An Introduction to Simulink® in Simulation of Instrumentation and Process Control Systems

By Dr. Hung Nguyen
(C) 2011 Hung Nguyen, Australian Maritime College/University of Tasmania
Email: nguyenhd@amc.edu.au
URL: http://academic.amc.edu.au/~hnguyen

Overview

Simulink® is a graphical application program accompanied with MATLAB and working in the MATLAB environment. Although the MATLAB package is useful for linear system analysis Simulink is far more useful for control system simulation. Simulink enables the rapid construction and simulation of control block diagrams.

Learning Outcomes

After completion of these tutorials, you should be able to

• Get started with Simulink®
• Explain how a Simulink model works
• Create and run a Simulink model for simulation of a temperature measurement system
• Simulate measuring systems
• Solve differential equations with Simulink®
• Simulate zero-, first- and second-order systems

Contents

• Introduction – Getting Started with Simulink
• Tutorial 1 – First Simulink Model and Simulation of Temperature Measurement System
• Tutorial 2 – Simulation of Zero-order, First-order and Second-order Systems
1. Products of MathWorks

http://www.mathworks.com/

Figure 1 Overview of MathWorks products

2. Toolboxes for Instrumentation and Process Control


MATLAB (Current Version 2011a 7.12 Released in March 2011)

Control System Design and Analysis
- Control System Toolbox
- System Identification Toolbox
- Fuzzy Logic Toolbox
- Robust Control Toolbox
- Model Predictive Control Toolbox
- Aerospace Toolbox

Test & Measurement
- Data Acquisition Toolbox
- Instrument Control Toolbox
- Image Acquisition Toolbox
- SystemTest
- OPC Toolbox
- Vehicle Network Toolbox
Simulink (R) (Current Version 7.7)

Physical Modeling
- Simscape
- SimMechanics
- SimPowerSystems
- SimDriveline
- SimHydraulics
- SimElectronics

Control System Design and Analysis
- Simulink Control Design
- Aerospace Blockset
- Simulink Design Optimization

Code Generation
- Real-Time Workshop
- Real-Time Workshop Embedded Coder
- Stateflow Coder
- Simulink HDL Coder

Rapid Prototyping and HIL Simulation
- xPC Target
- xPC Target Embedded Option
- Real-Time Windows Target

Control Engineering Lab (G51)

MATLAB (R2007b)
Math and Optimization
- Symbolic Math Toolbox

Control System Design and Analysis
- Control System Toolbox
- System Identification Toolbox

Test & Measurement
- Data Acquisition Toolbox

Simulink
Code Generation
- Real-Time Workshop

Rapid Prototyping and HIL Simulation
- xPC Target
3. Background of Simulink®

3.1 What Is Simulink?

Simulink, a companion program to MATLAB (the current version of MATLAB at AMC: R2010b), is an interactive system for simulating linear and nonlinear dynamic systems. It is a graphical mouse-driven program that allows you to model a system by drawing a block diagram on the screen and manipulating it dynamically. It can work with linear, nonlinear, continuous-time, discrete-time, multivariable, and multi-rate systems.

3.2 Simulink Library Browser

The first step is to start up MATLAB on the computer you are using. Type `>>simulink` in the Command Window or click the button in the Toolbar menu of the MATLAB Window. The Simulink Library Browser window will appear as that shown in Fig. 1.

![Simulink Library Browser](image)

**Figure 1** Simulink library browser (MATLAB R2008b (7.7), Simulink 7.2)
Simulink® 7.2 has a number of additional options. There are several groups of Simulink blocks in the Simulink icon such as Commonly Used Blocks, Continuous, Discontinuities, Math Operations, Sinks and Sources, etc. Selecting Commonly Used Blocks will provide a list of blocks shown in Figure 2.

Figure 2 A list of blocks in Commonly Used Blocks group
Selecting Continuous will provide a list of blocks shown in Fig. 3. The ones that we often use are Transfer Fcn, State-space and Integrator.

