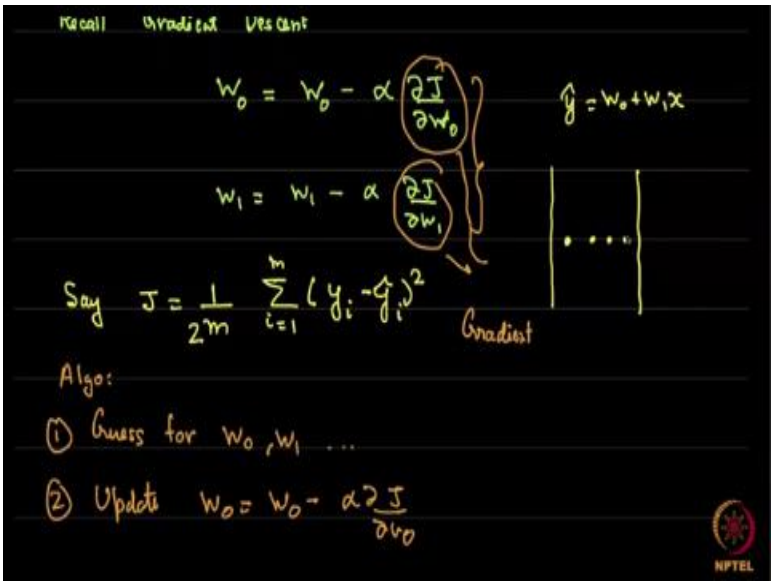
Inverse Methods in Heat Transfer
Prof. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology, Madras

Lecture - 47
Gradient Descent Batch, Stochastic and Mini Batch

(Refer Slide Time: 00:19)



Welcome back, this is week nine of inverse methods in heat transfer. In this video I will be talking about some variations of gradient descent. These are called batch gradient descent, mini batch gradient descent and stochastic gradient receptor. Now all these are particularly relevant for two reasons. So, if you recall gradient descent, so if you remember what gradient descent was. It was a simple idea that when you have a parameter W or a set of parameters W you basically update them as $w = w - \alpha \frac{\partial J}{\partial w}$.

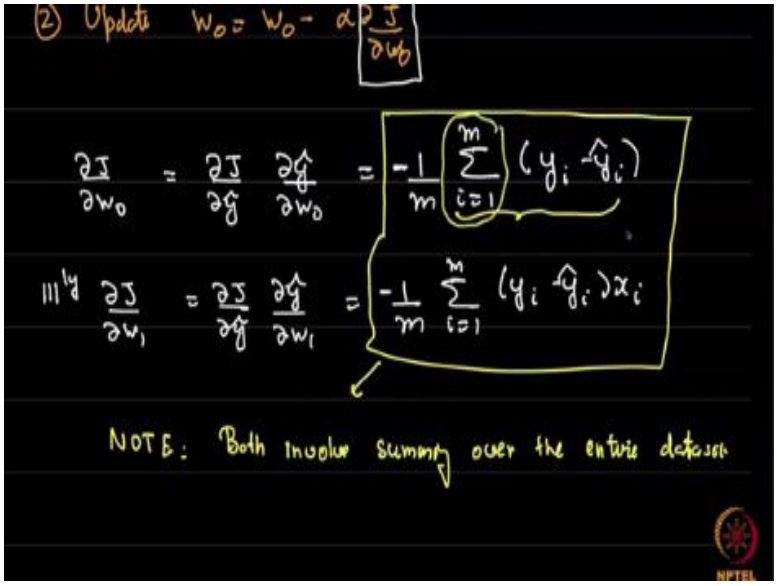
So, if you have two parameters in particular you would have something like this and $w_1 = w_1 - \alpha \frac{\partial J}{\partial w_1}$, we saw this early on in the course. So, for example let us say

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

So, for example our slab case suppose I am solving for these parameters w_0 and w_1 and $\hat{y} = w_0 + w_1 x$, then all I need to do is to minimize J and for minimizing J, I basically use these expressions.

