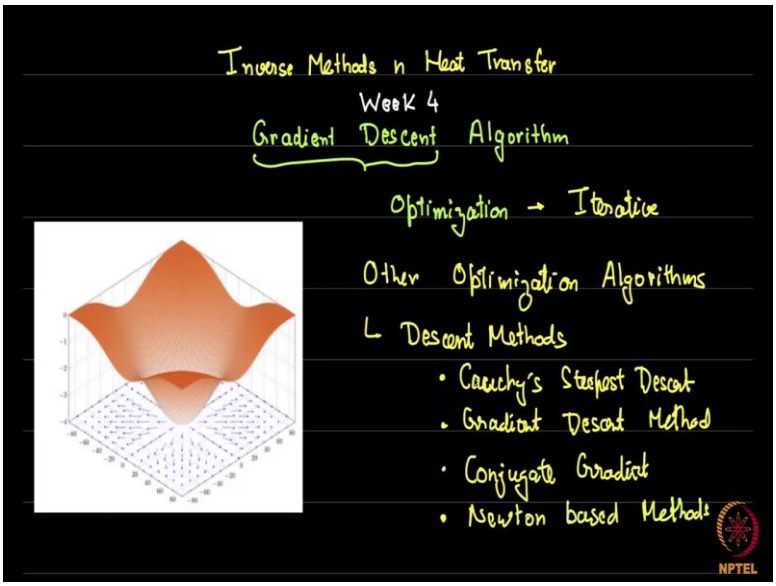
Inverse Methods in Heat Transfer
Prof. Balaji Srinivasan
Department of Mechanical Engineering
Indian Institute of Technology – Madras

Lecture – 21
Gradient Descent Algorithm

(Refer Slide Time: 00:19)



Welcome back. This is week 4 of inverse methods in heat transfer. We are going to discuss an important algorithm. This is not directly in inverse makers that it is an important algorithm, but it seems to be a very important algorithm currently within machine learning. So, this algorithm here gradient descent is an optimization algorithm. It is a very simple optimization algorithm actually, far simpler than the normal equations method that I showed you which I said was a direct method.

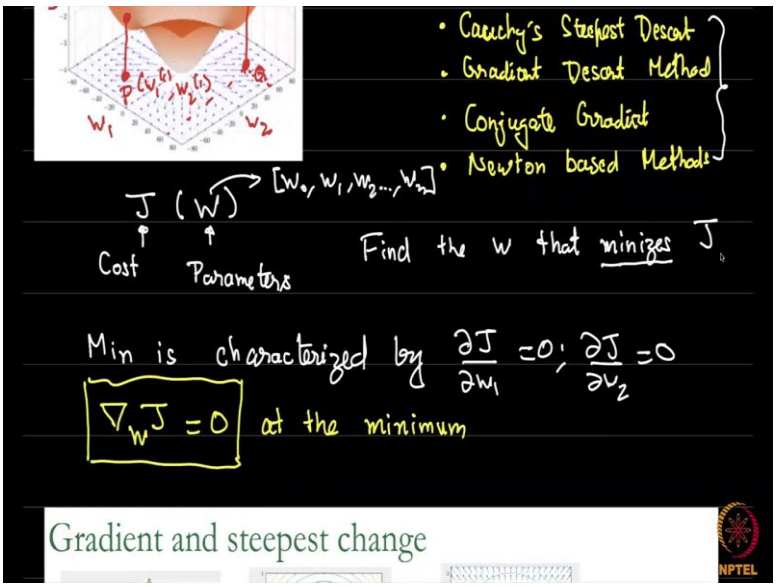
But this is what is known as an iterative method. That is, you make some guesses for the parameters you are optimizing for and you keep on improving them as I had said this is a standard process within any inverse problem within machine learning and without machine learning else. So, the idea here is to give you an introduction to a very simple algorithm and we will also go for a more complex algorithm later on in this week, what is known as a Gauss Newton algorithm.