Selecting the Sources icon yields the library shown in Fig. 4. The most commonly used sources are Clock (which is used to generate a time vector), Step (which generates a step input), and Constant (that generate a constant function).

The Sinks icon as shown in Fig. 5 provides a set of Sinks blocks that are used to display simulated results. The most often used blocks may be To Workspace (to which a variable passed is written to a vector in the MATLAB Workspace), Scope (to represent data graphically).

Figure 3 A list of blocks in Continuous group
**Figure 4** A list of blocks in the Sources group
Figure 5 A list of blocks in Sinks group
3.3 Concept of Signal and Logic Flow

In Simulink, data/information from various blocks is sent to another block by lines connecting the relevant blocks. Signals can be generated and fed into blocks (dynamic/static). Data can be fed into functions. Data can then be dumped into sinks, which could be scopes, displays or could be saved to a file. Data can be connected from one block to another, can be branched, multiplexed etc. In simulation, data is processed and transferred only at Discrete times, since all computers are discrete systems. Thus, a SIMULATION time step (otherwise called an INTEGRATION time step) is essential, and the selection of that step is determined by the fastest dynamics in the simulated system.

3.4 Connecting Blocks

A Simulink model is created based on the idea of block diagram and it consists of several blocks. Necessary blocks for a Simulink model can be selected from the Simulink Library Browser and they are connected together. To connect blocks, left-click and drag the mouse from the output of one block to the input of another block. Figure 6 shows the steps involved. The Simulink model in Fig. 6 consists of the following blocks: Clock, To Workspace1, Step Input, Transfer Fcn and To Workspace2.

![Connecting blocks diagram](image)

**Figure 6** Connecting blocks

4. Procedure for Simulink Simulation

With Simulink we can develop block diagram algorithms to solve ODEs or do scientific computation. Although computer simulations can be used to model a large variety of systems, it can be seen that all computer simulations must embody the following components:

1. **Structure of the mathematical model:** This is the complete set of differential equations that describe the system behaviour (dynamics) and reflect the fundamental physical laws governing the behaviour of the system.
2. **Model parameter values:** Model parameters refer to numerical constants that usually do not change over the course of the simulation. Typical parameters for mechanical systems are mass, damping coefficient, and spring stiffness.

3. **Initial conditions:** Initial conditions are important for determination of the solution of ODEs. Therefore in the simulation program initial conditions must be set.

4. **Inputs:** Typically a system responds to one or more inputs. The simulation must embody the inputs as well. In simulation tools, there are some blocks that generate test input signals such as a step function, a ramp function or a sine-wave function.

5. **Outputs:** Although a simulation does not require that the user explicitly define outputs, it is assumed that the goal of a computer simulation of a dynamic system is the time history of specific physical variables in the system under study. The time history of output variables can be stored to the computer hard drive for later analysis or displayed as graphs on the screen. The simulation tools often have blocks for displaying simulated outputs.

6. **Simulation solution control parameters:** Simulation solution control parameters define the values and choices made by the designer/engineer of the simulation tool who dictates how the numerical methods behind the simulation operate. These include values that determine the step size, output interval, error tolerance, and choice of numerical integration algorithm.

**Example:** in order to calculate \( y = au + b \) we develop a block diagram as follows:

![Block diagram](image)

**Figure 7** An example of block diagram algorithm

5. **Commonly Used Simulink Blocks**

Table 1 summarises a list of commonly used Simulink blocks which we will the most often use in our course.
Table 1 Summary of Commonly Used Simulink Blocks

