MTE204

Introduction to Simulink

1.0. What Is Simulink

Simulink is a software package for modeling, simulating, and analyzing dynamic systems (systems whose state varies with time). It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Simulink is a companion program to MATLAB and it contains two windows:

1. Library window: this contains several library and each library contains blocks
2. Workspace window: this is where the Simulink model is constructed

1.1. Tool for Interactive Simulation

Models can easily be constructed from scratch, or an existing model can be taken and addition made to it. Simulations are interactive, so one can change parameters in real time and immediately see the outcome.

1.2. Tool for Model-Based Design

Simulink turns the computer into a lab for modeling and analyzing systems. The workspace window in Simulink provides a graphical user interface (GUI) for building models as block diagrams, using click-and-drag mouse operations. Here one can draw the models just as you would with pencil and paper. Simulink includes a comprehensive block library of sinks, sources, linear and nonlinear components, and connectors. One can also customize and create his own blocks. After drawing a model, one can simulate it, using a choice of integration methods, either from the Simulink menus or by entering commands in the MATLAB Command Window. Using scopes and other display blocks, you can see the simulation results while the simulation is running. The simulation results can be put in the MATLAB workspace for post-processing and visualization. MATLAB and Simulink are integrated, so one can simulate, analyze, and revise your models in either environment at any point.

Simulink can be used to explore the behavior of a wide range of real-world dynamic systems, including electrical circuits, shock absorbers, braking systems, and many other electrical, mechanical, and thermodynamic systems.

2.0. Building a Model

The model below integrates a sine wave and displays the result along with the sine wave. The block diagram of the model looks like this.

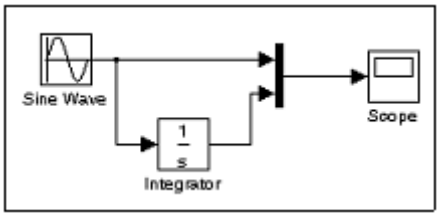
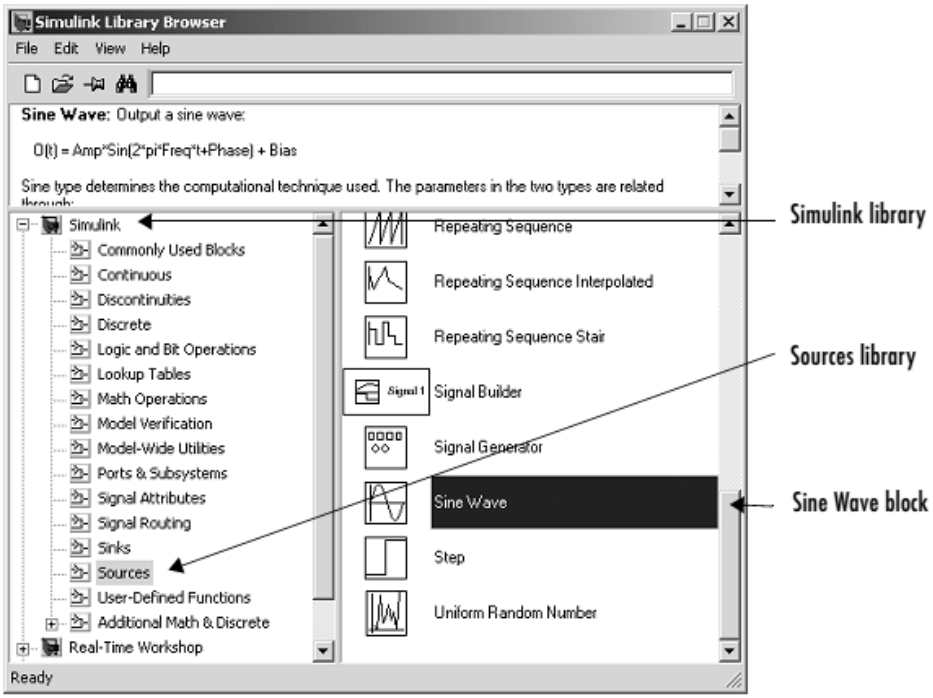


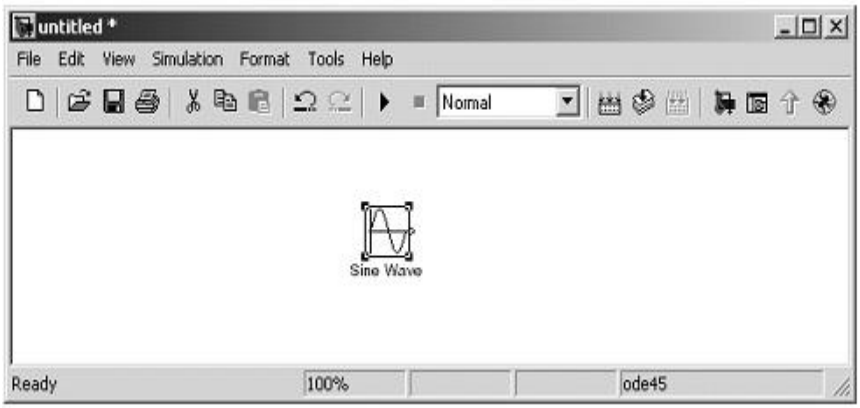
Fig. 1

2.1.Model creation steps

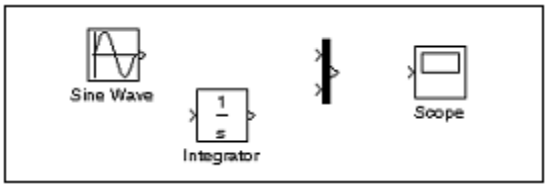
1. Enter Simulink in the MATLAB Command Window or click the Simulink library on the MATLAB toolbar, the Simulink Library Browser appears.
2. Click the New Model button on the Library Browser's toolbar, Simulink opens a new model window.
3. Create the model by copying blocks into the model from the following Simulink block libraries:
   - Sources library (the Sine Wave block)
   - Sinks library (the Scope block)
   - Continuous library (the Integrator block)
   - Signal Routing library (the Mux block)



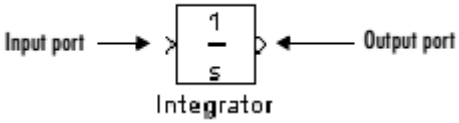
Now drag a copy of the Sine Wave block from the browser and drop it in the model window.