So, we would first guess so the general algorithm for gradient descent was we would guess for w_0, w_1 etcetera and then update. But all this of course requires this expression the gradient expressions. So, we are going to call these the gradients; which is why we call this the gradient descent.

(Refer Slide Time: 02:30)



Now our key task here is to evaluate this $\frac{\partial J}{\partial w_0}$ and as you will see both this week as and next week this is a very important ingredient of any optimization algorithm. But what was $\frac{\partial J}{\partial w_0}$. Again, you might remember from our earlier classes that $\frac{\partial J}{\partial w_0}$ was calculated as $\frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_0}$ which came to $\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)$, I would request you to check out your earlier notes by $(y_i - \hat{y}_i)$.

Similarly, $\frac{\partial J}{\partial w_1}$ turned out to be $\frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_1}$ which was $\frac{-1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)\, x_i$, in fact we have calculated these several times $(y_i - \hat{y}_i)x_i$, there is a negative sign also in front. Now notice these terms, there is a special feature in these terms. Both involve summing over the entire data set. So, notice this entire data set the sigma that we have is over the entire data set.

Again, in earlier programs or in case you used an Excel sheet this would have become clearer what we would have done remember each one of these terms is the error. So, what this term says is w_0 is updated by the difference between $(y_i - \hat{y}_i)$ and the average error over the various examples

that you take. So, once again let us go back to our model, in case we are fitting a temperature model and the actual temperature is something else.

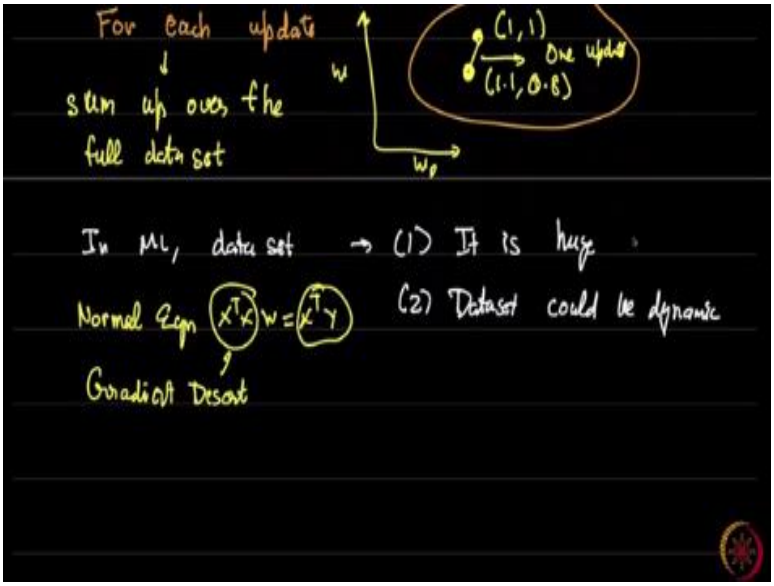
We see the difference between what we are predicting the model versus, what reality is that is what is written here $(\hat{y}_i - y_i)$ and sum can take an average. But what this requires is summing over the entire data set. For what? For each update what does that mean. So, remember I had given you a short example of gradient descent earlier also suppose you start in w_0, w_1 axis and you are starting with some initial guess.

Let us say this is the guess 1, 1 and you move to a different point and say this is the optimal point which is 1.1, let us say 0.8. This is one update; one update means w_0 has been updated you have made a new guess for w_0 and w_1 has been updated and you have made a new guess for w_1. But for one update we require to sum up over the full data set. Now this is fine when we are doing the sort of examples that we did in class with five or six thermocouples.

But as we will see in two weeks from now or even a little bit in the next week though I will not be showing any major examples in real machine learning. Usually, data set has problems first it is huge. So, in some of the image data sets you could have millions of images. So, in case you make a prediction you would actually have to do this summation over a million images before you make one small update.

Now we also saw earlier on that with iterative methods you might not converge in just one update. You might require like millions of updates. So, this becomes really expensive.

