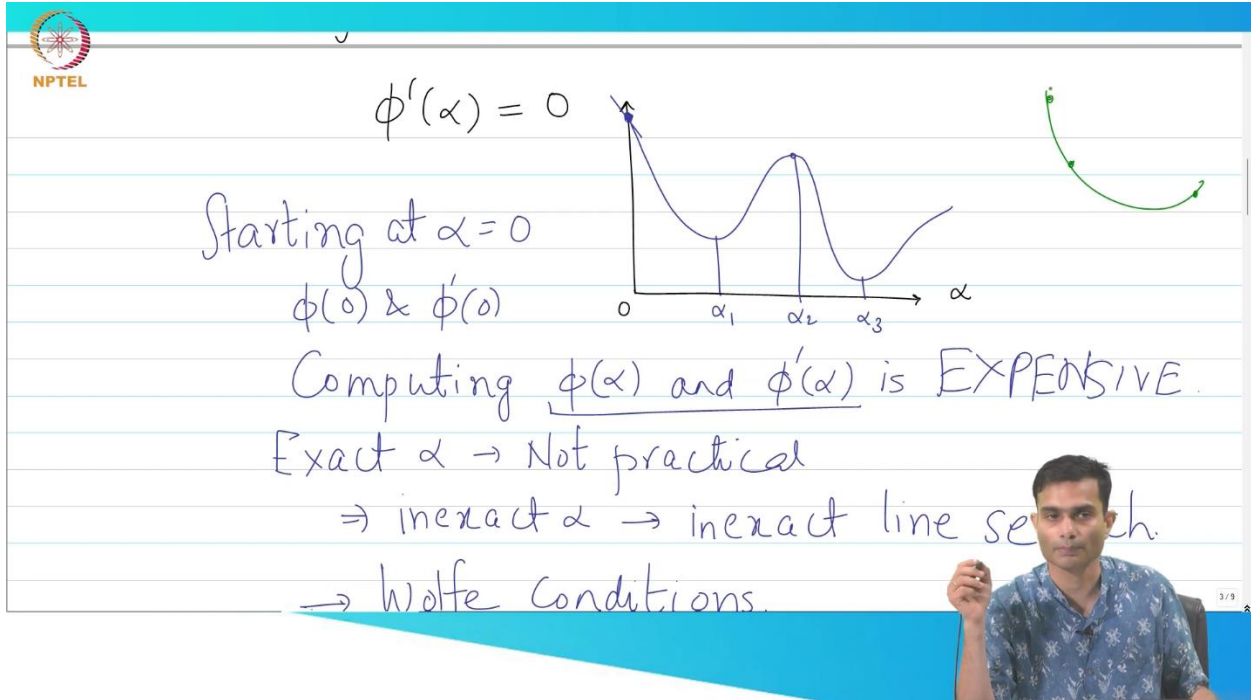


### Line Search - Analysis

We will start nevertheless. So I am going to read out some of the doubt sheet questions. There was one question about the Armijo rule. So the question is, can you give a better intuition for having  $C_1$  in the Armijo's rule? So the Armijo rule is, you can see it in front of you right now. What did it say? That the function value should be below the linear approximation of the function, but not exactly the linear approximation of the function; we modified the slope to be this orange line over here. I put  $C_1$  to reduce the slope a little bit from the original slope of the linear approximation.



$\phi'(\alpha) = 0$

Starting at  $\alpha = 0$   
 $\phi(0)$  &  $\phi'(0)$

Computing  $\phi(\alpha)$  and  $\phi'(\alpha)$  is EXPENSIVE.  
Exact  $\alpha \rightarrow$  Not practical  
 $\Rightarrow$  inexact  $\alpha \rightarrow$  inexact line search.  
 $\rightarrow$  Wolfe Conditions.

NPTEL

3/9

The intuition behind it is, as I mentioned in the previous class, that I can have a little bit more relaxed criteria. The most relaxed criteria would be  $C_1 = 0$ . That means the function value should just be less than the value of the function at  $x_k$ ; that is really, I mean, that is anyway going to happen because of sufficient decrease and it being a descent direction. So, we are trying to — this is like a tuning knob you have in your hand; there is no theoretical guarantee or proof that it should be some value or the other. This is something that you will locally use to fine-tune your rate of convergence.

