Intelligent Systems and Control

Prof. Laxmidhar Behera

Indian Institute of Technology, Kanpur

Module-1 Lecture-1

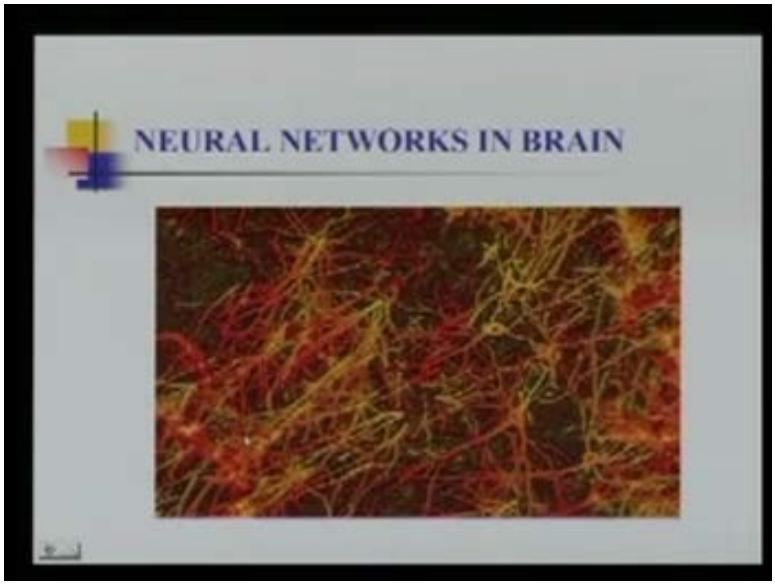
Linear neural networks

This is a course on intelligent control. Today, we will have the lecture 1 of module 1, neural networks. What is a neural network? What you are looking at is human brain, the structure of the human brain. It may appear in the beginning or at the very outset that it is a very homogenous substance. But in reality, it has various diversities.

(Refer Slide Time: 00:54)

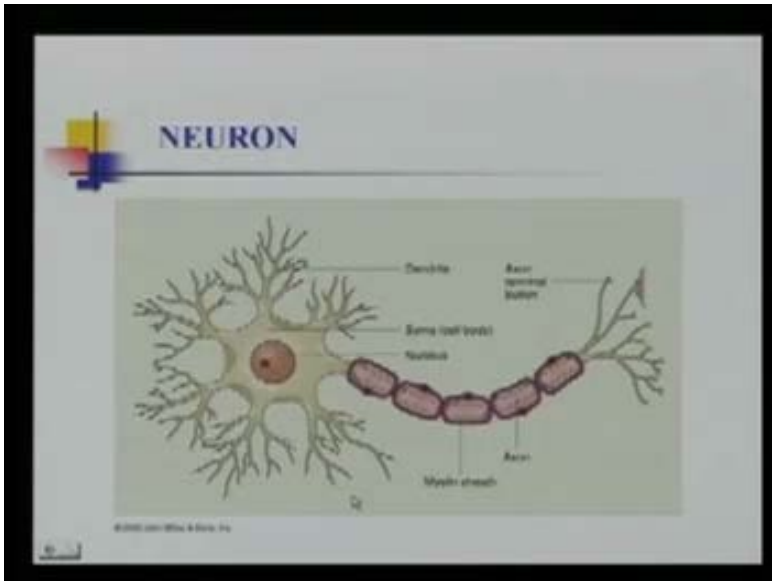
Each section of the brain functions differently. They have different functionalities and the field of neural networks, the artificial neural network, although may not have any connection with the real neural network in the real sense, nonetheless, the field of artificial neural network is inspired by the studies in real neural network and real neural networks are sitting on this brain. There are very complex connections among the neurons of this brain that you are looking at. Researchers over time have developed various models of the neural networks just by studying this human brain.

(Refer Slide Time: 02:12)



Look at the complex structure of the connections among neurons. We can see, these are all neurons and each neuron is connected to other neurons through these internal connections which we will call as dendrites and the whole structure is very complex.

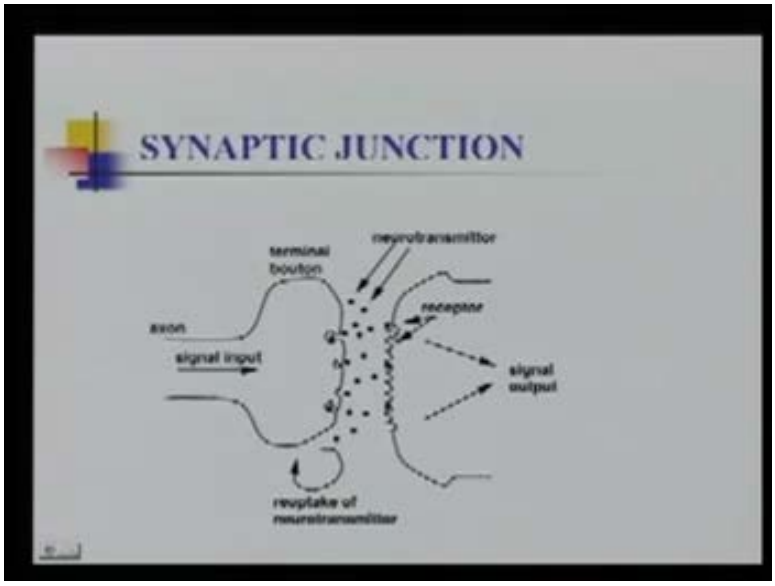
(Refer Slide Time: 02:44)



What is a neuron? A neuron is the basic processing unit in a neural network sitting on our brain. We can see that this is the structure of neuron where, this is the nucleus, this is the cell body and

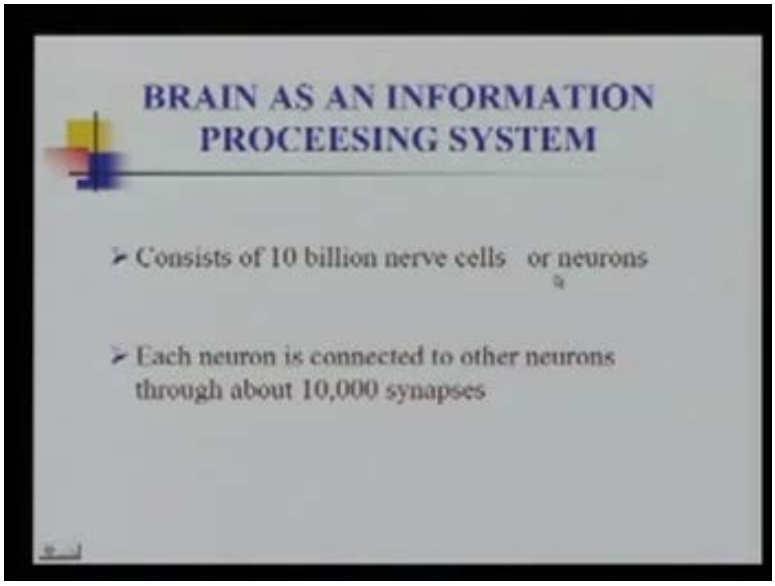
these are all dendrites. This is the Axon. The Axon is like an output node of the neural network and these are all inputs. They carry signals from other neurons. Such neurons, when they are connected with other neurons, they give rise to processing of information in a unique way inside the brain. When one neuron is connected with another neuron, they are connected through synaptic junctions.

Refer Slide Time: 03:51)



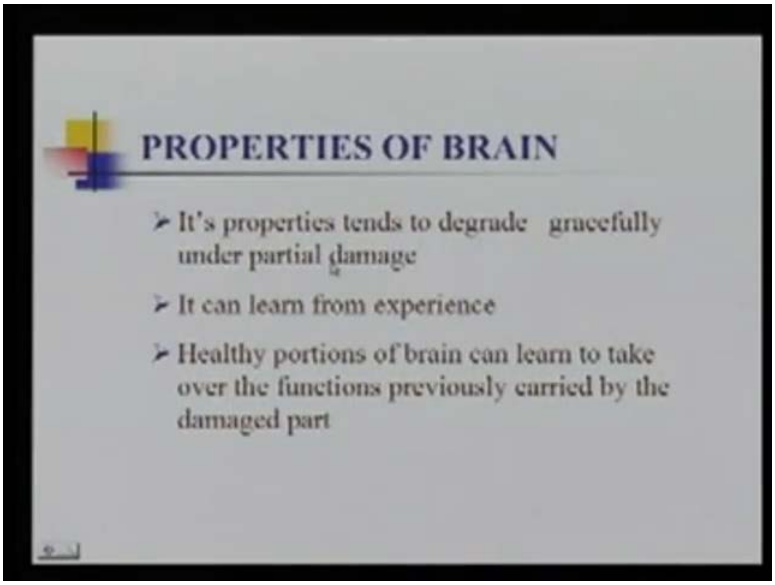
Again the dynamics of this synaptic junction is complex. We can see the signal inputs from the action of a neuron and through synaptic junction an output is actuated which is carried over through dendrites to another neuron. Here, these are the neurotransmitters. We learned from our experience that these synaptic junctions are either reinforced or in the sense they behave in such a way that the output of synaptic junction may excite a neuron or inhibit the neuron. This reinforcement of the synaptic weight is a concept that has been taken to artificial neural model.

(Refer Slide Time: 05:06)



The brain that we showed consists of 10 billion nerve cells or neurons. Each neuron is connected to other neurons through about 10,000 synapses. It may not be an exact figure, but estimation.

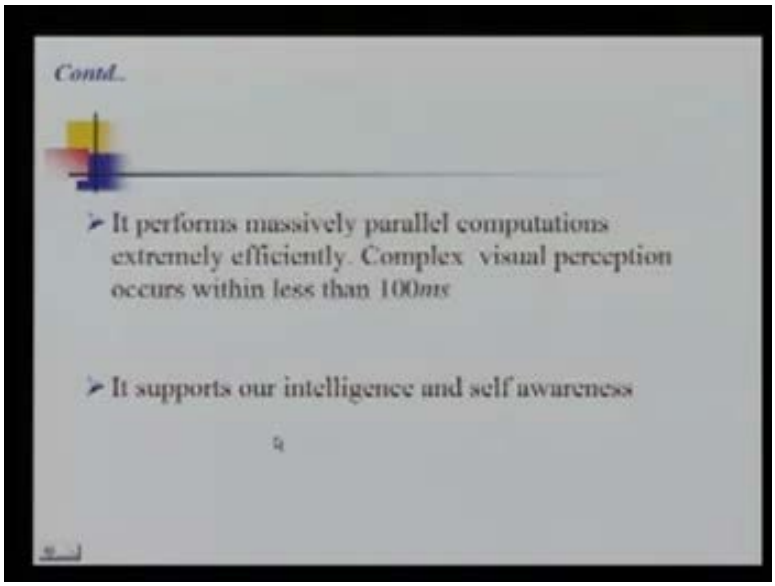
(Refer Slide Time: 05:32)



While saying so, let us look at some of the properties of the brain or neural networks inside the brain. The properties tend to degrade gracefully under partial damage. A part of the brain, as we saw before, if this part of the brain becomes dysfunctional, then another part of the brain takes

over the function of this. The damage of this part of the brain will not completely destroy the total functionality of the brain. The brain will continue to function and while functioning, the healthy part of brain tries to take over the functionality of the damaged part. This is a very important property of the real neural network or a real brain. Second one, it can be learnt from experience; it is one of the key properties of the brain that through synaptic junction reinforcement, the brain learns. As I said, healthy portions of brain learn to take over the functions previously carried out by the damaged part.