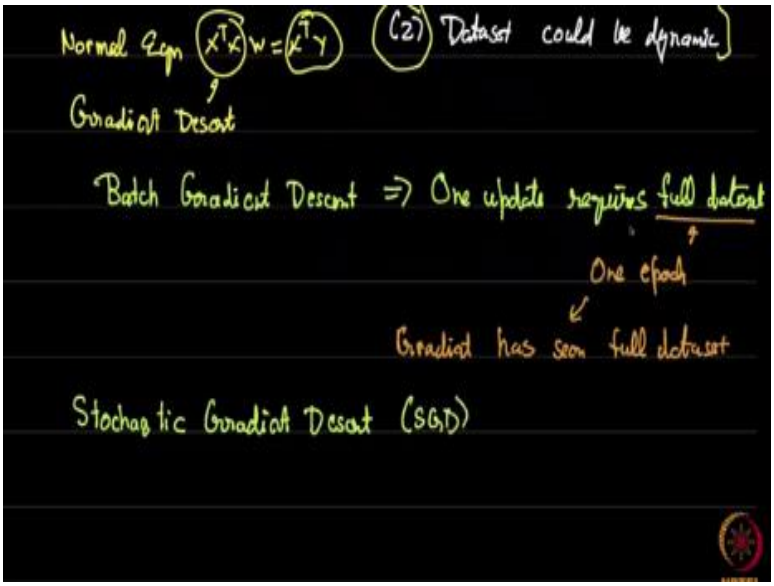
(Refer Slide Time: 07:05)



Second data set would be dynamic. That is so let us say you are in an industrial setup and you have instruments constantly making measurements. Now if you use the normal equation approach, normal equation approach has this problem remember our $X^T X \Delta W$ in our case I wrote $X^T XW = X^T Y$ now this requires matrix setup and the moment I give a new X you cannot update it you have to solve the entire problem again.

But even in gradient descent you have to do this entire sum again because the summation is from i = 1 to m.

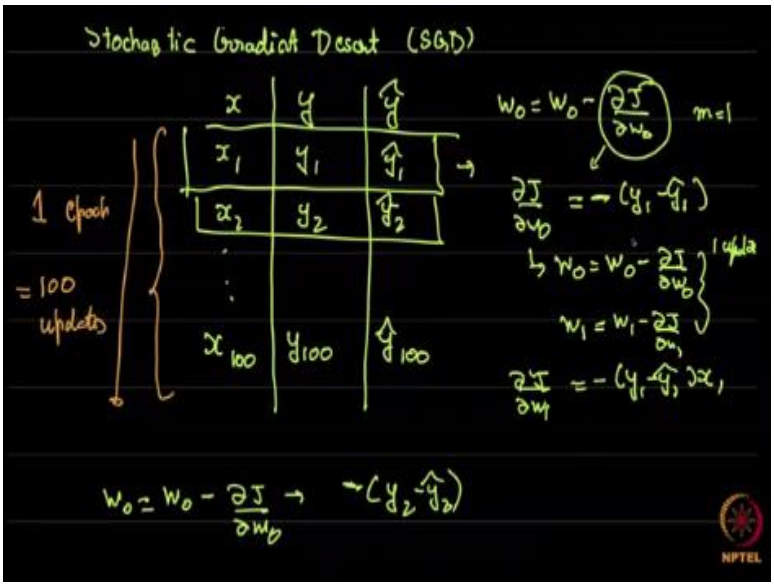
(Refer Slide Time: 08:05)



So, in both these cases it is somewhat disadvantages to use gradient descent. The gradient descent works kind of with dynamic data sets but it does not necessarily work well with huge data sets. So, we have a couple of other variants and these are very natural variants of what we do. So, what we have done is this version of gradient descent the vanilla version or the normal version of gradient descent is called batch gradient descent.

Batch gradient descent means that one update requires full data set. Another way to say it is this is called one epoch. One epoch means the update or the gradient has seen full data set. Now let us go to the other extreme. The other extreme is what is known as stochastic gradient descent SGD for short. Now how does this work?