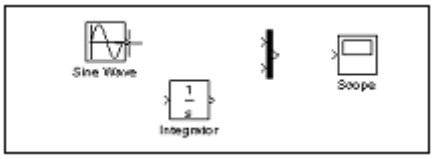
Repeat same for the other blocks.



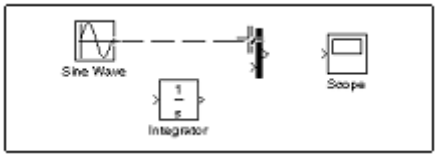
Examine the blocks above; observe the angle bracket on the right of the Sine Wave block and two on the left of the Mux block. *The > symbol pointing out of a block is an output port; if the symbol points to a block, it is an input port.* Signal travels out of an output port and into an input port of another block through a connecting line. When the blocks are connected, the port symbols disappear.



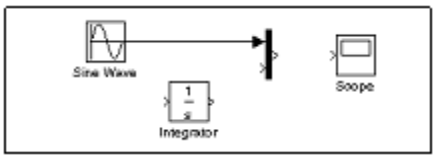
4. To connect the blocks, connect the Sine Wave block to the top input port of the Mux block. Position the mouse pointer over the output port right side of the Sine Wave block. Notice that the cursor shape changes to *crosshairs*.



Hold down the mouse button and move the cursor to the top input port of the Mux block.

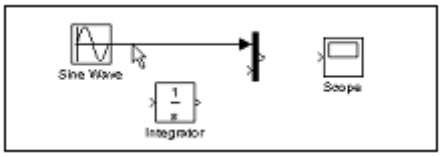


Now release the mouse button. The blocks are connected.

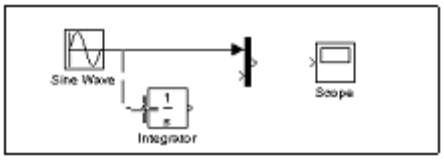


However, to create the branch line; such as that connects the Sine Wave output to the Integrator block in Fig.1 above;

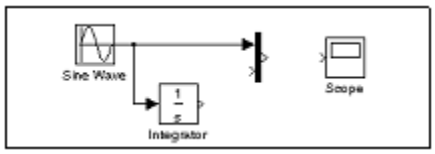
i. First, position the pointer on the line between the Sine Wave and the Mux block.



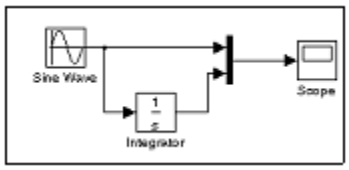
ii. Press and hold down the Ctrl key (or click the right mouse button). Press the mouse button, and then drag the pointer to the Integrator block's input port or over the Integrator blocks itself.



iii. Release the mouse button. Simulink draws a line between the starting point and the Integrator block's input port.



Finish making block connections. When you're done, your model should look something like this:



Note; the *branch line* carries the same signal that passes from the Sine Wave block to the Mux block.

2.2. Setting up to run the simulation

Now we set up Simulink to run the simulation for 10 seconds. First, open the Configuration Parameters dialog box by choosing Configuration Parameters from the Simulation menu. On the dialog box that appears, notice that the Stop time is set to 10.0 (its default value).

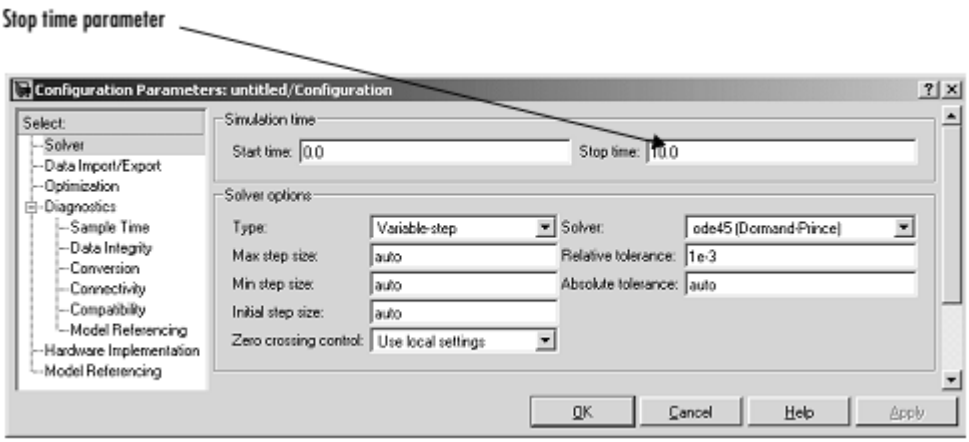


Fig. 2

Close the Configuration Parameters dialog box by clicking the OK button. Simulink applies the parameters and closes the dialog box. Now double-click the Scope block to open its display window. Finally, choose Start from the Simulation menu and watch the simulation output on the Scope.