(Refer Slide Time: 07:08)

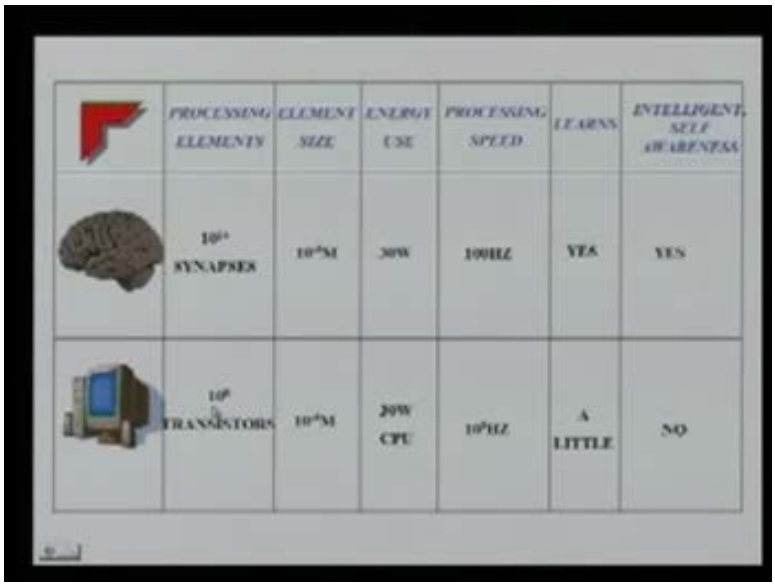


The other property is that it performs massively parallel computations extremely efficiently. For example, complex visual perception occurs within less than 100 milliseconds. Imagine an artificial machine or a computer trying to remember or trying to recollect an image or trying to recognize the image. It takes a long time, whereas the brain does this job within an order of 100 milliseconds. This is something unique to the brain as to how parallel massive computational activities are being carried on. As I said earlier, these massively parallel computations are because of this complex, parallel connection amongst neurons. The final one is it supports our intelligence and self-awareness.

Nobody knows the actual basis of intelligence and self-awareness. All of you must be very aware about your own experience. When you try to solve a problem, at some point of time you solve

the problem and at some other point of time, you fail to solve it. From where does the intelligence come to solve the problem? This is something we have still not answered. Although scientists believe this brain is the source of intelligence, but we are not very clear where this intelligence sits. Similarly, self-awareness: when I solve this problem, I am very much aware about the problem that I am going to solve, or when I see an image I am very clear that I perceive the image. These are the issues that have not been answered even in a real neural network. Nobody knows where from intelligence comes and where self-awareness comes from.

(Refer Slide Time: 09:46)

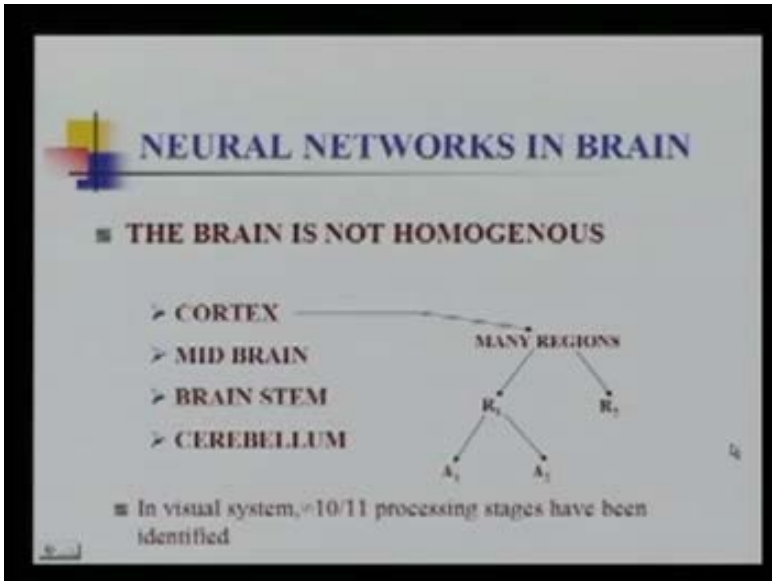


Now, I have given a comparison between a brain and computer. As I said, the brain has 10 to the power 10, approximately, nerve cells or neurons and each neuron is connected to other neurons through around 10 to the power 4 synaptic weights. In that way, the total number of synaptic weights sitting in our brain is approximately 10 to the power of 14. This is approximate, not exact figure; whereas in our computer, we have transistors whose order is 10 to the power 8. The element size of synapses is 10 to the power minus 6 whereas, in computer the transistor size also 10 to the power minus 6 units. The energy that is being consumed by our brain is 30 watts, whereas the CPU in a computer also consumes 30 watts. But the contrast is the processing speed. The normal frequency of the neuron, the operational frequency of neurons is 100 hertz, whereas the operation of the CPU is in the form of Giga hertz; pretty fast. But as I said, although the

computer has such fast speed, a traditional computer, however, when it comes to certain processing like pattern recognition and language understanding, the brain is very fast.

Although, present day computers has almost taken over popular human operation in terms of multiplication, arithmetic calculation and problem solving, there are certain fields, like pattern recognition, language understanding, where brain is very fast. When you compare in terms of learning, the ability to learn is very much there in human brain. We have very little idea about how we learn. We have some ideas, but not in completeness, like when a small baby grows the way she learns things fast. Probably, we have now developed a robot, an over artificial machine that learns like a little baby that grows and learns. In that sense, if you compare, we have not developed a machine that really learns like the way we learn. As I said, intelligence and self-awareness, first of all we do not understand these subjects as to how they come about although they are there with us. But these features are absent in an artificial machine.

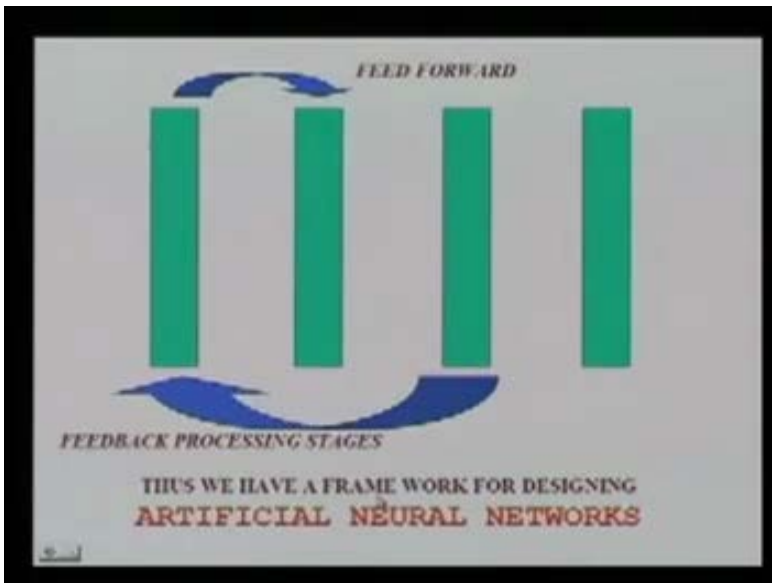
(Refer Slide Time: 14:14)



We talked about the brain. The real brain of the human, it is not homogeneous. You know it has various regions. The cortex, midbrain, brain stem and cerebellum are the various parts of the brain. Each part of the brain has many regions. If we take cortex, I think this is a major processing area in the brain. In the cortex, you have the visual cortex, the auditory cortex and like that there are various regions. In each region there are many areas. For example, one of the

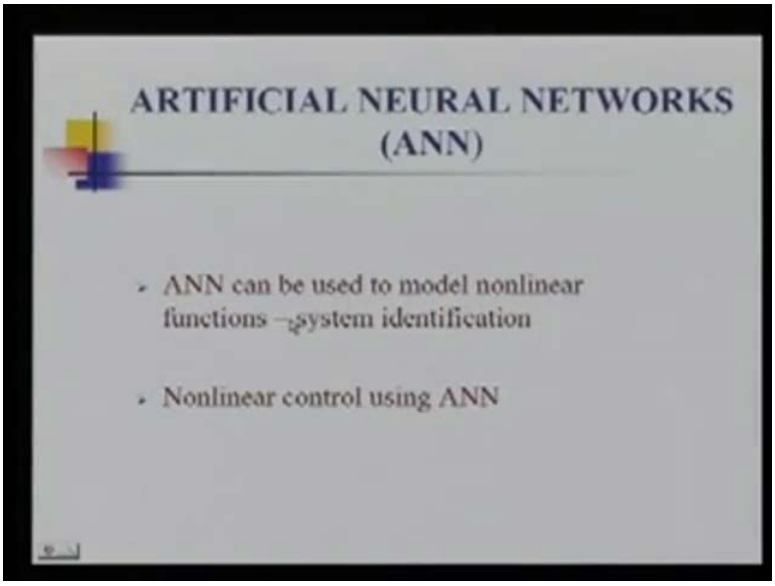
regions of the cortex is the visual cortex. It has been studied in detail by researchers and they have found out around 10 to 11 processing stages in this visual cortex. This is the most studied region in the human brain, because visual processing has attracted the attention of the researchers at large.

