



الجامعة الهاشمية

Submitted to the  
Mechatronics Engineering Department  
The Hashemite University

The project of Artificial Intelligence  
(Calculator Find Summation & Average Of Two Number)

Supervised by:  
Dr. Mohammed Abu Mallouh

Student Name

ID Number

1. Mu'men Fayez Abu-Zeneh

1636716

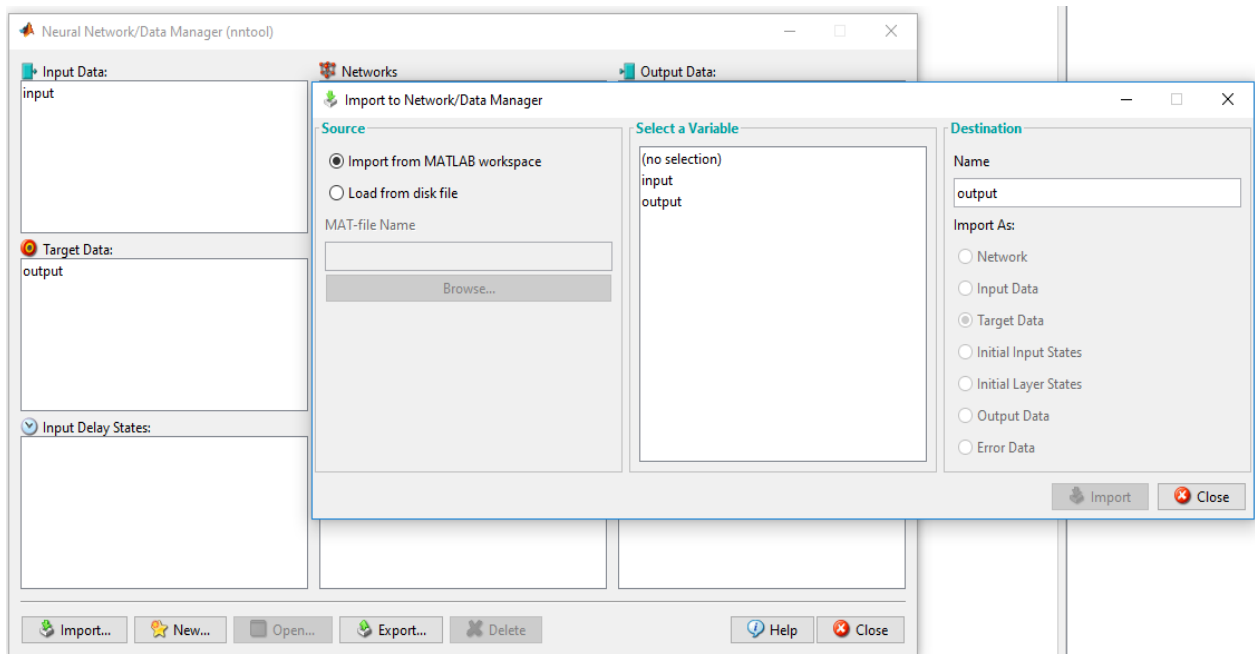
We get some correct data

H	G	F	E	D	C	B	A	
EX7	EX6	EX5	EX4	EX3	EX2	EX1		1
0	0	0	0	0	0	0	0 IN 1	2
6	5	4	3	2	1	0	0 IN 2	3
								4
6	5	4	3	2	1	0	0 SUM	5
3	2.5	2	1.5	1	0.5	0	0 Avg	6

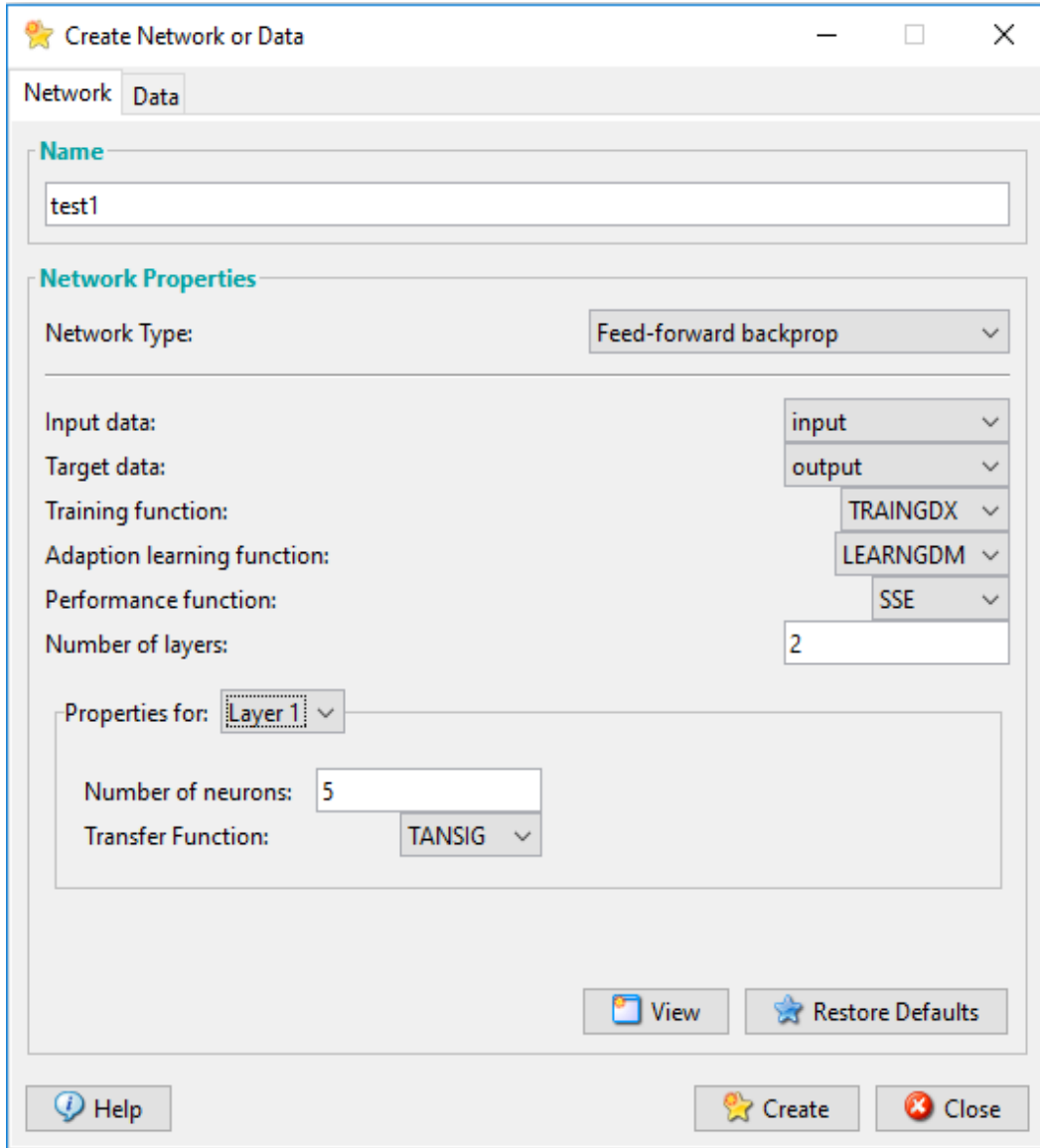
Include it to workspace in matlab as to group input & output