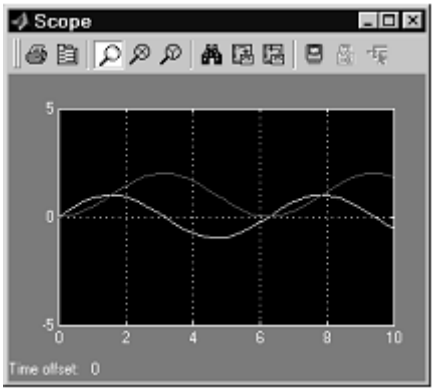


Fig. 3

The simulation stops when it reaches the stop time specified in the Configuration Parameters dialog box or when you choose Stop from the Simulation menu or click the Stop button on the model window's toolbar (Windows only).
To save this model, choose Save from the File menu and enter a filename and location. That file contains the description of the model.
To terminate Simulink and MATLAB, choose Exit MATLAB

2.2.1. Solving Differential Equations

Simulink can be used to solve evaluate various types of dynamic systems. Consider the first order differential equation:

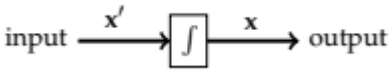
$$\frac{dx}{dt} = 2\sin 3t - 4x.$$

With initial condition $x(0) = 0$.

The system can be viewed as $x' = 2\sin 3t - 4x$, then fed into an integrator we have the output $x(t)$.

$$x(t) = \int x'(t)\,dt.$$

This can generally be depicted by



The schematics of the model of solving the system problem is presented in Figure 1

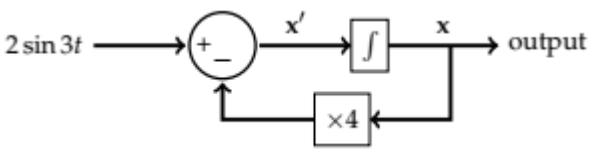


Figure 1 Model schematics 1

In Simulink the blocks required to model this system is as follows:

- Integrator block from the Continuous group;
- Sum block from the Math Operations group,
- Gain block from the Math Operations group,
- Sine Wave block from the Math Operations group; and,
- Scope block from the Sink group.

Therefore, the Simulink model is presented in Figure 2 after inputting the parameters of the sine wave, sum and integrator blocks.

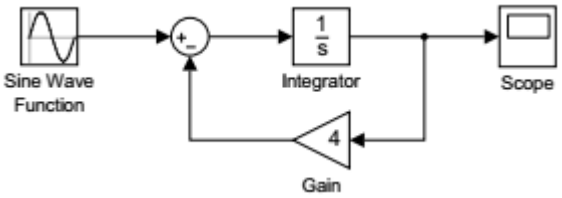


Figure 2 Simulink model 1

The simulation of the model gives the output of the system as presented in Figure 3. Obtained by double-clicking the Scope to see the solution. Figure 3 shows the Scope plot after using the auto-scale ( ) feature to rescale the scope view.

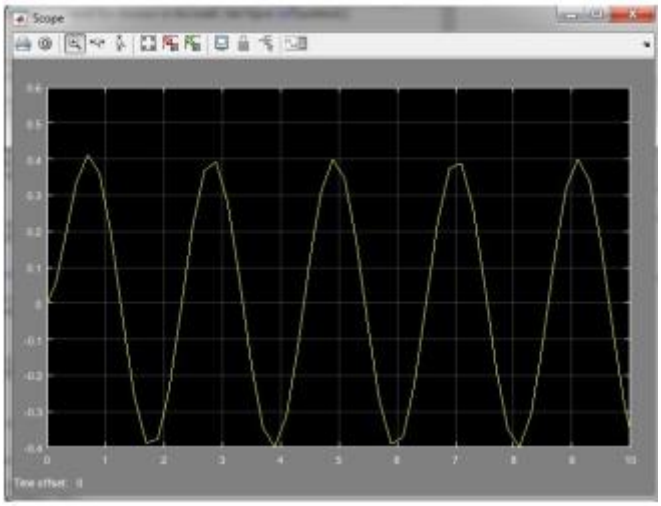


Figure 3 Oscilloscope output of model 1

However, we can make further changes to the system by checking the Configuration Parameters under the Simulation menu item. See Figures 4. In particular, changing the Refine Factor to 10 units can lead to smoother solutions.

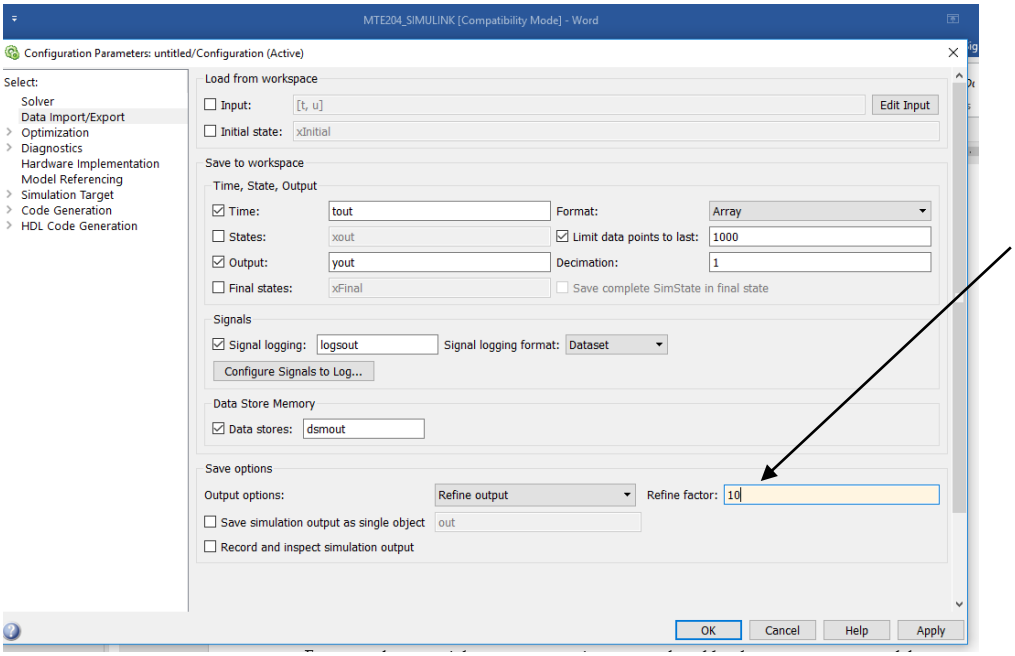


*Figure 4 System configuration parameter - Data Import/Export Parameters. Changing the **Refine Factor** to 10 units for smoother solution*

Hence, we obtain the scope output in Figure 5.