So, this is the first introduction, even though you will not find gradient descent within inverse methods books. mostly you will not find them you will find other algorithms. So, there are

other optimization algorithms within inverse methods or in general in optimization methods. So, these are part of what is known as descent methods. So, you can see the word gradient descent. So, gradient descent is a specific class of a descent method.

So, there is something called Cauchy steepest descent method. of course, there is gradient descent and all these are gradient based methods. And there is something called conjugate gradient this is not strictly speaking a descent method but that is so, conjugate gradient. Then there are Newton methods one of those which we will. Now what is the idea? the idea is this you have some function.

(Refer Slide Time: 02:38)



So, you have some function J which is a function of w. So, this is the cost or the objective and this is the parameter or set of parameters, because w itself could be made up of w_0, w_1, w_2 lot of things till wn as you saw already. So, there are a bunch of parameters and what you want to do is to find out find the w that minimizes J. Now you might ask optimization means I could maximize also, of course all the problems we are doing are trying to minimize the gap between reality and our model.

So, typically we use minimization but as it turns out if instead of minimizing J, you have a maximum problem you just maximize minus or minimize minus of J and that also is a basically any maximization problem by taking a negative can be turned into a minimization problem but that is an optimization course. Let us come back to our purposes, where we are typically always minimizing and that are where we use all these algorithms.

There are multiple other algorithms which are not based on the gradient. In the last video I showed you what a gradient means, at least I reviewed what a gradient means and as we discussed what a gradient means is suppose I have this parameter space and I am looking at x y for let us say w_1, w_2 let me correct this a little bit. So, suppose I am looking at w_1 and w_2, 2 parameters and let us say this is J.

So, at this value let us call this some something let us say ${w_1}^{(1)}$ and ${w_2}^{(1)}$ at this point p, let us say we are moving up and trying to find out the value here. So, the z axis here gives you J at 1. Similarly at another Point Q this gives you let us say J at Q or J at the second Point at this is a different value of w_1 and a different value of w_2. Now let us say we are somewhere here at this point you can imagine that you are on this hill which is represented by this complex surface.

You are on this hill and the lights are off and the only thing you know is your current position. But somehow you want to make your way up to the bottom of the hill which is where boom is but all you can look at is maybe you keep your foot out and sort of measure you know what is the slope around me. And the general idea of a decent algorithm is very simple, see we are looking at this whole surface but the computer does not know it only knows the value of the loss function at the particular point.

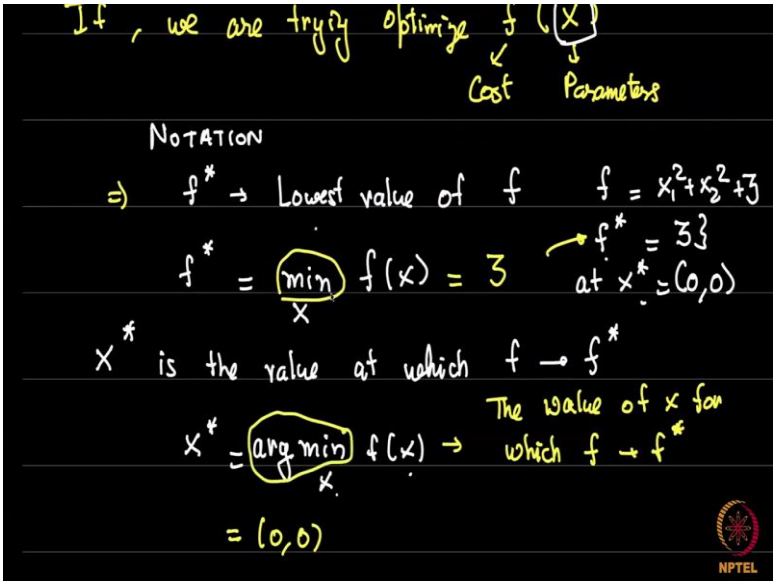
Given this w_1 which is say minus 40 and this w_2. which is minus 60 what is J and it will figure out what the J is and it will say it is minus 2 but it does not know whether this is good enough or not unless it looks here. What is the final goal? the final goal for all these is to reach at a place which is the bottom and what is this bottom characterized by the minimum, is characterized by the fact that wherever you look around the minimum.