Workspace	
Name	Value
input	2x275 double
output	2x275 double

Run nntool & import input & target data from workspace



Create new neural network & name it test1 type of it feed-forward backprop with training function variable learning rate gradient decent and performance function is square of sum and two layer with 5 neurons has transfer function TANSIG



The image shows the 'Create Network or Data' dialog box in MATLAB. It has two tabs: 'Network' and 'Data'. The 'Network' tab is active. The 'Name' field contains 'test1'. Under 'Network Properties', 'Network Type' is 'Feed-forward backprop'. 'Input data' is 'input', 'Target data' is 'output', 'Training function' is 'TRAINGDx', 'Adaption learning function' is 'LEARNGDM', and 'Performance function' is 'SSE'. 'Number of layers' is set to 2. Under 'Properties for: Layer 1', 'Number of neurons' is 5 and 'Transfer Function' is 'TANSIG'. At the bottom are buttons for 'View', 'Restore Defaults', 'Help', 'Create', and 'Close'.

Create Network or Data

Network Data

Name

test1

Network Properties

Network Type: Feed-forward backprop

Input data: input

Target data: output

Training function: TRAINGDx

Adaption learning function: LEARNGDM

Performance function: SSE

Number of layers: 2

Properties for: Layer 1

Number of neurons: 5

Transfer Function: TANSIG

View Restore Defaults

Help Create Close

## Select input & target

Network: test1

View Train Simulate Adapt Reinitialize Weights View/Edit Weights

Training Info Training Parameters

**Training Data**

Inputs	input
Targets	output
Init Input Delay States	(zeros)
Init Layer Delay States	(zeros)

**Training Results**

Outputs	test1_outputs
Errors	test1_errors
Final Input Delay States	test1_inputStates
Final Layer Delay States	test1_layerStates

Train Network

## Initialize weight

Network: test1

View Train Simulate Adapt Reinitialize Weights View/Edit Weights

**Directions**

Click [REVERT WEIGHTS] to set weights and biases to their last initial values.  
Click [INITIALIZE WEIGHTS] to set weights and biases to new initial values.  
Use the "Input Ranges" area below to view and edit input ranges.

**Input Ranges**

[0 700;  
0 10]

Get from input: ▾

Revert Input Ranges Set Input Ranges

Revert Weights Initialize Weights

## Chose epochs max fail min grad & learning rate

Network: test1

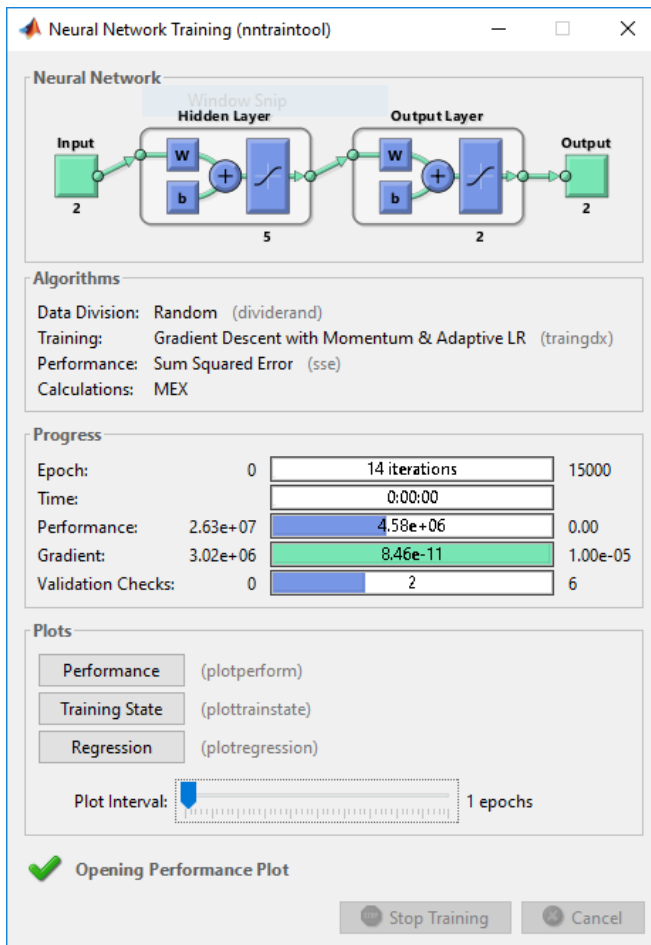
View Train Simulate Adapt Reinitialize Weights View/Edit Weights

Training Info Training Parameters

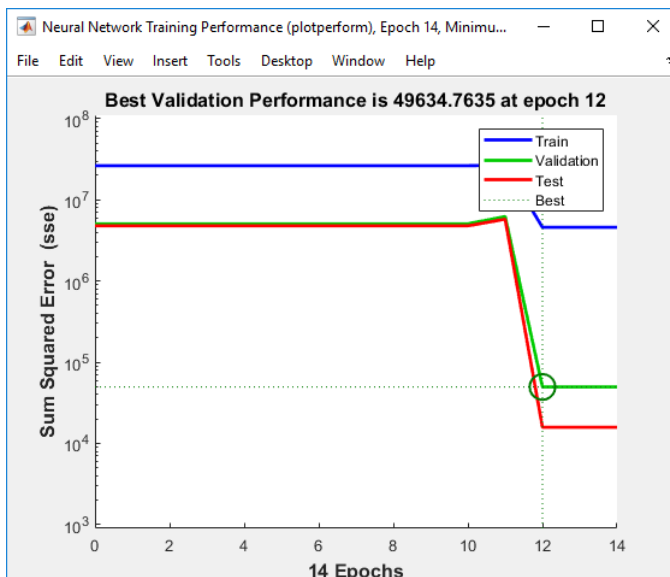
showWindow	true	lr	0.005
showCommandLine	false	lr_inc	1.05
show	25	lr_dec	0.7
epochs	15000	max_perf_inc	1.04
time	Inf	mc	0.9
goal	0		
min_grad	1e-12		
max_fail	12		

