

<https://www.halvorsen.blog>



Simulation Examples

Hans-Petter Halvorsen

Introduction

- We want to simulate a 1. order differential equation using LabVIEW
- We will use 3 different alternatives:
 - MathScript Window (similar to MATLAB)
 - MathScript Node inside LabVIEW
 - LabVIEW Control Design and Simulation Module
- We should of course expect the same simulation results using the 3 different alternatives



Model

Hans-Petter Halvorsen

Mathematical Model

In this example we will use the following 1. order differential equation:

$$\dot{x} = -ax + bu$$

Note that $\dot{x} = \frac{dx}{dt} = x'(t)$

Different notation is used in different textbooks and examples

We can set, e.g., $a = 0.25$ and $b = 2$ in the simulations

Simulation

- We want to simulate this differential equation by applying a step in the input signal $u = 1$ at $t = 0s$
- Then we will observe the simulation results (Step Response) by plotting the results

Alternative #1



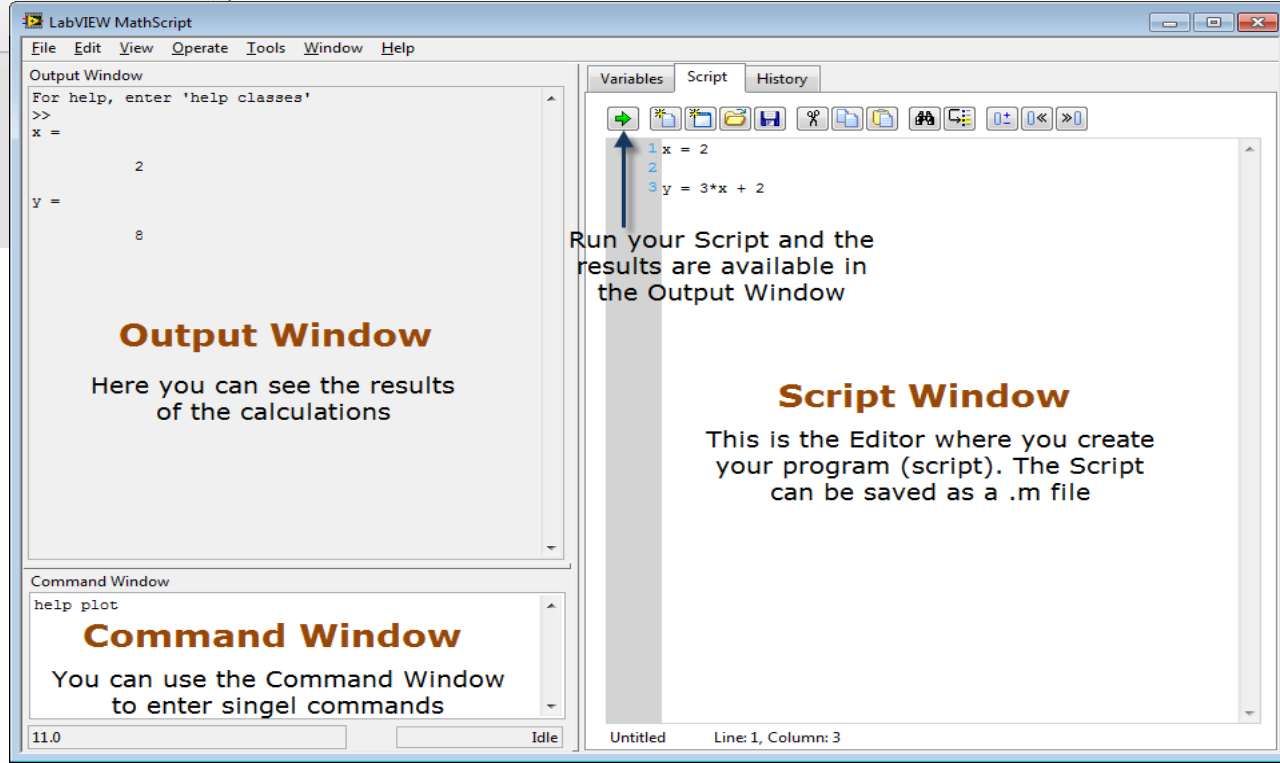
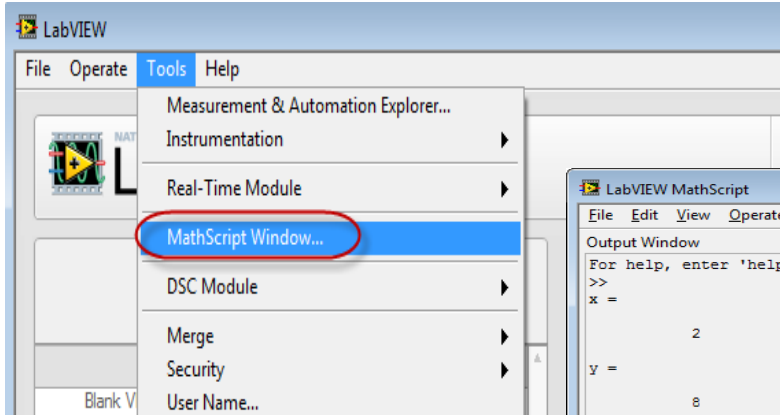
MathScript Module

