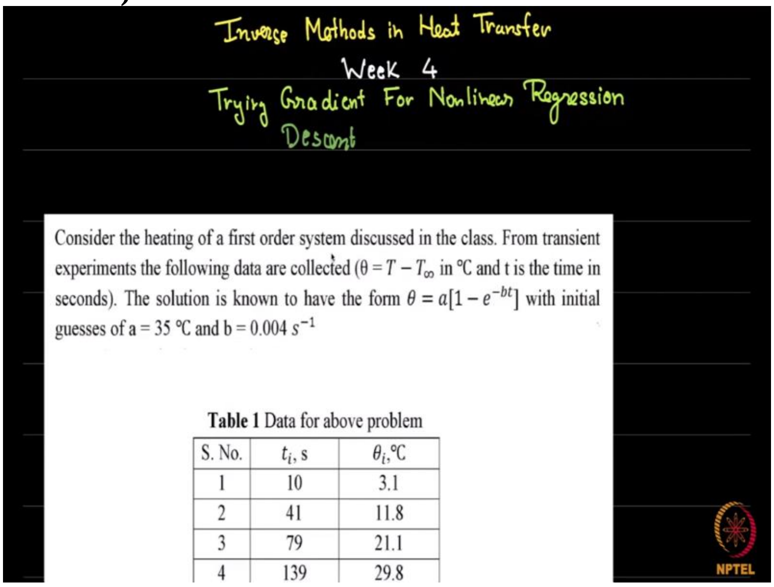
Inverse Method in Heat Transfer

Prof. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology – Madras

Lecture - 23
Gradient Descent for Nonlinear Inverse Problem Theory

(Refer Slide Time: 00:18)



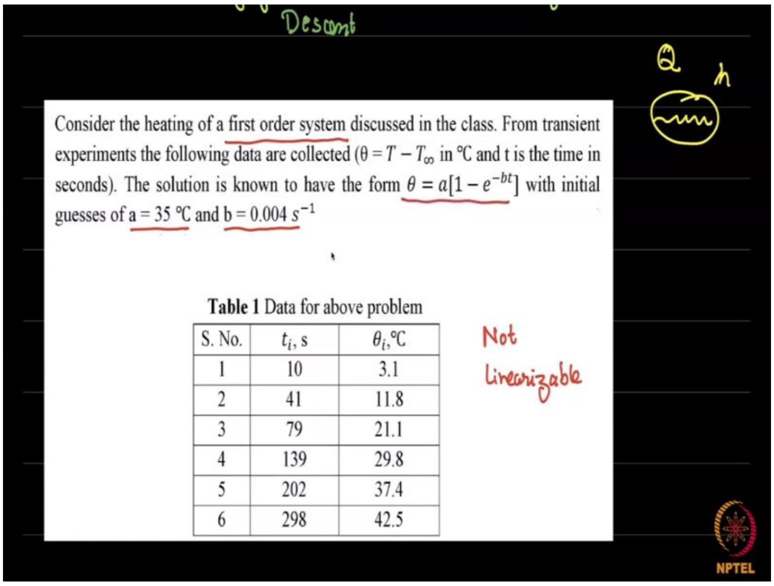
Consider the heating of a first order system discussed in the class. From transient experiments the following data are collected ($\theta = T - T_\infty$ in °C and t is the time in seconds). The solution is known to have the form $\theta = a[1 - e^{-bt}]$ with initial guesses of a = 35 °C and b = 0.004 s^{-1}

Table 1 Data for above problem

S. No.	t_i, s	θ_i,°C
1	10	3.1
2	41	11.8
3	79	21.1
4	139	29.8

Welcome back. This is week 4 of inverse method of heat transfer. In the last video, we looked at gradient descent and looked at a very simple example on how to do gradient descent, which was somewhat successful if we played with the value of the learning rate alpha. In this video, we would like to try gradient descent for I forgot to write descent here. So, we would like to try gradient descent for a nonlinear regression problem, which is our initial name. So I am going to introduce the data set here, we have a slightly modified data set from here with the new assignments for this week.

(Refer Slide Time: 01:05)



But let us just consider it and see whether we can figure out these constants n. So as usual, that is not as usual, but as we discussed earlier on this week, let us consider the heating of a first order system. So, first order of system remember was a system which had some amount of heat transfer Q, which was a heat generation Q, which was being added and there was also some convection h.