Train Network

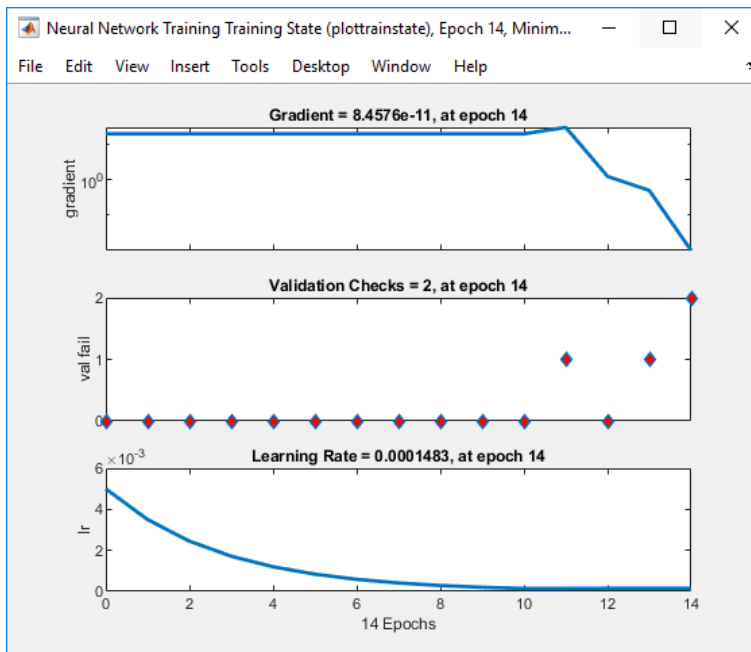
After train 14 epochs gradient stop learning



