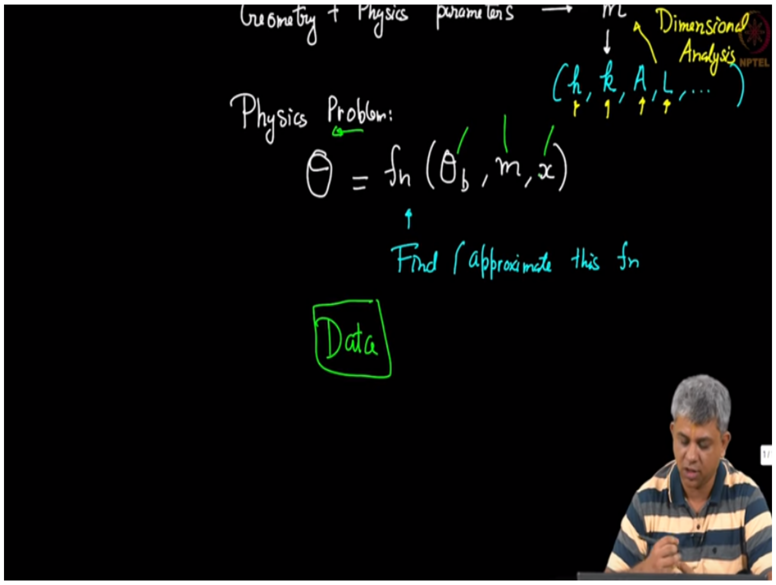
Machine Learning For Engineering and Science Application

Professor Dr. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology, Madras

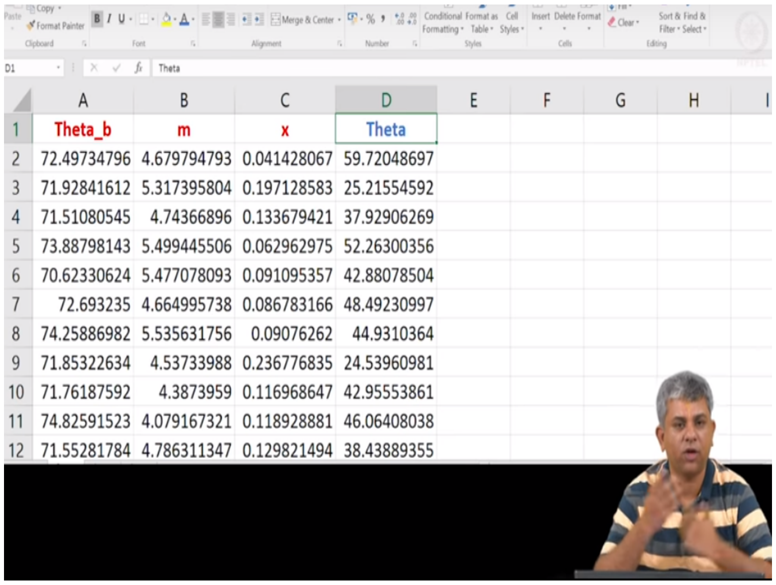
Application 1 Solution

(Refer Slide Time: 00:13)


Geometry + Physics parameters → m
(h, k, A, L, ...)
Dimensional Analysis
Physics Problem:
$\theta = fn(\theta_b, m, x)$
Find / approximate this fn
Data

Welcome back when I set up a problem I describe the simple problem of fin heat transfer this is typically a simple extension coming from a base and we wanted to find out given some data given some historical data of the dependence of theta on theta b, m and x how we generally set a function this is sort of a trivial task it is close to the task that we took up in the initial weeks of this course.

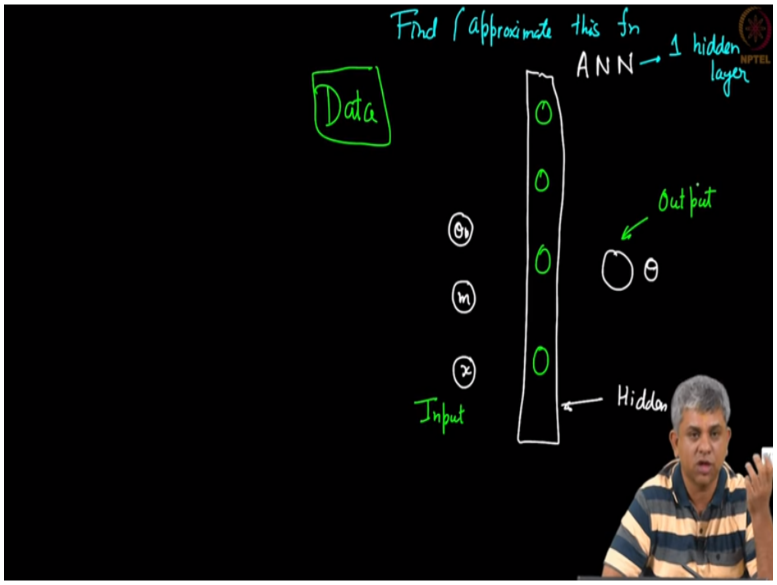
(Refer Slide Time: 00:47)