(Refer Slide Time 14:32)



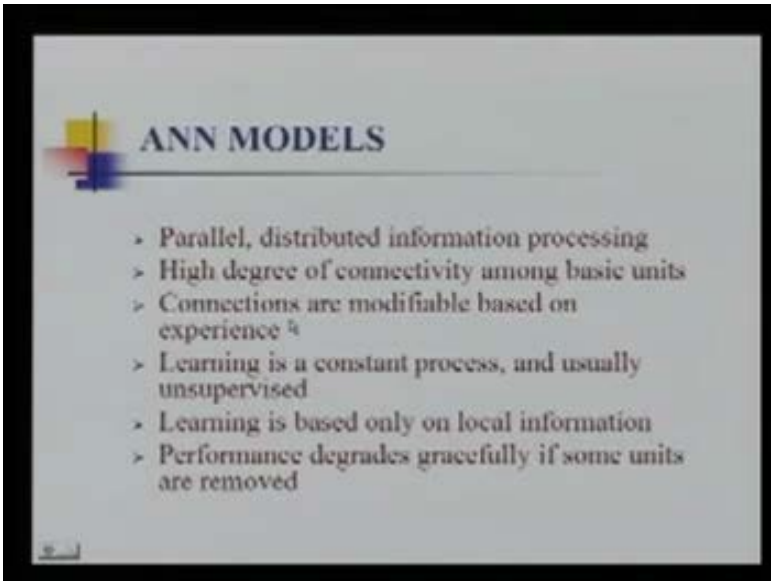
These stages they have identified. There are 10 stages and they found that there are connections called feed forward. For example: stage 1, stage 2, stage 3 and stage 4 and so on. We have been able to identify that there are connections in a feed forward manner; stage 1 to stage 2, stage 2 to stage 3 and so on and also feedback manner lies; stage 3 to stage 1, stage 3 to stage 2, like that. What we gave in this lecture until now is some idea as to what is real neural network. As I said, the brain is very complex.

(Refer Slide Time: 15:35)



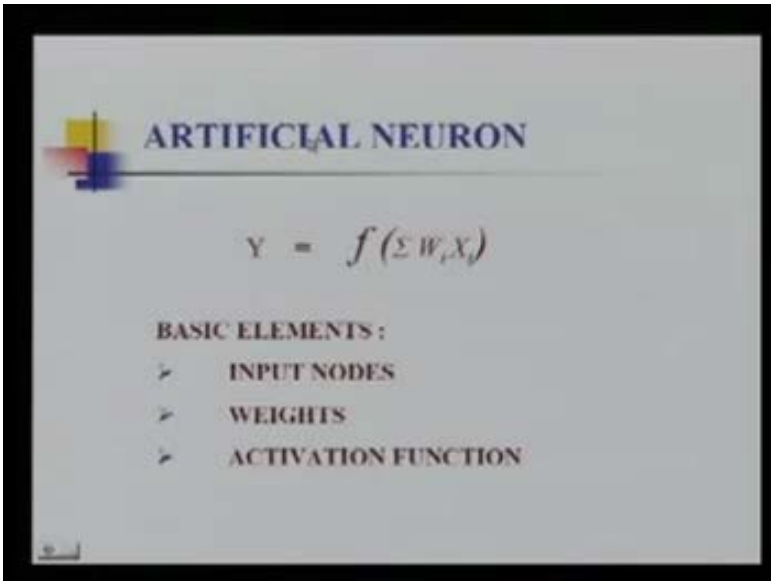
We have very little idea about the brain, although we have understood some of the properties of the brain through studies during the last few centuries. Based on the studies, researchers have tried to develop an artificial neural system. The objective is to create artificial machine and this artificial neural networks are motivated by certain features that are observed in human brain, like as we said earlier, parallel distributed information processing. The brain consists of 10 to the power of 10 nerve cells approximately and they are connected among each other in a very complex manner. Each neuron is a processing unit and each processing unit is connected to other processing units. Simultaneously, one unit is connected to another 10 to the power 4 units approximately. That gives the connection structure of the real neural network - a structure called parallel and distributed structure.

(Refer Slide Time: 16:00)



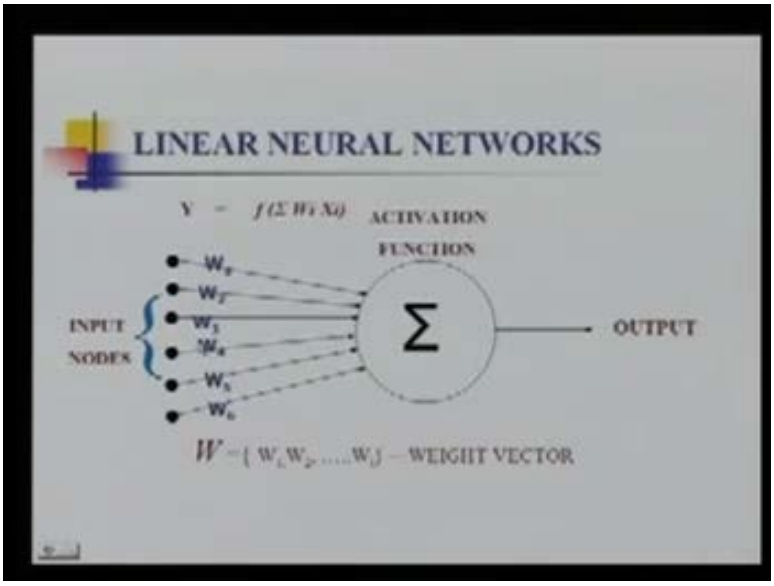
Information processing is carried out inside the brain in parallel and also in distributed fashion. It is not centralized like the CPU that you see in an artificial computer in which the computation takes place centrally in the central processing unit which is different in its architecture in comparison to the brain neural network. A high degree of connectivity among the basic units, connections are modifiable based on experience, learning is a constant process and usually unsupervised, learning is based only on local information and performance degrades gracefully, if some units are removed. These are some of the properties that we have seen in real neural network. Now, can we bring these properties while developing an artificial neural network model- that is a fear.

(Refer Slide Time: 18:20)



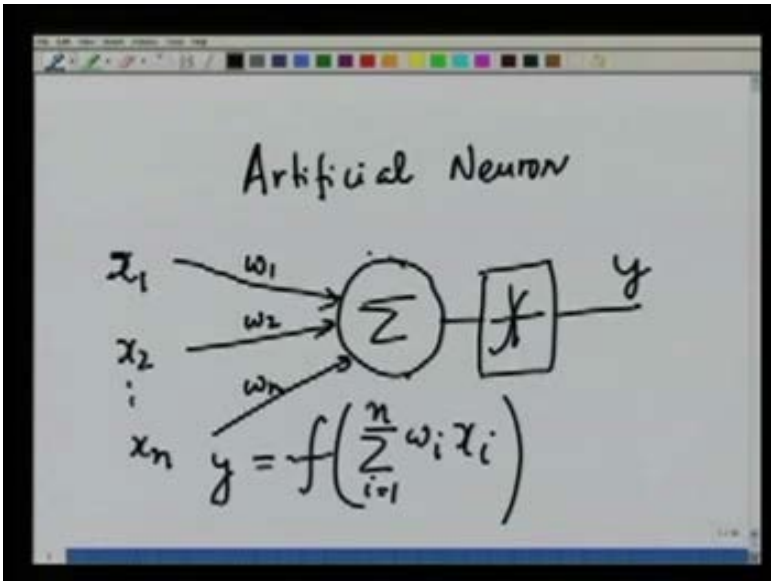
Now, being biologically inspired, we are talking about a field called artificial neural network and we have set some agenda on how to build a thinking machine or a learning machine which has the capability or which can be mimic the abilities of biological organism like a human being. Now, let us go to the most basic computational unit in an artificial neural network. Obviously, it has to be an artificial neuron.

(Refer Slide Time: 19:22)



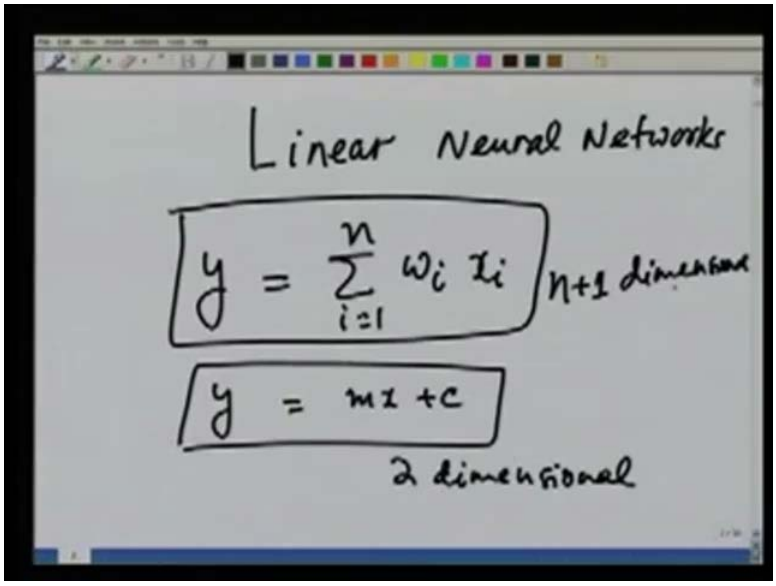
This artificial neuron has three basic elements: nodes, weights and activation function. This is a node and these are input signals. Between input nodes and output nodes, there are synaptic weights w_1, w_2, w_3, w_4, w_5 and w_6. There can be as many weights and these weights are multiplied with the signal as they reach the output unit, where the output is simply sum of the signal multiplied with the weights and then this output goes to an activation function; this is f.

(Refer Slide Time: 22:12)



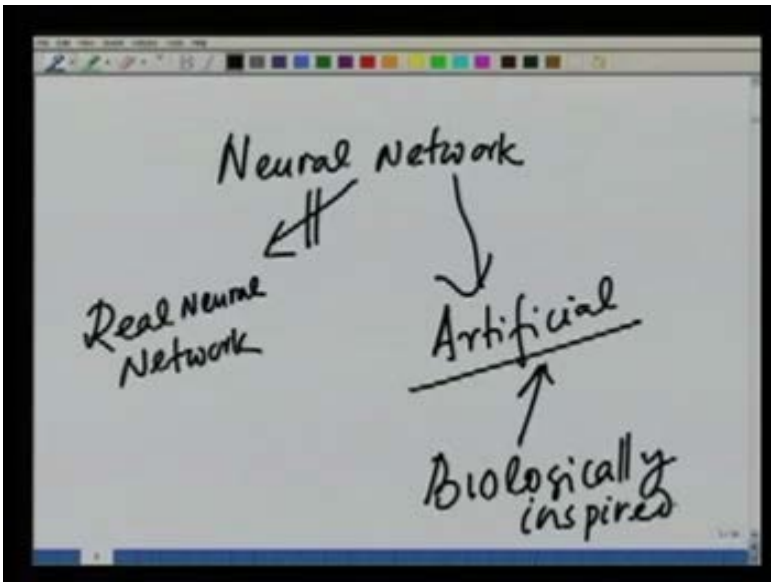
What we are talking about is the artificial neuron- a simple basic processing unit. As I showed you in the slide, we have an output node. At this output node, signals reach. Let my signals be x_1, x_2, x_n and then I put an activation function here and I have y. As you can see here, I can easily write down; these are weights w_1, w_2 and w_n. The total signal that is reaching here is the summation being $w_i x_i$, i is equal to 1 to n. This is your total output reaching here and you activate this total input by a function f. That is your output. What you are seeing is actually a nonlinear map from input vector x_i to output y. Here, in this case, we have single input and we also have multi output if we take more nodes here. What we are talking about is a single neuron. A single neuron has single output but multiple inputs. Inputs are multiple for a single neuron and the output is unique, y and this y and the input bear a nonlinear relationship, this f. Neural networks can be built using this single neuron. We can use the single neuron and build neural networks.