So, if I am at the minimum here what you will notice is it is flat any direction you move you will always move up. So, if you are here at the bottom and you keep your feet out it will be relatively flat here and that is the intuition for the physical intuition behind saying $\frac{\partial J}{\partial w_1}$ is 0, $\frac{\partial J}{\partial w_2}$ is 0. And of course, minimum also has certain implications on the second derivative but those are more complex.

So, we will leave that out for at least for any optimum you automatically know these 2 should be true and another way of writing it based on our discussion of gradient is to say gradient of J

with respect to w is a 0 at the minimum. So, this one condition is what we will try to satisfy this is the basic idea. Now we are going to use another intuition here which we also discussed in the previous video and let us go and do that and start looking at what else information the gradient has.

(Refer Slide Time: 07:27)



Now you might remember from again from your college days that if you are given any function f(x), I am going to again use f(x) to recollect what I did in college in the first couple of years, that the direction of the maximum rate of change is given by the direction of I have written given by this but given by the direction of the gradient. Now you might not remember this if you do not that is fine, I am just going to prove it.

So, this is a claim. Let us explain this claim first and then see what the proof is a very quick proof once again go back to the hint. So, this is sort of an inverted hill shape you are traveling from somewhere and you are coming to the bottom. Now let us say once again you are at some point and you do not know you really do not know which direction you should move in, you just have the value at that point more specifically we can look at this figure where this is at least a little bit more complex.

So, you are at this point you want to know which direction you have to move in and you have really nothing other than the position here you have your feet and you're feeling around. And each direction you put your feet, you know that in some directions you are moving up in some directions, you are moving kind of down and some directions, you are moving really down. So,

the heuristic here is I will always move in the direction where locally I am going to move rapidly down.

Now it is possible that you might be fooled that the surface might go up a little bit later but we do not care about that. This is the heuristic with which we are working heuristic means a rough idea. So, the idea is this optimization algorithm is decided by the direction of the steepest change wherever I am feeling the maximum change, I will move in that direction. Why is this important I will show it to you uh shortly but let us say that we are only trying to find out at any given point which direction should I move in.

So, that I change the most rapidly. So, the claim here is move parallel to the direction of the gradient. If you move along the gradient, you will either go up really fast and if you go exactly to the opposite of the gradient you will go rapidly negative. why is this? Because in the last video I showed you this that the rate of change in a given Direction is given by $\frac{\partial f}{\partial v}$, remember we had looked at partial derivatives.

So, if you have the partial derivative in Direction one that is given by $\frac{\partial f}{\partial x_1}$ partial derivative in direction 2 it is $\frac{\partial f}{\partial x_2}$ if you want how much change does the function face in any direction in this direction, in this direction let us say this is the vector V. What we are saying is, is the maximum rate of change is given along the gradient and just any particular V is given by $\frac{\partial f}{\partial v}$ just like $\frac{\partial f}{\partial x}$ is along x, $\frac{\partial f}{\partial y}$ is along y, $\frac{\partial f}{\partial v}$ is along the direction v.

And how is that given this 2 we had seen the last time we take the ∇f and dot it with the direction v cap once again same example if I take $\nabla f . \hat{e_1}$ this gives me $\frac{\partial f}{\partial x_1}$ because ∇f is nothing but,

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix}$$

This is the way we represent it. So, from here you can get this so. Now look at this expression

$$\frac{\partial f}{\partial v} = \nabla f . \vec{v}$$

Now ∇f is a vector as you know gradient is a vector this has a magnitude and a direction.

So, let us say the magnitude of ∇f is,

$$|\nabla f| = |G|$$

So, if you take the dot product of 2 vectors magnitude of the first Vector multiplied by magnitude of the second vector multiplied by cosine Theta.

$$|G||V| \cos\theta$$

So, as I have written here um on the left-hand side G is grad f Theta is the angle between the gradient and V.