Theta_b	m	x	Theta
72.49734796	4.679794793	0.041428067	59.72048697
71.92841612	5.317395804	0.197128583	25.21554592
71.51080545	4.74366896	0.133679421	37.92906269
73.88798143	5.499445506	0.062962975	52.26300356
70.62330624	5.477078093	0.091095357	42.88078504
72.693235	4.664995738	0.086783166	48.49230997
74.25886982	5.535631756	0.09076262	44.9310364
71.85322634	4.53733988	0.236776835	24.53960981
71.76187592	4.3873959	0.116968647	42.95553861
74.82591523	4.079167321	0.118928881	46.06408038
71.55281784	4.786311347	0.129821494	38.43889355

So I showed you the actual data also, so this had a thousand data points and you had a variation of theta be M and X and the output here even was theta now of course the simplest structure when you have such heterogeneous data by heterogeneous data I mean that it is not all theta B it is not all M and it is not all X you actually have three different physical quantities one is one physical quantity is temperature another physical parameters and the third is location.

(Refer Slide Time: 01:24)



And you want to combine this three and you want to get theta as out the obvious structure to use here is the very simple artificial neural network now I am going to do something given the simplicity of the problem actually this is probably the best thing to try when you have some such simple problem engineering practice over the last several year show that simply try one hidden layer we have not any end and example.

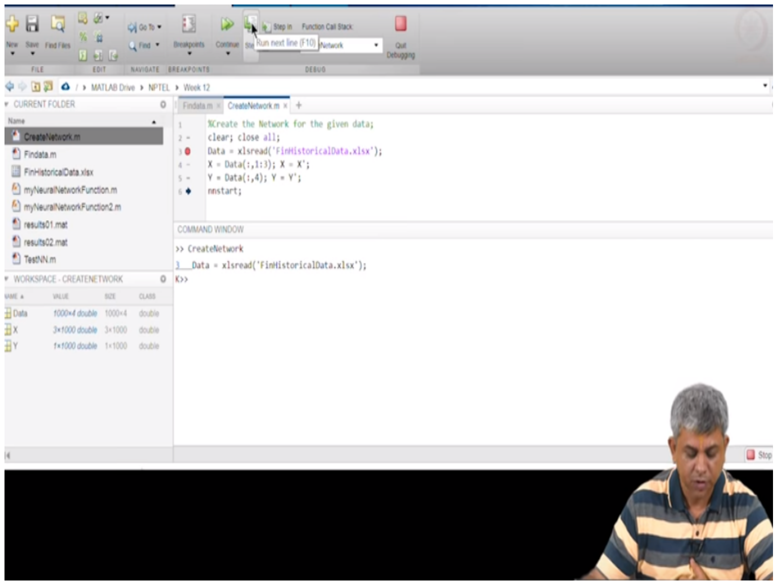
So far I am going to show you this as a very simple example just use one hidden layer and see how well it does only if it does not do well should you asked add more complex structure especially when you are dealing with ANN's, so if I have this as a hidden layer we will also do something like change the number of neurons middle, so let us say a few neuron here this is the input layer the hidden layer and this of course the output so we will keep some such structure and try to see how well it fits this thousand data points.

So we are going to try that now I am going to use MATLAB for this case I know that throughout the course people who have been asking for python it is actually fairly straight forward to do this python but the MATLAB implementation especially simple artificial neural network is very-very powerful I would very strongly recommended especially if you are going engineering practice this works extremely well of course it is not the hard to program this within python also within either tensor flow or crus are whatever it is that you are using currently there is another reason for me to use MATLAB.

It has a very nice graphical input to it has very nice GUI the interface is very nice also I have found surprising during the running of the score itself that it can do several thing that are actually quite hard to with a tenser flow we are not going to see examples of that today but it can do several things it also have a few functionality that tensor flow does not have because tensor flow is not bother about a single heat and layer any way they have built at specifically deep matter not for CHALU networks of this for that.