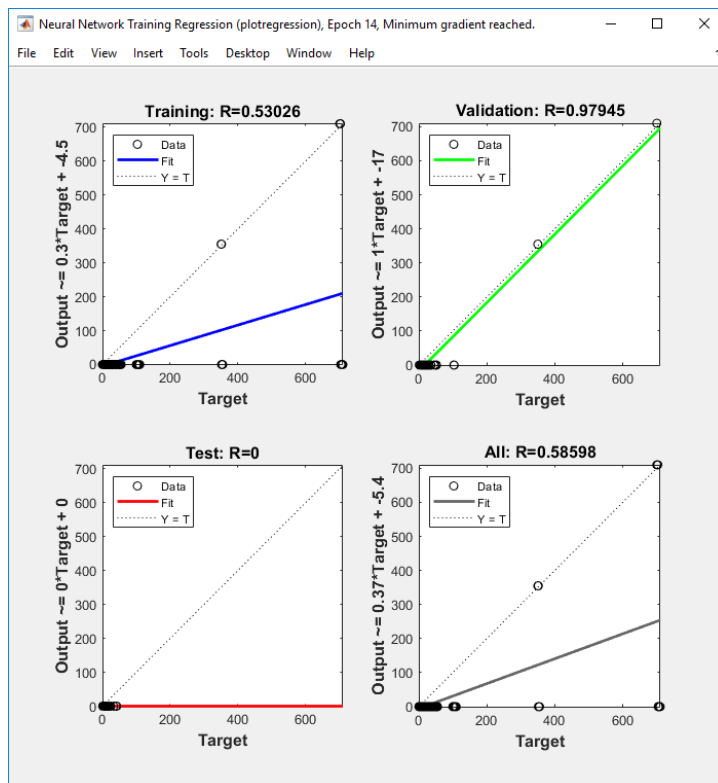
Train performance



## Train state

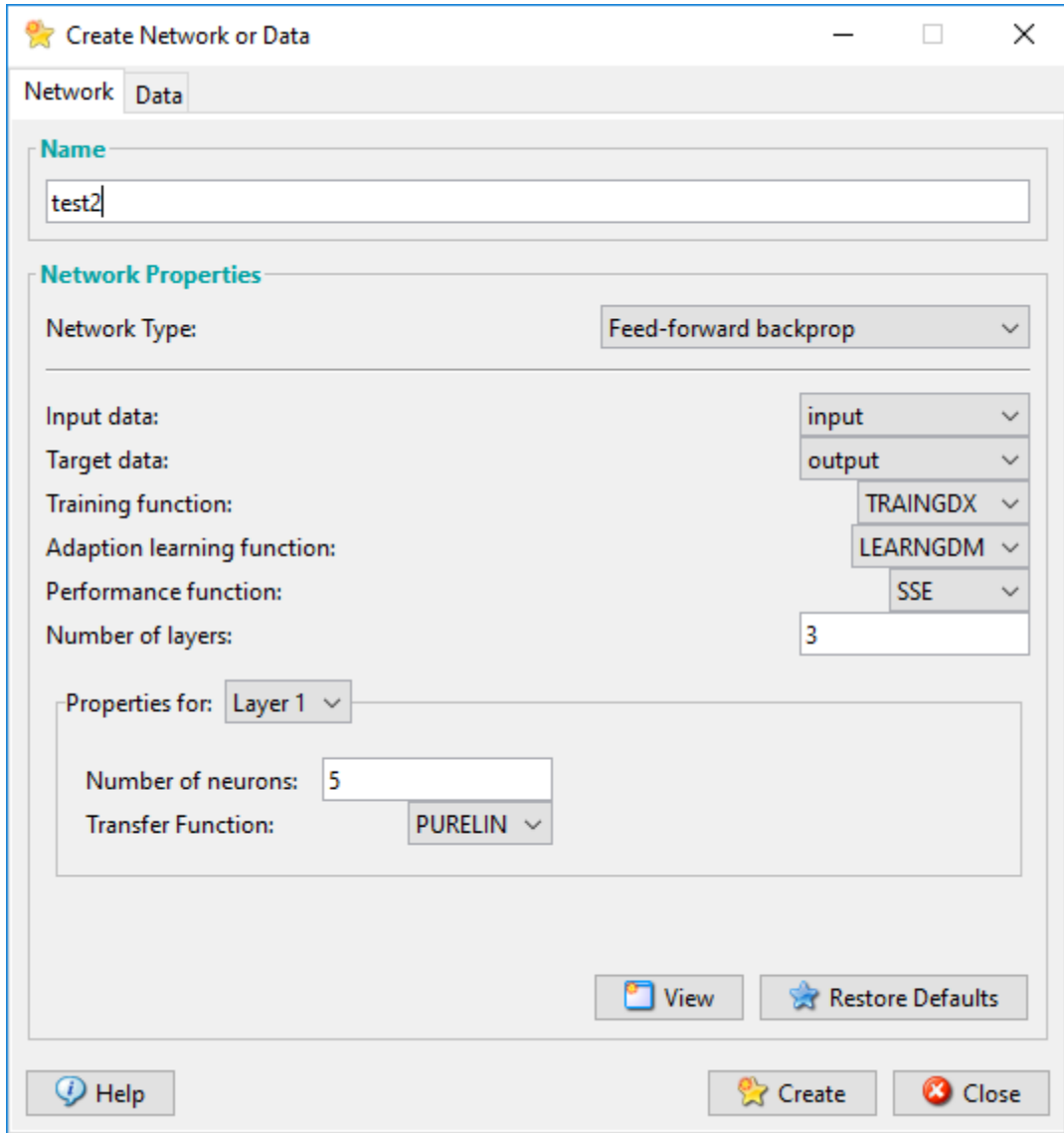


## Train regression



That's bad performance

We create new neural network named test2 type of it feed-forward backprop with training function variable learning rate gradient decent and performance function is square of sum and 3 layer with 5 neurons in layer 1 & 3 neurons in layer 2 has transfer function PURELIN



The image shows the 'Create Network or Data' dialog box in MATLAB. The 'Network' tab is selected. The 'Name' field contains 'test2'. Under 'Network Properties', 'Network Type' is 'Feed-forward backprop'. 'Input data' is 'input', 'Target data' is 'output', 'Training function' is 'TRAINGDX', 'Adaption learning function' is 'LEARNGDM', and 'Performance function' is 'SSE'. 'Number of layers' is set to 3. The 'Properties for: Layer 1' section shows 'Number of neurons' as 5 and 'Transfer Function' as 'PURELIN'. At the bottom, there are buttons for 'View', 'Restore Defaults', 'Help', 'Create', and 'Close'.

Property	Value
Name	test2
Network Type	Feed-forward backprop
Input data	input
Target data	output
Training function	TRAINGDX
Adaption learning function	LEARNGDM
Performance function	SSE
Number of layers	3
Properties for: Layer 1	
Number of neurons	5
Transfer Function	PURELIN

## Select input & target

Network: test2

View Train Simulate Adapt Reinitialize Weights View/Edit Weights

Training Info Training Parameters

**Training Data**

Inputs: input

Targets: output

Init Input Delay States: (zeros)

Init Layer Delay States: (zeros)

**Training Results**

Outputs: test2\_outputs

Errors: test2\_errors

Final Input Delay States: test2\_inputStates

Final Layer Delay States: test2\_layerStates

Train Network

## Initialize weight

Network: test2

View Train Simulate Adapt Reinitialize Weights View/Edit Weights

**Directions**

Click [REVERT WEIGHTS] to set weights and biases to their last initial values.  
Click [INITIALIZE WEIGHTS] to set weights and biases to new initial values.  
Use the "Input Ranges" area below to view and edit input ranges.

**Input Ranges**

[0 700;  
0 10]

Get from input: ▾

Revert Input Ranges Set Input Ranges

Revert Weights Initialize Weights

## Chose epochs max fail min grad & learning rate

Network: test2

View Train Simulate Adapt Reinitialize Weights View/Edit Weights

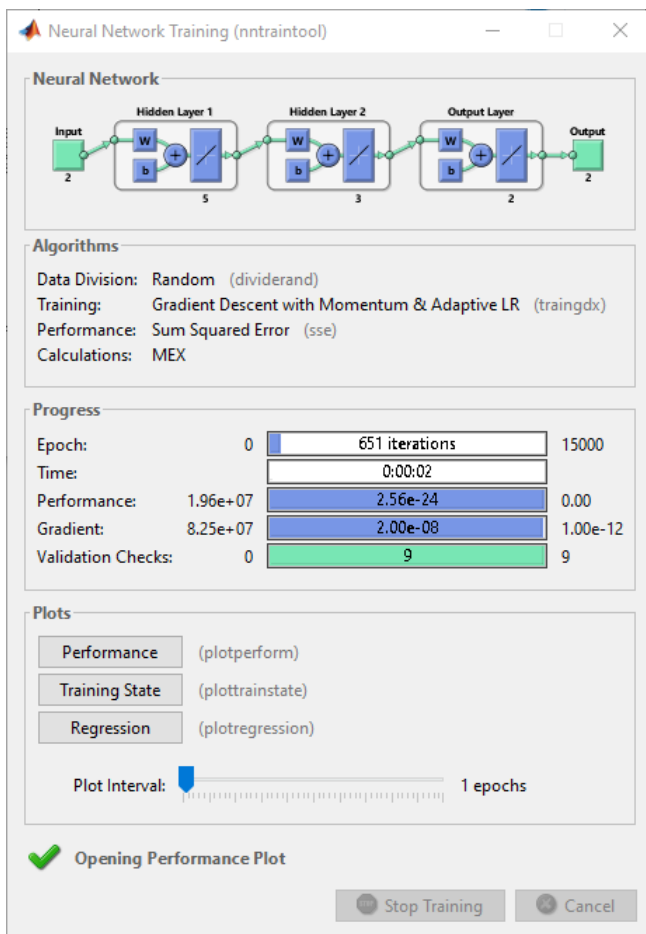
Training Info Training Parameters

showWindow	true	lr	0.001
showCommandLine	false	lr_inc	1.05
show	25	lr_dec	0.7
epochs	15000	max_perf_inc	1.04
time	Inf	mc	0.9
goal	0		
min_grad	1e-012		
max_fail	9		