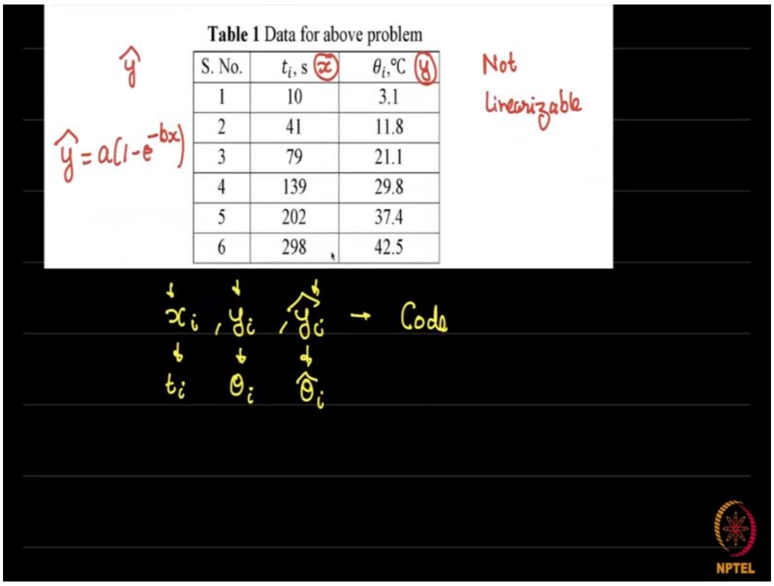
And finally, we saw that in such a case, the temperature distribution or the temperature excess distribution, in case outside temperatures, T_∞ looks to be in the form,

$$\theta = a(1 - e^{-bt})$$

Now, the problem with this was that, this you could not solve using linear regression, because this is not a linearizable form. so, this was not linearizable, and this is why we started looking for some methods that could generalize to non-linear equations also or non-linear optimization problems and 1 such example was gradient descent.

As I said, we will be using it again, when we come to the machine learning portions of this course. Now, let us say we have 2 initial guesses, we have already seen gradient descent and we have 2 initial guesses a = 35 degrees, and b is 0.004 per unit second, you can check what the units are by just trying to make dimensional analysis on this equation here. So, you can easily check dimensional analysis here. Now, here is the data set.

(Refer Slide Time: 02:37)



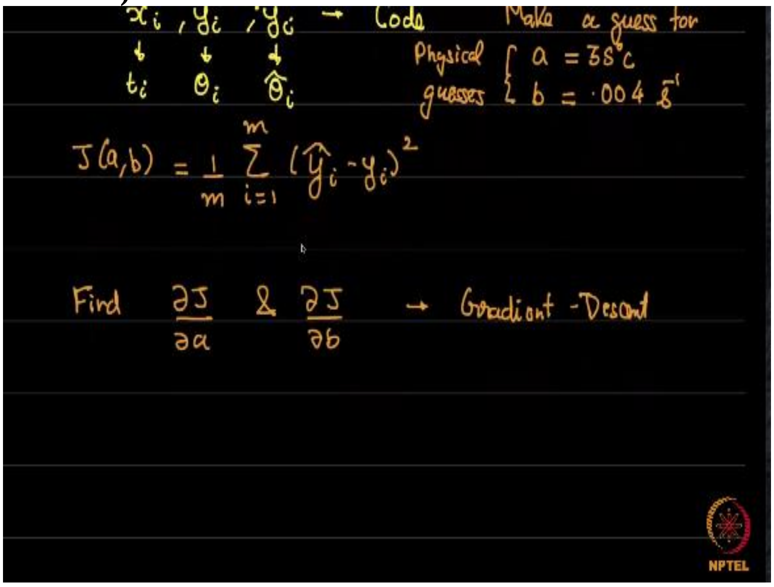
What I am going to do, of course, is after some time, I am going to refer to the first variable as x, and the second variable as y and of course we are we have our model $\hat{y}$ and $\hat{y}$ would look like it is $a(1-e^{-bt})$because it is just easy to code it once and for all. Now, word of warning for those of you who will be following me along here, even though I will use a code, it is it makes no sense unless it is a live class for me to actually show you a hand calculation on the board here.

