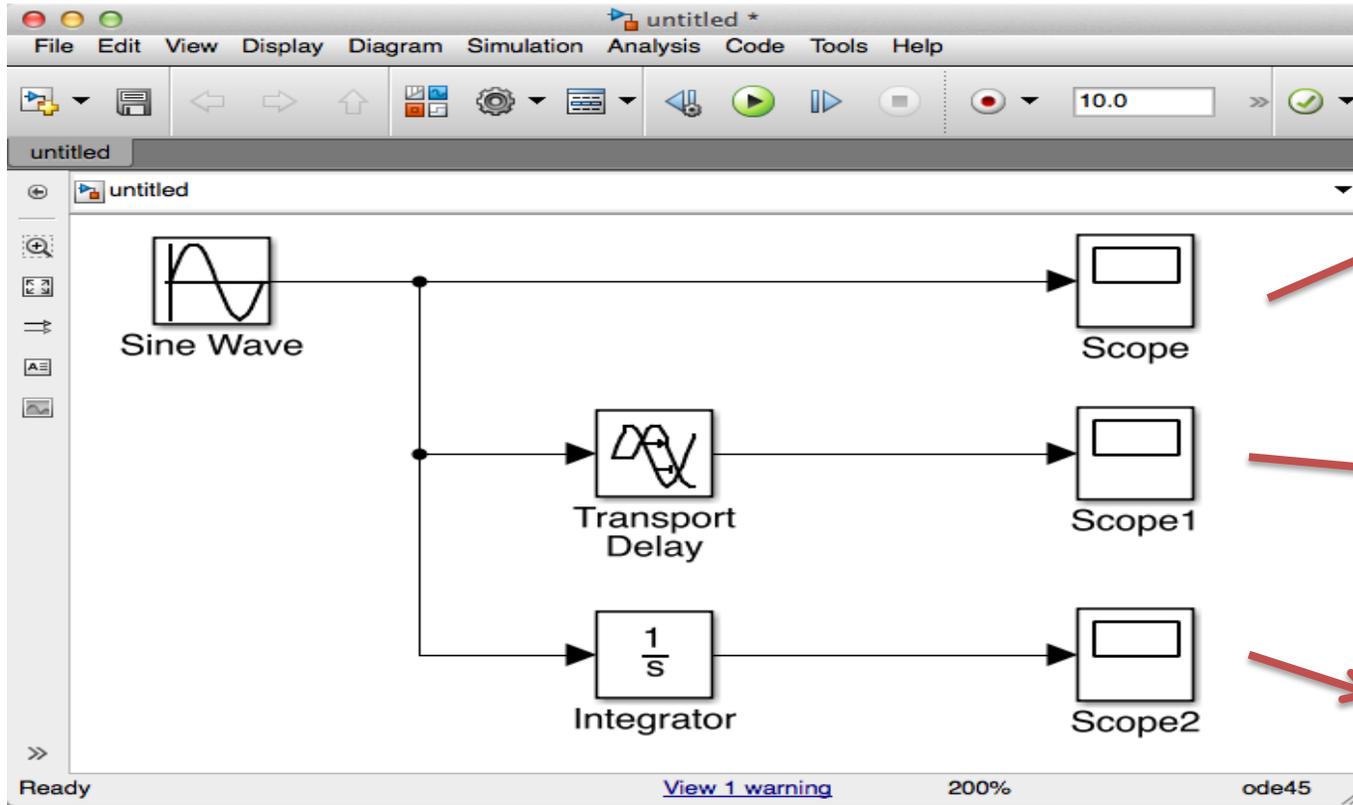
- Bus Creator
- Bus Selector
- Constant
- Data Type Conversion
- Delay
- Demux
- Discrete-Time Integrator
- Gain
- Ground
- In1
- Integrator
- Logical Operator
- Mux
- Out1
- Product
- Relational Operator
- Saturation
- Scope
- Subsystem
- Sum
- Switch
- Terminator
- Vector Concatenate

Simulink Example



DEMO

Simulink Example II



DEMO



Example

My First Simulink Model

Model:

$$\dot{x} = ax$$

Where

$$a = -\frac{1}{T}$$

T is the Time constant

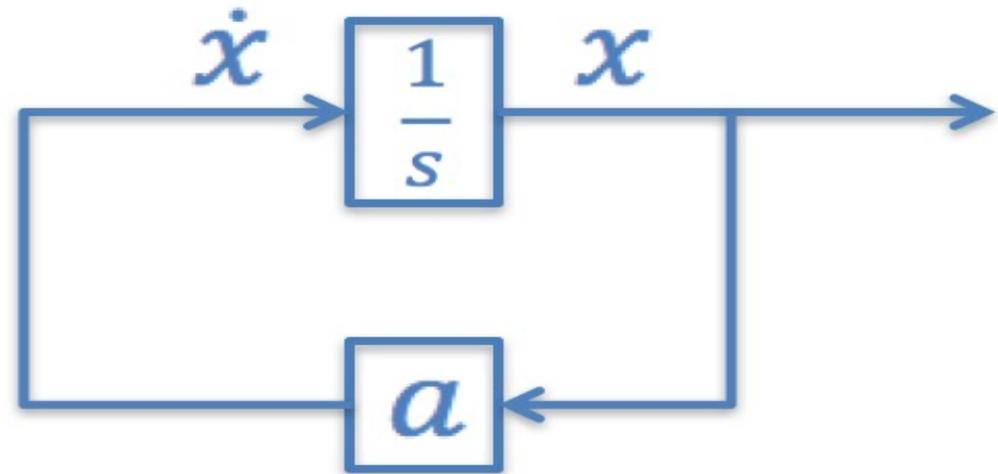
We will use the following:

$$T = 5$$

$$x(0) = 1$$

$$0 \leq t \leq 25$$

We start by drawing a simple Block Diagram for the model like this (“Pen & paper”):



We will create and simulate this block diagram with Simulink

Using ODE Solvers in MATLAB

$$\dot{x} = ax$$

Step 1: Define the differential equation as a MATLAB function (`mydiff.m`):

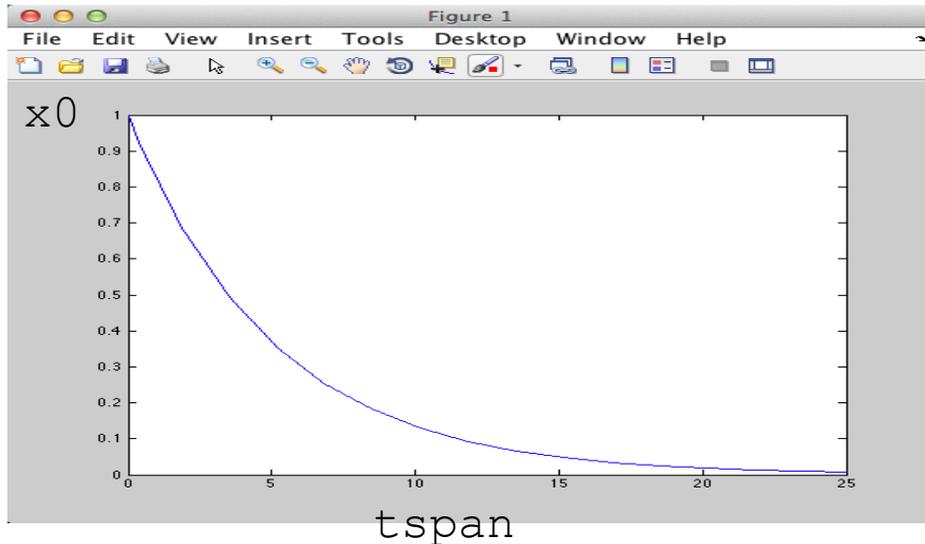
```
function dx = mydiff(t,x)
T = 5;
a = -1/T;
dx = a*x;
```

Step 2: Use one of the built-in ODE solver (`ode23`, `ode45`, ...) in a Script.