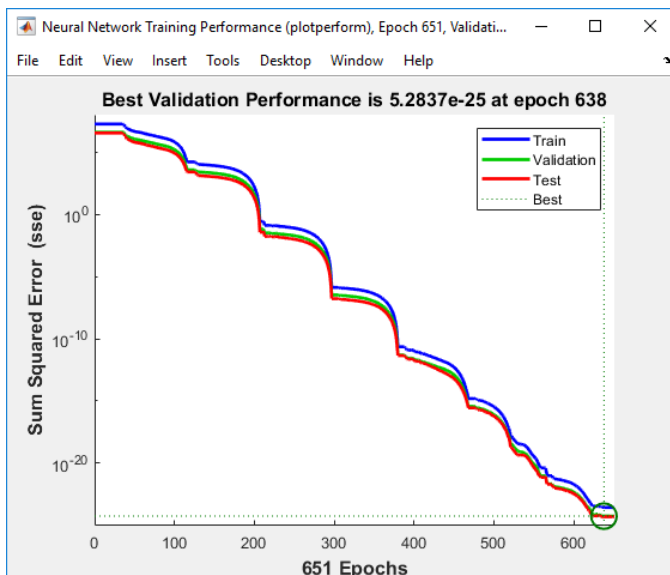
Train Network



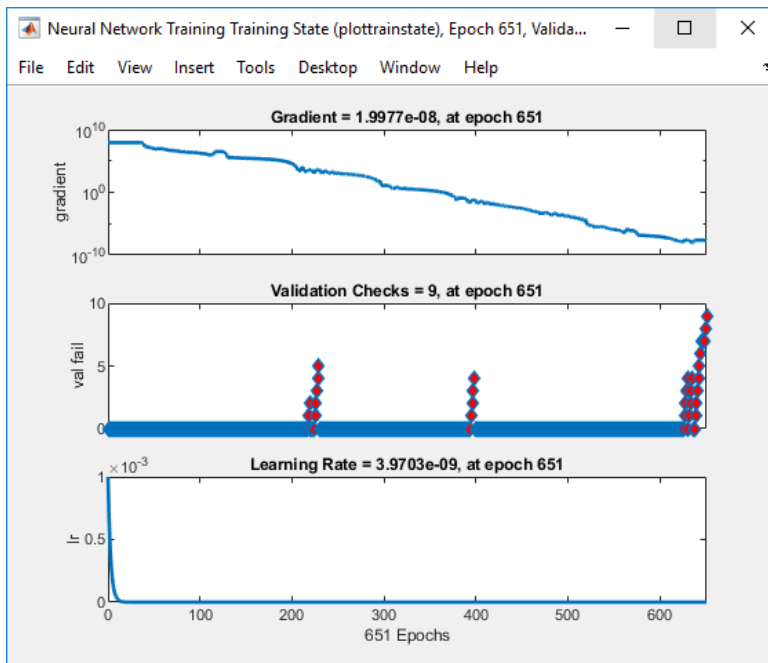
After train 651 epochs validation check stop learning