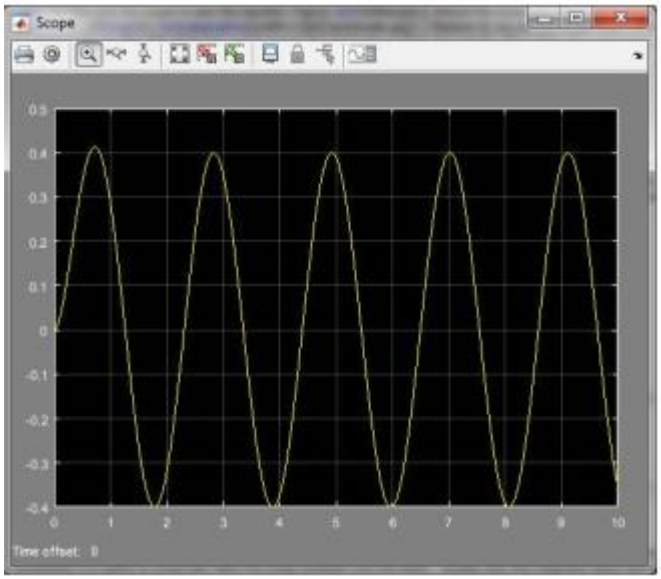


Figure 5 Scope plot of the solution, with **Refine Factor=** 10

Until now we have inputed the initial condition for the integrator internally. However, there are instances whereby we will linke to do same externally. Double-click the Integrator block and change the initial condition source from internal to external Figure 6. This adds another input to the block Figure 7. Drag a Constant block from the Sources group into the model, connect it to the new input, and change the constant value to the desired initial value. This results in the simulation shown in Figure 8.

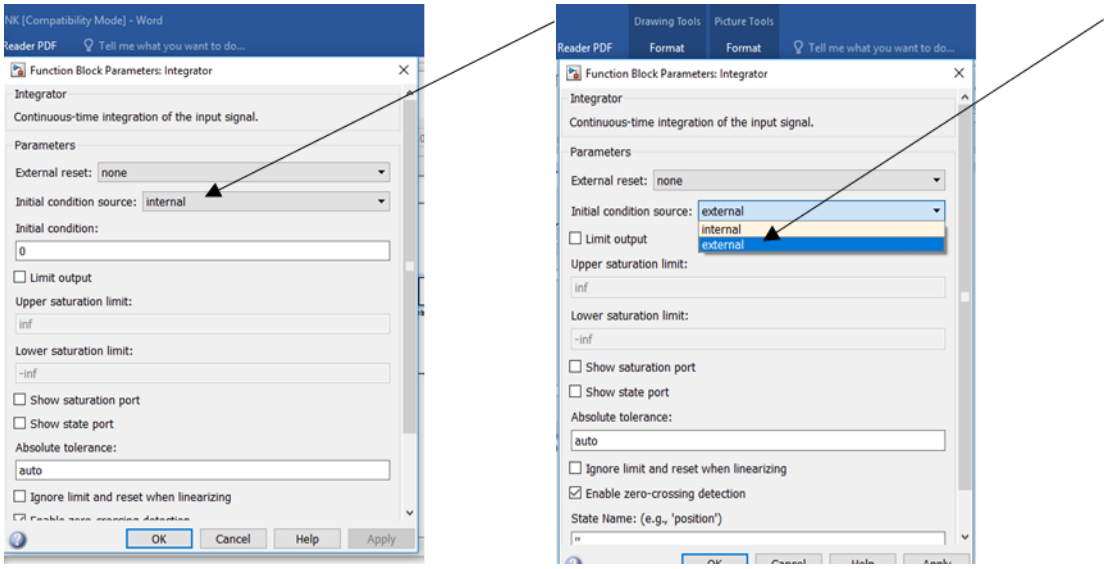


Figure 6 integrator block initial condition source

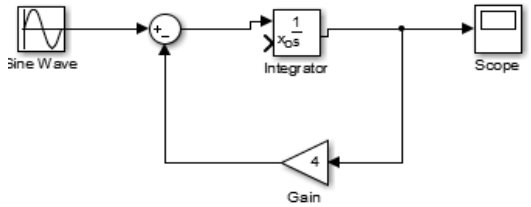


Figure 7 Additional input to integrator block

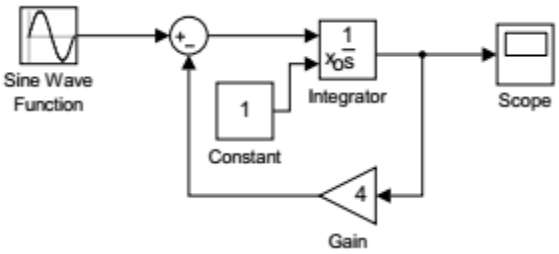


Figure 8 Constant block addition as external initial condition source for the integrator