(Refer Slide Time: 22.40)



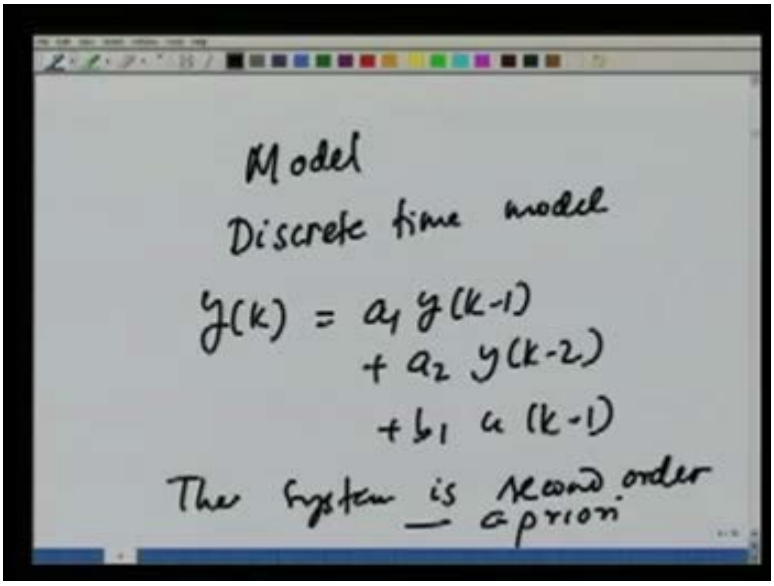
Today, we will discuss the linear neural networks. What is linear neural network? I have a neuron whose output is simply $w_i x_i$, i is equal to 1 to n. I have this expression. That is I do not have a nonlinear activation function. That is my y is a linear relationship with weight. All of you know, if I write y is equal to mx plus c in a two-dimensional place, this is a linear equation. This is two-dimensional. Similarly, this is an equation. Input …… is n dimensional, output is one more; this is an equation with n plus 1 dimensional plane.

(Refer Slide Time: 24:02)



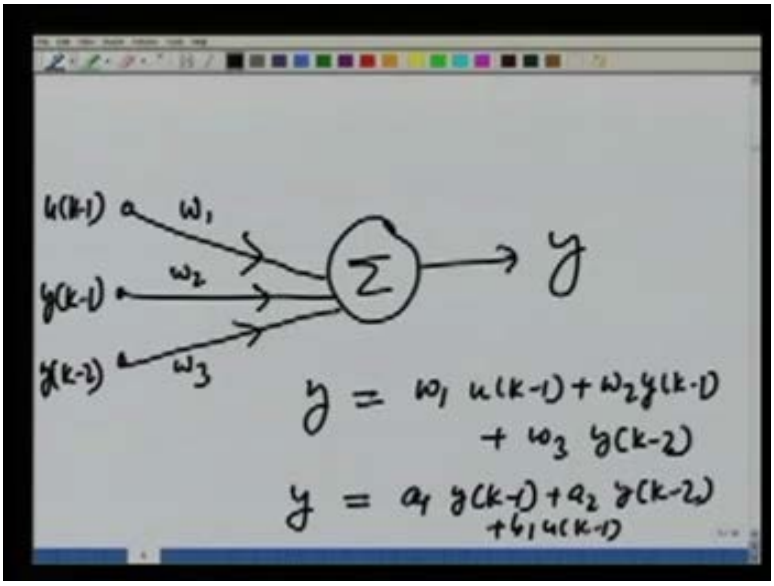
When my neural network output When I say neural network, kindly remember, it does not have connection with real neural network. There is no connection, whereas when I say neural network in this class, it means always artificial. The only connection between the artificial neural network and real neural network is that these artificial neural networks are biologically inspired and that is only the difference.

(Refer Slide Time: 25:37)



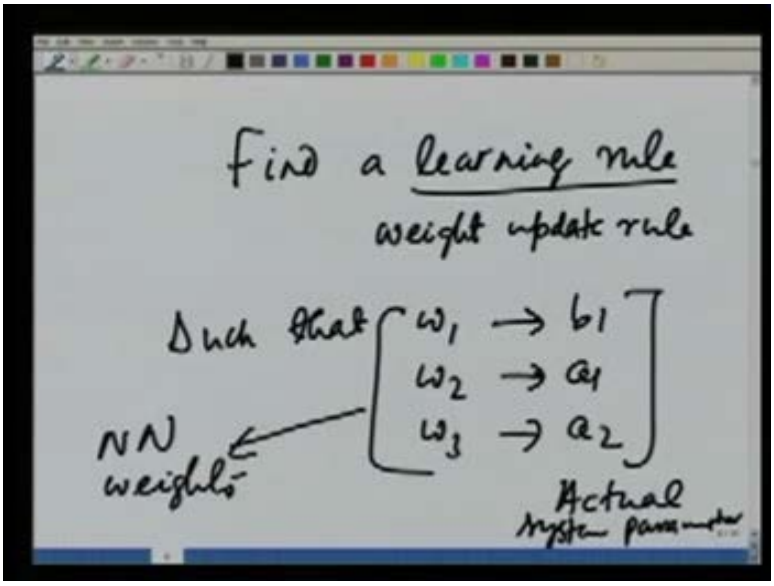
Let us see how we develop a linear neural network. All of you want to study in this course- intelligent systems and control. It will be nice, if I take an example of a control system and then explain to you what is a linear neural network? All of you know that in a control system we always talk about model. Let us talk about a discrete time model. y k is equal to a_1 y k minus 1 plus a_2 y k minus 2 plus b_1 u k minus1. If I have such a dynamical model and I want to feed this dynamical model to a system about which I have some apriori knowledge that the system is second order. This is apriori; this knowledge is aproiri and I use this apriori knowledge, I put forth a model. But what I do not know is a_1, a_2 and b_1. These parameters I do not know and I want to identify. What do I do? I observe the input and output from the system and try to fit this model using this data and try to identify a_1, a_2 and b_1.

(Refer Slide Time: 27:28)



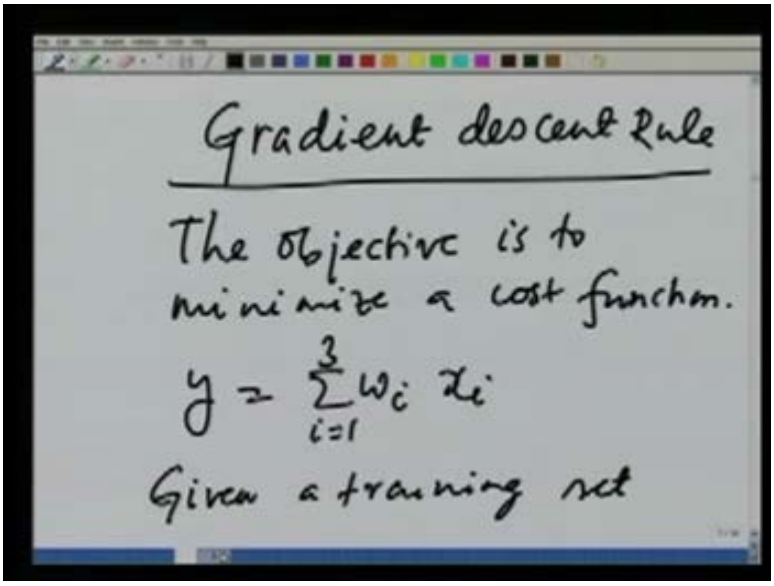
The objective is how do I solve this problem using a neural network? Since this model is linear, I will use a linear neural network. What I will do in my neural network is I will have a single neuron, because this is only a single output. I do not need more number of neurons here, only a single neuron and I have three inputs. As I said before, the inputs are y k minus1 y k minus 2 u k minus 1. These are the inputs, I put it here. u k minus1 y k minus1 y k minus 2 ; these are three inputs. I connect these three inputs through three weights w_1, w_2 and w_3 and I add them. My y is w_1 u k minus 1 plus w_2 y k minus 1 plus w_3 y k minus 2, whereas my actual system is supposed to be a_1 y k minus 1 plus a_2 y k minus 2 plus b_1 u k minus 1. a_1, a_2, b_1 represent the system parameters. I would like to know the parameters a_1, a_2, b_1. What I do is I keep this model and this model is sitting in my computer and to the computer, I feed this data u k minus 1 y k minus 1 y k minus 2 at every instant to the model that is sitting in my computer and what is the actual output of the system? y, given y k minus1 y k minus 2 u k minus1. I give this data to this model and I have to develop a methodology to update my weights w_1, w_2, w_3 in such a manner and finally w_1 becomes b_1, w_2 becomes a_1 and w_3 becomes a_2.

(Refer Slide Time: 30.05)



The objective is to find a learning rule which you can also say as weight update rule such that w_1 converges to b_1, w_2 converges to a_1 and w_3 converges to a_2. These are actual system parameters and these are neural network weights. I hope the problem is very clear to all of you.

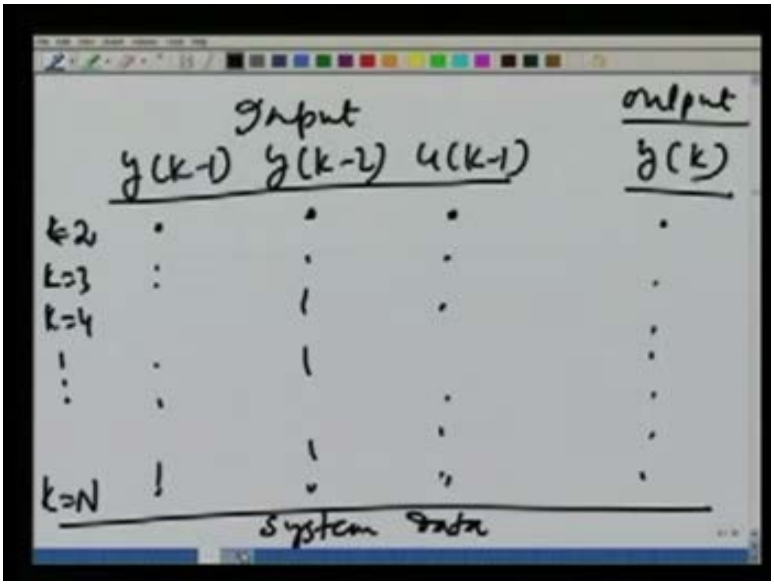
(Refer Slide Time: 31:30)



Now, I want to identify the weights w_1, w_2 and w_3. How do I do it? This is called a popular This learning rule is derived using the popular gradient descent rule. What is gradient descent rule?