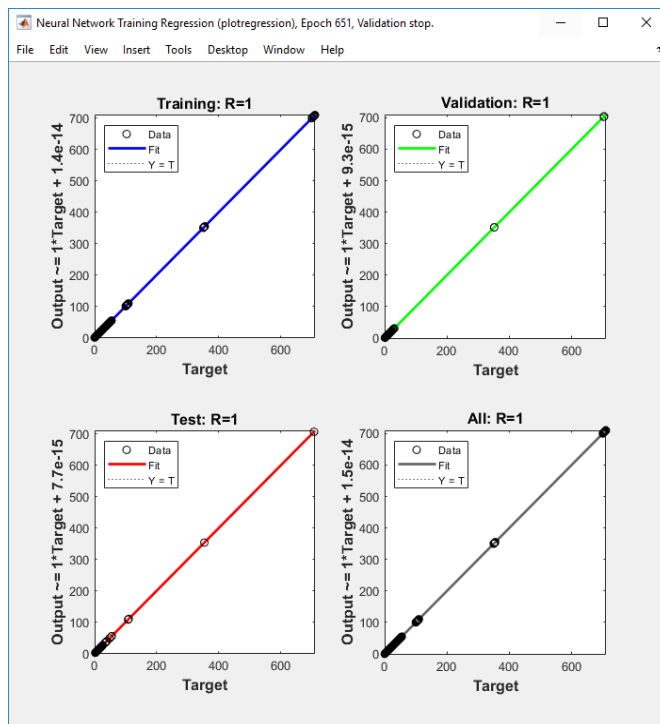
Train performance



## Train state

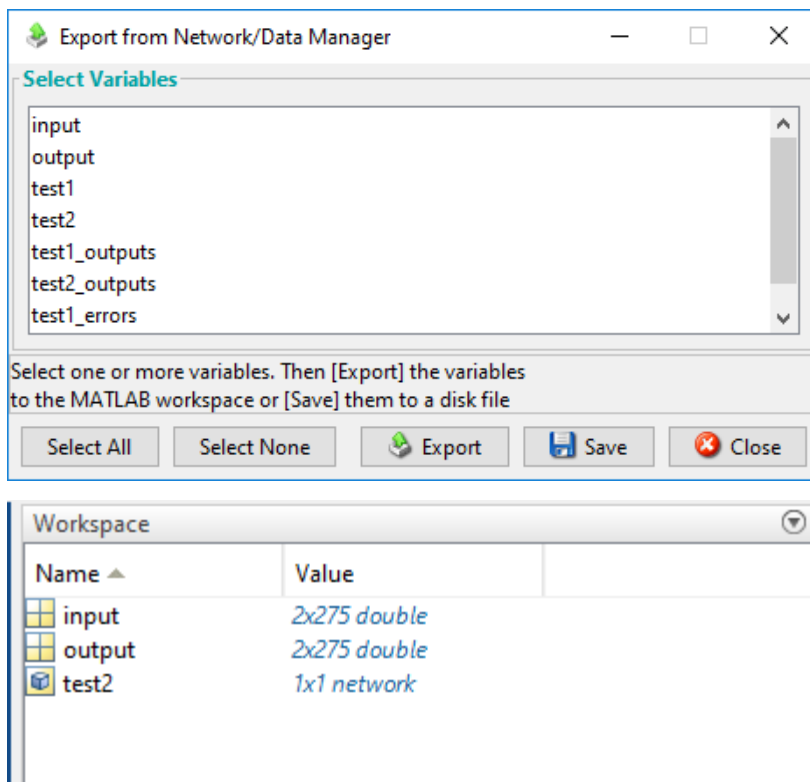


## Train regression

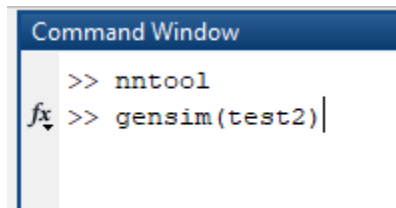


That's good performance let's test it

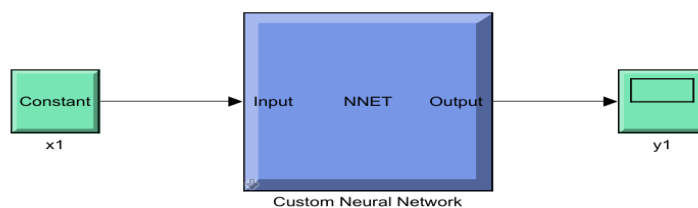
We export test2 to workspace



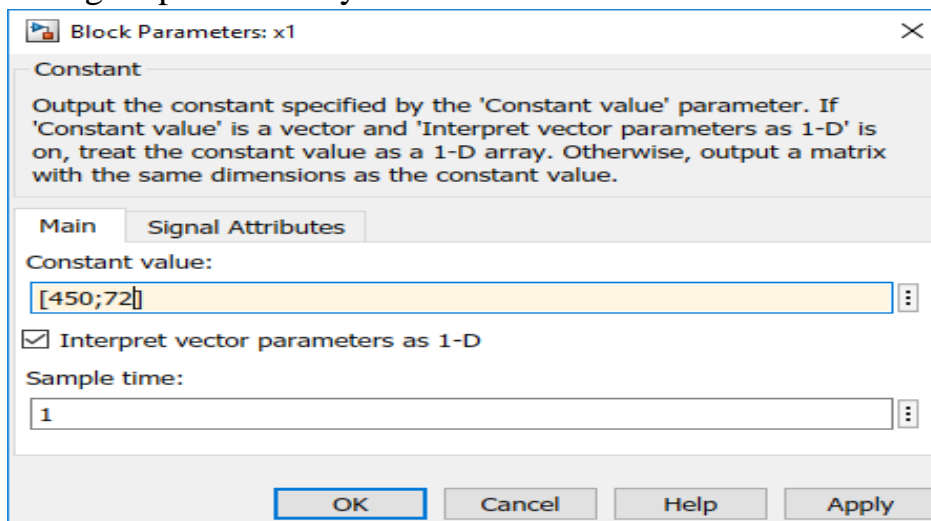
Write { gensim(test2) } in command window



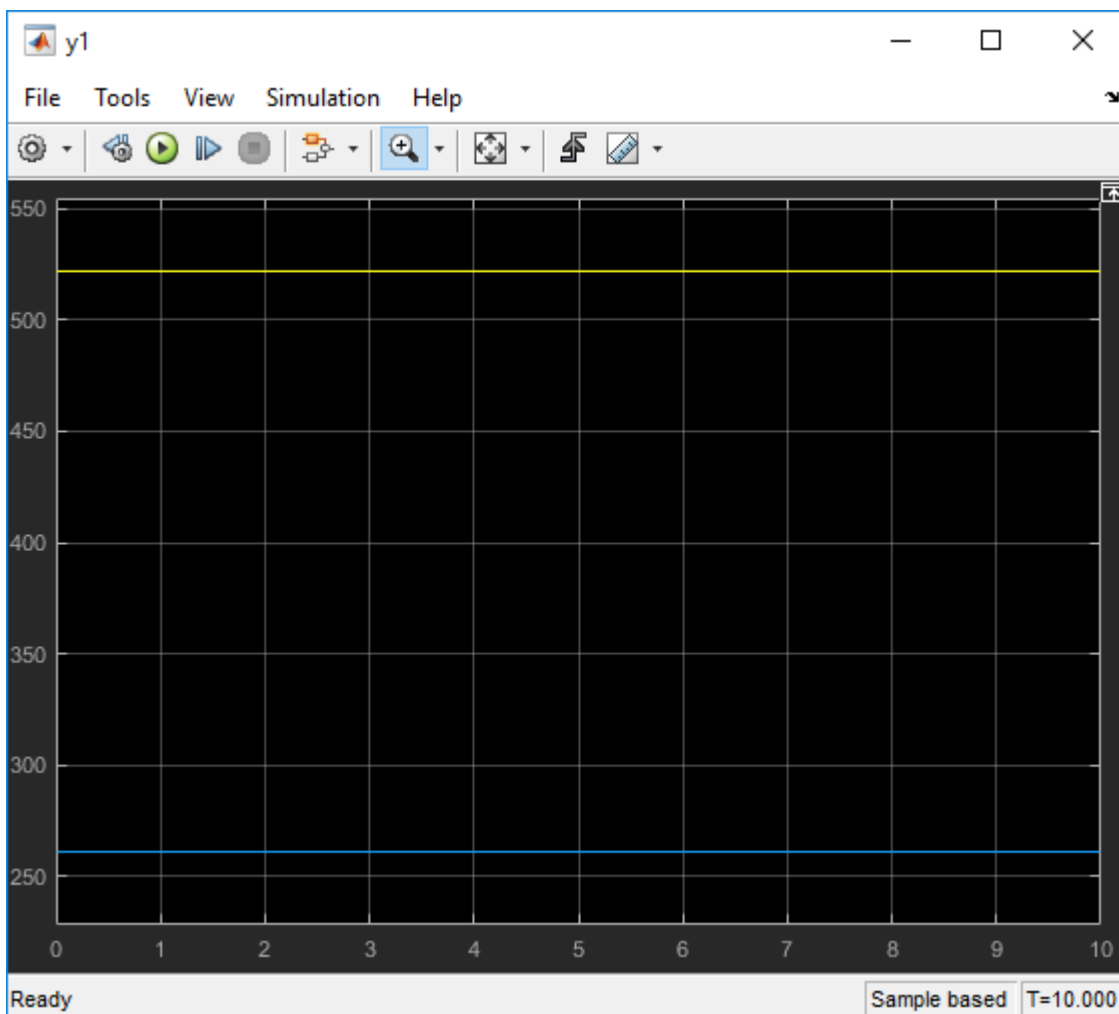
Run Simulink as black box has input & output