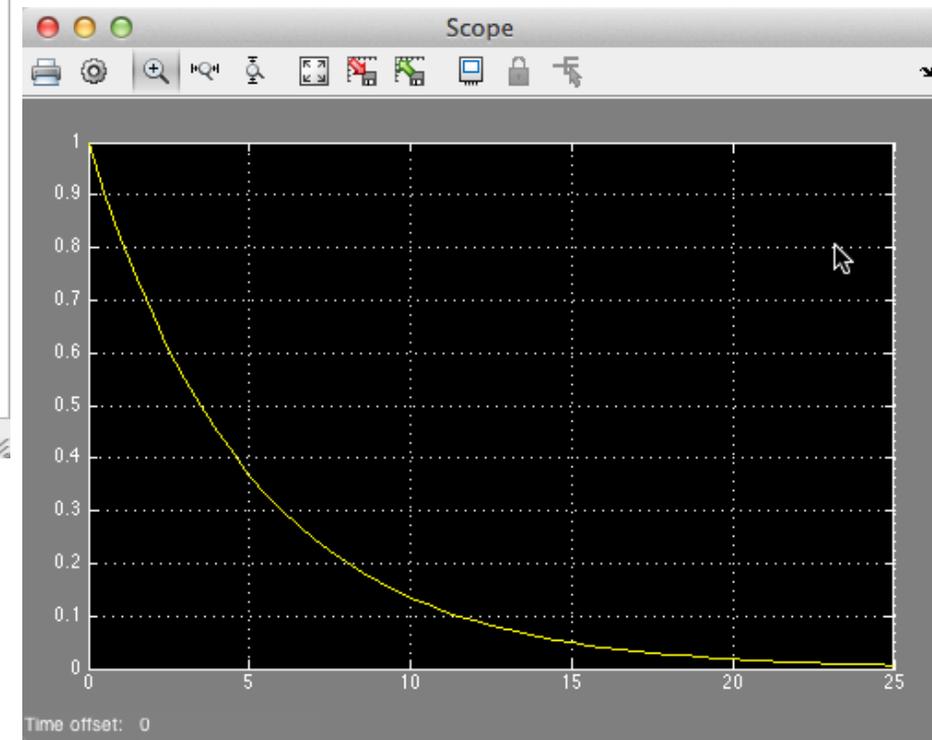
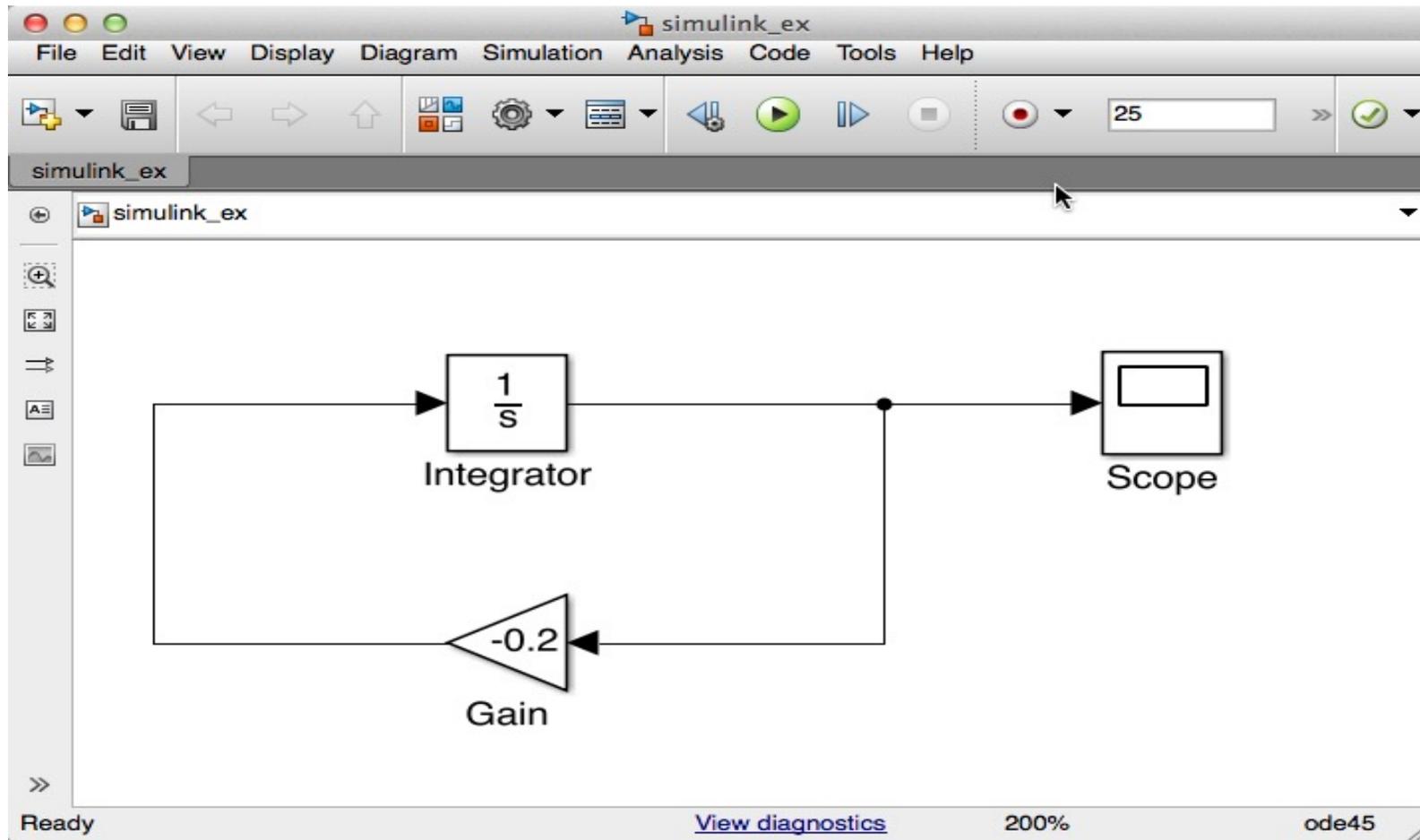
```
clear
clc

tspan = [0 25];
x0 = 1;

[t,x] = ode23(@mydiff,tspan,x0);
plot(t,x)
```