So, suppose you decide at this point to go in this direction V and the gradient has this direction and this is θ, then the amount of change that you will face in v is gradient multiplied by this magnitude multiplied by $\cos\theta$ that is right we have just shown. Now we want to maximize this or minimize this. we want to maximize if we want to climb up really fast, we want to minimize it we want to climb down really fast.

Now this is simple if grad G is modulus and V are fixed, V is a unit Vector. So, this is just one this is the ∇f, we cannot change that, but $\cos\theta$ we can manipulate by changing the direction of V then all we need to do is ensure that $\cos\theta$ is minus 1. So, that it is minimum. So, maximum increases along gradient and maximum decreases opposite to the gradient. So, very simple idea I will come to the last line a little bit later but maximum decrease at a point is along minus ∇f.

So, for example if f is,

$$f(x_1, x_2) = {x_1}^2 + {x_2}^2$$

then ∇f is you can now see,

$$\nabla f = x_1\hat{e_1} + x_2\hat{e_2}$$

So, let us say at some (x_1, x_2). So, let us say at (x_1, x_2) equal to (3, 4), this means ∇f will be,

$$\nabla f = (3\hat{e_1} + 4\hat{e_2})$$

or you can call it the vector (3, 4). So, maximum decrease now maximum decrease locally will be along (minus 3 minus 4). So, for example let me show you here or let us see this one.

So, let us say you are at the point (3, 4) is somewhere here, this is not showing here on this graph but let us say this is this point (3, 4) and you want to find out in which direction do I

move the fastest, it will be here, it will be (minus 3 minus 4) something of that sort. So, for example let me take another point if I take the (0.1, 0.5) then along (minus 1, minus 0.5) which will be exactly perpendicular to these contours here.

Now one other thing it is very common to express these contours and it is important for you to understand the meaning of these contours. Now contours are lines of constant function constant f. For example, since this is the function $x_1^2 + x_2^2$everywhere on a circle whether it is of radius 1 Radius 2 radius 3 or the function is going to have a constant value. So, let us see it here, where it is a little bit more obvious if you come here this was also created using $x_1^2 + x_2^2$ are actually something like yeah it was created using $x_1^2 + x_2^2$.

Now you can see that if I come at a particular point let us say here. So, if I come here and if I move a little bit up, I get a value. Now if I look at a circle around this value I am going to be at the same height. For example, if I keep on moving along the circle, I will be stuck here itself all the while this has some important consequences. The consequences are that if you have any surface at all.

And you start looking at these lines or these paths where the function is a constant. you can sort of think of collapsing, imagine that each of these circles that I was drawing here is sort of made up of a spring and you collapse that, then you see this figure. This is basically what is known as the contour plot. The contour plot is just a 2-dimensional representation of this 3-dimensional plot which is a surface plot.

Now how did we reduce this 3D to a 2D we reduced it by giving different colors as you can notice. So, you can see that as it gets bluer, this is a typical Convention, as it gets Bluer it means you are going to lower and lower heights as it gets a rendered it is going to higher and higher height. So, yellow is slightly higher compared to blue and you will see all this gradation from yellow orange green right till blue.

So, this is typically what you will see you will see red data really high the big height. So, here what you can imagine is each of these blue lines is at the bottom and I would like you to physically imagine pulling up uh you just imagine that you are looking at it top down and there are all these lines at each height which is what you represent as a contour here. So, typically

when I say that we are going to lower and lower values along this, it is an easy way to visualize that actually you are coming down the hill.

So, it is you will have to project a little bit of imagination using your color, but that is for human beings that is actually something that is possible. So, we try to represent from a 3D figure to a 2D figure. So, the point here is, you start at some arbitrary point and somehow you should move in a direction which is always optimal that is not possible but with gradient descent which is what I am going to describe right now.

We are going to use this simple idea that instead of ensuring an always use along the best path, but at least local best path we will move. so now let us say we are trying to optimize. once again, I will go back to the original question, if we are trying to optimize the function f and f is a function of x and x are the parameters. Now I will rewrite it in our usual form. So, this is our cost function and x are the parameters then we use a little bit of notation which I will write down now.