(Refer Slide Time: 09:40)



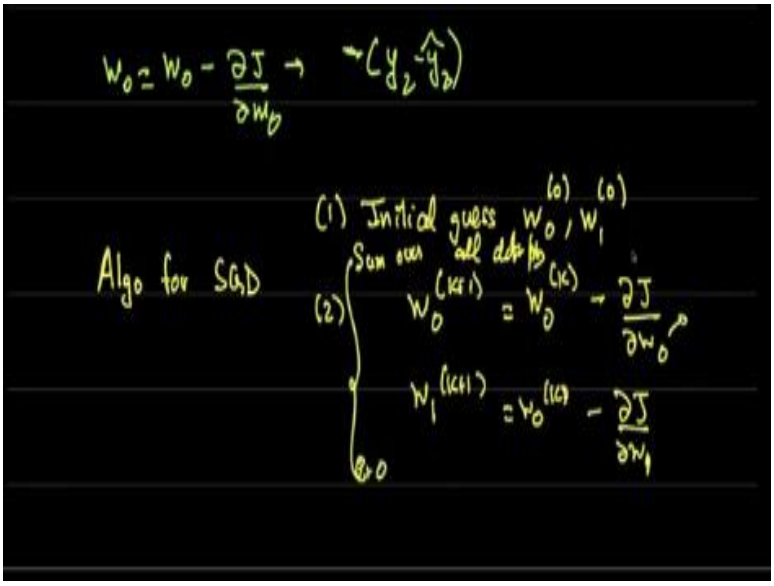
So, let us take our data set the usual x, y, $\hat{y}$ let us say we have x_1, y_1, $\widehat{y_1}$ up till let us say x_{100}, y_{100} and $\widehat{y_{100}}$. So, now when we do $w_0 = w_0 - \frac{\partial J}{\partial w_0}$, we take only one data point for this calculation. That is $\frac{\partial J}{\partial w_0}$ is simply calculated as $(y_1 - \widehat{y_1})$. So, notice here this was $(y_1 - \widehat{y_1})$, sorry negative of that always keep on making this mistake $-(y_1 - \widehat{y_1})$.. Remember you have to sum up over a certain set of data points and then do 1 by m.

And I am basically setting this m = 1. So, and then I will do $w_0 = w_0 - \frac{\partial J}{\partial w_0}$. Similarly, $w_1 = w_1 - \frac{\partial J}{\partial w_1}$ where $\frac{\partial J}{\partial w_1}$ is only calculated for the single data point $-(y_1 - \widehat{y_1})x_1$. Now how does that help

us. Now once I do this, I get one update. So, instead of getting stuck at this point, I actually move a little bit. maybe I will move in a different direction because this is an average of the entire data set this is just what one data point predicts but I am moving.

So, I am making shorter steps but I am at least moving more rapidly. by the time I see all data points I would have perhaps move further down. So, what I do is this the second step will be $w_0 = w_0 - \frac{\partial J}{\partial w_0}$ where this delta w_0 will be now calculated for the second data point. So, one more update so this will be $-(y_2 - \widehat{y_2})$ so, on and so forth.

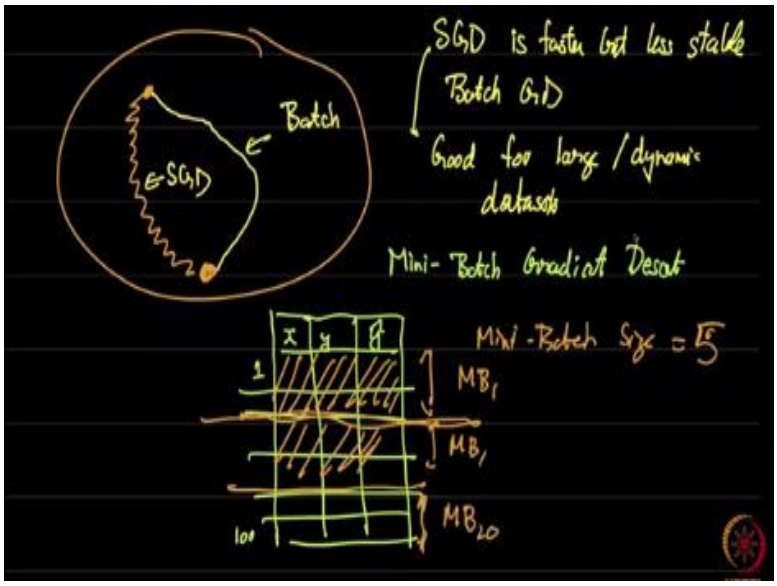
(Refer Slide Time: 12:05)