I can do it a couple of times, but it quickly gets boring. It is best for you to however do it by hand, in case you are taking this course for credit, because you will have to use a calculator in the final exam, we will try to make the problem simpler but handle able by hand. But nonetheless, we would not have access to programming languages when we look at the final exam. So nonetheless, I will motivate the problem, then I will show you a code for gradient descent, which for those of you who are taking it for credit.

It should be available to you through the NPTEL or Swayam platform, whichever they are taking this course. So, in general, we have this data, I am still going to call it, x_i, y_i and $\widehat{y_i}$ and you can translate it in your mind to actually time, the temperature excess and you can also think of this as $\widehat{\theta_i}$. The reason I am using this is I would like to employ it in a code and when I write it in a code, it is generally a good idea for the entire course for us to use the same notation.

So that we remember that this is the input this is the output and this is the model, so let us start some data as given here. I will of course, put it explicitly within the program when we come to the program.

(Refer Slide Time: 04:37)



So, we have this data and our problem is this as usual, we have to specify some J. Now J is going to be a function of a and b now. I could call it w also but I will retain a and b because a and b are a little bit clearer as far as this example is concerned. So you have a summation, we make a guess for a and b as is given in the problem, so, we said a is 35 degrees this is the initial guess and b is 0.004, if I remember right 0.004 per unit second.

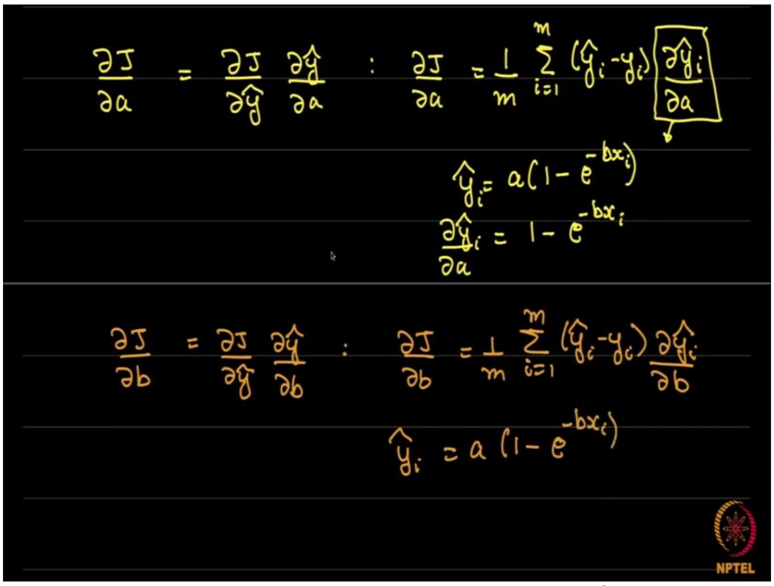
How did we get these values of course, we have some physical idea for what these mean and if you look at the original forward model for the problem, these come from some physical constraints. So, some of these are physical guess and this is once again where your knowledge of heat transfer will play into getting the right initial conditions for your model parameters. So, as usual we have the same model you can call it,

$$J(a,b) = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

$(y_i - \hat{y}_i)^2$or $(\hat{y}_i - y_i)^2$ which is the same thing.

And what we want to do is to find remember $\frac{\partial J}{\partial a}$ and $\frac{\partial J}{\partial b}$ in order to use for gradient descent. Even if you want to optimize it analytically, you still need $\frac{\partial J}{\partial a}$ and $\frac{\partial J}{\partial b}$.

(Refer Slide Time: 06:16)



Now, the way this will be done, so let us say I want to find out $\frac{\partial J}{\partial a}$ that is going to be,

$$\frac{\partial J}{\partial a} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a}$$

just like we did for linear regression. So, this will basically give us $\frac{\partial J}{\partial a}$ is,