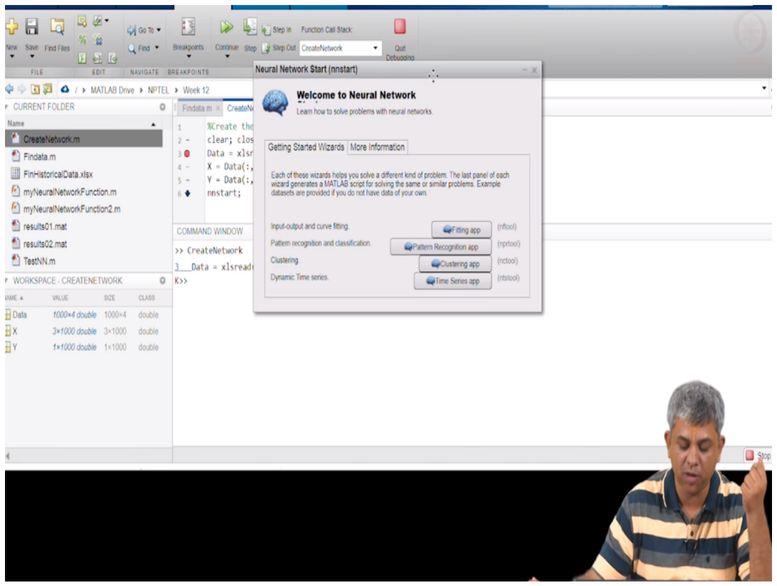
I have shown so that my lesson that I would like to point about here is in case you have access to MATLAB the biggest problem with MATLAB of course is that is not open source and it is not free but to the extent that we have access to it during the duration of this course please you try it you will find it surprisingly good at many task even for deep learning task I have found surprisingly good performance with MALTAB it is actually quite impressive.

(Refer Slide Time: 04:34)



```
%Create the Network for the given data;
clear; close all;
Data = xlsread('FinHistoricalData.xlsx');
X = Data(:,1:3); X = X';
Y = Data(:,4); Y = Y';
nnstart;
```

```
>> CreateNetwork
3   Data = xlsread('FinHistoricalData.xlsx');
K>>
```



Learn how to solve problems with neural networks.

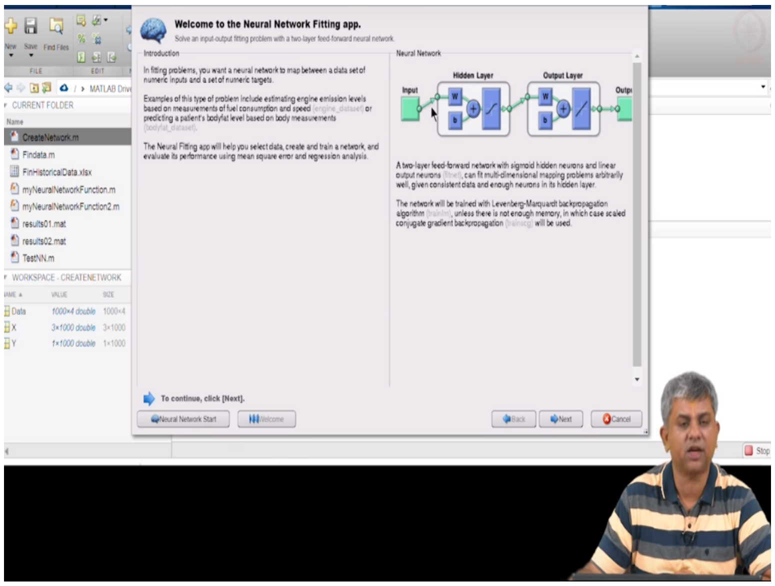
Each of these wizards helps you solve a different kind of problem. The last panel of each wizard generates a MATLAB script for solving the same or similar problems. Example datasets are provided if you do not have data of your own.

Input-output and curve fitting.	Fitting app	(nftool)
Pattern recognition and classification.	Pattern Recognition app	(nprtool)
Clustering.	Clustering app	(nctool)
Dynamic Time series.	Time Series app	(ntstool)

So here is what we are going to do I have called my file fin historical data and let X, so let us run our course very simply I have called my code create network we are here and I have written out the data in as well as we format basically this rates our input data the as well as so the excel file that I showed you now the data is right we would like to label the inputs and outputs separately first three label the first three columns remember were the inputs and then the next column was the output.

So this is not a very complex code I need not have actually shown this at all now this thing here NN start actually starts the neural networks GUI you can do it without the GUI but for demonstration specially in a video I find it particularly convenient to do this this is one other reason with that we are trying to show this within MATLB rather than python, so suppose we start so you will see this, this is actually an old tool box MATLAB now has a deep learning tool box that us far many ore thing much like tensor flow or anything else, so now we are going to use the fitting app within this so please see this.

(Refer Slide Time: 06:06)



Welcome to the Neural Network Fitting app.

Solve an input-output fitting problem with a two-layer feed-forward neural network.

Introduction

In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating engine emission levels based on measurements of fuel consumption and speed (engine_dataset) or predicting a patient's bodyfat level based on body measurements (bodyfat_dataset).

The Neural Fitting app will help you select data, create and train a network, and evaluate its performance using mean square error and regression analysis.