LabVIEW MathScript Window

Hans-Petter Halvorsen

MathScript Window

LabVIEW MathScript RT Module



The image shows the LabVIEW MathScript window with three main sections:

- Output Window:** Displays the results of calculations. The text shows:

```
For help, enter 'help classes'  
>>  
x =  
    2  
y =  
    8
```
- Script Window:** The editor for creating and saving scripts. It contains the following code:

```
1 x = 2  
2  
3 y = 3*x + 2
```
- Command Window:** Used for entering single commands. The text shows:

```
help plot
```

Annotations in the image include:

- An arrow pointing to the 'Run' button (a green play icon) in the Script Window toolbar, with the text: "Run your Script and the results are available in the Output Window".
- The text "Output Window" is centered in the Output Window section, with the subtext: "Here you can see the results of the calculations".
- The text "Script Window" is centered in the Script Window section, with the subtext: "This is the Editor where you create your program (script). The Script can be saved as a .m file".
- The text "Command Window" is centered in the Command Window section, with the subtext: "You can use the Command Window to enter single commands".

The status bar at the bottom shows "11.0", "Idle", "Untitled", and "Line: 1, Column: 3".

Add-on Module for LabVIEW
where we can do text-based
programming and simulations
– very powerful!

Discretization

- In order to simulate this system, we typically need to find the discrete differential equation (difference equation)
- We can use e.g., the **Euler** Approximation:

$$\dot{x} \approx \frac{x(k+1) - x(k)}{T_s}$$

Discrete Model

We have the continuous differential equation: $\dot{x} = -ax + bu$

We apply Euler: $\dot{x} \approx \frac{x(k+1) - x(k)}{T_s}$

Then we get:

$$\frac{x(k+1) - x(k)}{T_s} = -ax(k) + bu(k)$$

This gives the following discrete differential equation (difference equation):

$$x(k+1) = (1 - T_s a)x(k) + T_s bu(k)$$

This equation can easily be implemented in any text based programming language

MathScript Code

This is an example.