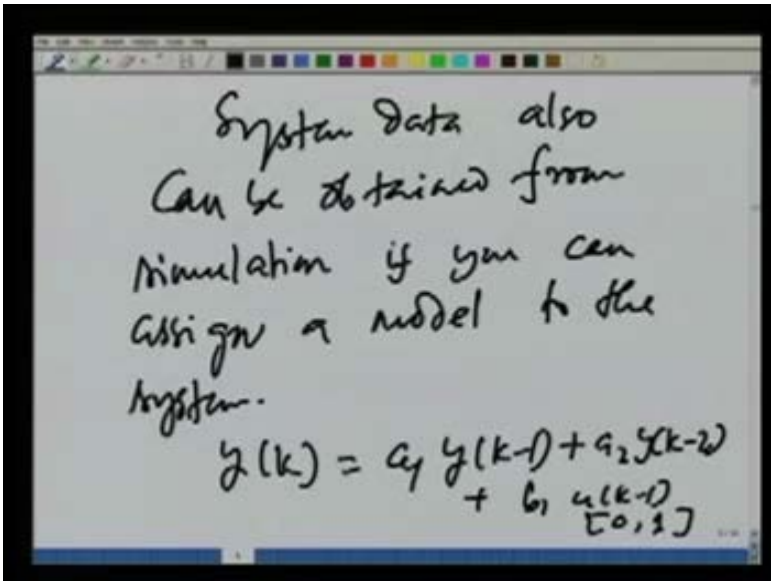
When I try to learn the weights, the objective is always to minimize a cost function. That is the objective. I want to minimize a cost function. If I can minimize this, now your y is equal to w_i x_i and in this case i is equal to 1 to 3, for this example. But generally, this i is equal to 1 to n; you have n weights. But in our case the example that we took for a second order model, you have only three parameters to identify; that is w_i. What you are given? You are given a training set.

(Refer Slide Time: 32:43)



What is a training set? Now I have y k minus 1 y k minus 2 u k minus 1 and y k. This is my input and this is my output y k. I do some experiments in my system. I give various inputs. You can see here I can define k equal to 2, k equal to 3, k equal to 4 and so on. What am I doing? I generate some random input or some input I generate here and correspondingly, I have some initial values of y k minus1, some initial values of y k minus 2. When I put that values, my system gives When I give u k minus 1 my system gives y k and I go on giving to the system various inputs and I get the various outputs here. I know also the previous output; y k minus1 y k minus 2. Once I know what y k is that my system is actuating or the system output is, I know also y k minus 1 and y k minus 2. This data I am getting from my system. This is called system data.

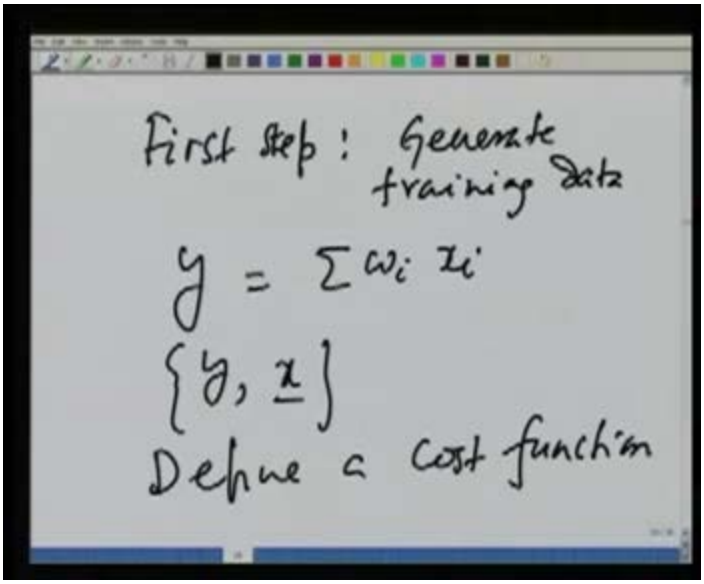
(Refer Slide Time: 34.28)



This system data also can be obtained from simulation if you can assign a model to the system. This also you can do. That is in computer, what I will do is given y k equal to a_1 y k minus1 plus a_2 y k minus 2 plus b_1 u k minus 1, I will generate some random input here, u k minus1, say 0 to 1 and I give this input to the system. I have some values a_1, a_2 and b_1, because I have some idea about the system. I know the system a_1, a_2 and b_1; so, I generate data by just simulating this system. I just put the value k equal to 2, 3, 4, 5. Then I will compute what are y_2, y_3 and y_4.

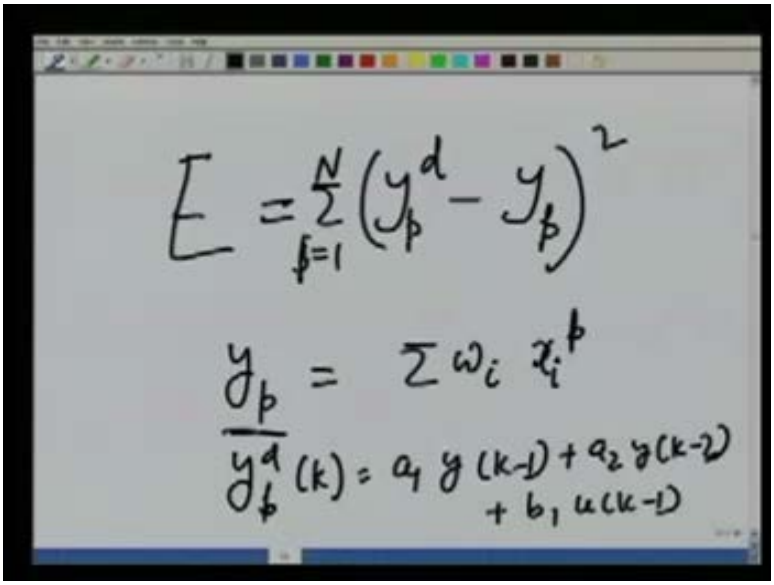
So, this point is very clear to you.

(Refer Slide Time: 36:00)



The first part is to generate the training data. Our model is sigma w_i x_i. I am generating data y and x. Once I have this data, I define a cost function. What is the cost function?

(Refer Slide Time: 36:49)

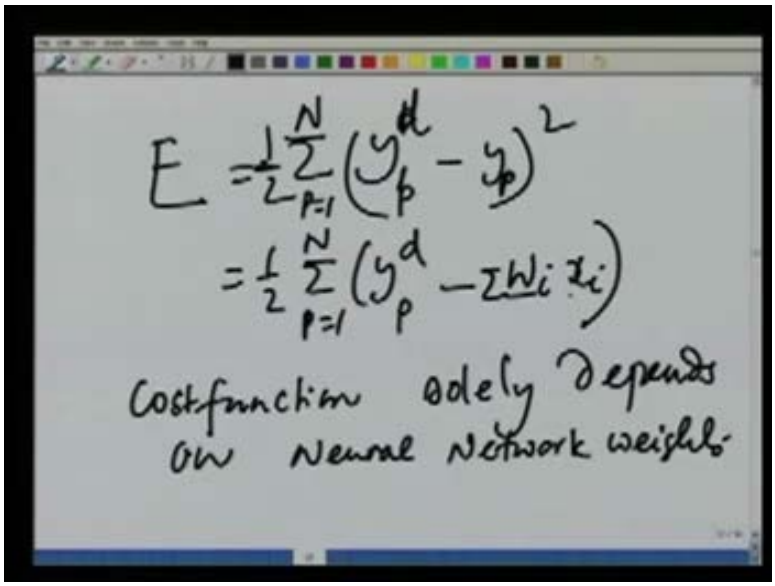


The cost function E is normally given as y_p d minus y_p. This is p whole square sigma; p equal to 1 to N. What is the meaning of that? My actual system gives me the output y d p and the neural network gives y_p. Again, to help you to remind, my y_p is sigma w_i x_i p. This is my p x input

pattern and p x input pattern gives me output y_p using the neural network weights w_i. These weights do not correspond to actual weights. We want to identify this weight and the system provides an output y p d, the actual system. In our case, the actual system was that is y p d k is a_1 y k minus1 plus a_2 y k minus 2 plus b_1 u k minus 1. This is my actual system and this actual system gives me a value y p d k and the neural network provides me y_p, corresponding to these values y k minus1 y k minus 2 u k minus 1 and a specific constant k.

What is the objective? The objective is that I give my neural network these weights, the various input patterns. Let us look at what the neural network output y_p is and I compute this output y_p for p equal to 1 to N and I find what is E? The objective is E should be minimum.