Neural Network

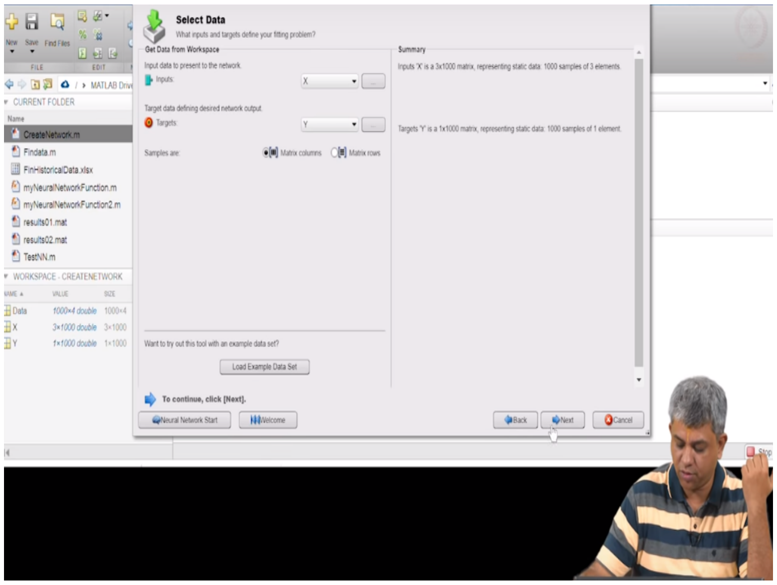
A two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (fitnet), can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer.

The network will be trained with Levenberg-Marquardt backpropagation algorithm (trainlm), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (trainscg) will be used.

To continue, click [Next].

So the general structure that is being used with the fitting app is of course you have one input hidden layer and directly the output layer it is a sitting layer.

(Refer Slide Time: 06:19)


Select Data

What inputs and targets define your fitting problem?

Get Data from Workspace

Input data to present to the network.

Inputs: X

Target data defining desired network output.

Targets: Y

Samples are: [✓ Matrix columns] Matrix rows

Summary

Inputs 'X' is a 3x1000 matrix, representing static data: 1000 samples of 3 elements.

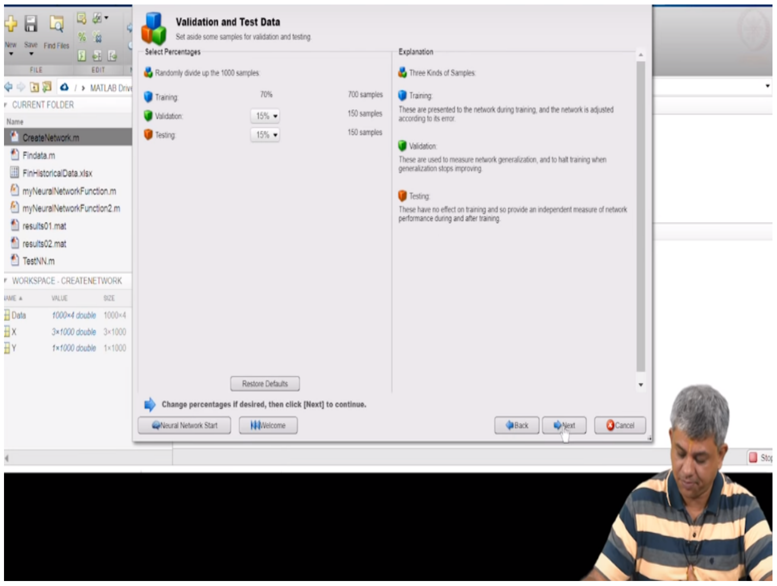
Targets 'Y' is a 1x1000 matrix, representing static data: 1000 samples of 1 element.

Want to try out this tool with an example data set?

To continue, click [Next].

Now it ask for inputs in fact if somebody is interested you should probably make an interface of this sort for tensor flow or something that all just so that beginners can actually easily get into a problem, so I am going to give X as input Y is output.

(Refer Slide Time: 06:41)


Validation and Test Data

Set aside some samples for validation and testing.

Select Percentages

Randomly divide up the 1000 samples:

Training:	70%	700 samples
Validation:	15%	150 samples
Testing:	15%	150 samples

Explanation

Three Kinds of Samples:

Training:
These are presented to the network during training, and the network is adjusted according to its error.