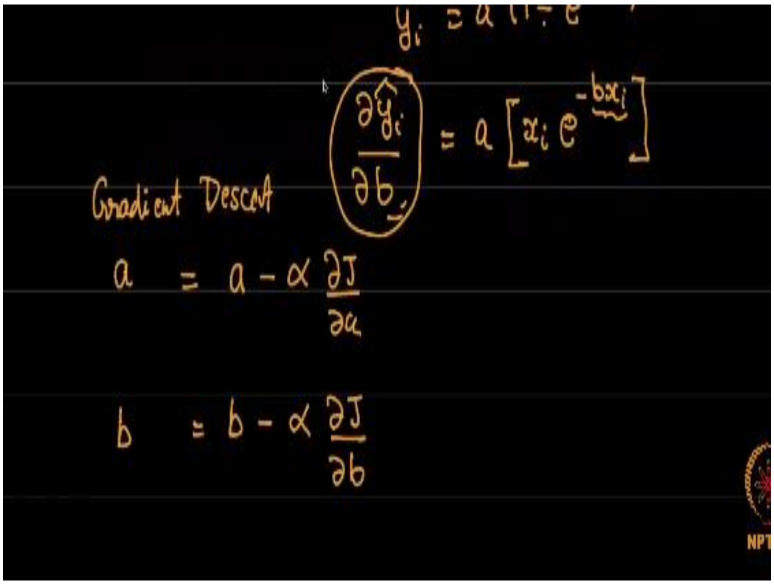
$$\frac{\partial J}{\partial a} = \frac{1}{m} (\hat{y}_i - y_i) \frac{\partial \hat{y}_i}{\partial a}$$

$\frac{1}{m} (\hat{y}_i - y_i)$. you can multiply you can look up your notes for linear regression also multiplied by $\frac{\partial \hat{y}_i}{\partial a}$. Now, what is this quantity remember, $\hat{y} = a(1 - e^{-bt})$ again, I am warning you I am using x whereas I actually mean t, but I am going to use x just for convenience of notation.

So, $\frac{\partial \hat{y}_i}{\partial a}$ is simply $1 - e^{-bx_i}$, in case this is $\hat{y}_i$, so we can substitute that here and it is easy to programming again, this is easier when you actually have to program, it is a little bit tougher when you have to evaluate it. So, as you can imagine, if you have to evaluate by hand, you will have to do a summation over the 6 points is you can write it in column, you can write $\partial \hat{y}$, you have ∂a in 1 column, this in another column, etcetera.

So, maybe during the summary for this course, at the end of the last week, I will write that down. So, the people taking it for credit can employ it a little bit more easily. Similarly, when you have $\frac{\partial J}{\partial b}$, that is $\frac{\partial J}{\partial \hat{y}}$, which remains the same $\frac{\partial \hat{y}}{\partial b}$ and once again you can calculate this expression as $\frac{1}{m} (\hat{y}_i - y_i) \frac{\partial \hat{y}_i}{\partial b}$ and now given that $\hat{y} = a(1 - e^{-bx_i})$.

(Refer Slide Time: 08:30)



$\frac{\partial \hat{y_i}}{\partial b}$ = a times now, this is a constant because we are taking a partial derivative with respect to b. So, this is just a derivative of this term which is e^{-bx_i} multiplied by you have to be careful, you are differentiating with respect to b. So, the derivative actually becomes $-x_i$ and the minus and minus cancel and you simply have x_i here. So, be please b is careful about this term here, it is often I see b is get confused here.

$$\frac{\partial \hat{y_i}}{\partial b} = a[x_i e^{-bx_i}]$$

They will put a b here instead of putting an x_i, we are differentiating with respect to b. So, just a slightly tricky thing that you have to negotiate because of what you are used to usually. So, once again, you can program it this way, $(\hat{y_i} - y_i)\frac{\partial \hat{y_i}}{\partial b}$. So, this is what is going to go into our program we are simply going to say that any update for a the gradient descent update is simply going to be whatever current value it has minus some learning rate multiplied by $\frac{\partial J}{\partial b}$.

Similarly, b is going to be whatever current value of b it is alpha multiplied by $\frac{\partial J}{\partial b}$.

$$b = b - \alpha\frac{\partial J}{\partial b}$$

$$a = a - \alpha\frac{\partial J}{\partial a}$$

Notice that $\frac{\partial J}{\partial a}$ and $\frac{\partial J}{\partial b}$ earlier to a bit more complicated than what we had in our previous examples, that is because they are summations over the data set, so this entire data set has to be summed up and that is how you get $\frac{\partial J}{\partial a}$ and $\frac{\partial J}{\partial b}$. So, I will just show you how to implement in a program and then we will see whether it works or not, so let us go ahead and try it now.