You can implement it in many different ways

```
% Simulation of discrete model
clear, clc

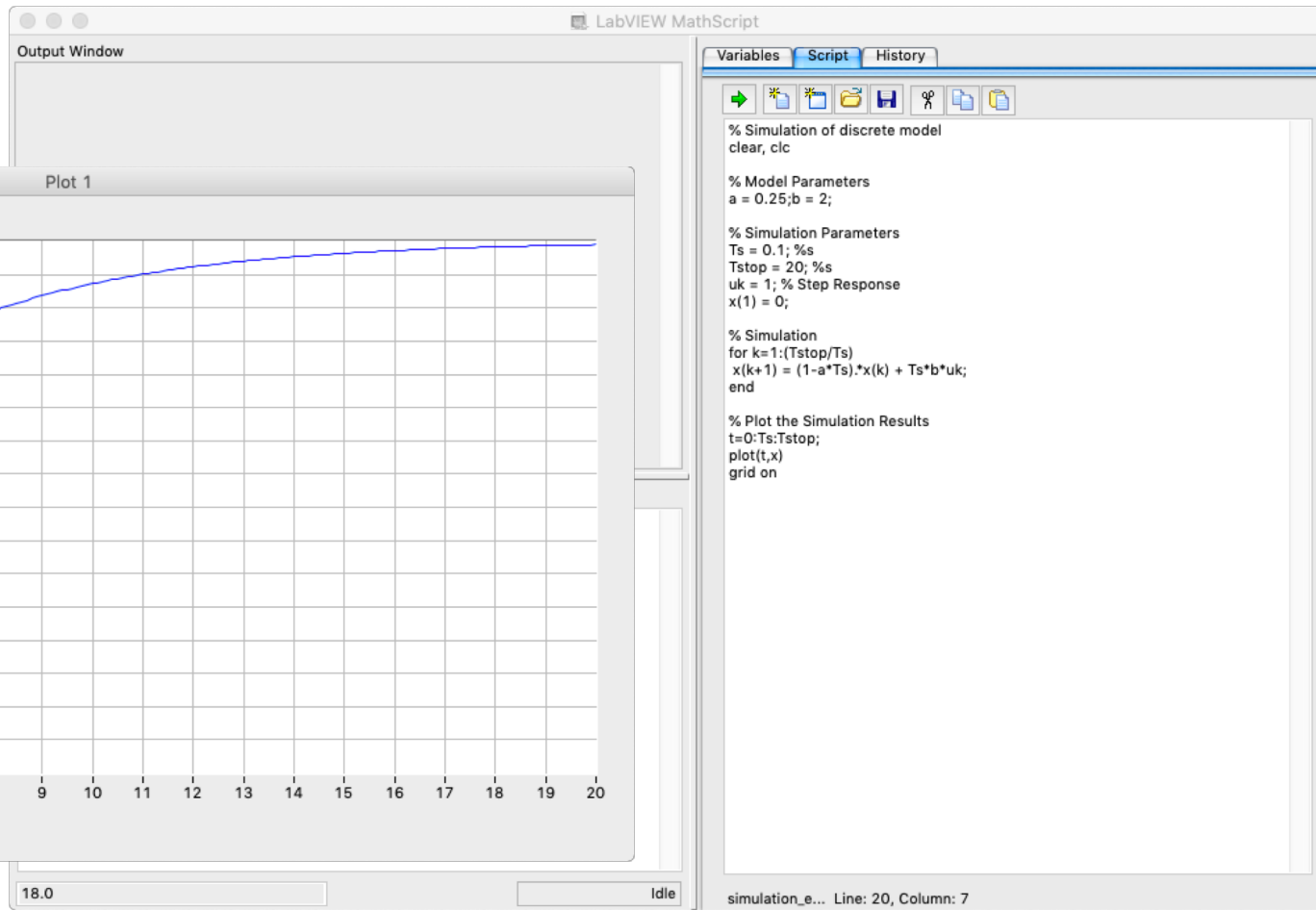
% Model Parameters
a = 0.25;b = 2;

% Simulation Parameters
Ts = 0.1; %s
Tstop = 20; %s
uk = 1; % Step Response
x(1) = 0;

% Simulation
for k=1:(Tstop/Ts)
    x(k+1) = (1-a*Ts).*x(k) + Ts*b*uk;
end

% Plot the Simulation Results
t=0:Ts:Tstop;
plot(t,x)
grid on
```

MathScript Code and Simulation Results



The image shows a screenshot of the LabVIEW MathScript environment. On the left, a window titled "Plot 1" displays a graph of a discrete-time signal. The x-axis is labeled from 0 to 20, and the y-axis is labeled from 0 to 8. The plot shows a blue curve that starts at (0,0) and rises asymptotically towards a value of approximately 8. On the right, the MathScript code editor is open, showing the following code:

```
LabVIEW MathScript
Output Window

Variables Script History
[Icons]
% Simulation of discrete model
clear, clc

% Model Parameters
a = 0.25; b = 2;

% Simulation Parameters
Ts = 0.1; %s
Tstop = 20; %s
uk = 1; % Step Response
x(1) = 0;

% Simulation
for k=1:(Tstop/Ts)
x(k+1) = (1-a*Ts).*x(k) + Ts*b*uk;
end

% Plot the Simulation Results
t=0:Ts:Tstop;
plot(t,x)
grid on

simulation_e... Line: 20, Column: 7
```