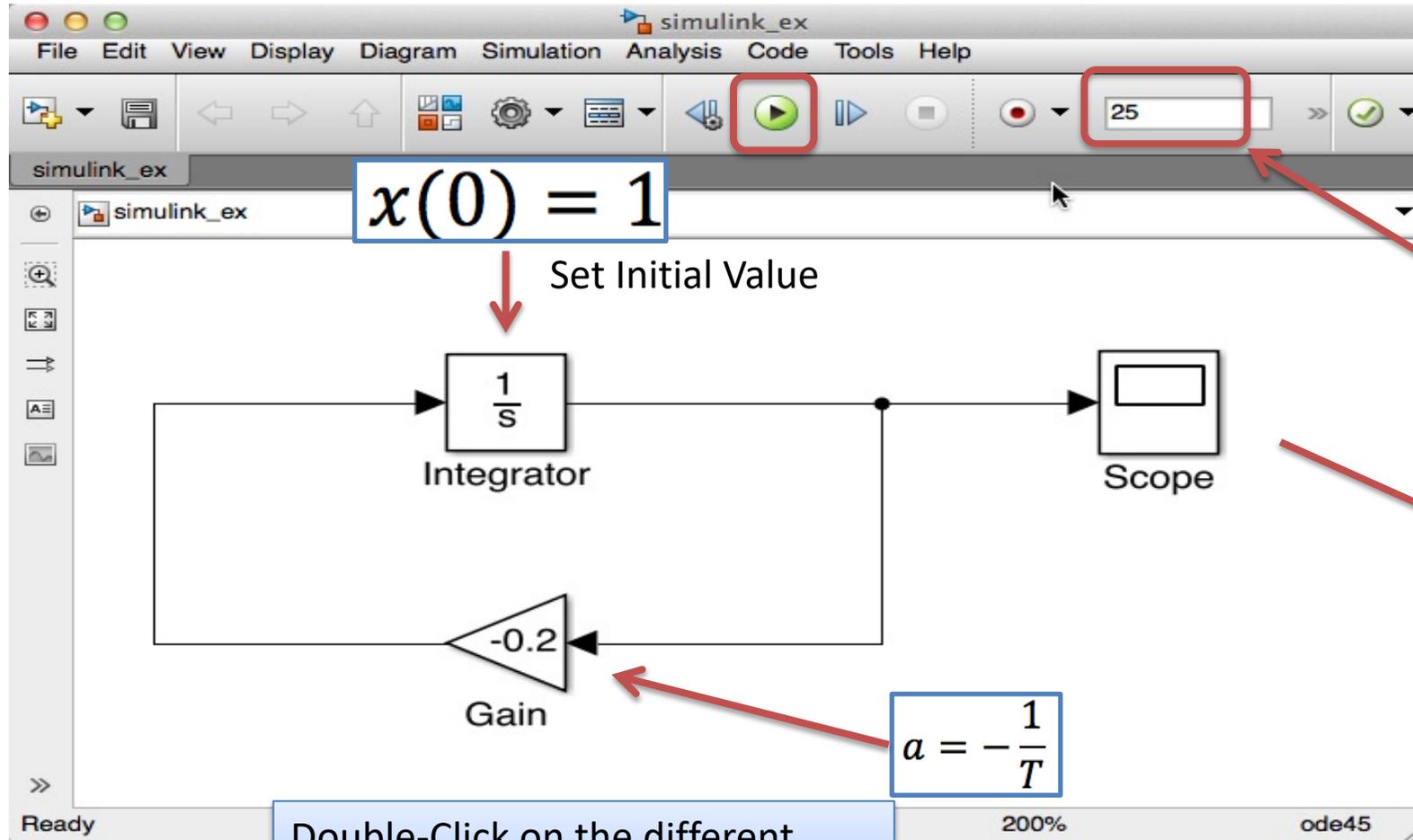
My First Simulink Model



DEMO

Solution

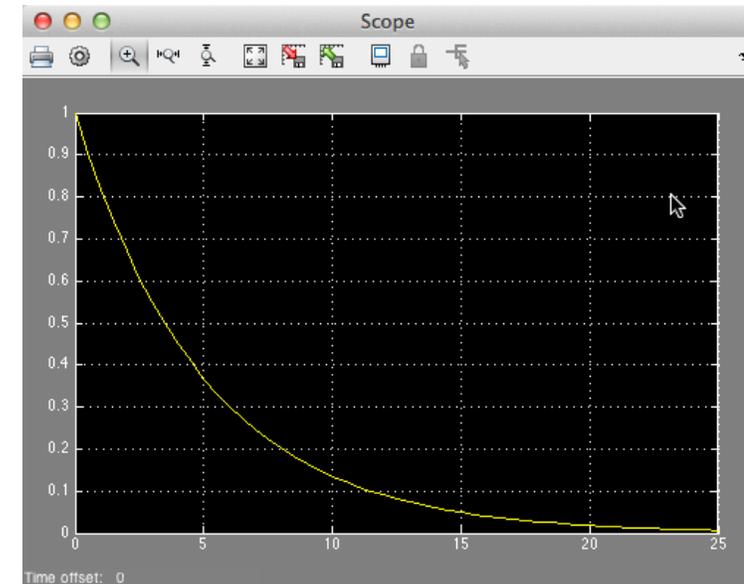
My First Simulink Model

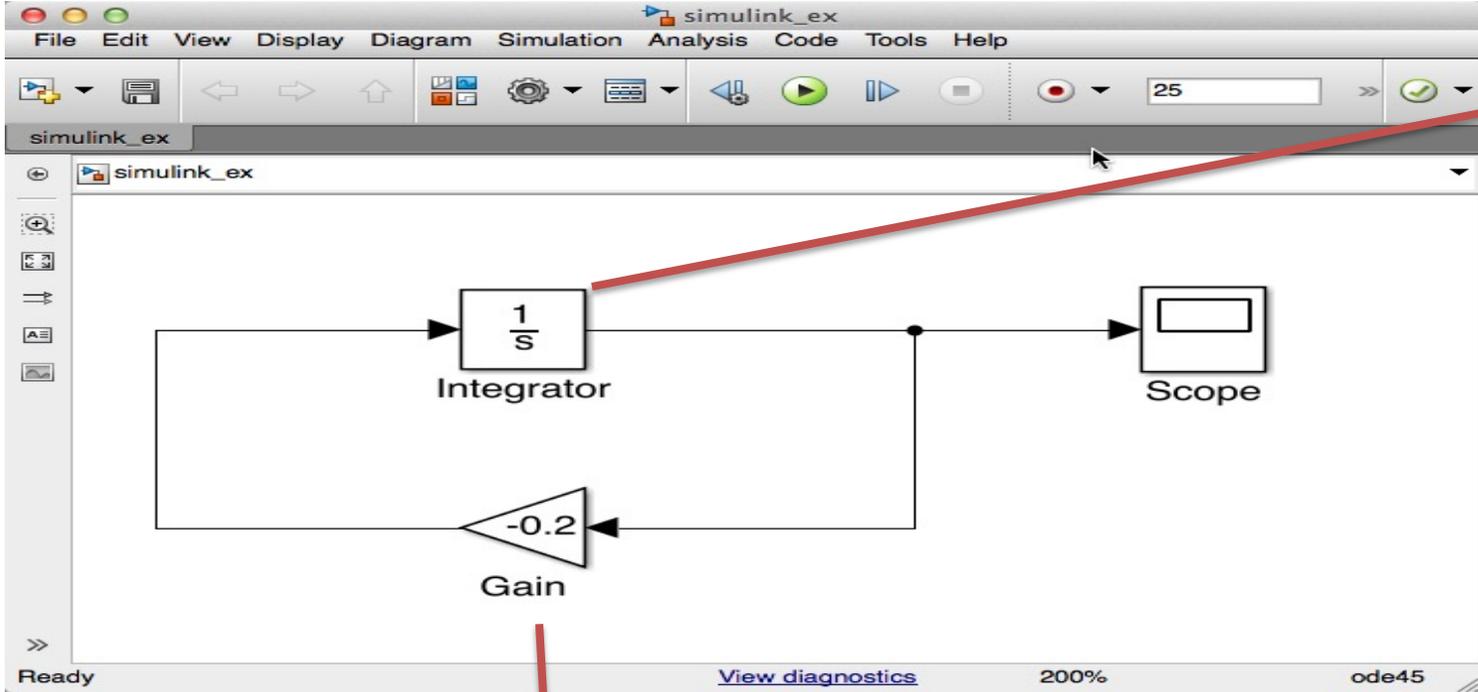


 Start the Simulation by clicking this icon

Simulation Time
 $0 \leq t \leq 25$

See the Simulation Results





Function Block Parameters: Gain

Gain

Element-wise gain ($y = K \cdot u$) or matrix gain ($y = K \cdot u$ or $y = u \cdot K$).

Main | Signal Attributes | Parameter Attributes

Gain:

Multiplication:

Sample time (-1 for inherited):

Buttons: ? OK Cancel Help Apply

Function Block Parameters: Integrator

Integrator

Continuous-time integration of the input signal.

Parameters

External reset:

Initial condition source:

Initial condition:

Limit output

Upper saturation limit:

Lower saturation limit:

Show saturation port

Show state port

Absolute tolerance:

Ignore limit and reset when linearizing

Enable zero-crossing detection

State Name: (e.g., 'position')

Buttons: ? OK Cancel Help Apply



Bacteria Population

Here we will simulate a simple model of a bacteria population in a jar.

The model is as follows:

$$\text{birth rate} = bx$$

$$\text{death rate} = px^2$$

Then the total rate of change of bacteria population is:

$$\dot{x} = bx - px^2$$

Set $b=1$ /hour and $p=0.5$ bacteria-hour

→ We will simulate the number of bacteria in the jar after **1 hour**, assuming that initially there are **100 bacteria** present.