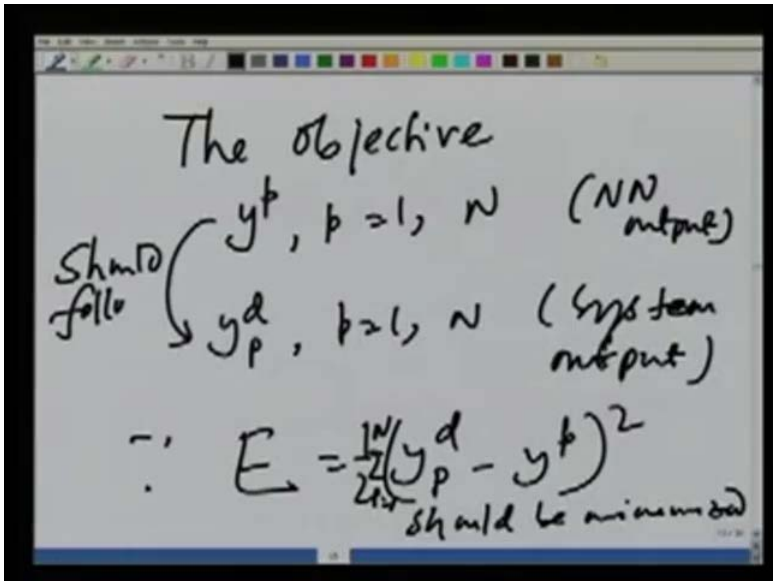
(Refer Slide Time: 39:34)



This E I am writing as sigma y_p d minus y whole square. I missed this half earlier. This is customary because, normally the energy function is written as half mv square. From that this half is coming here; so, half y_p d minus y square p equal to 1 to N patterns. This is actually a function p equal to 1 to N and this is a constant, because this comes from the system y_p d, system provides this output and my actual output is a function of weight. So, that is weight factor W_i. This I can write as W_i x_i. In essence, the cost function solely depends on neural network weights. I hope you understand now that the cost function solely depends on neural network weights. Why? Because the given input and output everything is known except the weights of the

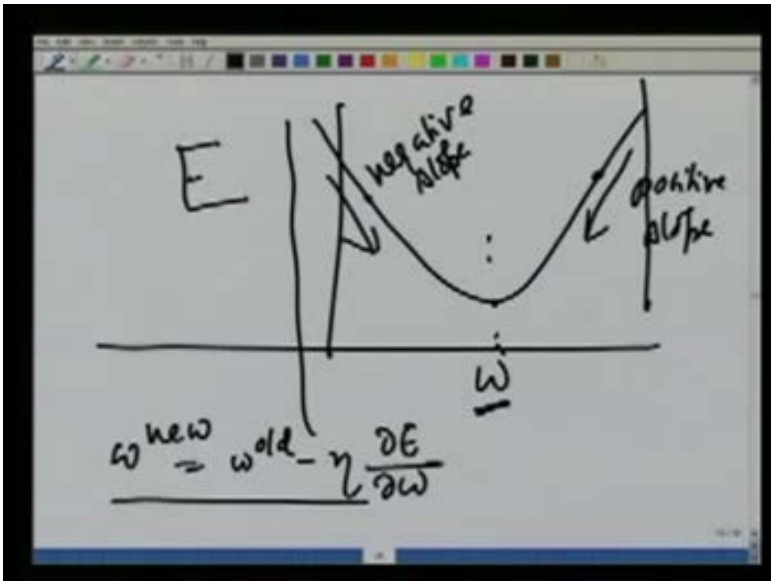
neural network. The cost functions because y_p d is known, it is given; x_i is known, because it is given. What else is left unknown here? Only weights. How do I update these weights such that my cost function..(Refer Slide Time: 41:45 min) What would you like? You would like given input x_i, my y_p should be exactly ….. as that of the y_p d. That is my neural network output should exactly follow what is the desired output of the system. That is the objective.

(Refer Slide Time: 42:30)



The objective is that is I have the neural network output, y_p from p equal to 1 to N. This is neural network output and my system output is y_p d, p equal to 1 to N. This is my system output and the neural network output should follow the system output. This implies my E as I defined earlier equal to y_p d minus y_p whole square half sigma p equal to 1 to N. This function should be minimized. I hope you understand this. This is the key. You must know before designing a neural network, what you are going to do? That is the most important point and here, what I am saying is this y_p the neural network output, should follow the system output y_p d. When you understand this, the cost function that I have defined is minimized. How do you do it?

(Refer Slide Time: 44:00)



Question is how do we minimize E? We follow gradient descent rule. What is gradient descent rule? Gradient descent rule, I plot E with respect to W. I already told you that E is solely a function of weight. This cost function is solely a function of weight and I want to see, how this E varies with W and what are we considering now? Remember we are only considering a linear neural network and for linear neural network, the normal kind of curve that you will find for any kind of linear system, E versus W. This we will also discuss later in detail, but now you just understand, you can also try; you try some linear function and define E and W; do a computer simulation and plot E versus W.

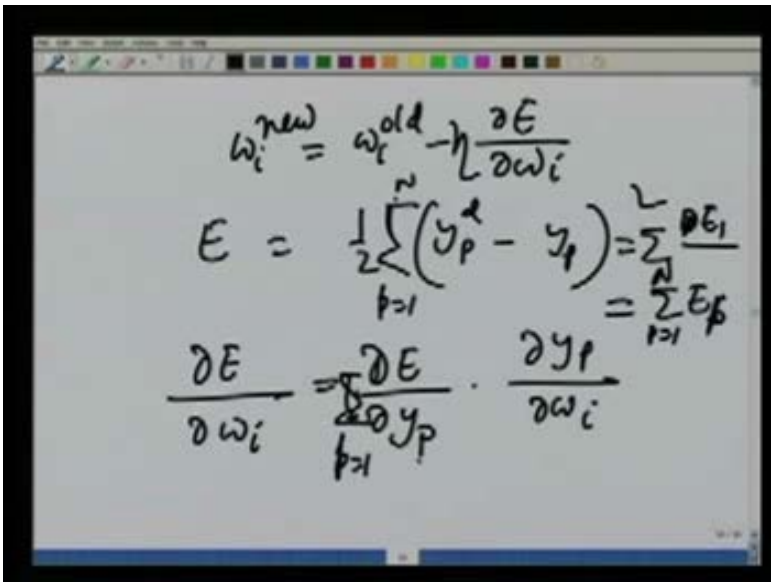
You will find a kind of concave function having one global minimum. I hope this is clear to you. This is a normal curve that I get when I plot E versus W. W is N dimensional, because I have N dimensional weight and that means this whole plot will be N plus 1 dimensional. But I cannot see things in N plus 1 dimensional. For the sake of clarity, I am showing you how E is the function of w in two dimensions; only in two dimensions. Let us see the weight. What I am doing here is at this point, I am drawing a slope and at this point I am drawing a slope. You can see that if I draw a slope here, this is a positive slope and this is a negative slope; a positive slope and a negative slope. The objective is how do I update my weight so that I can find this weight. This is the global minimum and this global minimum, this weight I have to find out. How do I find?

Imagine that you have a neural network. You start from blind; you are completely blind. What are the weights? You may be anywhere here or you may be anywhere here; either to the right or left of the actual minimum weight that you are looking for. The objective is that what is the method by which you can come from any position you are situated in the beginning to the actual position, the minimum weight.

The gradient descent rule says if I update my weight, w new equal to w old eta, in this manner if I do it, then I can easily come to this particular point. How do I do it? You just look at this learning rule. What is this learning rule? I am now situated here. According to this learning rule, del E by del w is a positive quantity here. Because it is a positive quantity, whatever is my old w new w will be less, because you will subtract. If I am here, my old w is here and I have negative slope. Because of that, this weight update algorithm will add something positive to the old point, because of the negative slope. So, this is negative. The negative sign becomes positive.

What is happening is whether you are here, if you are here, you are moving in this direction; if you are here, you are moving in this direction. Using this algorithm, you always tend to come to a point or towards the direction where my optimal weights are there.

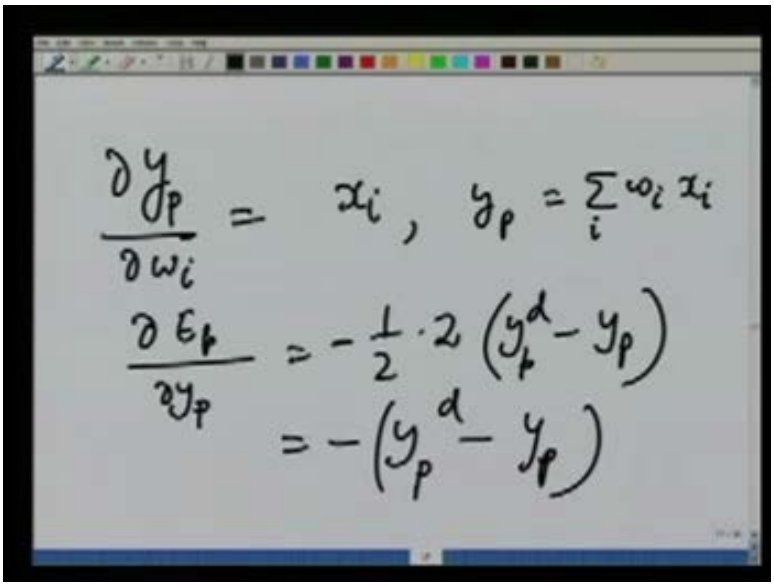
(Refer Slide Time: 49:05)



Now, the simple formula is how do I derive my algorithm? This is my algorithm number. I gave you the principle. Now, I will derive. Given E equal to half y_p d minus y_p whole square p equal

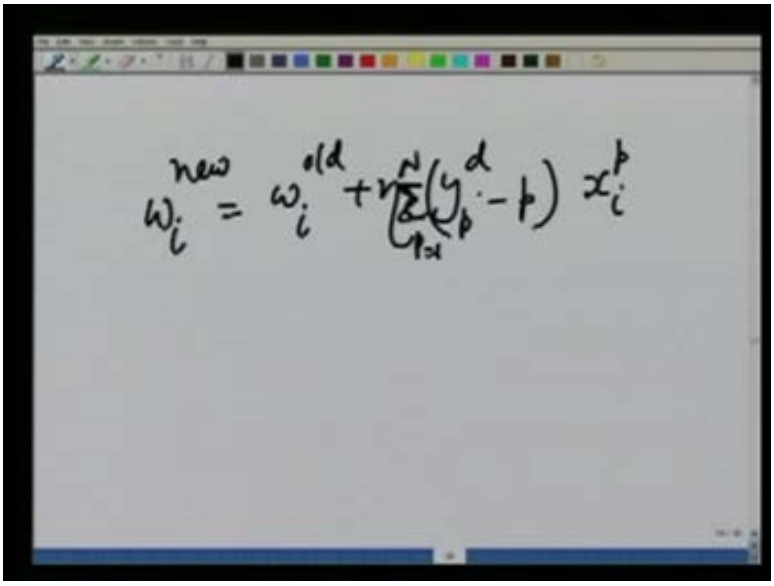
to 1 to N. What will I do? del E by del w. I put i here, because this is a vector. Del E by del w_i is half; I write like this. This is my del y_p; del y_p over del w_i and this has to be summed. I have a specific pattern p, I have a term and I differentiate that term with respect to w_i; that is del E by del y_p del y_p by del w_i, because y_p d is a constant term and I have N such terms. Naturally, I will try to compute this term p is equal to 1, 2. I can write this one as w del E_1; sorry. This is simply E_1. So, this term I can say simply say, E_i or $E_{p,}$ p is equal to 1 to N. This is my term. E is equal to my error square for N patterns. When I differentiate the total E with respect to $w_{i,}$ I get a term like this. Hope you are clear.