<table>
<thead>
<tr>
<th>Block icon</th>
<th>Name</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="State-Space" /></td>
<td>State-Space</td>
<td>Implement a linear state-space system</td>
</tr>
<tr>
<td><img src="image2.png" alt="Transfer Fcn" /></td>
<td>Transfer Fcn</td>
<td>Implement a linear transfer function</td>
</tr>
<tr>
<td><img src="image3.png" alt="Derivative" /></td>
<td>Derivative</td>
<td>Merge scalar, vector or matrix signals</td>
</tr>
<tr>
<td><img src="image4.png" alt="Divide" /></td>
<td>Divide</td>
<td>Multiply or divide inputs</td>
</tr>
<tr>
<td><img src="image5.png" alt="Function" /></td>
<td>Function</td>
<td>Apply a specified expression to the input</td>
</tr>
<tr>
<td><img src="image6.png" alt="Gain" /></td>
<td>Gain</td>
<td>Multiplies the input by a constant value (gain)</td>
</tr>
<tr>
<td><img src="image7.png" alt="Integrator" /></td>
<td>Integrator</td>
<td>Integrate the input signal</td>
</tr>
<tr>
<td><img src="image8.png" alt="Math Function" /></td>
<td>Math Function</td>
<td>Perform a mathematical function</td>
</tr>
<tr>
<td><img src="image9.png" alt="Product" /></td>
<td>Product</td>
<td>Multiply inputs</td>
</tr>
<tr>
<td><img src="image10.png" alt="Sum" /></td>
<td>Sum</td>
<td>Add or subtract inputs</td>
</tr>
<tr>
<td><img src="image11.png" alt="Transport Delay" /></td>
<td>Transport Delay</td>
<td>Delay the input by a given amount of time</td>
</tr>
<tr>
<td><img src="image12.png" alt="Demux" /></td>
<td>Demux</td>
<td>Split vector signals into scalars or smaller vectors</td>
</tr>
</tbody>
</table>
12

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Mux" /></td>
<td>Mux: Extract and output the elements of a bus or vector signal</td>
</tr>
</tbody>
</table>

**Sinks**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Scope" /></td>
<td>Scope: Display signals generated during a simulation</td>
</tr>
<tr>
<td><img src="image" alt="To Workspace" /></td>
<td>To Workspace: Write data to the workspace</td>
</tr>
<tr>
<td><img src="image" alt="XY Graph" /></td>
<td>XY Graph: Display an X-Y plot of signals using a MATLAB figure window</td>
</tr>
</tbody>
</table>

**Sources**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Clock" /></td>
<td>Clock: Generate a time vector</td>
</tr>
<tr>
<td><img src="image" alt="Constant" /></td>
<td>Constant: Generate a constant</td>
</tr>
<tr>
<td><img src="image" alt="Ramp" /></td>
<td>Ramp: Output a ramp signal</td>
</tr>
<tr>
<td><img src="image" alt="Sine Wave" /></td>
<td>Sine Wave: Generate a sine wave signal</td>
</tr>
<tr>
<td><img src="image" alt="Step" /></td>
<td>Step: Generate a step signal</td>
</tr>
</tbody>
</table>

6. Summary

Simulink is a very powerful block diagram simulation language. Simple simulations can be set up rapidly. The aim of this tutorial was to provide enough of an introduction to get you started on the development of simulations for dynamic systems. With experience, the development of these simulations will become second nature. It is recommended that you perform the simulations shown in these tutorials as well as the follow-up exercises, to rapidly acquire these simulation skills.

Further information

http://www.mathworks.com/help/toolbox/simulink/
http://www.mathworks.com/products/simulink/
Simulink® Tutorial 1
Simulation of a temperature measuring system

1. Aims
- To use Simulink as a simulation and diagnostic tool for dynamic systems
- To simulate instrumentation systems with Simulink
- To do scientific computation with Simulink

2. Learning Outcomes
Upon completion of this Tutorial you will be able to:
- Create and run a Simulink model
- Compute value of a function and display computed results with Simulink
- Use blocks in Simulink Library Browser
- Configure system parameters
- Configure simulation parameters
- Store simulated data to Workspace and save Workspace to files

3. Hands-on Exercises

Hands-on Exercise 1: Create and Run a simple Simulink model

Statement of Problem
Create a Simulink model to simulate a signal that have the following form:
\[ y(t) = 2 + 3\sin\left(\frac{\omega t + \pi}{2}\right) \quad (t = 0 \text{ to } 10, \quad \omega = 2 \text{ rad/s}) \]  

(1)

In the Simulink simulation program, do the following:
- Compute value of \( y \) vs \( t \) (from 0 to 10 seconds)
- Display value of \( y \) (with digital display and graph)
- Save data to Workspace

SOLUTION

Block diagram algorithm:
Block diagram algorithm for function (1) is given in Fig. 1.