In MATLAB We would do like this

```
function dx = bacteriadiff(t,x)
% My Simple Differential Equation

b=1;
p=0.5;

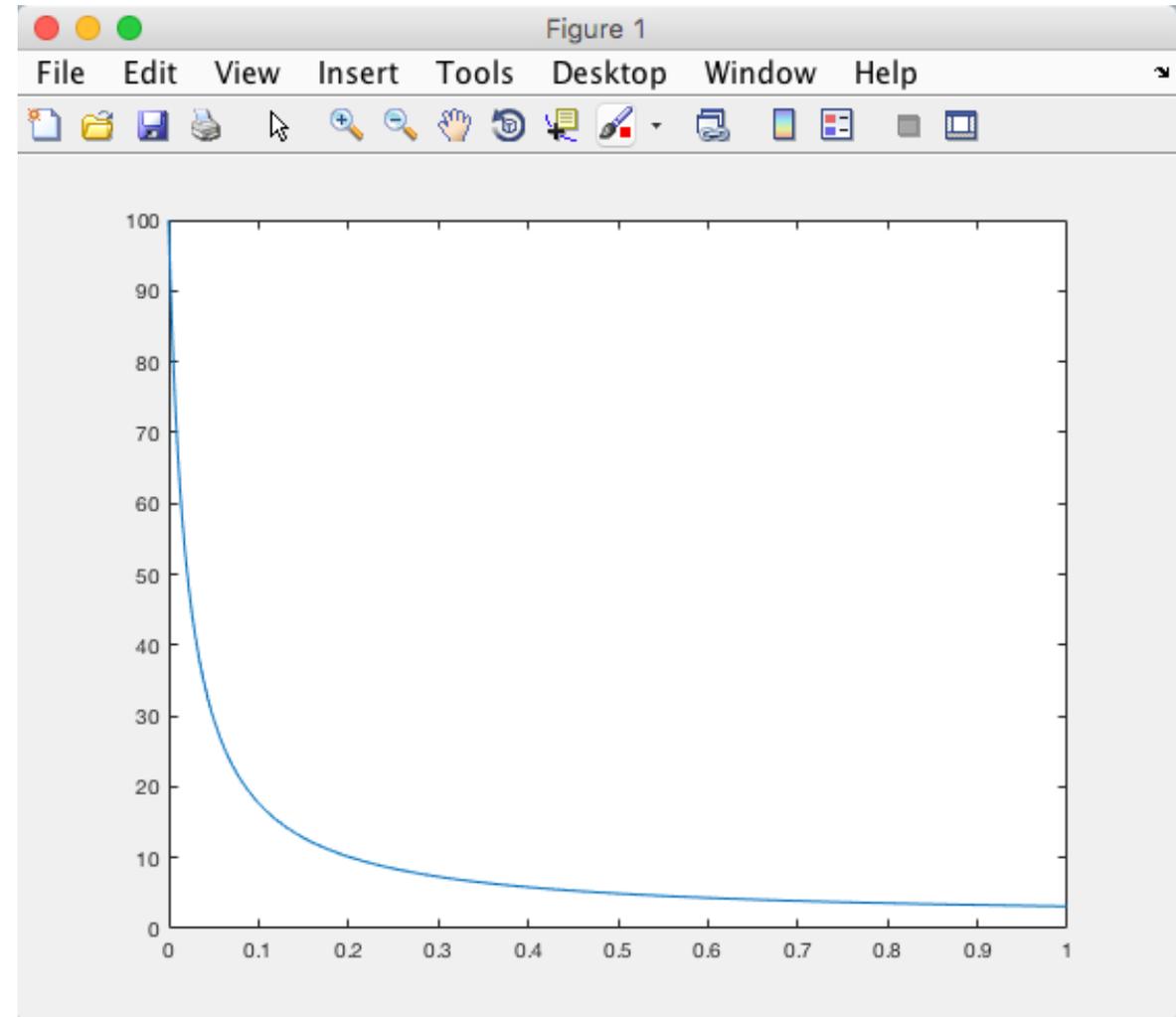
dx = b*x - p*x^2;
```