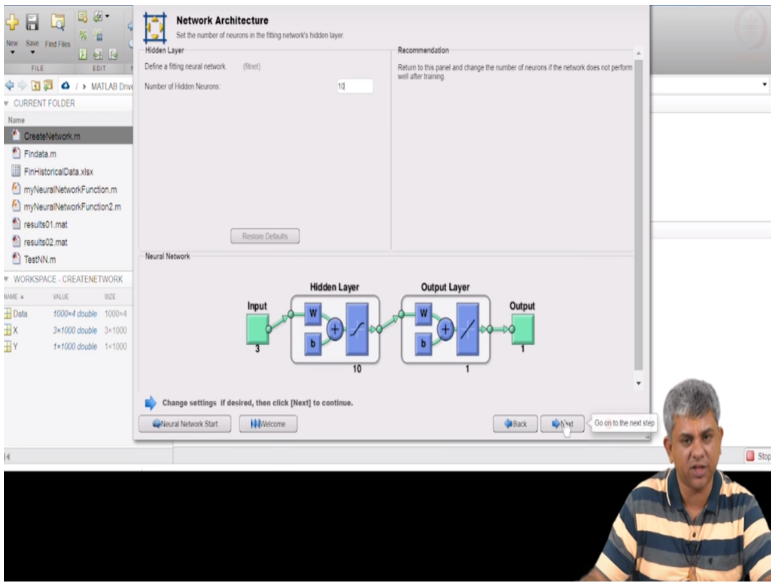
Validation:
These are used to measure network generalization, and to halt training when generalization stops improving.

Testing:
These have no effect on training and so provide an independent measure of network performance during and after training.

Change percentages if desired, then click [Next] to continue.

Now please see here within MATLAB they have a default set of values for splitting the data as training, testing and validation all of you would have seen this during the earlier weeks whenever you have data you use training data in order to get your parameters validation data in order to get your some of your hyper parameter and the testing data in order to finally report how good or bad were results were.

(Refer Slide Time: 07:11)



Network Architecture

Set the number of neurons in the fitting network's hidden layer.

Hidden Layer

Define a fitting neural network. (fitnet)

Number of Hidden Neurons: 10

Recommendation

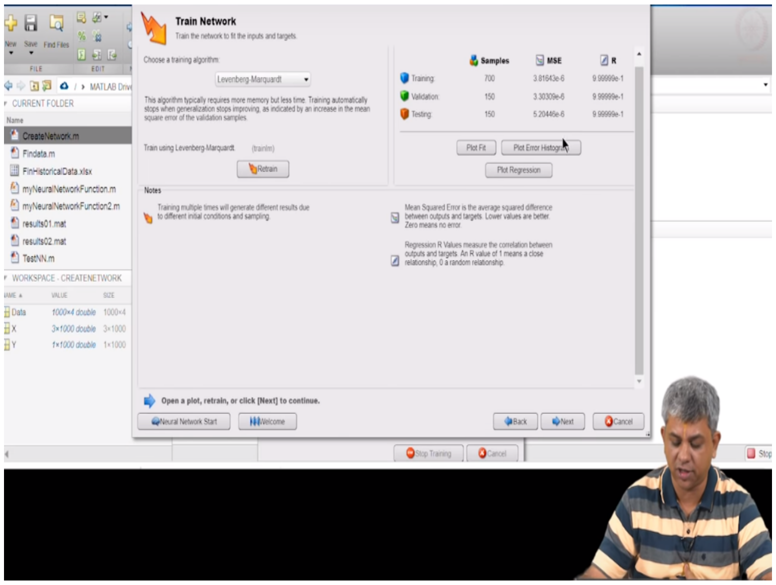
Return to this panel and change the number of neurons if the network does not perform well after training.

Neural Network

Change settings if desired, then click [Next] to continue.

So now it is asking for the number of hidden neurons now what I am going to do this you might seen on the screen is that the current number of hidden neurons this is very first layer are set to be ten we will leave it as it is we will leave it to be ten hidden neurons so currently we have inputs three hidden ten and the output layer is just one output, so I will leave it we will change it a little bit later just to see what the effect is.

(Refer Slide Time: 07:41)



Train Network

Train the network to fit the inputs and targets.

Choose a training algorithm: Levenberg-Marquardt

This algorithm typically requires more memory but less time. Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Train using Levenberg-Marquardt. (trainlm)

	Samples	MSE	R
Training:	700	3.81643e-6	9.99999e-1
Validation:	150	3.30309e-6	9.99999e-1
Testing:	150	5.20446e-6	9.99999e-1

Notes

Training multiple times will generate different results due to different initial conditions and sampling.

Mean Squared Error is the average squared difference between outputs and targets. Lower values are better. Zero means no error.

Regression R Values measure the correlation between outputs and targets. An R value of 1 means a close relationship, 0 a random relationship.

Open a plot, retrain, or click [Next] to continue.

NAME	VALUE	SIZE
Data	1000×4 double	1000×4
X	3×1000 double	3×1000
Y	1×1000 double	1×1000