[Diagram]

Figure 1 Block diagram algorithm for equation (1)
Create a Simulink model:
Do the following steps:
• Start Simulink
• Open a new model – Save the Simulink model as MyFirstSimTute01Ex01.mdl
• Select a Constant block (Sources)
• Select a Sine Wave block (Sources)
• Select a Sum block (Math Operations)
• Select a Clock block (Sources)
• Select a Display (Sinks)
• Copy Display to Display 1
• Select a Scope (Sinks). The Simulink model should look like the one in Fig. 2.
• Save the Simulink model.
• Connect/wire blocks as shown in Fig. 3.

Figure 2 Start a simple Simulink model
Figure 3 The Simulink model ‘MyFirstSimTute01Ex01.mdl’

System Parameters:
- Double-click the Constant, change the Constant value to 2 as shown in Fig. 4.
- Click the OK button.

Figure 4 Change parameter of the Constant block

- Double-click the SineWave block, do some changes as shown in Fig. 5.
- Click OK button.
Configuration of Simulation Parameters:

- Before running the Simulink model, we need to set simulation parameters by selecting the Simulation menu → Configuration Parameters:
The Configurations dialog box will appear as shown in Fig. 7. In this dialog box we can set some parameters such as Start time 0.0, Stop time: 10.0 (seconds), and Solver: ode45. Click OK button. The Simulink model is ready to run.

![Configuration Parameter dialog box](image)

**Figure 7** Configuration Parameter dialog box

**Run Simulink model:**

- Double the Scope block and move it to a desired position on the screen.
- Run the Simulink model by selecting Simulation → Start or click the button.

The resulting Simulink model should look like...
Figure 8 Resulting Simulink model
Store Data to Workspace

1. Use of To Workspace and a Mux

Simulated data (including time, t, and value of the function (1), y in a matrix)
- Launch a To Workspace block from the Simulink Library Browser > Sinks
- Launch a Mux block (from Commonly-Used Blocks)
- Wire them as shown in the following figure:

![Simulink diagram showing To Workspace and Mux blocks](image)

**Figure 10** Launching To Workspace and Mux blocks

- Double click the To Workspace, and set:
  - Variable name: data
  - Save format: Array
- Click the Apply button then the OK button.
Figure 11 Sink Block Parameters: To Workplace Window

- Save the Simulink model
- Run the Simulink model. The simulated data are in Workspace (matrix “data”).
- In the Command Window, type the following commands:

\[
\begin{align*}
\text{>>t} &= \text{data}(:,1) \\
\text{>>y} &= \text{data}(:,2) \\
\text{>>plot(t,y);grid}
\end{align*}
\]

Figure 12 Graph of \( y = 2 + 3 \sin \left( \omega t + \frac{\pi}{2} \right) \)
2. Use of Scope

- Save the Simulink model as “MyFirstSimTute01Ex0102.mdl”.
- Double click the Scope block
- Click the Parameter button:

![Scope Parameters](image1)

**Figure 13** Scope > Parameters button

- Select the History tab
- Set the ‘Scope’ parameters as in the following figure:

![Scope Parameters](image2)

**Figure 14** Data history tab

- Click Apply button
- Click OK button
- Save the Simulink model
- Run the model
- View Workspace... there must be a variable named “ScopeData”
In the Command Window, type the following command:

```matlab
>> ScopeData
```

that is resulting...

```matlab
ScopeData =
```
```matlab
    time: [10001x1 double]
    signals: [1x1 struct]
    blockName: 'MyFirstSimTute01Ex0102/Scope'
```

Let’s extract signals from the ScopeData variable by typing the following commands in Command Windows:

```matlab
>> t = data.time
( to retrieve time)
```

```matlab
>> y = ScopeData.signals.values
(to retrieved signal y)
```

```matlab
>> plot(t, y); grid
```

The resulting plot looks like...