18.0

Idle

simulation_e... Line: 20, Column: 7

Alternative #2

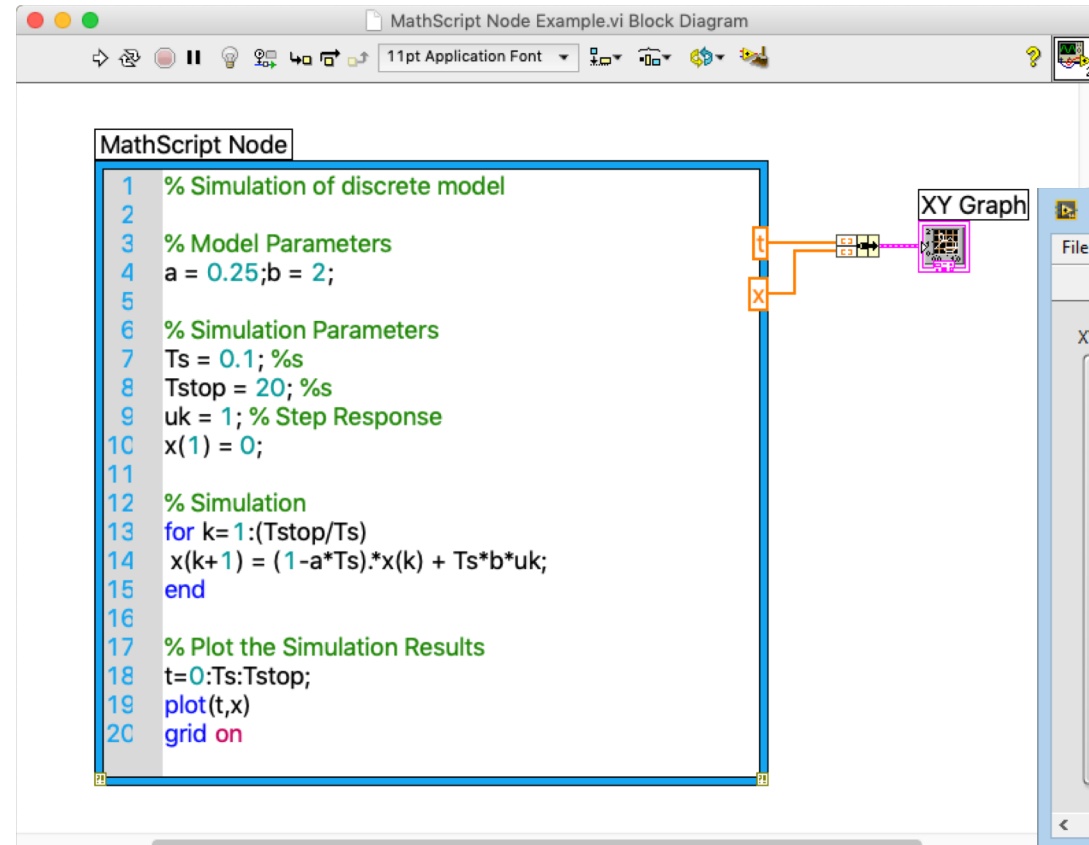


MathScript Node

Hans-Petter Halvorsen

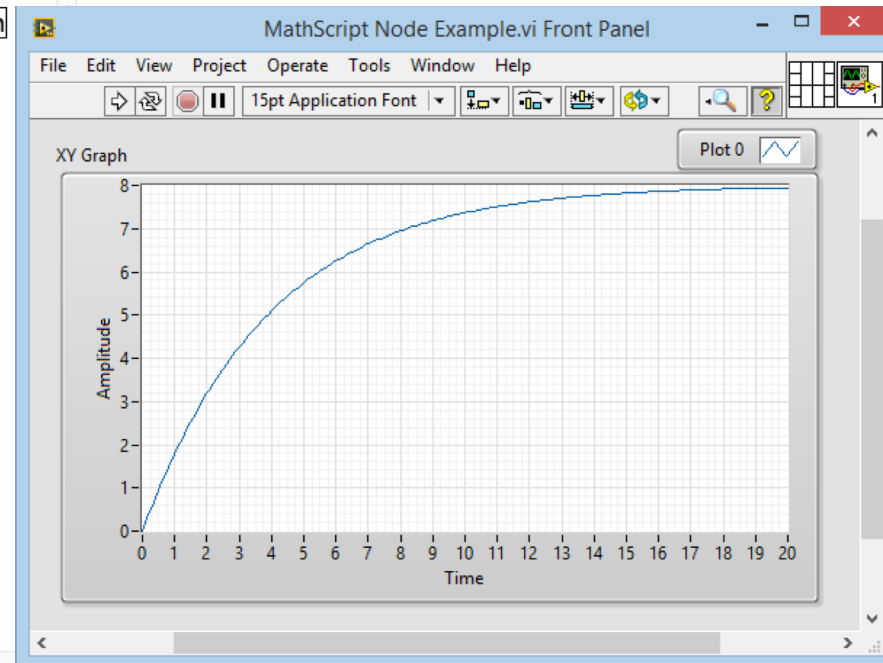
MathScript Node Code Example

We just copy the previous code into a MathScript Node inside LabVIEW:



The screenshot shows the LabVIEW Block Diagram for a file named "MathScript Node Example.vi". A "MathScript Node" is connected to an "XY Graph" block. The MathScript Node contains the following code:

```
1 % Simulation of discrete model
2
3 % Model Parameters
4 a = 0.25;b = 2;
5
6 % Simulation Parameters
7 Ts = 0.1; %s
8 Tstop = 20; %s
9 uk = 1; % Step Response
10 x(1) = 0;
11
12 % Simulation
13 for k=1:(Tstop/Ts)
14 x(k+1) = (1-a*Ts).*x(k) + Ts*b*uk;
15 end
16
17 % Plot the Simulation Results
18 t=0:Ts:Tstop;
19 plot(t,x)
20 grid on
```



Alternative #3



LabVIEW

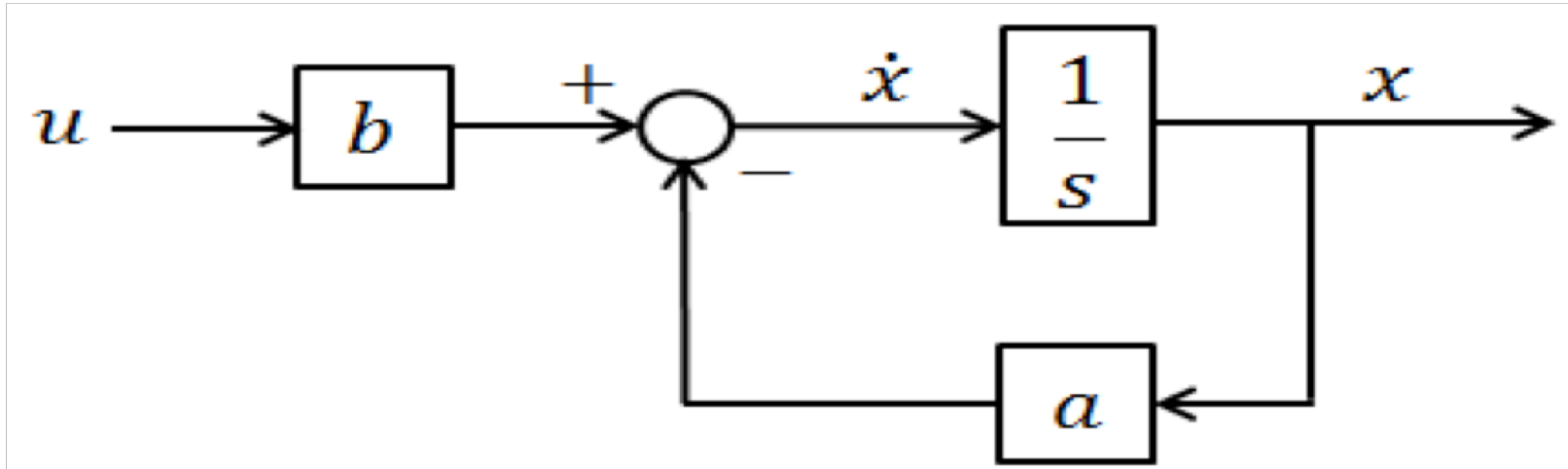
LabVIEW Control and Design and Simulation Module