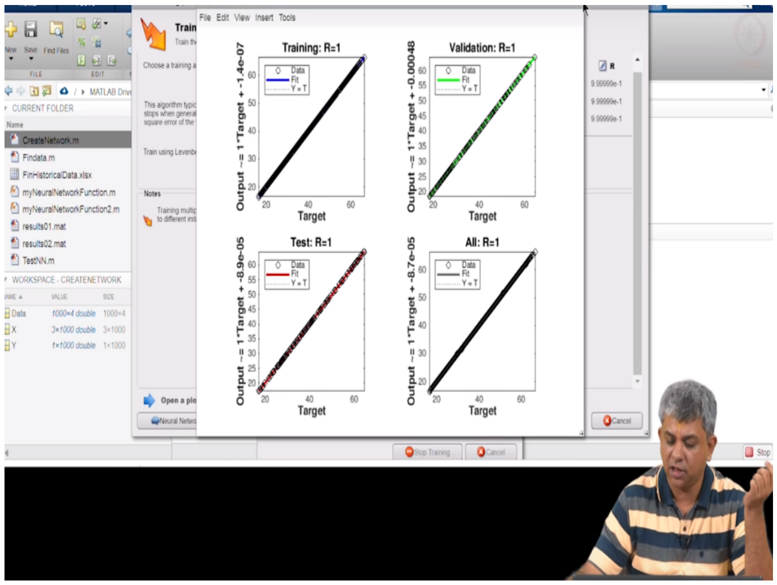
Now here is one other place for a single hidden layer that MATLAB been sower it ask for a training algorithm now remember all we have been using, so far is gradient descent one or the other version of gradient descent if you check the options here there is something called Bayesian regularization which we have not done nor have we done scaled conjugate gradient nor have we done Levenberg Marquardt, Levenberg Marquardt algorithm works very well.

If you have like single layer and your loss function is a least square lost function which is basically what we want to use for our problem I have all this predicted values what would be a lost function it would be a least square lost function it is a regression problem in not a classification problem if you try gradient descent on the current problem A you will find that you will have to of course normalize the input which we have not done you might not this that all the columns that we had as input and output were not normalize.

When we did the linear regression problems to might remember that when we did not normalize we had lots of trouble with gradient descent whether we use gradient descent stochastic gradient descent add an etcetera typically they will be much slower this problem then Levenberg Marquardt for this problem, so I will just move the training here and you will see that within the a few second it has actually fully trained this simple network.

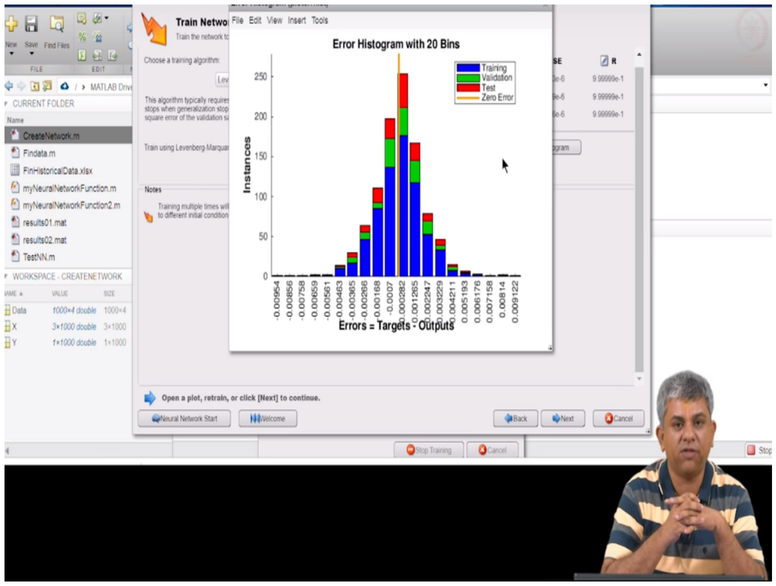
If you write a code within python my suspicion is that gradient descent will actually work a little bit slower than what MATLAB even for this, so you notice a few things that have been reported remember the statistical measure called R which shows you the amount of correlation.

(Refer Slide Time: 09:32)


Training: R=1
Validation: R=1
Test: R=1
All: R=1
Output ~= 1*Target + -1.4e-07
Output ~= 1*Target + -0.00048
Output ~= 1*Target + -8.9e-05
Output ~= 1*Target + -8.7e-05
Target
Data
Fit
Y = T

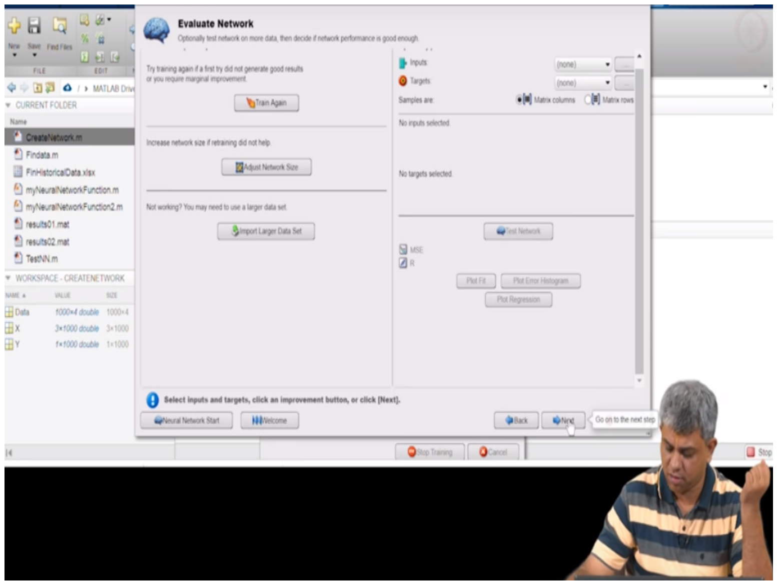
Let me just show the correlation here this is normalize correlation you will see that a the output is matching really really well if it is a straight line it basically means that your output are matching very well that is your predicted output versus the actual output is matching very very well.