![Figure 15 Plotting data (y vs t) from ScopeData](image)
Selection of Solver and Step Size:
- Simulation menu > Simulation parameters > see the following figure:

Figure 16 Change Variable step to Fixed-step and Solver to Ode4 (Runge-Kutta)

- Click Apply button, then OK button. Run the Simulink model and plot y vs t:

Figure 17 Smoother graph of \( y = 2 + 3\sin(\omega t + \pi/2) \)

Save Workspace to a MAT-formatted files:

In some cases it is necessary to save all data in Workspace to files. In order to view the contents of Workspace, select the Workspace tab as follows:
Figure 18 Workspace tab

Or type the following command in Command Window:

```matlab
>> whos
```

That results in

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScopeData</td>
<td>1x1</td>
<td>161080</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>tout</td>
<td>1000x1</td>
<td>8000</td>
<td>double</td>
<td></td>
</tr>
</tbody>
</table>

Type the following command to save all data in Workspace to “Tute01Ex01.MAT” file

```matlab
>> save Tute01Ex01
```
The file will be in the Current Directory as follows:

![Current Directory](image)

**Figure 19** A new MAT file has been created

To load the MAT file, type the following commands:

```matlab
>> clear
>> whos
>> load Tute01Ex01
>> whos
```

That results…

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScopeData</td>
<td>1x1</td>
<td>161080</td>
<td>struct</td>
<td></td>
</tr>
<tr>
<td>tout</td>
<td>1000x1</td>
<td>8000</td>
<td>double</td>
<td></td>
</tr>
</tbody>
</table>
2. Hands-on Exercise 2: Simulation of a temperature control system

Statement of Problem: Temperature measurement

A PC-based temperature measuring system is shown in Fig. 14 in which the signal is processed as: 0-100°C > mV > mA (4-20 mA) > SC-2345: 1-5 [V] – DAQ : 1-5V (software).

Figure 14 PC-based temperature measuring system

In the Simulink model, perform signal processing (conditioning) and display:
- Temperature in V (uin: 1-5 V) – simulated by a signal generator block
- Temperature in mA (u: 4-20 mA)
- Temperature in °C (y: 0-100°C)

SOLUTION

The signal processing in PC/software will be:

\[
\begin{align*}
\text{uin [V]} & \quad \rightarrow \quad K_1 \rightarrow \quad u_1 [A] \\
1-5 \text{ V} & \quad \rightarrow \quad 0.004-0.020 \text{ A} \\
\text{K_2} & \quad \rightarrow \quad u [mA] \\
\text{K_3} & \quad \rightarrow \quad y [^\circ\text{C}] \\
0-100 \text{ ^\circ C} & \quad \rightarrow \quad \text{% of span 0-100\%} \\
\end{align*}
\]

Therefore we have the following algorithm

Ohm’s law:

\[
u_1 [A] = \frac{\text{uin} \times K_1}{R} = \text{uin} \times \frac{1}{R} = \text{uin} \times \frac{1}{250} (R = 250 \text{ Ω}, \text{ thus } K_1 = 1/R = 1/250)
\]

Convert A to mA:

\[u = u_1 \times K_2 (K_2 = 1000)\]

Convert mA to °C:

\[
K_3 = \frac{\Delta y}{\Delta u} = \frac{y_{\text{max}} - y_{\text{min}}}{u_{\text{max}} - u_{\text{min}}} = \frac{100^\circ \text{C} - 0^\circ \text{C}}{20\text{mA} - 4\text{mA}} = \frac{100^\circ \text{C}}{16\text{mA}} = 6.25 \frac{^\circ \text{C}}{\text{mA}}
\]