Hans-Petter Halvorsen

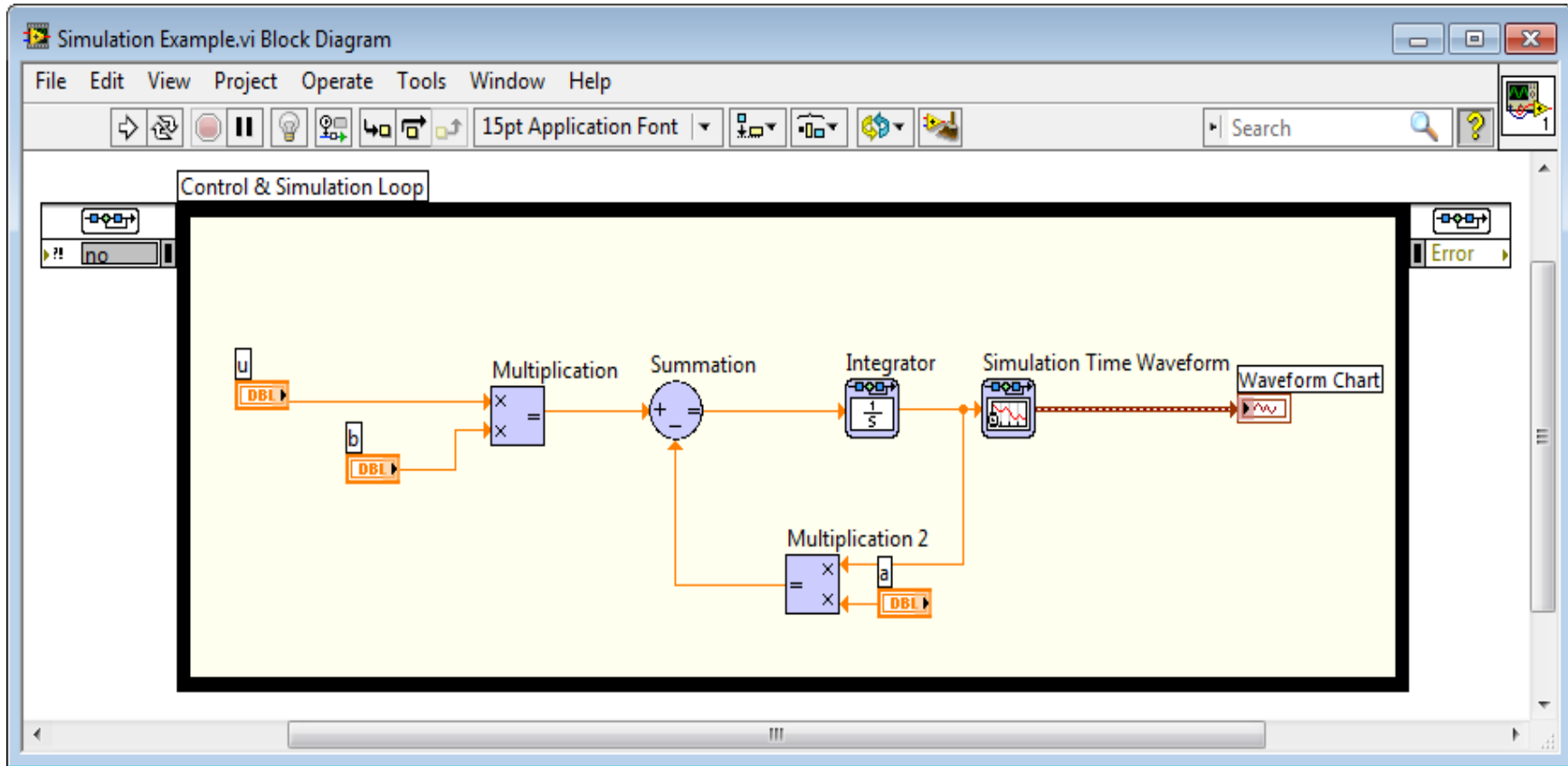
Block Diagram

$$\dot{x} = -ax + bu$$

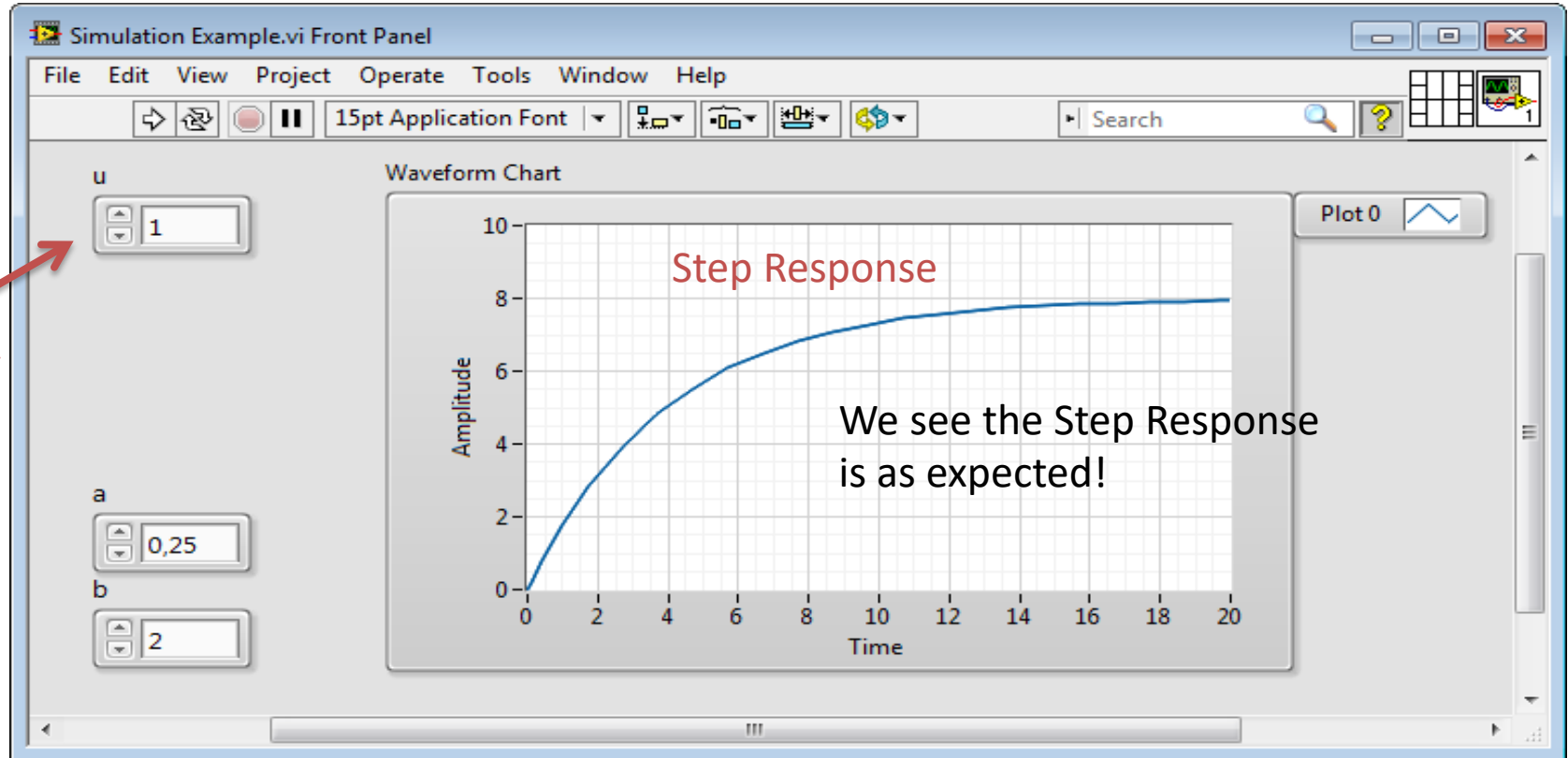
A Block Diagram for the model/differentia equation above becomes:



LabVIEW Code (Block Diagram)



LabVIEW GUI (Front Panel)





Summary

Hans-Petter Halvorsen

Discussion/Summary/Conclusion

- We have simulated a 1. order differential equation using LabVIEW
- We have used 3 different alternatives:
 - MathScript Window (similar to MATLAB)
 - MathScript Node inside LabVIEW
 - LabVIEW Control Design and Simulation Module
- We got (of course) the same simulation results using the 3 different alternatives
- What to do next: Do the same for your differential equation

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

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