So, there is nothing very deep about it. Choosing  $\alpha$ , choosing the perfect  $\alpha$  implies  $\phi'(\alpha) = 0$ ; this results in taking a step that blows past our minima, and then we take turn around to take a small step. Well, if you chose your perfect  $\alpha$ , you would land up at the correct solution, right? So

you will not go past. But this is always an issue that if I choose too big of a step size, I can go past my solution, in which case I need to come back.

line search - analysis

Recap ->

① Backtracking line search

Repeat until  $\left[ f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla f_k^T p_k \right]$

$\alpha \leftarrow \alpha \rho$  ,  $\rho \in (0, 1)$

②  $\phi(\alpha) \rightarrow \phi(\alpha) \leftarrow$  calculate pick any  $\alpha$

$\left. \begin{matrix} \phi(0) \\ \phi'(0) \end{matrix} \right\}$  fit a quadratic & minimize

at  $\alpha=0$

So the backtracking line search algorithm which I gave you works on that principle. I start with a large step size and start walking back until some of the conditions are satisfied, and I say, ok, this is a good step length. I think there was a small typo in the curvature condition. So, let us look at the curvature condition. Slope of the linear approximation at 0 in the at  $x_k$ ; instead of — well, I think it is correct. It is exactly what is written here.

If we do not know the exact function that is, all the data points that we have are  $(x, f(x))$ , then we do not know what  $f$  is. Why not just keep track of the minima while collecting all the values? The question is clear. So, we said that when we spoke about the intuition over here, which was, I think, this graph right? That we do not know the shape of this graph because this graph is expensive to calculate. So, the question is, if we do not know the exact function and only we only have the data points, why do not we just sort of pick points and keep track of the minima?

Does the optimization part come when we model a random function and compare the minima as a metric of how good the function approximation is?

So, and related is: I do not understand the point of  $\alpha$ ; if function evaluation is expensive, then using derivatives Hessians as a condition to check if  $\alpha$  is a good learning rate is contradictory. If the bounds on  $\alpha$  are found by function evaluation, we might as well evaluate the function at a lot of points and find the minima, right? As always, as they say, the devil lies in the details. Right now, if your function — if for example, I take three points — so supposing I pick three random points and with three points and a function of single variable, what is the best I could do in terms of fitting a polynomial? Three points I can fit a parabola. Right now, if I can fit a parabola, so you know, supposing — let us say, right? So, I pick one point here, one point here, one point

here. Now with these three points, I fit a parabola, and I automatically know that this is the best point, right? This is the minima point.

Observe?

Assume  $f_1 > f_2 > f_3$

$p_k \rightarrow -\nabla f_k$   
(grad descent)

Exact line searches

**OPTIMIZATION THEORY AND ALGORITHMS**

Now, if I were only taking function values and samples, it will take me forever to reach this point, and even when I reach a point, I will not have any guarantee that there is no other better point available. So one of the assumptions that we are making throughout this course is that the functions that we are dealing with are continuous, and not just continuous, they are also differentiable. So, calculus gives us nice ideas of smoothness which we can use. That is why a few function evaluations, like three function evaluations, is giving us a good approximation, a second order approximation.

So, in this case, it is justified. Going purely by function values is always going to be more expensive without any guarantee that you have hit the correct point, right? So, it is a lot of trial and error, and we do not want to rely on trial and error, ok. So, the sort of underlying assumption to be kept in mind is continuity and differentiability, and calculus gets you a lot more than just simple function evaluations.

So, does that clarify? Whoever this student's doubt was, ok, I guess. So, today we are going to talk about line search, the analysis part. Convergence is what we are going to talk about. So, let us write that out. Line search analysis.