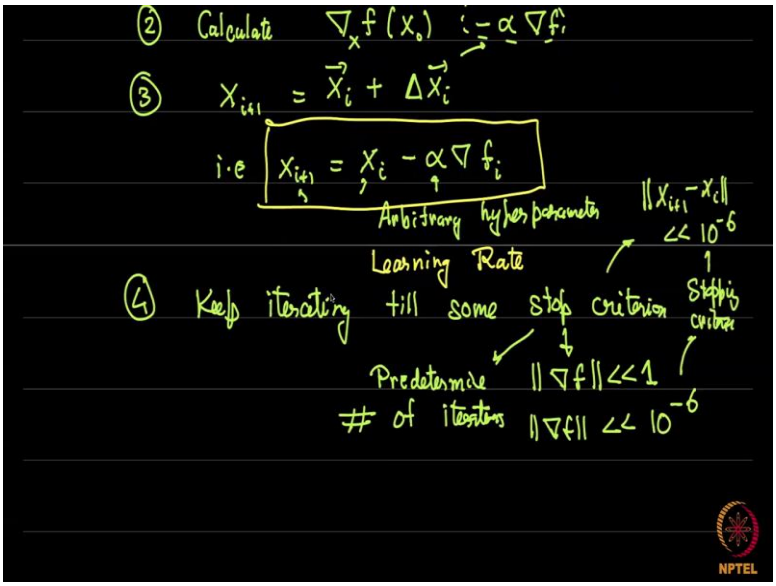
So, let us say f^* is the lowest possible value of f that is I keep on changing x. So, that somehow, I come to the lowest point in this figure. let us say f^* would be the value right at the bottom here that would be f^*. So, we would say something like f^* is minimum of f over all possible values of x. So, that is the way it is written just like with limit we have a notation. but where does it reach a minimum?

So, x^* is the value at which f goes to f^*. So, for example if I look at the function f equal to ${x_1}^2 + {x_2}^2$ we know that if star is 0 at x^* being x_1 equal to 0 and x_2 equal to 0. So, x^* is (0, 0) and if star is also 0. So, suppose I made this function as ${x_1}^2 + {x_2}^2 + 3$ you can easily see that if star is 3 and x^* is still (0, 0) but can we call x^* as minimum over f(x) we cannot do that.

The reason we cannot do that is minimum actually is 3 the minimum possible value of f over x is actually 3. So, we have a different notation we call this argument. What that means is the value of x for which f reaches f^*. So, for example here f star is 3 in this example and x^* is 0, 0 for Which f reaches f reaches the value 3. So, please remember this because sometimes I will use this notation min versus R min.

Min is the value and R min is the argument or the parameter value at which that minimum is reached.

(Refer Slide Time: 21:07)



Having said that let us now come back to the gradient descent algorithm. The idea is very simple. first step you take a random guess for the parameter. So, in this case our parameter was x, typically we will call our parameter w within inverse methods but for. Now let us human mean let us say this is x. So, x is typically a vector, it is going to have multiple components just like our w would have a whole bunch of components. Now calculate at this step you calculate the gradient.

Gradient of that function with respect to x at this step, at this value of x let us call this x, x0. meaning at the very first case. So, now the third step is this, improve and you say my new x that is I am going to move my point and I am going to look somewhere within the neighborhoods of where I am in the hill and I will say this is x plus some Δx. Now this Δx is a vector as you can see it is going to have both the position as well as a magnitude it is going to have a direction as well as a magnitude.

And this Δx we know should be parallel to $-\nabla f$. So, we will say this is going to be,

$$\Delta \overrightarrow{X_i} = -\alpha \nabla f$$

Another way of writing it is,

$$X_{i+1} = X_i - \alpha \nabla f_i$$

very simple algorithm what is alpha? α is an arbitrary I should call it parameter but we call it hyper parameter because x are the parameters which we are solving for is called a hyper parameter.