```
clear
clc

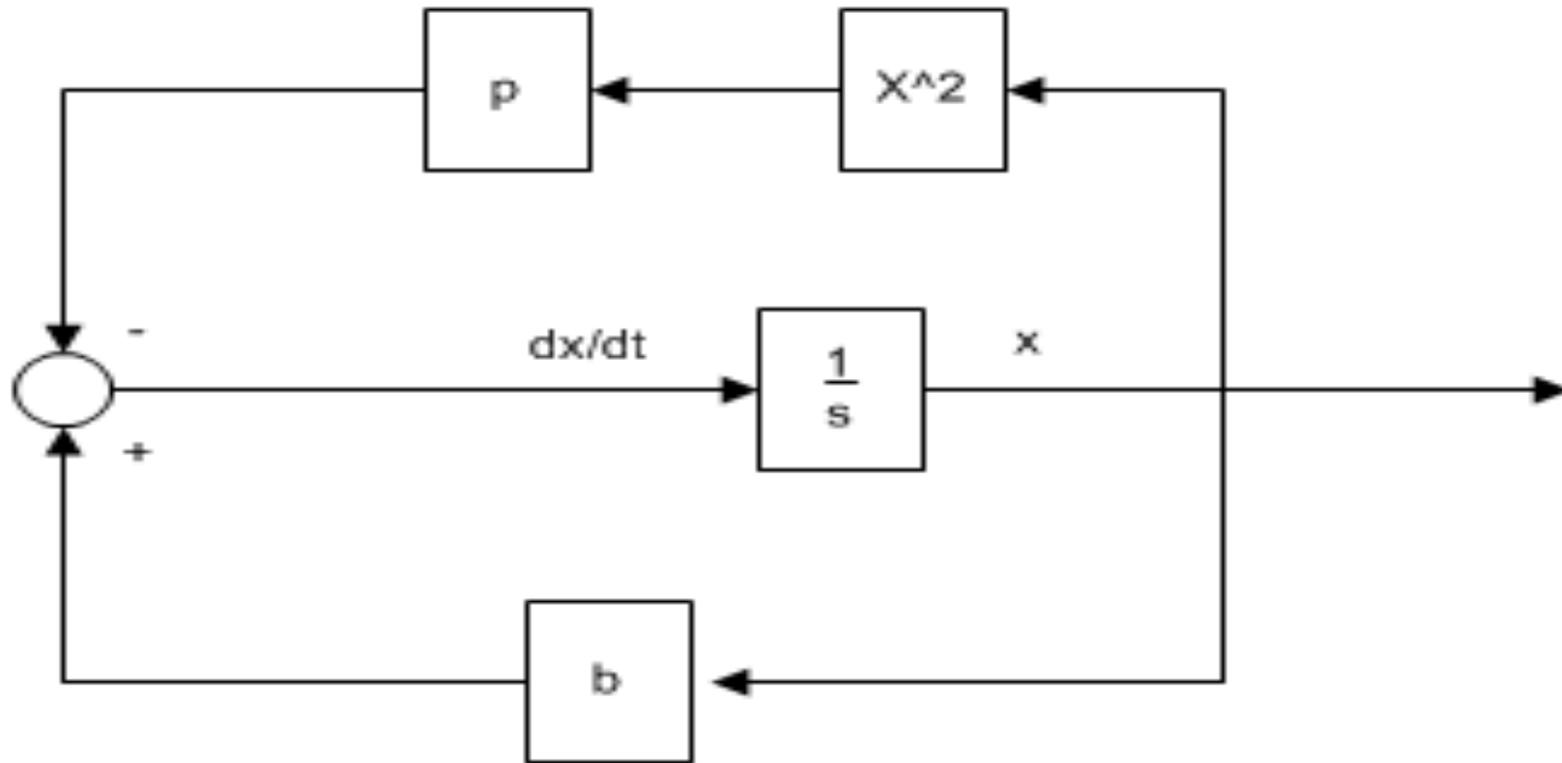
tspan=[0 1];
x0=100;

[t,y]=ode45(@bacteriadiff, tspan,x0);
plot(t,y)

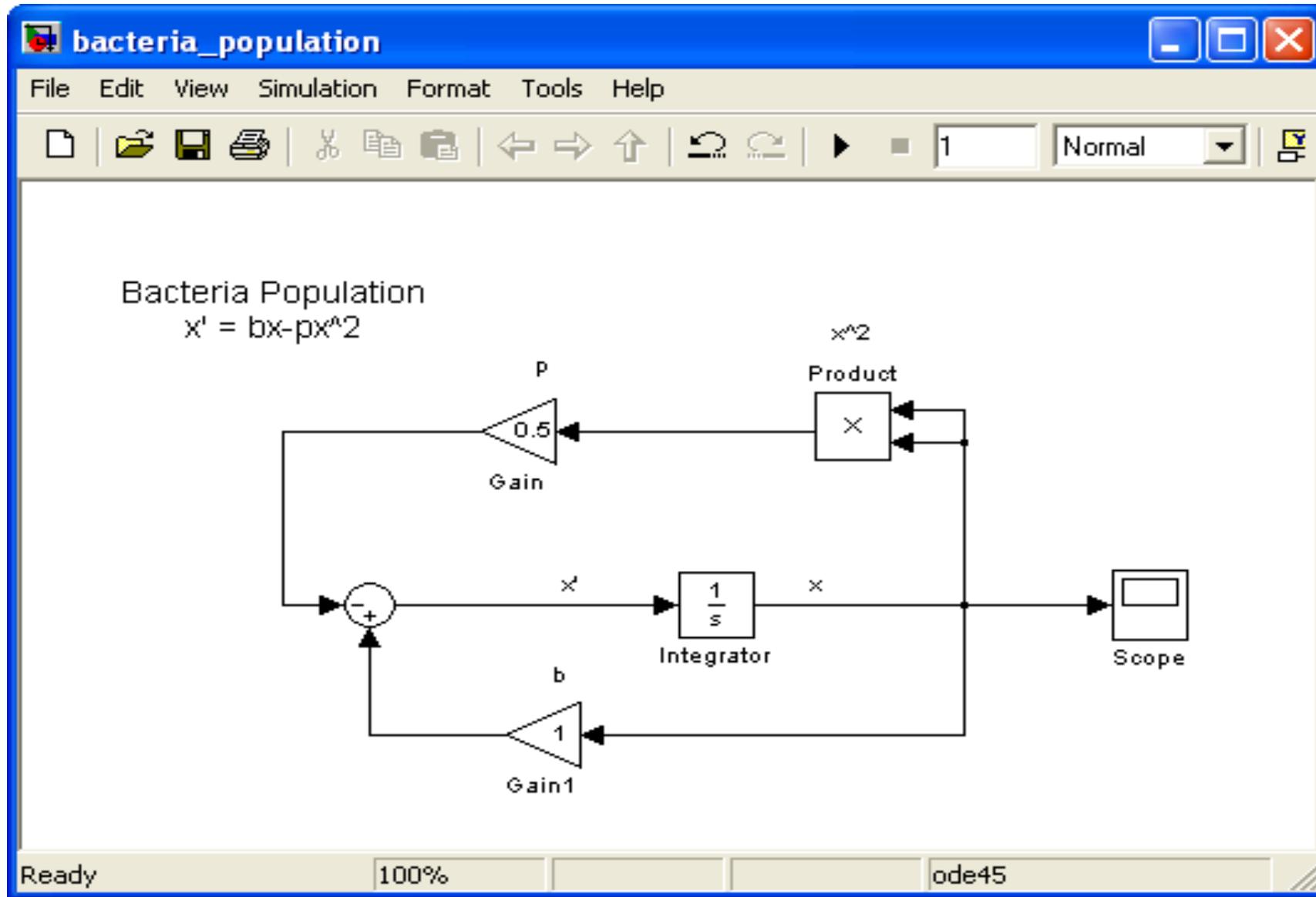
[t,y]
```