So, we can write the algorithm this way, the algorithm for stochastic gradient descent. First is initialize or take a guess for w_0 and w_1 or how many other parameters we have, we will call this the 0th iteration. And then we will say ${w_0}^{(k+1)} = {w_0}^{(k)} - \frac{\partial J}{\partial w_0}$. Now this one is calculated sum over all data points. So, similarly for w_1 and then you repeat this so, what does that mean?

It means that I update w_0 once here, w_1 once here, once again I update it here and by the time, I see the entire data set. So, remember I call that the epoch 100 updates. So, gradient descent updates for each data point separately.

(Refer Slide Time: 13:37)

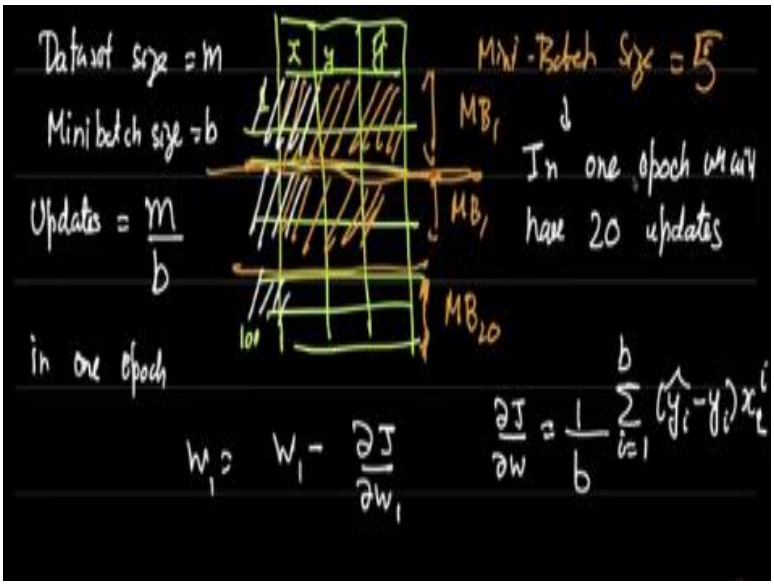


So, when you do that typically we notice that you will go a little bit like this to the final point. Whereas batch gradient descent would have gone a little bit more smoothly. so, this could be batch this could be SGD. But SGD is typically faster but less stable. It might go this just like a very active person, they tend to get things fast, but they will make a little bit more mistake. So, SGD is faster but it is a little less stable than batch gradient descent.

The advantage also with SGD is it is good for large and or dynamic data sets. I will show you a demonstration of SGD versus batch gradient descent in the next week, when we come to a full treatment of neural networks. Now there was a third thing which is kind of an obvious measurement or on a modification of the two which is called mini batch gradient descent. Now mini batch gradient descent works like this.

Suppose this is our block again of x our data set x, y, $\hat{y}$ now, instead of let us say again we have 100 points. Now instead of taking either completely hundred or completely one we can go somewhere in the middle. Let us say we take mini back size of let us say 10 or let me call it five. So, I take 20 sections yes so this is mini batch 1, mini batch 2 again I take five data points, data points 6 to 10, 11 to 15 so on and so forth till mini batch 20.