(Refer Slide Time: 51:34)



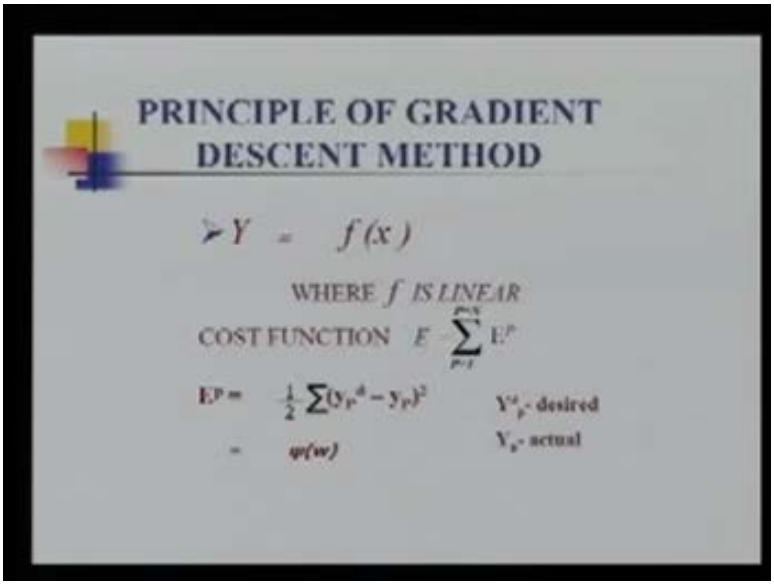
If I do that I find out del y_p by del w_i is simply x_i. That is because y_p is equal to sigma w_i $x_{i.}$ Then del E_p by del y_p is half is there; then the 2 will come from the square and y_p d minus y_p because with respect to y_p, you have to multiply minus. So, finally you land with a term like this.

(Refer Slide Time: 52:21)



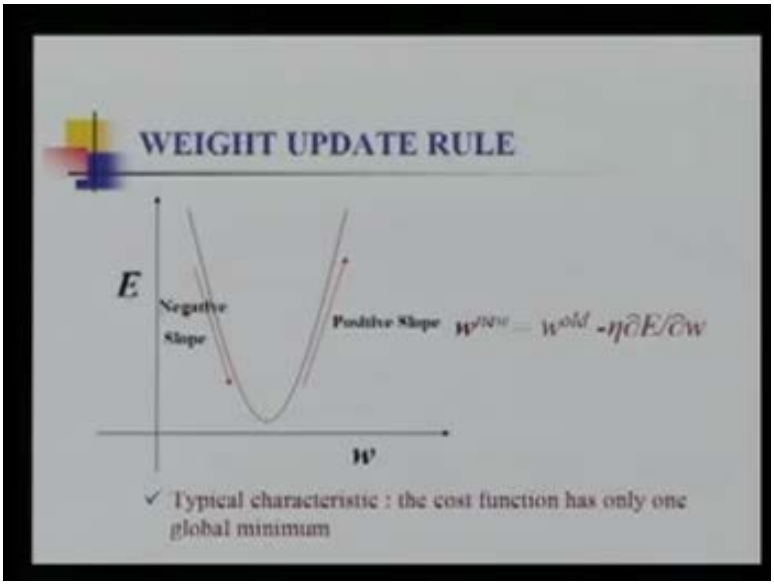
The final equation is w_i new is w_i old plus eta. You remember this eta becomes plus because I have a negative sign; y_p d minus p into x_i and this pattern will be there. This is the p th pattern and sigma p equal to 1 to N. This is my total the learning algorithm that I talked about.

(Refer Slide Time: 53:12)



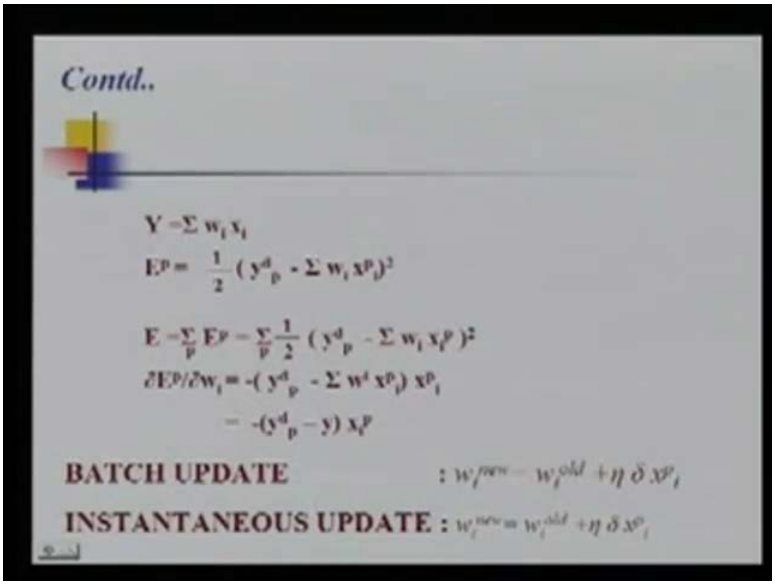
Let me summarize what we discussed today. We had a cost function of this form. This is for individual cost corresponding to each pattern.

(Refer Slide Time: 53:29)



This is the total cost for N patterns and we use the gradient descent rule.

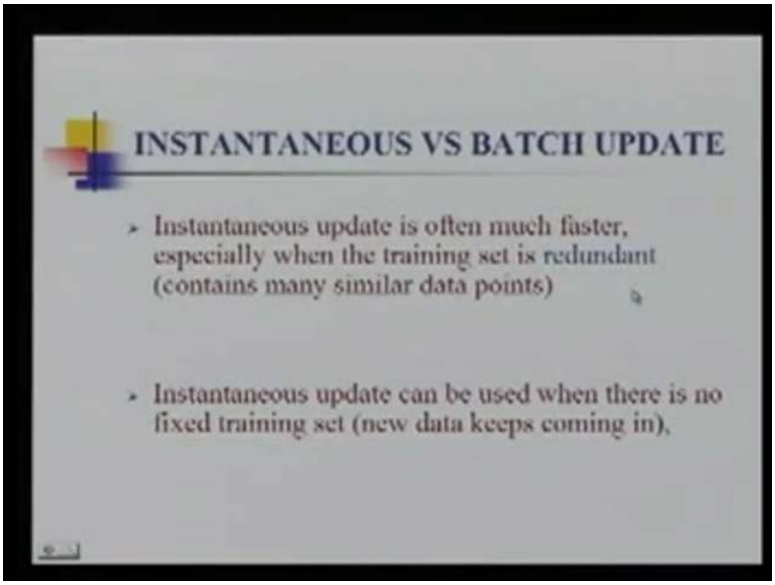
(Refer Slide Time: 53:44)



I explained to you, how using the gradient descent rule, wherever you may be in the weight space, you will always go to the global minimum and we derived the learning rule. That is for batch update, w_i new equal to w_i old plus eta into delta. delta is y_p d minus y_p that is delta and this is input x_i p. This is my general form. Here is a mistake. There is a sigma and when I make it

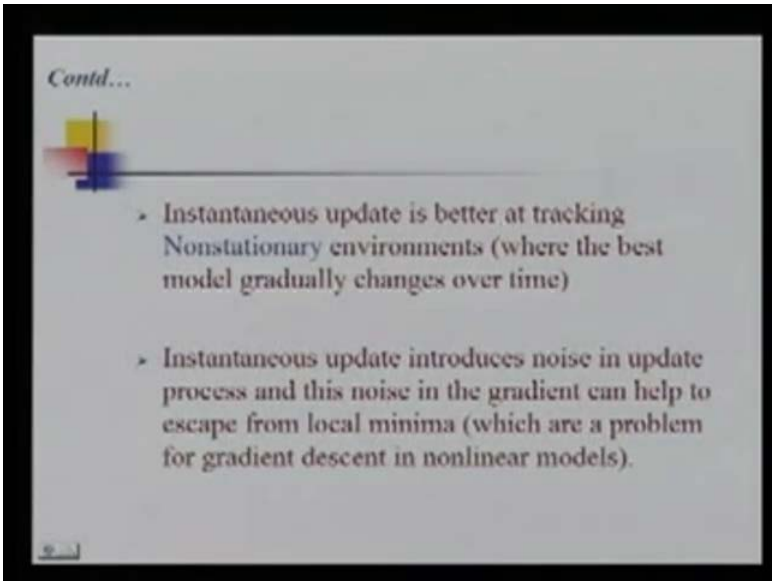
instantaneous update, that means I simply use this del E_p by del $w_{i,}$ then I have this simple rule. That is my instantaneous update.

(Refer Slide Time: 54:22)



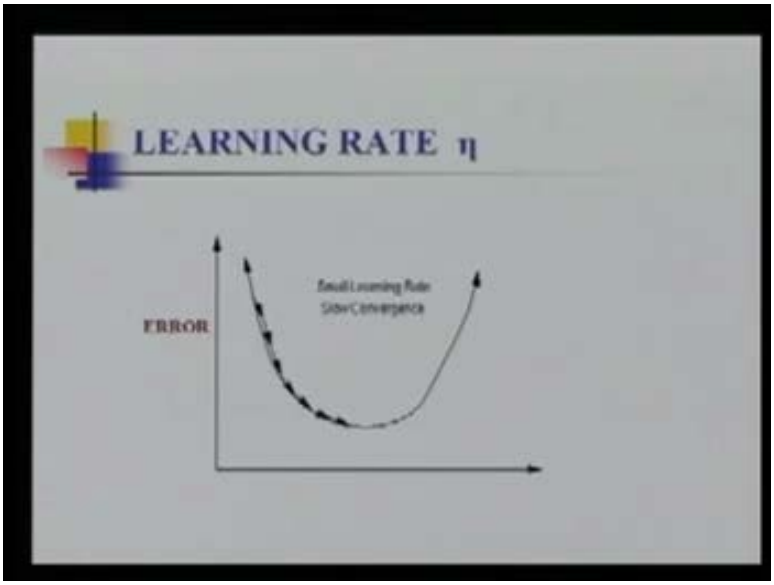
What is the difference between instantaneous update and batch update? Instantaneous update is often much faster, especially when the training set is redundant. It contains many similar data points. Instantaneous update can be used when there is no fixed training data; new data keeps coming.