This is also called in machine learning; this is called the learning rate and why it is called the learning rate is something we will come to later on when we come to the machine learning chapter. For now, just assume it is a constant and we arbitrarily decided, how much we keep α decides, how large the steps we take are again going back to this picture here. let me go to this picture let us say you are trying to move from some point to the minimum.

Now let us say the gradient is here negative, I will show you this with an example. let us say the gradient is negative here let me take a large step you might land up here. If you take a very small step you will slow only go towards the minimum if you take a large step we might land up at really bad places, you can keep on jumping just because you are looking at local gradients but the local gradients you are taking a large step again imagine you are walking down on a hill, you don't know where the hill is going to go back up.

So, you might take a small step locally where you know it is sloping down but maybe a little bit later, it is going to slow back up again. So, because of that you have to adjust the alpha a little bit and we will come to more details about this when we come to machine learning. As of Now I just want to show you an iterative simple iterative algorithm. So, that is it, you keep on improving this keep iterating, either you use some stop till some stop criterion.

So, what is the stop criterion? it is like when do you stop iterating. So, stop Criterion could be you can predetermine the number of iterations. So, for example you will say I will take a thousand steps and wherever I am at the minimum another more satisfactory one is we know where ∇f remember that Norm which I had told you about earlier you take ∇f is very small. So, let us say

$$\|\nabla f\| \ll 10^{-6}$$

So, you know that the gradient has become very small, you know that at minimum gradient is going to be 0. but we cannot always reach perfect 0. So, you can decide on stopping there.

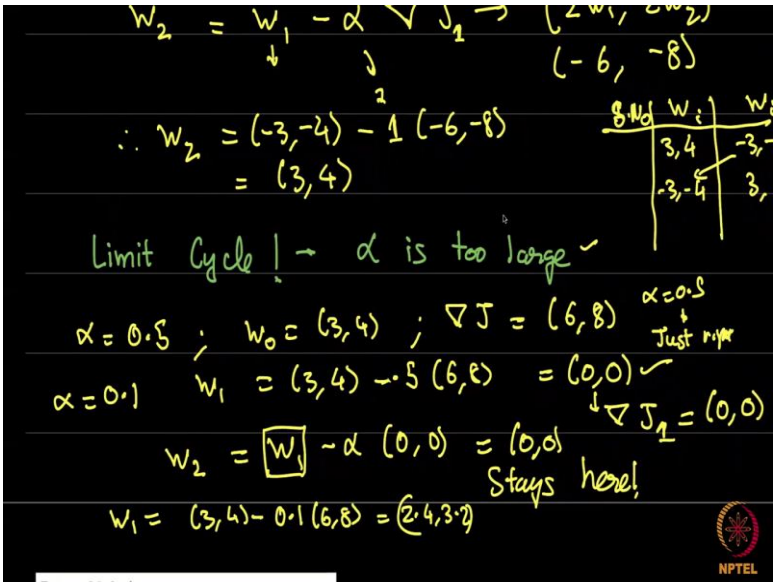
another possible way of stopping is to say that the difference between the current step and the previous step that is the update that you are giving to your parameter size is very small,

$$\|X_{i+1} - X_i\| \ll 10^{-6}$$

whatever these are arbitrary some stopping criteria.

So, these are called tolerances etcetera. we will come to this later on when we come to the gross Newton portion of this week's class. So, this is the gradient descent algorithm.

(Refer Slide Time: 26:41)



Let me show you a very quick demo with a simple example, where you will see the effect of alpha. So, let us take a simple case. Now I am going to switch notations here just for your convenience just. So, that you keep on thinking J and w instead of f and x. So, let us say J of w itself is made up of 2 variables or 2 features,

$$J(w) = w_1^2 + w_2^2 + 3$$

We already know the theoretical minimum for this as I said we should somehow converge to (0, 0) we already know theoretically.

And we could have done this using any of our theoretical techniques we need w_1^* is 0 and w_2^* is 0. So, we know that now instead of that for convenience we could also make this plus 3 just like I did a little bit earlier. Now what we are going to do is give an initial guess following our algorithm initial guesses we don't know where the minimum is I am just going to say it is 3, 4.