Now, you have already seen the backtracking line search algorithm. So, backtracking line search algorithm was basically — let us just recap that before we get into analysis. So, we spoke about backtracking line search, which was basically: I have  $f(x_k + \alpha p_k)$ ; this should be — if I am taking the sufficient decrease condition, this function value should be less than what? Right, so  $f(x_k) + C_1 \alpha \nabla f_k^T p_k$  right, and this was  $C_1$ , and I am going to say repeat until this holds true and you keep updating  $\alpha$  like this. That was a very simple update; it required function evaluations, only one gradient, and that is this gradient at  $\alpha = 0$ .

The second — so, if you look at the optimization literature, there are lots and lots of different tricks that people have tried instead of backtracking line search. So I am going to give you just one more example, which I referred to just a few minutes ago of another trick, which people will do. Let us say that  $\phi(\alpha)$  — we all know what  $\phi(\alpha)$  is. Now, I give you, for example,  $\phi'(\alpha)$ ; I calculate this, I calculate this, and I calculate  $\phi'(0)$ , pick some — pick any  $\alpha$  and calculate this. So, this gave me three points, three different values of either  $\phi$  or  $\phi'$ . What I could do with it is, next is to fit a parabola. Right? Once I fit a parabola, I could, analytically, come to the point at which that function is minimized, and I pick that to be my best  $\alpha$ .

The image shows a whiteboard with handwritten notes and a diagram. The diagram consists of three concentric ellipses representing contour lines of a function, labeled  $f_1$ ,  $f_2$ , and  $f_3$  from outermost to innermost. A zig-zag trajectory is drawn with blue arrows, starting from the outermost contour and moving towards the center. The trajectory consists of three steps, each perpendicular to the previous one, illustrating a method like the Nelder-Mead algorithm. To the right of the diagram, there are handwritten notes: "Assume  $f_1 > f_2 > f_3$ ", " $p_k \rightarrow -\nabla f_k$  (grad descent)", and "Exact line searches". To the left of the diagram, the word "Observe?" is written. Below the diagram, there are three numbered observations: "1 Zig-zag trajectory", "2 step sizes are decreasing", and "3 Each step is  $\perp$  to previous step!". In the bottom right corner, there is a small video inset of a man in a blue patterned shirt.

So, here I am not doing backtracking line search; I am doing quadratic fit, minimizing the quadratic function and choosing that as my best  $\alpha$ . So, this is going to be good as long as the quadratic approximation is good. So, if I pick a very large  $\alpha$ , it is not likely that the function is going to be a quadratic function starting from  $\alpha = 0$  all the way to  $\alpha =$  some large number, right? So, again, I have to pick a reasonable number. Right? So, fit a quadratic. Once you get a hang of this idea, you can go a little crazy with it. I can take — instead of 3 points, I could take  $n$  points and interpolate a polynomial.



③ Each step is  $\perp$  to previous step!  $\leftarrow$

$$s_k = x_{k+1} - x_k \quad \#3 \rightarrow s_k^T s_{k+1} = 0$$

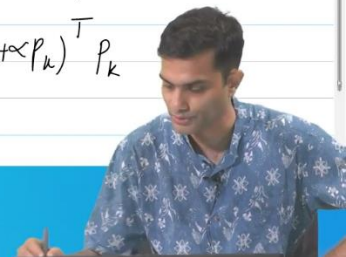
$$x_{k+1} = x_k + \alpha_k p_k \quad \underbrace{s_{k+1}^T}_{\perp} \underbrace{s_{k+1}}_{\perp} = 0$$

$$\Rightarrow s_k = \alpha_k p_k \quad \phi(\alpha) = f(x_k + \alpha p_k)$$

Exact line search  $\leftarrow \frac{d\phi}{d\alpha} = \nabla f(x_k + \alpha p_k)^T p_k$

$$\Rightarrow \frac{d\phi}{d\alpha} = 0 \Rightarrow \nabla f(x_k + \alpha_k p_k)^T p_k = 0$$

## OPTIMIZATION THEORY AND ALGORITHMS




Once I interpolate a polynomial, I can see are there multiple minima; I could fit a cubic, a quintic, whatever, and come to those points. So, you will find variations of this throughout the literature. Now, as it turns out, all of these tricks are not going to fundamentally change the rate of convergence. The rate of convergence, as we will derive later in this lecture or possibly in the next lecture, is linear; there is nothing that you can do to beat linear.