2.2.2. Exponential Growth and Decay

The simplest differential equations are those governing growth and decay. As an example, we will discuss population models. Let P(t) be the population at time t. We seek an expression for the rate of change of the population, $\frac{dp}{dt}$. Assuming that there is no migration of population, the only way the population can change is by adding or subtracting individuals in the population. The equation would take the form

$$\frac{dp}{dt} = Rate\ In - Rate\ Out.$$

$$Rate\ In\ =\ bP\ and\ Rate\ Out\ =\ mP.$$

This gives the total rate of change of population as

$$\frac{dp}{dt} = bp - mp$$

$$where, k = b - m$$

$$Then,\quad \frac{dp}{dt} = kp$$

The equation above can be modelled easily in Simulink. Using Integrator, Constant, Gain, and a Scope block. Given the initial value, $P(0)\ =\ 8$ and $k\ =\ -0.8$

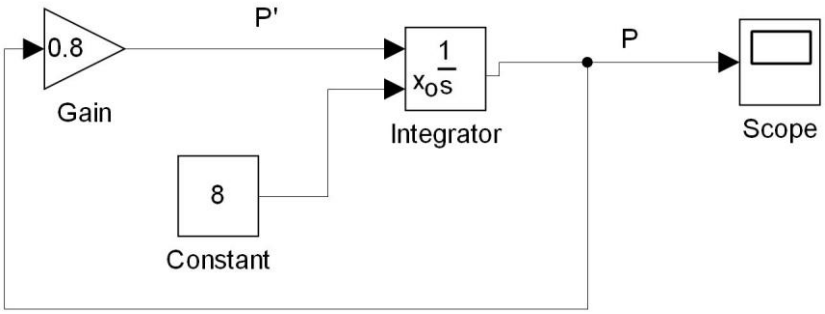


Figure 9 Simulink model for exponential growth and decay

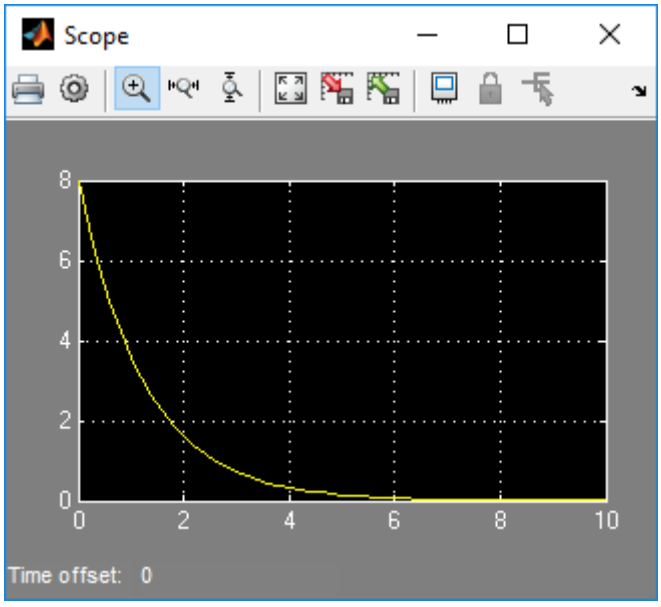


Figure 10 Solution for the exponential decay with $\boldsymbol{P}(0) = 8$ *and* $\boldsymbol{k} = -0.8$. *The simulation time was set at 10*

2.2.3. Newton's Law of Cooling

The law of cooling is attributed to Isaac Newton (1642-1727) who was probably the first to state results on how bodies cool. For instance A cup of hot tea, kept in a room will cool off and reach room temperature after a period of time. The main is idea is that a body at temperature $T(t)$ is initially at temperature $T(0) = T0$. It is placed in an environment at an ambient temperature of Ta. The goal is to find the temperature at a later time, $T(t)$.

Let assume that the rate of change of the temperature of the body is proportional to the temperature difference between the body and its surroundings. Thus, we have

$$\frac{dT}{dt} \propto \mathrm{T} - \mathrm{Ta}$$

The proportionality is removed by introducing a cooling constant,

$$\frac{dT}{dt} = -\mathrm{K}\,(\mathrm{T} - \mathrm{Ta})$$

Where, $K > 0$

Example 2.1. A cup of tea at 90^oC cools to 85^oC in ten minutes. If the room temperature is 22^oC, what is its temperature after 30 minutes?

Using the general solution with $T_0 = 90^o$C,

$$T(t) = 22 + (90 - 22)e^{-k} = 22 + 68e^{-kt},$$

we then find k using the given information, $T(10) = 85^o$C. We have

$$\begin{aligned} 85 &= T(10) \\ &= 22 + 68e^{-10k} \\ 63 &= 68e^{-10k} \\ e^{-10k} &= \frac{63}{68} \approx 0.926 \\ -10k &= \ln 0.926 \\ k &= -\frac{\ln 0.926}{10} \\ &\approx 0.00764\text{min}^{-1}. \end{aligned}$$

This gives the solution for this model as

$$T(t) = 22 + 68e^{-0.00764t}.$$

Now we can answer the question. What is $T(30)$?

$$T(30) = 22 + 68e^{-0.00764(30)} = 76^o\text{C}.$$

Example 2.2

Using Simulink; A cup of tea at 60°C is placed in a room with a temperature of 20°C. Find the temperature of a cup of tea at time 60s and 80s. given k = 0.1 s^{-1}