Block Diagram for the Model (“Pen and Paper”)



Simulink Block Diagram for the Model



DEMO



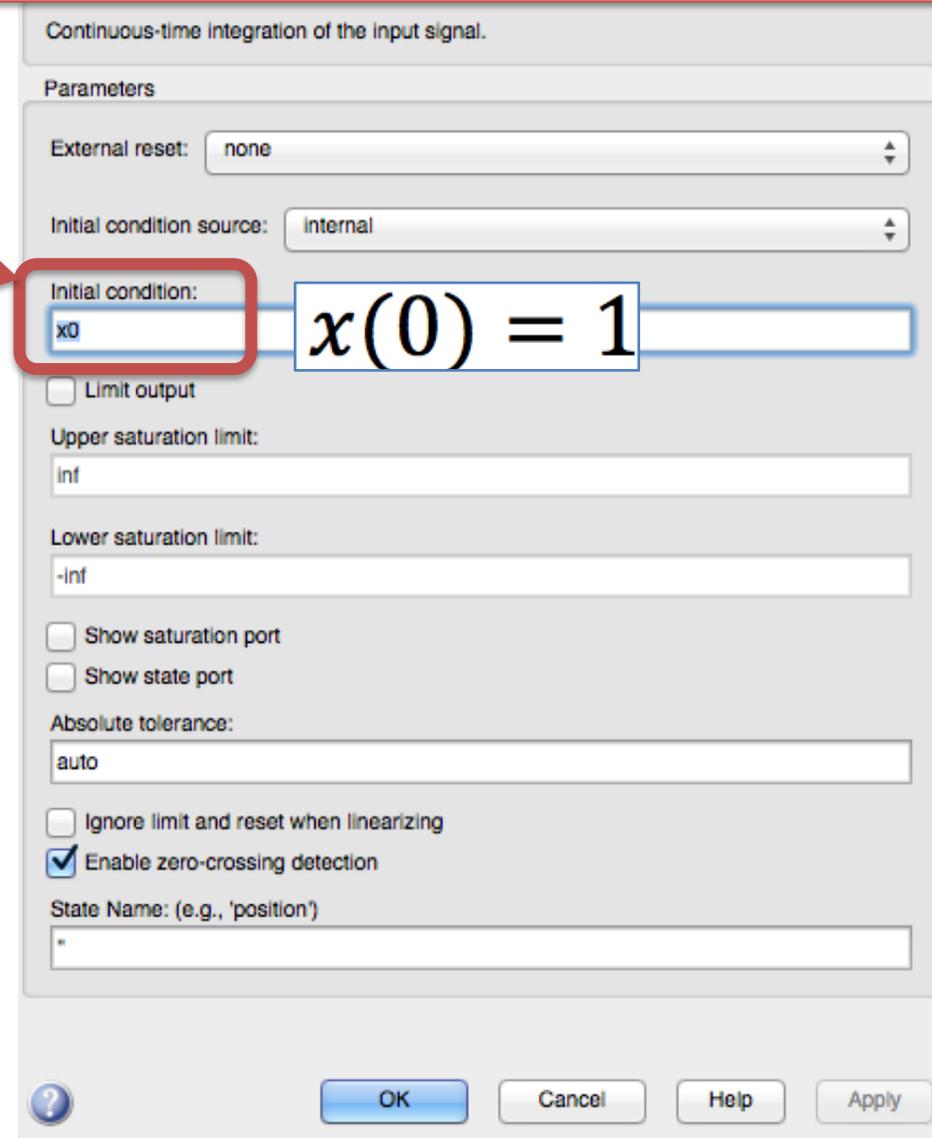
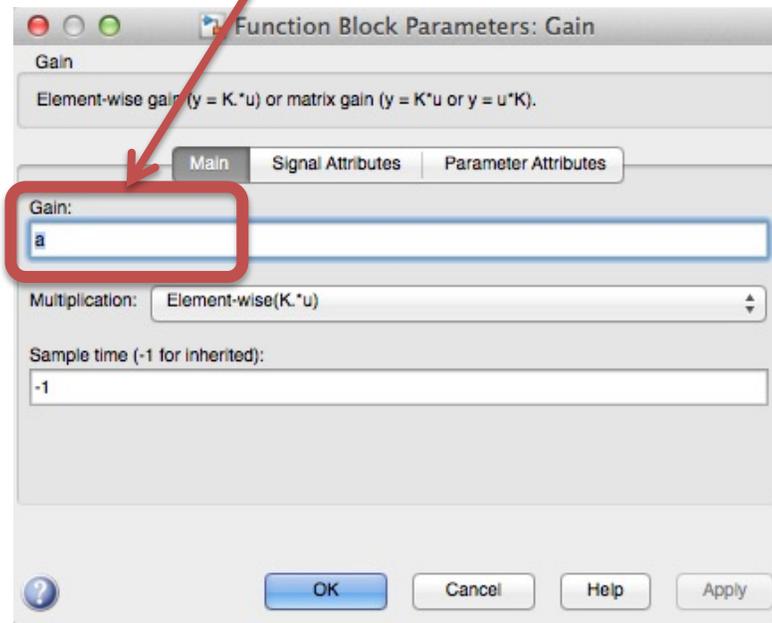
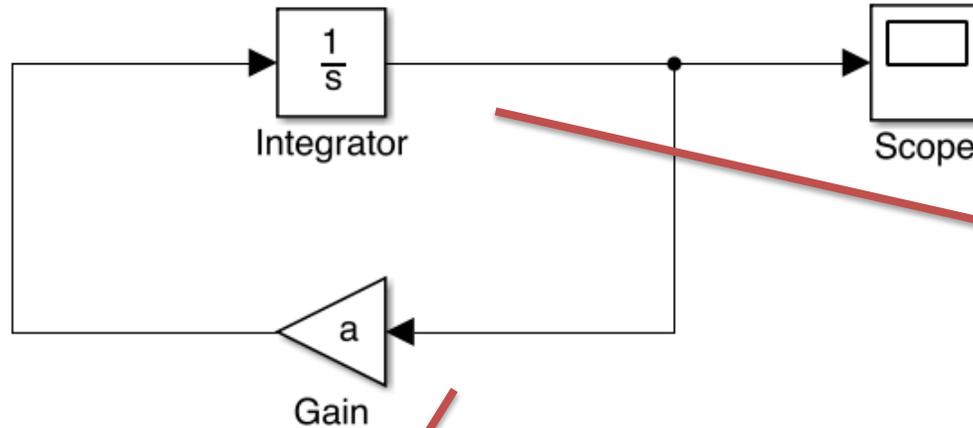
Data-driven Modelling