Let us now try to understand, first of all, does this line search method, for example, using steepest descent or gradient descent — does it converge? If it converges, at what rate does it converge?

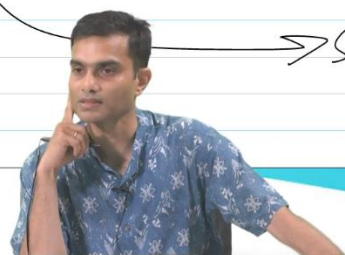
So, I am going to draw a graph over here; a better version of that is on the class notes, ok? I am going to try to draw it over here. So, let us — what I am drawing are the contours of the function. So, what are contours? Lines on which the function value is the same. So, think of them as cross sections of a hill. So, this is one; this is one.

So, I am trying to draw something elliptical. And let us say I start at this point. This is my starting point. Now, if this is the starting point, let us say that these contours are such in such a way that  $f$  — this is let us say  $x_1, x_2$  direction; let us say  $f$  is a function of two variables. So, starting at this point, let us say I go to the right to this point; I will have some function value here; then, I will go down here; so this function value would be lower than what I was there before.





$S_k = x_{k+1} - x_k \quad \#3 \rightarrow S_k^T S_{k+1} = 0$   
 $S_{k+2}^T S_{k+1} = 0$   
 $x_{k+1} = x_k + \alpha_k p_k$   
 $\Rightarrow S_k = \alpha_k p_k$   
 Exact line search  $\leftarrow \frac{d\phi}{d\alpha} = \nabla f(x_k + \alpha p_k)^T p_k$   
 $\Rightarrow \frac{d\phi}{d\alpha} = 0 \Rightarrow \nabla f(x_k + \alpha_k p_k)^T p_k = 0$   
 $\Rightarrow S_{k+2}^T S_{k+1} = \alpha_{k+2} \alpha_{k+1} p_{k+1}^T p_k = -\alpha_{k+2} \alpha_{k+1} \nabla f(x_k + \alpha_k p_k)^T p_k$   
 $p_{k+1} = -\nabla f_{k+1} = -\nabla f(x_k + \alpha_k p_k)$



**OPTIMIZATION THEORY AND ALGORITHMS**

So, steepest descent implies that I am always going to go in the direction of negative gradient. This is essentially what I am going to do: I will go this way. Now, you might think that every time I go, it is just going to get me closer and closer to the minima. However, if you see the trajectory I am going to take, it is going to zigzag like this rather than go directly to the minima. This is very interesting because, when I look at the trajectory, what I am going to find is that the step sizes appear to be decreasing, so they look something like this.

So, I go from this point to that point; then I go from that point to that point; then I go from that point to this point; this is essentially my trajectory that I am taking as the steepest descent is zigzagging down this way. The other thing that I observe is, you see, the trajectory appears to be roughly perpendicular to each of the previous steps. So, I can see that. So, I am going to define the distance between points as  $S_k$ .

So, let us just draw that here. Let me denote this as  $S_k$ , this as  $S_{k+1}$ . So, what I want to show is that if you observe the step, this is the one that I just drew above. So, I want to prove that  $S_k^T S_{k+1} = 0$ .

So, I will write  $S_k = \alpha_k p_k$ . Now, using that step, I can write the whole trajectory. So, every time I take a new step, I update that new point to be  $x_{k+1} = x_k + S_k$  right. So,  $x_k + \alpha_k p_k$ .

Now, once I have  $x_{k+1}$ , I can plug it in and plug it back to the function; I can write down the value of the function as the value of the function at the previous point plus the new step size. Now, using the derivative condition; using exact line search condition — so what we have is that the subsequent steps are actually going to be perpendicular.

In practice, when you implement backtracking line search, you do not really hit those exact perpendicular trajectories, but theoretically, there is a strong case for it.

## Conclusion

So, in summary, the backtracking line search is going to be a very good idea. We have talked about all of this; however, we do not have an exact convergence rate as a function of the original parameter; it may or may not yield a faster convergence in terms of the rate. The idea of line search, backtracking or otherwise, does have significant advantages, but the results may not always translate into practical applications.