\[
K_3 = \frac{\Delta y}{\Delta u} = \frac{y - y_{\text{min}}}{u - u_{\text{min}}} \Rightarrow y = K_3(u - u_{\text{min}}), u_{\text{0}} = u_{\text{min}} = 4 \text{ mA, thus: } y = K_3(u - u_{\text{0}})
\]
Convert $V$ to $\%$ of span:

\[
\% \text{ of span} = \frac{u_{\text{in}_{\text{max}}} - u_{\text{in}_{\text{min}}}}{5V - 1V} = 4V
\]

\[
K_4 = \frac{100\%}{4} = 25[\%]
\]
or

\[
\% \text{ of span} = K_4 \left( u_{\text{in}_{\text{max}}} - u_{\text{in}_{\text{min}}} \right) = 25\% \left( 5V - 1V \right)
\]

Block diagram algorithm:

![Block Diagram for Temperature Measurement](image)

**Figure 15** Block diagram for temperature measurement

**Programming with Simulink**

- Open a new Simulink model
- Save as "TempMeasurementSim_Tute01_Ex02.mdl"
- Refer to the block diagram algorithm in Fig. 15 select all necessary blocks from the Simulink Library Browser as in Fig. 16.
- Wire the blocks and set system parameters as in Fig. 17 (demonstrate each step).
Figure 16 Necessary blocks for block diagram algorithm in Fig. 15

Figure 17 System parameters
• Configure simulation parameters as in the following figure:

**Figure 18** Simulation parameters

• Save the program.
• Run the Simulink program… The resulting program looks like…

**Figure 19** Simulation program for temperature measurement
• Luckily so far there is no error!

You can test the functionality of the program by changing some values of uni below:

\[
\begin{align*}
\text{uni} &= 1 \, \text{V} \rightarrow 4 \, \text{mA} \rightarrow 0^\circ \text{C} \rightarrow 0\% \\
\text{uni} &= 2 \, \text{V} \rightarrow 8 \, \text{mA} \rightarrow 25^\circ \text{C} \rightarrow 25\% \\
\text{uni} &= 3 \, \text{V} \rightarrow 12 \, \text{mA} \rightarrow 50^\circ \text{C} \rightarrow 50\% \\
\text{uni} &= 4 \, \text{V} \rightarrow 16 \, \text{mA} \rightarrow 75^\circ \text{C} \rightarrow 75\% \\
\text{uni} &= 5 \, \text{V} \rightarrow 20 \, \text{mA} \rightarrow 100^\circ \text{C} \rightarrow 100\%
\end{align*}
\]

WELL-DONE!

Conclusions

At this point the following learning objectives have been met:

• Create and run a Simulink model
• Compute value of a function and display computed results with Simulink
• Use blocks in Simulink Library Browser
• Configure system parameters
• Configure simulation parameters
• Store simulated data to Workspace and save Workspace to files

Follow-up Exercise Simulation of a level measurement system

1. A PC-based level measuring system is shown in Fig. 20 in which the signal is processed as: 0-400mm > mA (4-20 mA) > SC-2345: 1-5 [V] – DAQ : 1-5V (software). Make a Simulink model to simulate the level measuring system (you can modify the above simulation program).

Figure 20 PC-based level measuring system
Tutorial 2
Simulation of Dynamic Systems

Aim
• To get familiar with Simulink
• To simulate zero-order systems, first-order systems and second-order systems
• To improve programming skills with Simulink
• To use simulation programs to learn dynamics of second-order systems

Learning Objectives
Upon completion of this tutorial we will be able to
• Explain dynamics of zero-order systems
• Explain dynamics of first-order systems and second-order systems
• Solve first-order ODEs with Simulink and second-order ODEs
• Create a subsystem

Hands-on Exercises
Exercise 1 Zero-order systems (ideal systems)
A zero-order system is expressed by the following equation:
\[ y(t) = Ku(t) \]  
where K is gain, y(t) is output and u(t) the input. Make a Simulink model to simulate (1).

SOLUTION
• Open a Simulink model and save as “ZeroOrderSysSimTute02_01.mdl”

---

**Figure 1** Simulink models for zero-order systems
The resulting Scopes for \( K = 4 \) (step function) and \( K = 5 \) (sine wave input) are shown in Fig. 2.