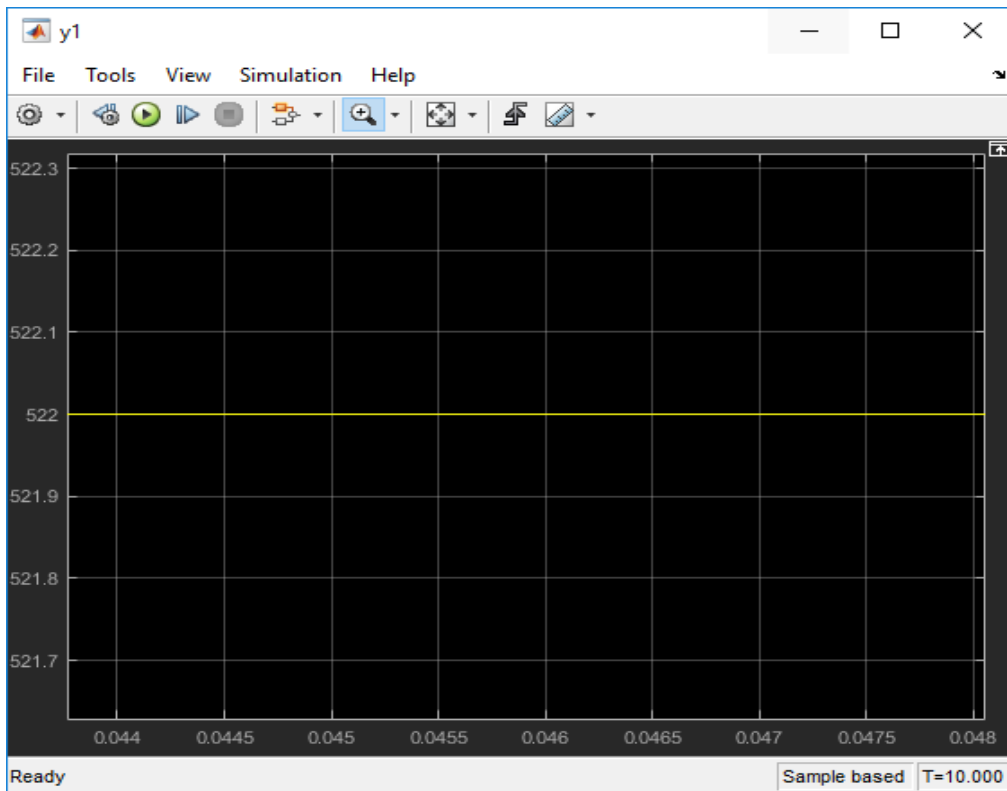
Change input randomly number



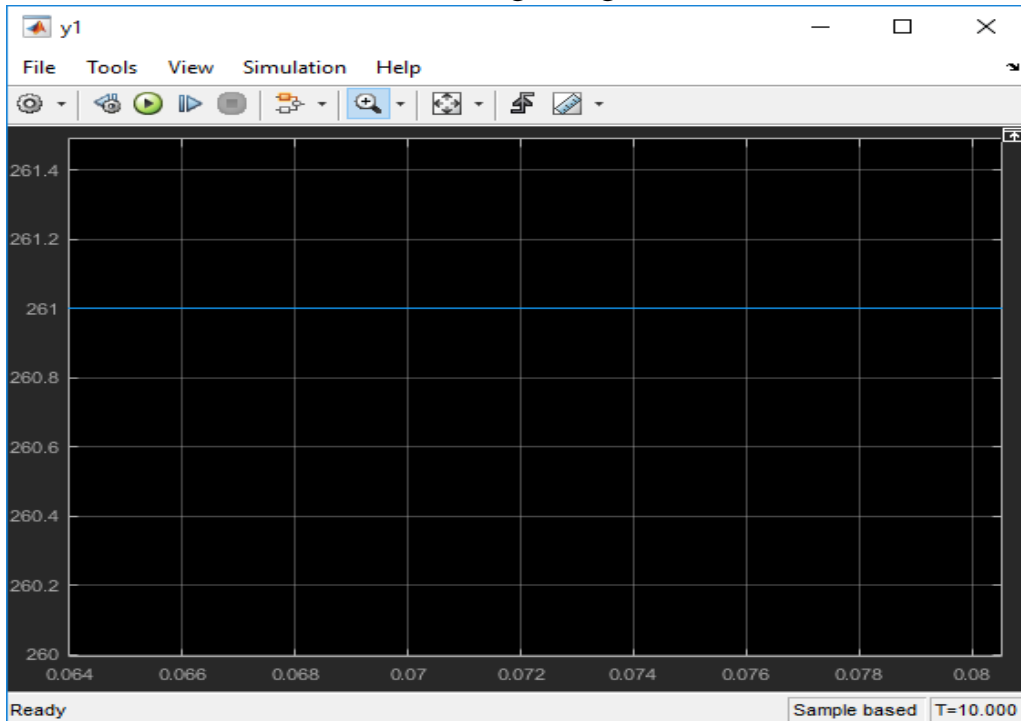
Then output get



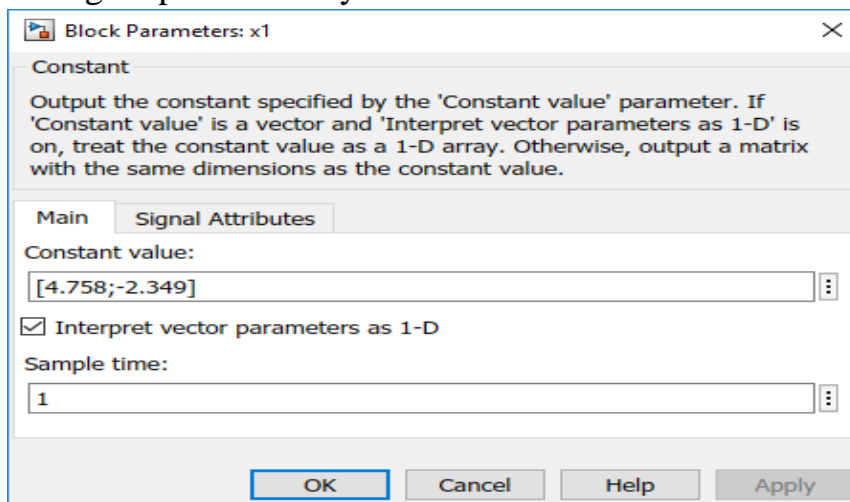
After zoom in to the yellow line we get sum