Now I want the next step I am starting at the point (3, 4), I am somehow having to go to some other point. So, the way to do it is to use the gradient update steps. So, first we find out $\frac{\partial J}{\partial w_1}$ which using this expression is 2 w_1 and of course find out $\frac{\partial J}{\partial w_2}$ it is $2w_2$. So, at the initial step or at the initial condition we know that grad of J with respect to w is going to be 2 w_1 which is 6, 2 w_2 which is going to be 8.

So, let us say α is 1. This gives us x or w, in this case w is w minus. So, this is of course a computational notation a coding notation, you can also say,

$$w_{i+1} = w_i - \alpha \nabla J_i$$

So, w_1 is our original value was (3, 4) minus α is 1 and this is (6, 8). So, this is w_1 the new value is (minus 3, minus 4). Now you can go one step ahead and find out what w_2 is and w_2 would be,

$$w_2 = w_1 - \alpha \nabla J_1$$

and what is ∇J_1 this is 2 w_1, 2 w_2 which is (minus 6, minus 8).

So,

$$w_2 = (-3, -4) - 1(6, -8)$$

So, if you calculate this, this comes back to 3, 4. So, what happened was very simple you started with our initial guess. So, serial number or iteration number then you have w and then you have w_i and you have w_{i+1}. So, we started with (3, 4) ended up at (minus 3 minus 4) then you started with (minus 3, minus 4) and you ended up (3, 4) and guess what you will keep on cycling.

So, this is an example of what is known as a limit cycle. So, this limit cycle happens because Alpha is too large. we can show it in a figure here. So, if you start here let us say you start at some point, you start here and you take a large step and somehow magically you landed up exactly on the same contour but at the opposite side. So, then you see the value in fact it will be the same and then you look at a local step I am supposed to move in this direction but instead of taking a small step you take a large step again and you go back here.

So, you are just oscillating between this and this. this is an example of taking a bad value of alpha. Now what happens if you take a smaller value, I will just show it to you quickly. So, if we take a smaller value of alpha say Alpha equal to 0.5, once again let us say w_0 is (3, 4). Now

grad J is once again (6, 8), but w_1 is (3, 4) minus halves of (6, 8) which is exactly (0, 0). Now this is good. Now what is grad J here? so grad J at the first values. Now again 2 w_1, 2 w_2, so, it is 0, 0.

So, w_2 is,

$$w_2 = w_1 - 0.5(0,0)$$

So, this is just 0. So, it stays here which is very good. So, α equal to 0.5 turns out to be ideal in one step you reach the actual minimum. remember the minimum of the function is at 0, 0. So, the reason we reach there is somehow we went here, we took exactly the right step we were somewhere here. So, we come somewhere here and we take a step which directly puts us to 0, 0 and of course here it is flat.

So, any place we keep our feet around and we see it is flat. So, we are just stuck here, which is perfect because it is the minimum it can happen that you can get stuck at the maximum also according to the formula, but luckily in this case we know it is a minimum. So, we are just stuck here. So, this is an advantage of the gradient descent algorithm. Once you come to the minimum or once you come close to the minimum you will always be there.

Coming back to the algorithm coming back to the algorithm, we saw that when Alpha is high, when Alpha is something like 1, it can actually get stuck, when α is 0.5 it is just right this typically never happens. but typically, we will take a small α. So, something like α equal to 0.1. So, if you take α equal to 0.1 you will move a little bit more slowly. So, you would do something like w1 is equal to (3, 4) minus 0.1 into (6, 8).

So, that would be something like (2.4, 3.2). So, it will move slowly towards the minimum in the next video I will show you another example since this video is a little bit too long already, I will show you another example with a slightly different function and I will show you the code also for this function. So, that you can see how this can be done obviously it is tedious to do by hand in the exam or something we might give you something like a couple of iterations.

And we have given you some such examples within this week's exercise also. but apart from that within the general practical realm we program this. So, I will show you a quick program

for programming this kind of gradient descent and how you can progress through various steps and reach convergence. So, I will see you in the next video, thank you.