![Scope Images]

(a) Step function response  
(b) Frequency response

**Figure 2** Zero-order system responses

### Exercise 2 First-order Systems

**State of Problem:** A storage tank system is shown in Fig. 3. By applying the mass balance principle in Fluid Mechanics the relationship between the level \( h \) and the inlet flow rate \( \text{m}^3/\text{min} \) is derived as follows:

\[
Ah + K_v \sqrt{h} = q_{in}
\]  

(2)

![Diagram of Tank Level System]

**Figure 3** Tank level system
Equation (2) is a non-linear differential equation. It is hard to solve ODEs with analytical methods. With the aid of computer and software a non-linear differential equation can be solved using numerical integration methods. Make a simulation program with Simulink to solve Equation (2). Use the following numerical values: $K_v = 0.000187$, $d = 150$ mm, level in range of 0 to 400 mm, $q_{in}$ in range of 0 to 0.0071 m$^3$/min. In the Simulink model, do the following tasks:

- Convert the flow rate $q_{in}$ from m$^3$/min to m$^3$/s
- Convert the level $h$ from m to mm
- Set low alarm limit (20 mm) and high alarm limit (380 mm) for the level
- Simulate a level transmitter providing a level signal in range of 4 to 20 mm.

**SOLUTION**

**Block Diagram Algorithm**

Using the SI units the tank level system is represented by the block in Fig. 4.

![Figure 4 Block diagram of the tank level system](image)

Equation (2) for the tank level system can be rewritten as follows for a Simulink program:

$$
\dot{h} = \frac{1}{A} \left( -K_v \sqrt{h} + q_{in} \right)
$$

(3)

In order to solve equation (3) we develop a block diagram algorithm below:

![Figure 5 Block diagram algorithm for solving a first-order differential equation with Simulink](image)

The cross-sectional of the tank is calculated by the following equation:

$$
A = \pi \frac{d^2}{4} \ [m]
$$

(4)
Programming

- Open a Simulink model and save as “FirstOrderSysSim_Tute02_02_01.mdl”.
- Using the above block diagram and make a Simulink model for solving equation (3). The Simulink model may look like that in Fig 6.

**Figure 6** Simulink model to solve a first-order ODE

- Save the model.

**System Parameters**

- Set all system parameters for the model.
- Set initial conditions for the Integrator block by double-clicking it and set:
  - \( h(0) = 0 \), check Limit output (upper: 0.4 m, lower: 0.0)

**Figure 7** Set initial conditions for the Integrator block
• Save the model. The resulting program looks like...

Figure 8 Simulink model after setting the Integrator block

Simulation Parameters

• Set the simulation parameters as in the following figure
• Click the OK button.

Figure 9 Configuration parameters
• Save the model.
• Run the model and test its functionality. The results (for $q_{in} = 0.0035$ m$^3$/min) are shown in Fig. 10 and Fig. 11.

![Simulink model](image)

**Figure 10** The Simulink model is running for $q_{in} = 0.0035$ m$^3$/min

![Resulting Scopes](image)

**Figure 11** Resulting Scopes (Level in mm and mA)

**Create a Subsystem** (for cross-sectional area calculating algorithm)

• Select blocks as in Fig. 12.
• Edit menu > Create Subsystem (or Right-click the selected blocks > Create Subsystem).

![Figure 12 Selecting blocks](image)

The resulting program looks like...

![Figure 13 Created a Subsystem](image)

• Change Subsystem to A, double click the A block, change the output port (Out1) to A as the figure below.
Figure 9 Subsystem A for cross-sectional area

- Save the model as “FirstOrderSysSimTutorial_02_02_02.mdl”.
- Run the Simulink model and test its functionality. The same results should be obtained.