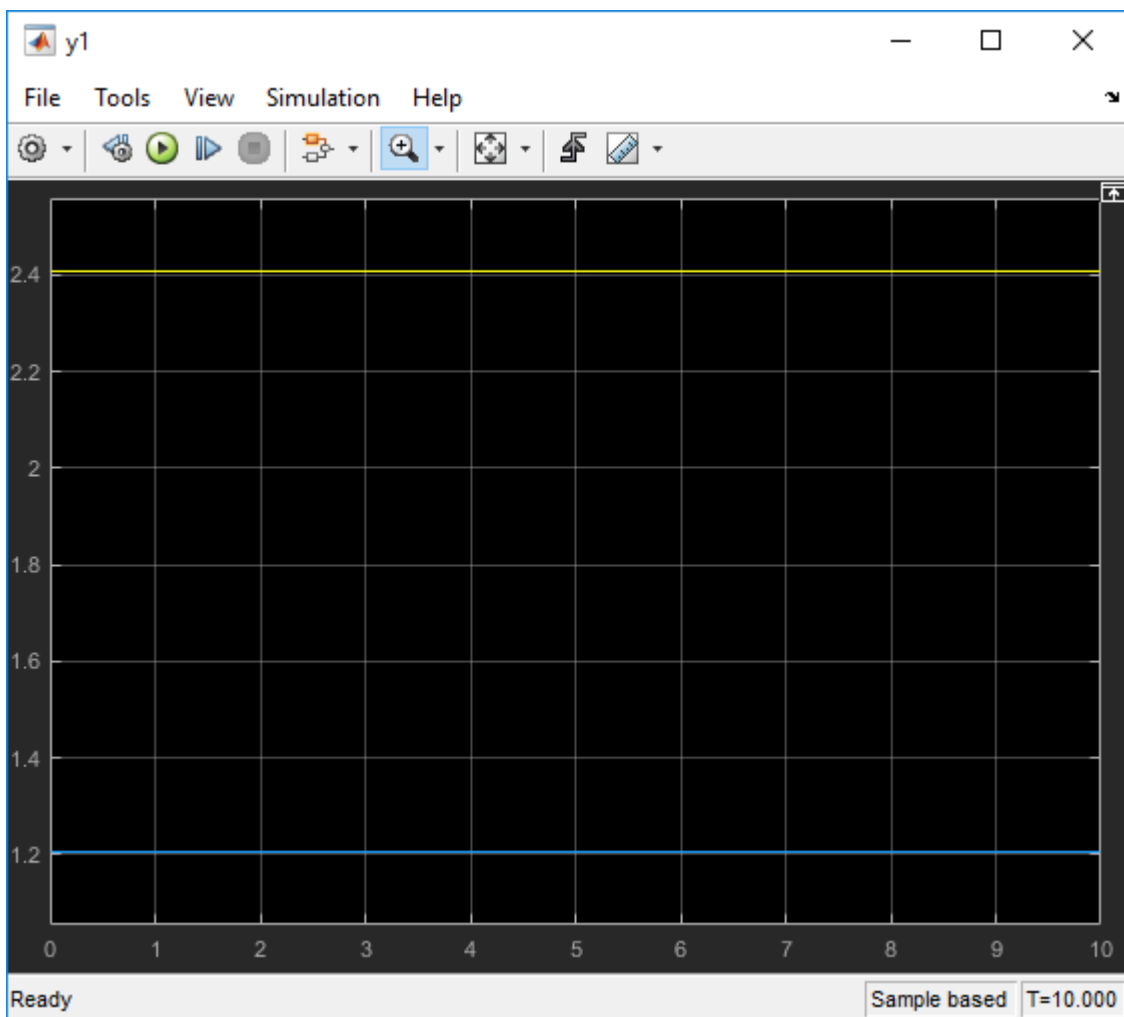
After zoom in to the blue line we get avg



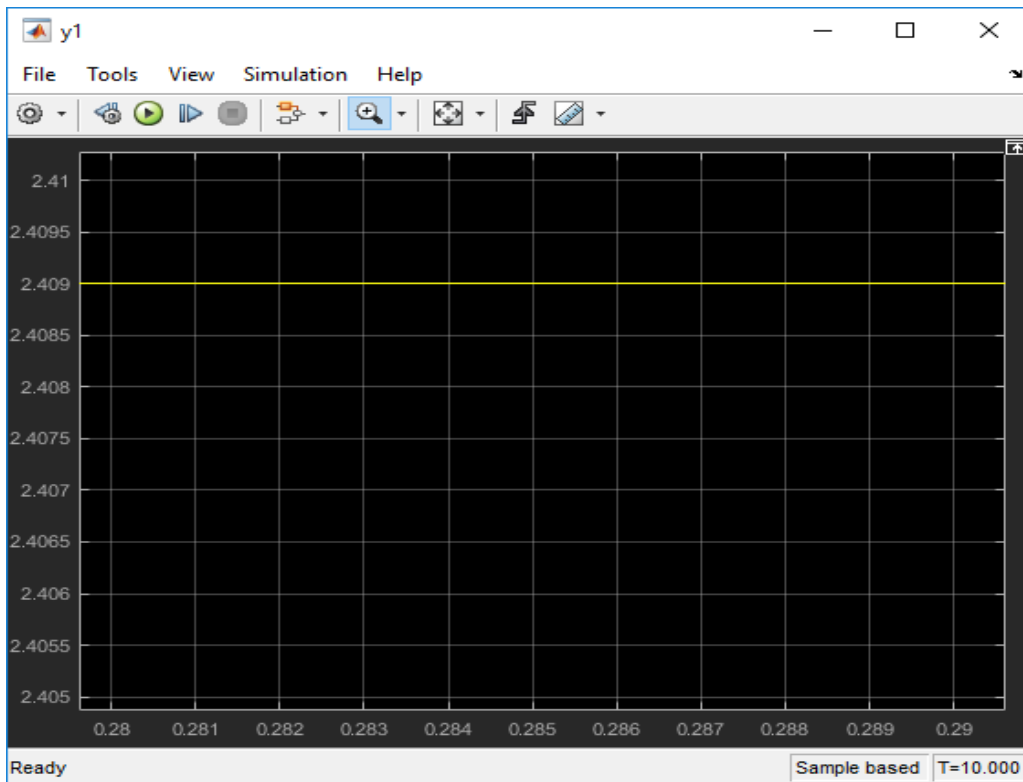
Change input randomly number



Then output get



After zoom in to the yellow line we get sum



After zoom in to the blue line we get avg