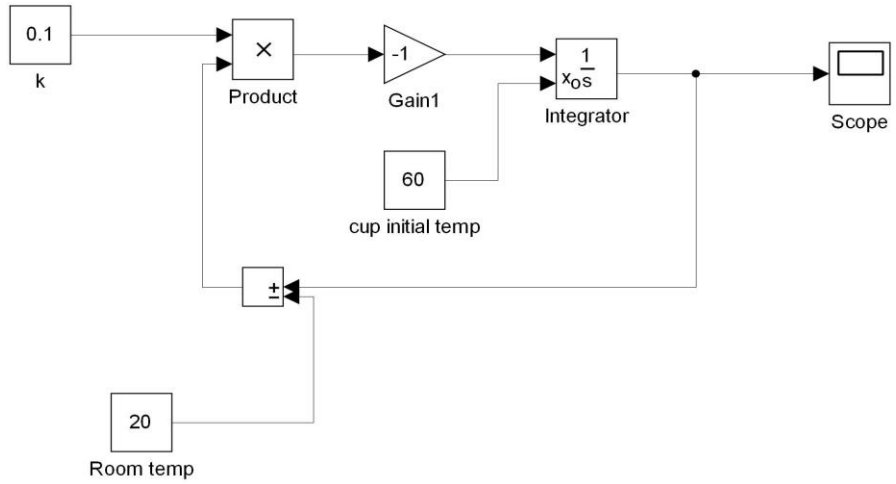


Figure 11 Simulink model Newton's law of cooling

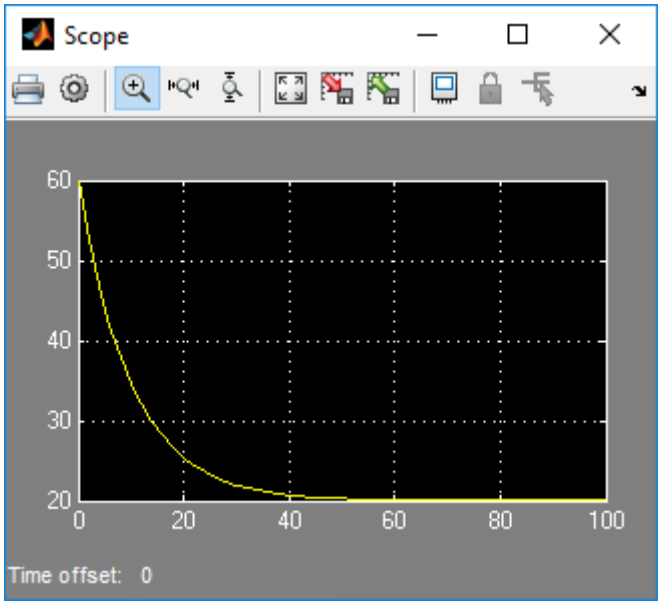


Figure 12 Solution for T0 = -k(T - Ta), T(0) = T0. Here we set k = 0.1 s-1, Ta = 20°C, and T0 = 60°C.

From the Figure 12, the cup temperature at 60s and 80s is 20°C

Note that example 2.1 shows the mathematical method where the equation to solve is $T(t) = T_a + (T_0 - T_a)e^{-kt}$

Where, T_a is the ambient temperature and T_0 is the body initial temperature.

2.2.4. Free Fall with Drag

Consider an object falling to the ground with air resistance? Free fall is the vertical motion of an object solely under the force of gravity. It has been experimentally determined that an object near the surface of the Earth falls at a constant acceleration in the absence of other forces, such as air resistance. This constant acceleration is denoted by −g, where g is called the acceleration due to gravity. The negative sign is an indication that we have chosen a coordinate system in which "up" is positive.

We are interested in determining the position, $y(t)$, of a falling body as a function of time. The differential equation governing free fall is have

$$\ddot{y}(t) = -g$$

However, the differential equation we need to solve is

$$\dot{v} = Kv^2 - g$$

Where, g is acceleration due to gravity, K is drag and v is velocity.

The Simulink model is presented in Figure 13 with K set as 0.00159

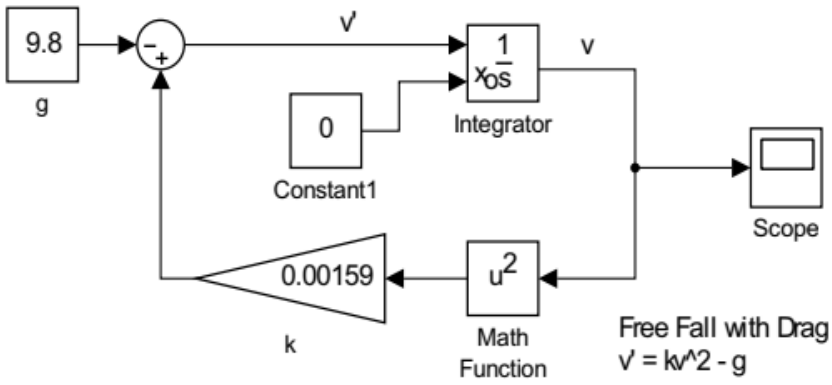


Figure 13 Simulation model for free fall

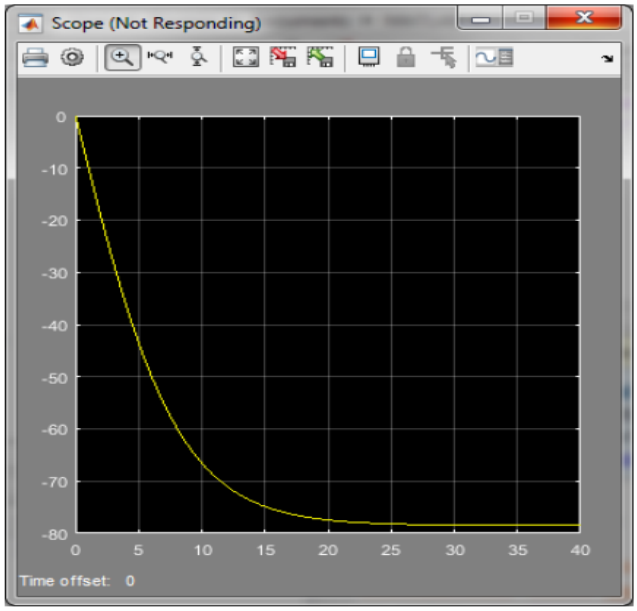


Figure 14 Solution for free fall with drag with **k** = 0.00159 starting from rest.