Figure 10 Simulink model with a Subsystem (A)
Exercise 3 Simulation of Second-order Systems

Background

A mass-spring-damper system is shown in Fig. 1 in which the input force $u$ is in N, $m$ is mass [kg], $k$ is spring stiffness, and $b$ is viscous damping coefficient, $y$ is the displacement of the mass. Use these numerical values: $m = 20$ kg, $k = 2.0$ N/m, and $b = 4.0$ N/(m/s) and $u = -8/+8$ N.

The mass-spring-damper is expressed by a second-order ODE below:

$$m\ddot{y} + b\dot{y} + ky = u$$  \hspace{1cm} (5)

Equation (1) can be rewritten as follows:

$$\ddot{y} = \frac{1}{m} \left(-ky - b\dot{y} + u\right)$$  \hspace{1cm} (6)

A block diagram algorithm for Equation (6) is shown in Fig. 12.

Programming

- Open a Simulink model and save as “SecondOrderSysSim_Tute_02_03_01.mdl”.
- Add necessary block and wire them as in the following figure.
System Parameters

\[ U = 5.2 \text{ [N]}, \quad m = 20 \text{ [kg]}, \quad b = 4 \text{ [N/(m/s)]} \quad \text{and} \quad k = 2 \text{ [N/m]} \]

Initial Conditions

- Double click Integrator, \( ydot(0) = 0 \)
- Double click Integrator 1, \( y(0) = 0 \).
- Save the program.

Simulation Parameters

- Fixed step: 0.1
- Run the simulation program.

The result (Fig. 14) is obtained.

You can test functionality by changing values of input force \( u \).

- Save the program (if it is not).

Conclusions

At this point, the following LOs have been met:

- Explain dynamics of zero-order systems
- Explain dynamics of first-order systems and second-order systems
- Solve first-order ODEs with Simulink and second-order ODEs
- Create a subsystem
Follow-up Exercises

1. Make a Simulink model to simulate the following RC Circuit expressed by the first-order differential equation:
   \[ RC \dot{e}_o + e_o = e_i \]
where \( e_o \) and \( e_i \) are output and input voltages, \( R \) and \( C \) are resistance and capacitance, respectively. This equation is rewritten in a standard first-order ODE below:
   \[ \dot{e}_o + \frac{e_o}{T} = K e_i \]
where \( T (= RC) \) is time constant (seconds), and \( K (=1) \) is sensitivity (gain, V/V).

![Figure 11 An RC network](image)

Use these numerical values: \( R = 10 \, k \Omega \); \( C = 10^{-4} \, \mu F \); \( e_i(t) = A_a \sin(2 \pi f_a) + A_b \sin(2 \pi f_b) + V_i \)

Note: Use may compare results with those of the following RC circuit simulator:
http://techteach.no/simview/rc_circuit/index.php

2. A temperature transmitter is represented by a first-order differential equation below:
   \[ \tau \dot{y} + y = Ku \]
where \( \tau \) is the time constant (5 sec), \( K \) the sensitivity/gain (10\(^{-7}\) V/°C), \( y \) and \( u \) are the output voltage (V) and input temperature (°C), respectively. The input temperature is a step function having initial value of 0 °C and final value of 100 °C. Make a Simulink model to simulate the temperature transmitter displaying temperature in mA and °C. Save the simulated results (input temp and output voltage). The resulting data should look like:

![Temperature data graph](image)

You can change the value of \( \tau \) (e.g. 10 sec, 15 sec), run the simulation program. What is the value of output (percentage of span) when \( t = \tau \)? Suggest a method to estimate \( \tau \) and \( K \) from the simulated data.
3. The following RLC network is represented by a second-order ODE below:

\[ LC\ddot{e}_o + RC\dot{e}_o + e_o = e_i \]

Figure 18 An RLC network

Make a Simulink model to simulate the network. Use these numerical values: \( R = 100 \, \Omega, \, L = 0.2 \, H, \) and \( C = 2000 \, \mu F, \) \( e_i(t) = 5 \, V. \)

4. Transfer Function Block: Investigate how to use the Transfer Function block and make simulation programs for the mass-spring-damper system in hands-on Exercise 2 and the RLC network in Problem 3.