- You may use Simulink together with MATLAB in order to specify data and parameters to your Simulink model.
- You may specify commands in the MATLAB Command Window or as commands in an m-file (Script).
- This is called data-driven modeling
- Instead of using values directly we use variables instead - This is more flexible because we can easily change Simulation Parameters without touching the Simulink Model

Example

$$\dot{x} = ax$$

Instead of using values directly we use variables instead – This is more flexible because we can easily change Simulation Parameters without changing the Simulink Model



DEMO

Data-driven Modelling

MATLAB Script for running the Simulink Simulation:

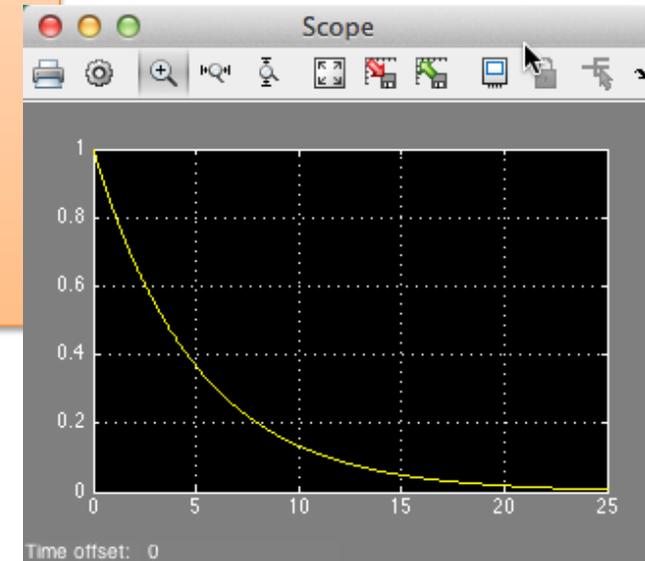
```
clear
clc

%Set Simulator Settings
x0=1;
T=5;
a=-1/T;
t_stop=25; %[s]
T_s=t_stop/1000; %[s]
options=simset('solver', 'ode5', 'fixedstep', T_s);

%Starting Simulation
sim('simulink_ex2', t_stop, options);
```

This is the Name for our Simulink Model

We get the same results:





Mass-Spring-Damper System

In this example we will create a mass-spring-damper model in Simulink and configure and run the simulation from a MATLAB m-file.

The differential equation for the system is as follows:

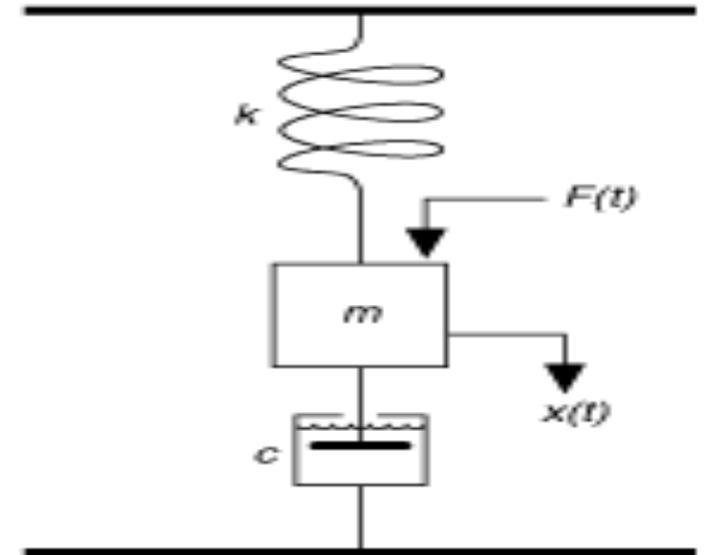
$$\ddot{x} = \frac{1}{m}(F - c\dot{x} - kx)$$

Where:

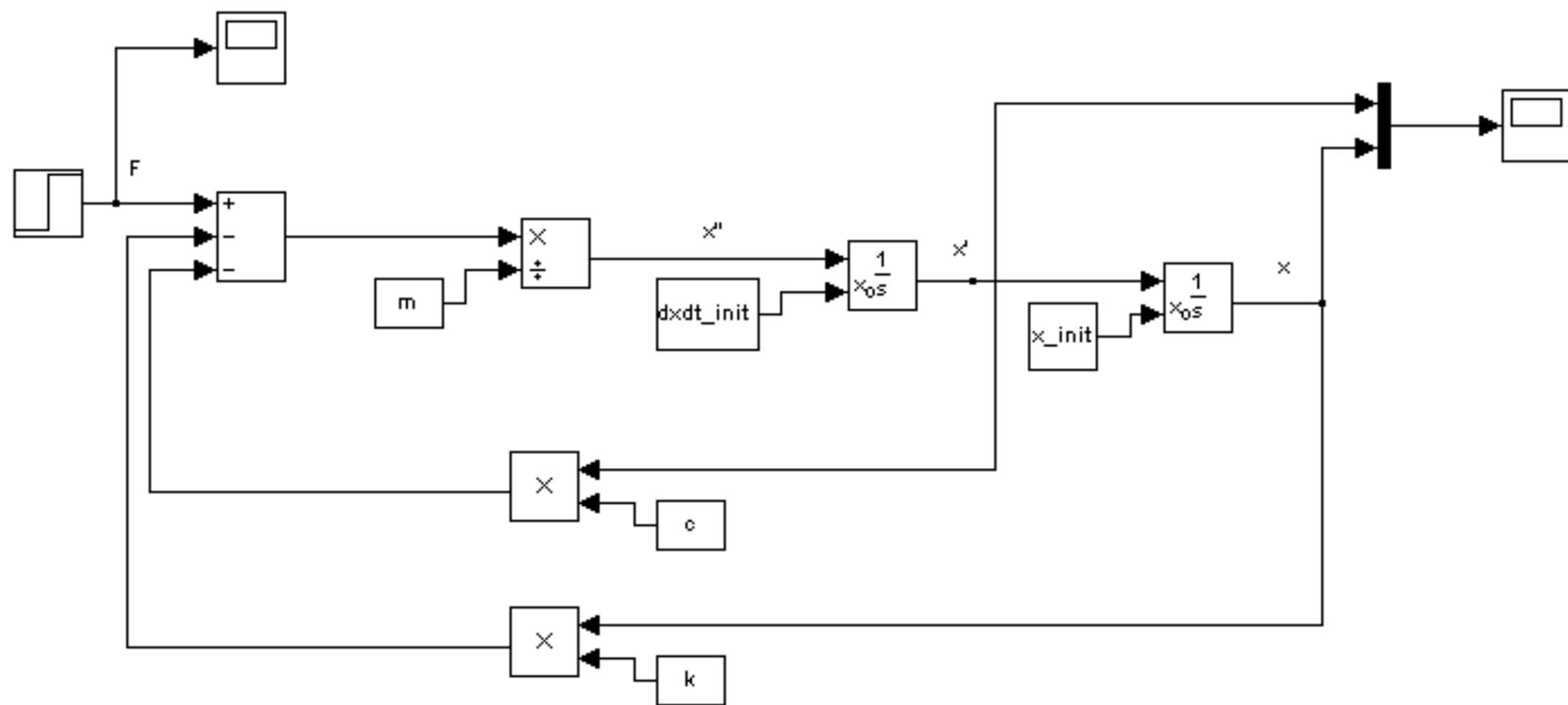
x - position

\dot{x} - speed

\ddot{x} - acceleration



Instead of hard-coding the model parameters in the blocks you should refer to them as variables set in an m-file.



The following variables should then be set in the m-file:

```
x_init = 4; %[m]. Initial position.  
dxdt_init = 0; %[m/s]. Initial Speed.  
m = 20; %[kg]  
c = 4; %[N/(m/s)]  
k = 2; %[N/m]  
t_step_F = 50; %[s]  
F_0 = 0; %[N]  
F_1 = 4; %[N]
```

DEMO

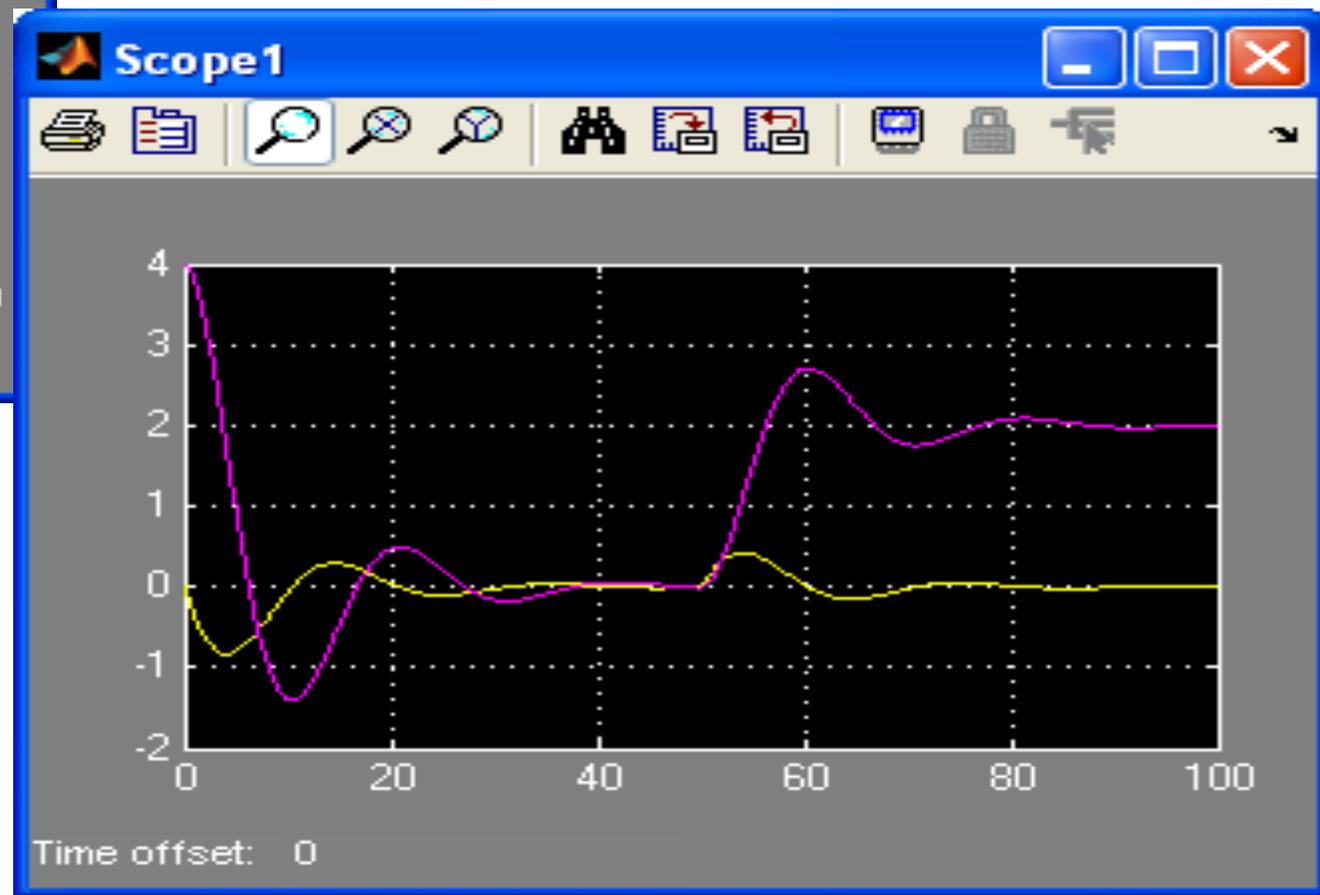
```
1 %Script of mass-spring-damper simulator.
2 %Hans-Petter Halvorsen. 20.11.2009
3
4 %Modell Parameters
5 - x_init=4; %[m]. Initial position.
6 - dxdt_init=0; %[m/s]. Initial Speed.
7 - m=20; %[kg]
8 - c=4; %[N/(m/s)]
9 - k=2; %[N/m]
10 - t_step_F=50; %[s]
11 - F_0=0; %[N]
12 - F_1=4; %[N]
13
14 %Simulator Settings
15 - t_stop=100; %[s]
16 - T_s=t_stop/1000; %[s]
17 - options=simset('solver', 'ode5', 'fixedstep', T_s);
18
19 %Starting simulation
20 - sim('mass_spring_damper', t_stop, options);
```

DEMO

Force F:



Position x and speed \dot{x} :





Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