(Refer Slide Time: 16:05)



So, what do we do in this expression? In this expression which is $w = w - \frac{\partial J}{\partial w}$ whether it is w_0 or w_1 it does not matter the $\frac{\partial J}{\partial w}$ is calculated let us say this is w_1 is calculated as $\frac{1}{b}\sum_{i=1}^{b}(\hat{y}_i - y_i){x_1}^i$. So, what we will do is first we will take an average over the first five points, next over the next five points, third time over the third point third five points.

So, basically in one epoch we will have 20 updates. So, in general if the data size is or data set size is m and mini batch size is b, then we will have m by b in one epoch. Of course, you want to what is the significance of one epoch? one epoch is when you have seen the entire data set that is for example you have 100 thermocouple measurements, you have made sure that your parameters have seen all these hundred measurements and then only you are happy.

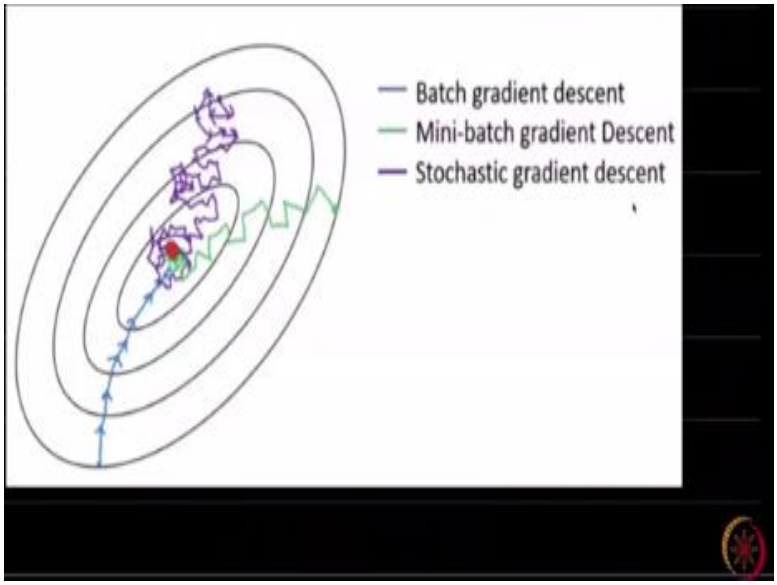
But nonetheless and that is also what is an apples-to-apples comparison. because if I compare batch gradient descent for each update, it is looking at all 100 points and summing up over it. If I look at stochastic gradient descent, it is only looking at it has a very quick calculation one data point. If you do it by hand, you will see you just have to find out one prediction, that you made versus one prediction which is of the ground proof.

But by the time I do 100 of those calculations I might actually move further than the batch gradient descent has, like I said you can think of these three people as some extremes. Batch gradient descent is a very careful person, who does not want to make a move or does not want to say where

I should go next before I have looked at all the data. Stochastic gradient descent is like a very judgmental person it is like one point of data it will move immediately.

Whereas mini batch is somewhere in the middle it is like I am not satisfied with just knowing one thing I will see at least a few points. Then I will make a move and then I will see the next data point and then I will make a move. So, this person looks at partial amount of data but is at least a little bit action-oriented SGD is fully action oriented that is like fully thought oriented.

(Refer Slide Time: 19:03)



So, these kinds of differences often shown between batch gradient descent, mini batch gradient descent and stochastic gradient descent. While reaching the optimum let us say the red point is the optimum you want to reach. Batch gradient descent will reach that but it will take a lot of computation, but it will read smoothly. Stochastic will go all over the place and mini batches somewhere in the middle.

Often, we either use in practice like I said if you have data sets which are coming very rarely or very dynamically something like mini batch or stochastic gradient this and it makes sense. If you have a small data set you want some smoothness, so, batch-gradient descent makes sense there. So, now what we will see starting from the next video is how these ideas gradient descent etcetera can be applied to the idea of classification.

Specifically, we look at two algorithms the logistic regression algorithm and the multi-class classification algorithm. So, I will see you in the next video, thank you.