(Refer Slide Time: 54:45)



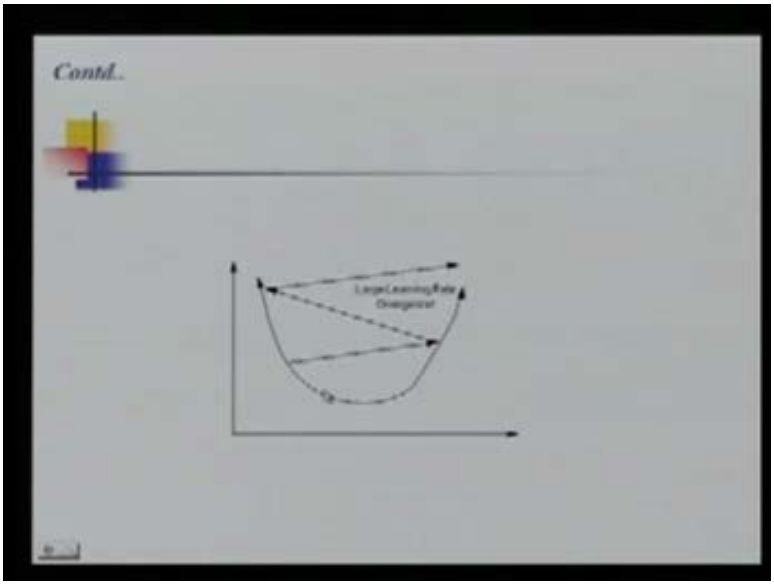
Also, instantaneous update is better at tracking non-stationary environments. Instantaneous update introduces noise in update process and this noise in the gradient can help to escape from local minima.

(Refer Slide Time: 54:56)



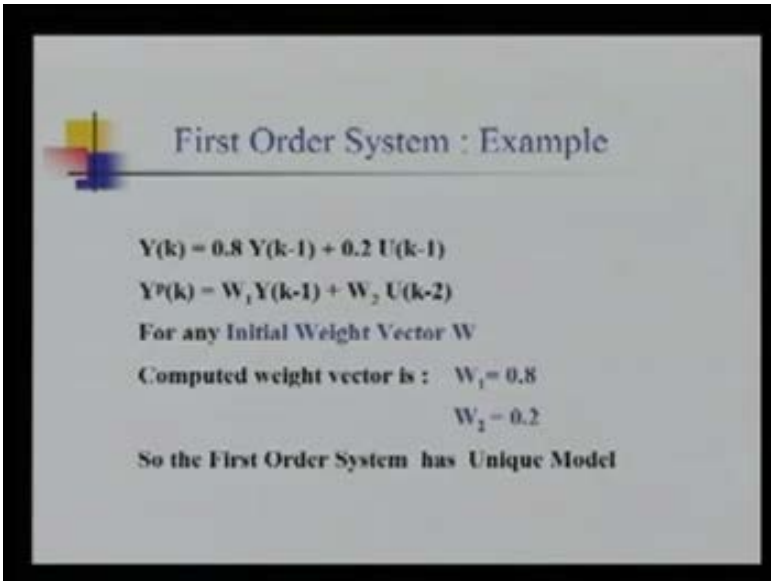
Let us see how to select this eta. If you take small eta, you can see, in this surface you slowly go towards the global minimum. This is called slow convergence.

(Refer Slide Time: 55:09)



In the contrary, if you take a very high value of eta you diverge. Eta has to be fixed drastically in such a way that your convergence speed is optimal as well as you go towards the global minimum; that is the objective.

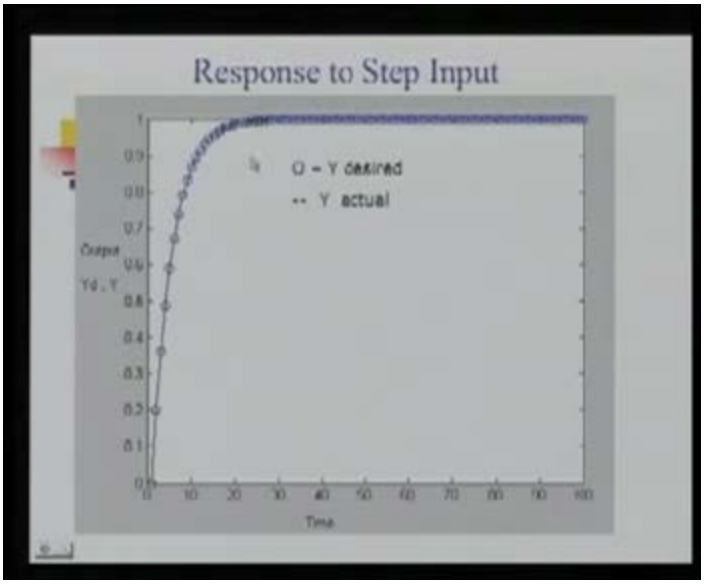
(Refer Slide Time: 55:30)



Now, we will end this lecture by taking an example. We will take a first order system.

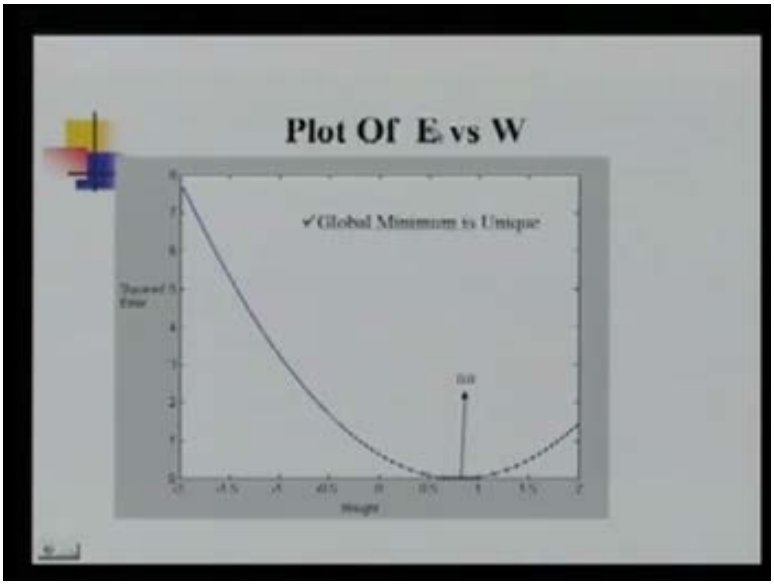
y k equal to 0.8 y k minus1 plus 0.2 u k minus1. This is my actual system. I generate data using this system. You can do a simulation experiment using this data from here. Then I have this neural network where W_1 and W_2 are unknown. In the beginning, I take the initial values to random, very small values between 0 and 0.1 and then, if you use the gradient descent rule, you finally reach W_1 equal to 0.8 and W_2 equal to 0.2 and you can see that W_1 0.8 matches, here 0.8 and W_2 0.2 matches, here 0.2. That is you have exactly identified the system.

(Refer Slide Time: 56:27)



Now, your actual system and neural network, the response is same.

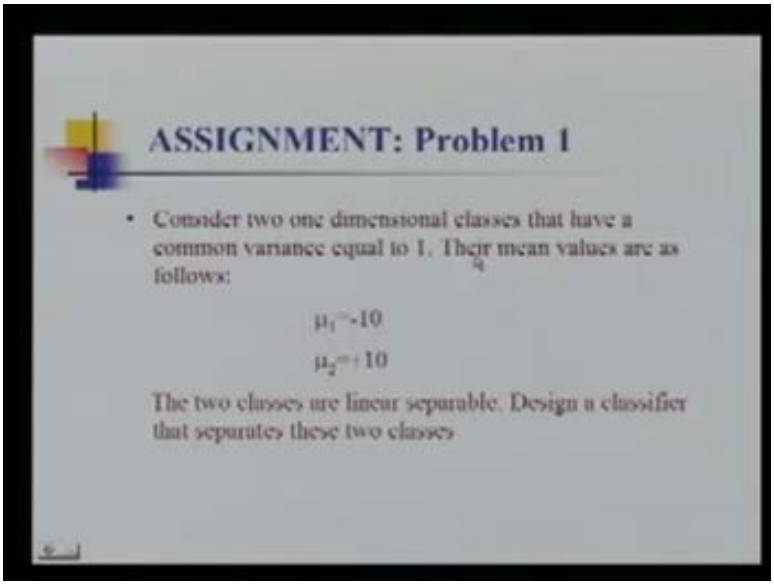
(Refer Slide Time: 56:32)



Let us see, if you plot E versus W, you will see you have a concave like function here and the minimum is exactly at 0.8. W, when I plot E versus weight which is the coefficient of y k minus 1.

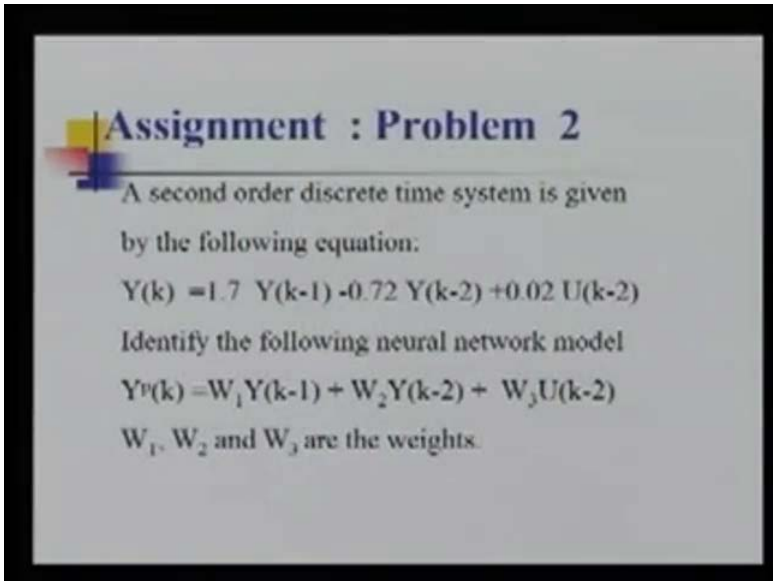
What we did in this class today? We explained how the artificial neural network models are being developed, being inspired biologically. Today, we discussed linear neural network. We discussed linear neural network in the context of system identification, because this is a course on intelligent control and I hope you are familiar with system identification problem.

(Refer Slide Time: 57:26)



I give you two assignments in this lecture. Problem one: consider two one dimensional classes that have a common variance equal to 1. Their mean values are as follows: mu_1 equal to minus 10 and mu_2 equal to plus 10. The two classes are linearly separable. Design a classifier that separates these two classes. This is the first problem.

(Refer Slide Time: 57:47)



The second problem is the extension of the example that we discussed today. A second order discrete time system is given by the following equation. You simulate this equation; generate data, input data and output data. Then identify the following neural network model where $W_{1,}$ $W_{2,}$ W_3 have to be identified and please solve this problem. Till then, good-bye!