2.3. Working with Simulink Output

Often we might want to access the solutions in MATLAB. Using the model in Fig. 8 add the *To Workspace block from* Sinks Library. Double-click and rename the *simout* variable name as *y* and change the output type to array Figure 15. Run the simulation. This will put *tout* and *y* data into the MATLAB *workspace*. In MATLAB you can plot the data using plot (tout,y). You can add labels with xlabel('t'), ylabel('y'), title('y vs t').

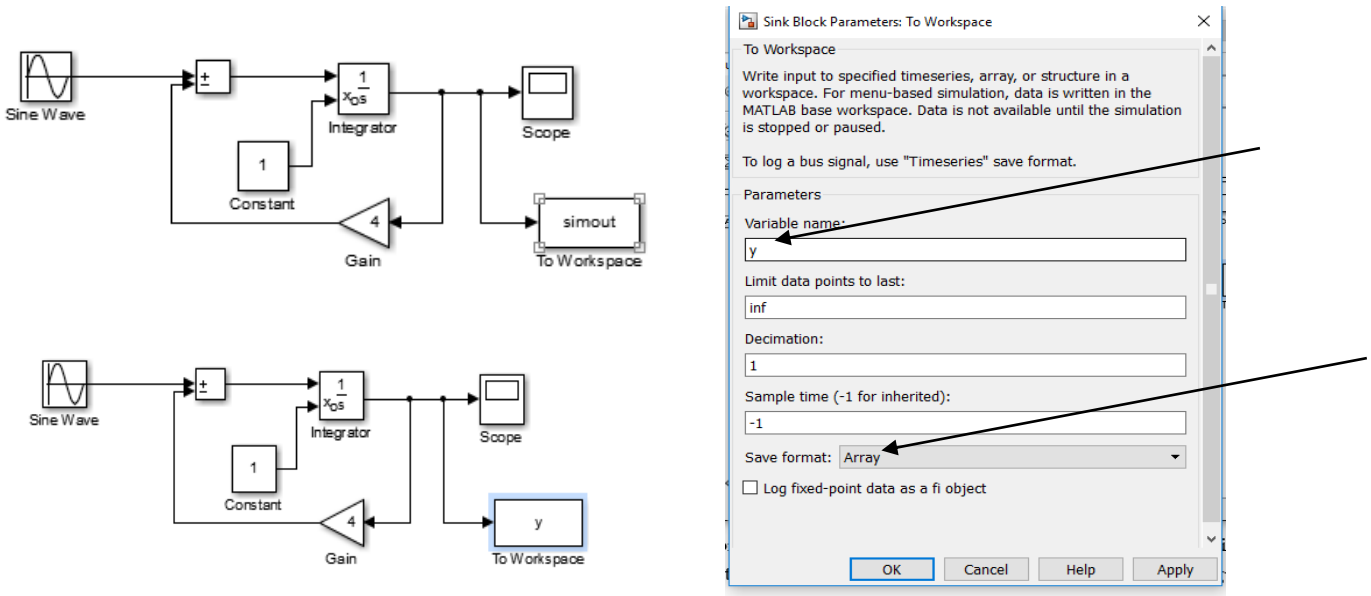


Figure 15 Outputting to MATLAB

2.4. Printing Simulink Scope Images

For example, one might want to copy images produced by the scope or your model into an MS Word document. Select the Scope Figure window in Fig. 5, then hit ALT+PrintScrn to copy the Figure to a clipboard and paste the Figure into your application.

2.5. Printing Models

Once you have made a model, you might want to include it in a report; this can be done by typing the following in the MATLAB command window:

1. To print the open model to an encapsulated postscript file:

```
print -s -deps -r300 mymodel.eps
```

2. For jpg files, you can use

```
print -s -djpeg -r300 mymodel.jpg
```

The picture file of the model will appear in MATLAB current folder.

3.0. How Simulink Works

Simulating a dynamic system is a two-step process with Simulink. First, a user creates a block diagram, using the Simulink model editor that graphically depicts time-dependent mathematical relationships among the system's inputs, states, and outputs. The user then commands Simulink to simulate the system represented by the model from a specified start time to a specified stop time.

3.1. Modeling Dynamic Systems

A Simulink block diagram model is a graphical representation of a mathematical model of a dynamic system.

3.2. Block Diagram Semantics

A classic block diagram model of a dynamic system graphically consists of blocks and lines (signals). The relationships between each elementary dynamic system in a block diagram are illustrated by the use of signals connecting the blocks. Collectively the blocks and lines in a block diagram describe an overall dynamic system.

There are two classes of blocks:

1. Nonvirtual blocks represent elementary systems. Blocks that plays specific role in the definition of the system of equations described by the block diagram model.
2. Virtual block is provided for graphical organizational convenience and plays no role in the definition of the system of equations described by the block diagram model. Examples of virtual blocks are the Bus Creator and Bus Selector which are used to reduce block diagram clutter by managing groups of signals as a "bundle." You can use virtual blocks to improve the readability of your models.

Because we use the term block diagrams in other fields, the term "*time-based block diagram*" is used to distinguish block diagrams that describe dynamic systems in Simulink from that of other forms of block diagrams.

Summary meaning of time-based block diagrams:

i. Simulink block diagrams define time-based relationships between signals and state variables. The solution of a block diagram is obtained by evaluating these relationships over time, where time starts at a user

specified "start time" and ends at a user specified "stop time." Each evaluation of these relationships is referred to as a time step.

ii. Signals represent quantities that change over time and are defined for all points in time between the block diagram's start and stop time.

iii. The relationships between signals and state variables are defined by a set of equations represented by blocks.

3.3. Time

Time is an inherit component of block diagrams in that the results of a block diagram simulation change with time. Simply put, a block diagram represents the instantaneous behavior of a dynamic system.