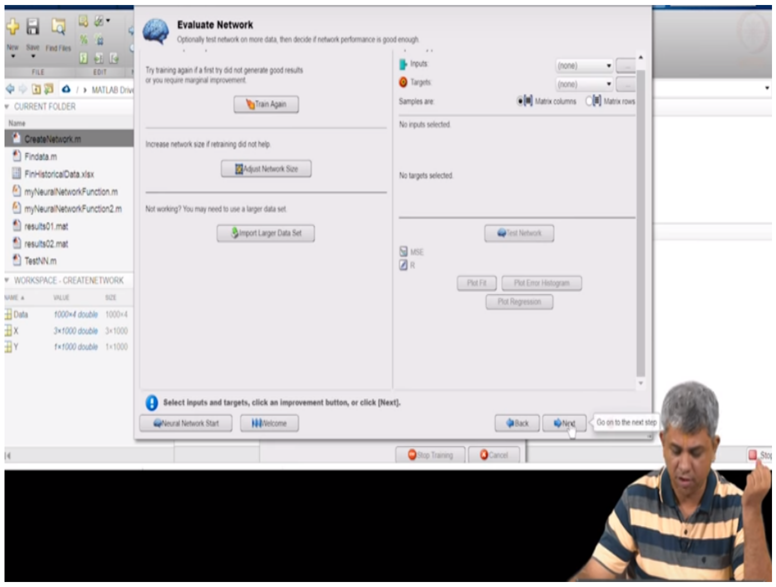
(Refer Slide Time: 09:52)


Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
Errors = Targets - Outputs
Train Network
Open a plot, retrain, or click [Next] to continue.
Neural Network Start
Welcome
Back
Next
Cancel

If you plot the error histogram you will see that there is the error here is of the scale of point zero zero nine etcetera etcetera you will see that a lot of view in fact find Gaussian this is a very nice example of central limit theorem and what we have been talking about even the error in the prediction of the neural networks are arranged very nicely as if they were from Gaussian if time permits I will show you how to use an inverse problem using idea the error would be distributed as a Gaussian probably later this week you can combine neural network with (())(10:30) Dr. Ganapati discuss the previous also very nicely if time permit I will do that.

(Refer Slide Time: 10:39)


Evaluate Network
Optionally test network on more data, then decide if network performance is good enough.
Train Again
Adjust Network Size
Import Larger Data Set
Test Network
Plot Fit
Plot Error Histogram
Plot Regression
Select inputs and targets, click an improvement button, or click [Next].
Neural Network Start
Welcome
Back
Next


Evaluate Network
Optionally test network on more data, then decide if network performance is good enough.
Try training again if a first try did not generate good results or you require marginal improvement.
Train Again
Increase network size if retraining did not help.
Adjust Network Size
Not working? You may need to use a larger data set.
Import Larger Data Set
Inputs:
Targets:
Samples are:
Matrix columns
Matrix rows
No inputs selected.
No targets selected.
Test Network
MSE
R
Plot Fit
Plot Error Histogram
Plot Regression
Select inputs and targets, click an improvement button, or click [Next].
Neural Network Start
Welcome
Back
Next
Go on to the next step.
Stop Training
Cancel

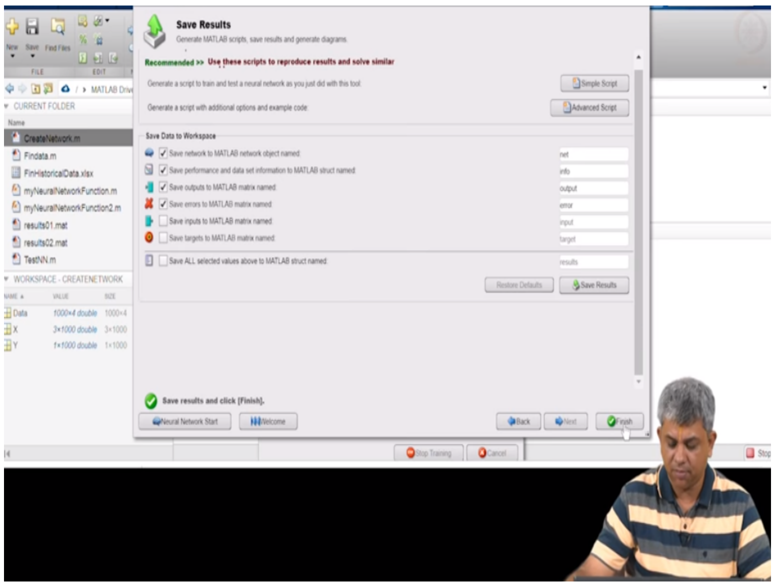
So this is just as an example that the error the answer was not exact but you do have a very nice error plot error is quite is low, so error you will see is around to three into to ten to power minus six now whenever you trained you look for a few things remember our over fitting and regularization and discussion you will see that training error is around three into ten the power minus six this is the main square error you will also see that the validation and the testing error are round about the same of the same order of magnitude.