3.4. States

Typically, the current values of some system, and hence model, outputs are functions of the previous values of temporal variables. Two types of states can occur in a Simulink model: discrete and continuous states. A continuous state changes continuously. Examples of continuous states are the position and speed of a car. A discrete state is an approximation of a continuous state where the state is updated (recomputed) using finite (periodic or aperiodic) intervals. An example of a discrete state would be the position of a car shown on a digital odometer where it is updated every second as opposed to continuously.

Assignment

Using Simulink, model, simulate and analysis the system below:

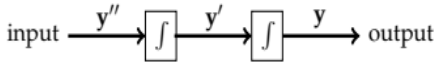
1. $\frac{d}{dt}y(t) + 5y(t) = x(t)$ where $x(t) = u(t)$, initial condition $y(0) = -2$
2. $\frac{dy}{dt} = \frac{6}{t}\, y$, where $y(1) = (1)$,
3. $\frac{dy}{dt} = \frac{6}{t}\, y + t^2$, where $y(1) = (1)$
4. Solution to the logistic equation, $y' = ry\,(1 - y)$, with $r = 1$ and $y(0) = 0.1$.
5. $\frac{dy}{dt} = y^2\,(1 + t^2)$ where $y(1) = (1)$
6. $\frac{ds}{dt} + 2s = st^2$ where $s(1) = (1)$
7. $x' + 2x = te^{2t}$ where $s(1) = (1)$
8. The temperature inside your house is 70°F and it is 30°F outside. At 1:00 A.M. the furnace breaks down. At 3:00 A.M. the temperature in the house has dropped to 50°F. Assuming the outside temperature is constant and that Newton's Law of Cooling applies, determine when the temperature inside your house reaches 40°F.

9. A body is discovered during a murder investigation at 8:00 P.M. and the temperature of the body is 70^{o}F. Two hours later the body temperature has dropped to 60^{o}F in a room that is at 50^{o}F. Assuming that Newton's Law of Cooling applies and the body temperature of the person was 98.6^{o}F at the time of death, determine when the murder occurred.

4.0. Second Order Differential Equations

These are equations involving the second derivative,$y''(x)$. Let's assume that we can write the equation as $y''(x) = F\ (x, y(x), y'(x))$.

We would like to solve this equation using Simulink. This is accomplished using two integrators in order to output $y'(x)$ and $y(x)$.



Example

Model the initial value problem $y''(x) + 5y' + 6y = 0,\ y'(0) = 1,\ y(0) = 0$ in Simulink

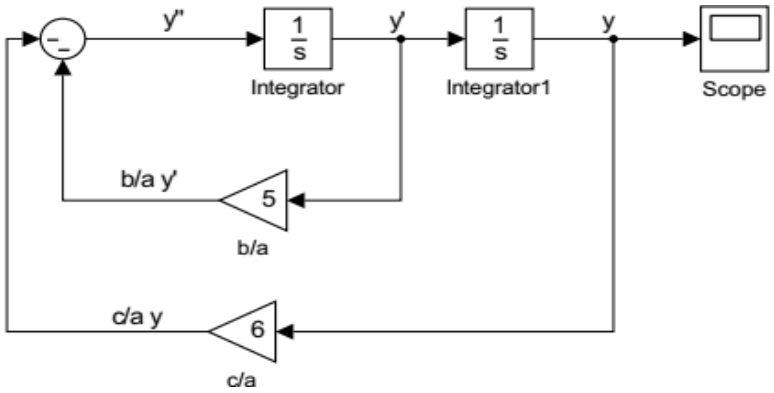


Figure 16 Model for the second order constant coefficient ODE $y''(x) + 5y' + 6y = 0$

4.1. Harmonic Oscillation

A typical application of second order, constant coefficient differential equations is the simple harmonic oscillator as shown in Figure 17. Consider a mass, m, attached to a spring with spring constant, k. According to Hooke's law, a stretched spring will react with a force $F = -kx$, where x is the displacement of the spring from its un-stretched equilibrium. The mass experiences a net for and will accelerate according to Newton's Second Law of Motion, F = ma. Setting these forces equal and noting that $a = \ddot{x}$, we have

$$m\ddot{x} + kx = 0$$

Hence, the differential equation we intend to solve with Simulink is $\ddot{x} = -\frac{1}{m}\ (kx)$.

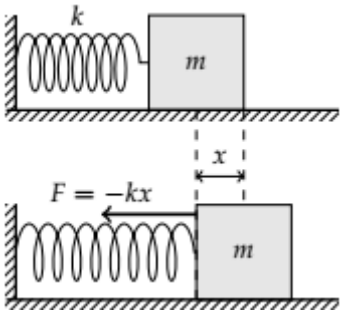


*Figure 17 A simple harmonic oscillator consists of a mass, **m**, attached to a spring with spring constant, **k***

Example

A Simulink model for simple harmonic motion where k = 5 and m = 2. We also specify the initial conditions $x(0) = 1$ and $\dot{x}(0) = 0$ is

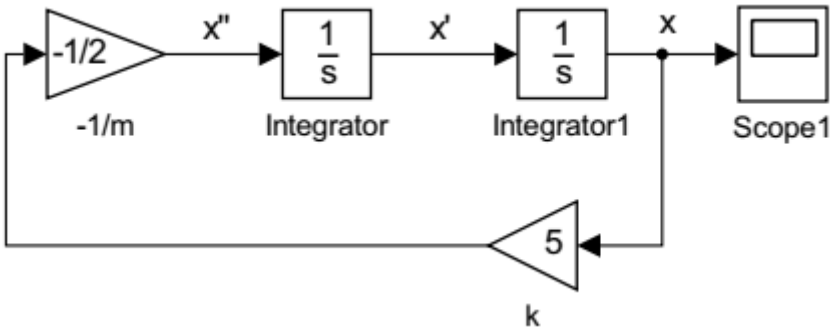


Figure 18 A model for damped simple harmonic motion, $m\ddot{x} + kx = 0$