Then that is the case we can kind of assume that we do not have too much variance problem and we do not have over fitting etcetera so we are reasonably ok we will also see that the correlation is extremely high which means ten neurons for this problem is actually pretty good and ten neurons is nothing as you know from CNN cases so with three inputs and one outputs you can actually fit very well just using ten neurons, now suppose we want to change our number of neurons and we bring it down to two remember.

That the main square error in the previous case was of the order ten power minus six and if I go here and I trained again you will see that the main square error actually gone up even though the correlation is still pretty good what you notice now is the errors have gone up in magnitude if we plot the error histogram you will see that it has come down by a couple of decimal place so reducing the number of neuron even though have only few weights right now it is still looks a little bit a Gaussian but you will see that the corresponding errors have actually gone up.

So if we come here and let me go back and instead of two suppose I use twenty and I trained this network you will see that the errors extremely low now it has gone to the order of ten to the power minus seven luckily still the training and the testing errors are approximately of the same size so is the validation so you know that we have not yet over fit the problem, so this is a simple example of neural network if you actually want to use this tool box let me show you an example here.

(Refer Slide Time: 13:13)



Save Results

Generate MATLAB scripts, save results and generate diagrams.

Recommended >> Use these scripts to reproduce results and solve similar

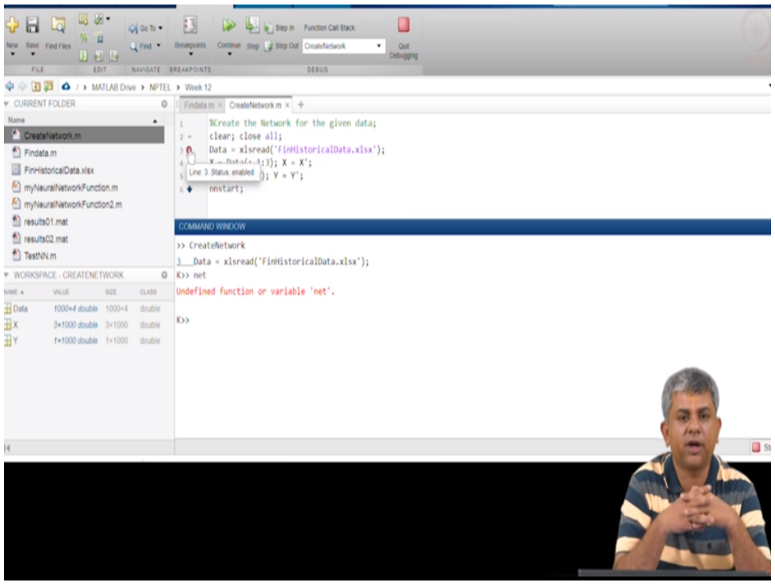
Generate a script to train and test a neural network as you just did with this tool: Simple Script

Generate a script with additional options and example code: Advanced Script

Save Data to Workspace

[✓] Save network to MATLAB network object named:	net
[✓] Save performance and data set information to MATLAB struct named:	info
[✓] Save outputs to MATLAB matrix named:	output
[✓] Save errors to MATLAB matrix named:	error
[ ] Save inputs to MATLAB matrix named:	input
[ ] Save targets to MATLAB matrix named:	target
[ ] Save ALL selected values above to MATLAB struct named:	results

Save results and click [Finish].



```
%Create the Network for the given data;
clear; close all;
Data = xlsread('FinHistoricalData.xlsx');
X = Data(:,1:3); X = X';
[illegible]); Y = Y';
nnstart;
```

```
>> CreateNetwork
3   Data = xlsread('FinHistoricalData.xlsx');
K>> net
Undefined function or variable 'net'.

K>>
```

You can save this network as net, so you will notice here so this is a very simple example of what can be done with neural networks we had some data set with thousand points but we did not know the physics we will pretend that we will did not know the physics but we had a lot of historical data we saw that we could easily fit it using a neural network now this has several uses as I said it can be use as a surrogate it can be used for inverse problem etcetera in the next video we will see a little bit of a more complex problem, thank you.