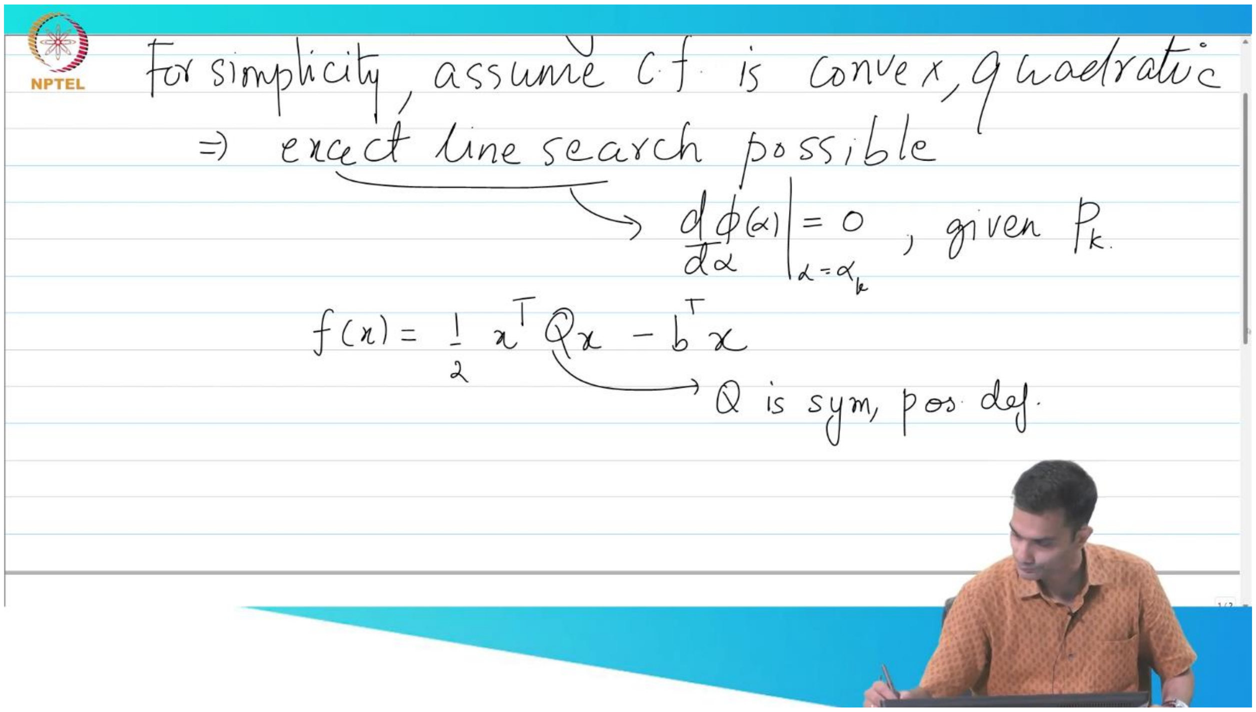
**Course Name: Optimization Theory and Algorithms**
**Professor Name: Dr. Uday K. Khankhoje**
**Department Name: Electrical Engineering**
**Institute Name: Indian Institute of Technology Madras**
**Week - 05**
**Lecture - 30**

# Convergence Analysis of a Descent Algorithm - 1

Okay, so let us begin. We will begin with the doubt sheets. The first doubt sheet actually is useful if there were more in number. It said I felt the class went a little fast. After the class got over, I asked the remaining few students who were there; one of them said this was the slowest class. So I don't know. If more of you feel that it's going fast, let me know; that will help me to modulate, right? But if one out of 70 says it's slow, I don't know if it's noise or not.

There were a couple of questions about this non-circular contours. It is not clear: does this perpendicular depend on the initial position, or is it true all the time for non-circular contours? If it is true for every initial point, is it only for exact line search? As we showed in the proof, it is true when I did exact line search. When I did exact line search, every step was perpendicular to the previous one. That just came out of simple algebra. It does not depend on the initial position.



For simplicity, assume c.f. is convex, quadratic

$\Rightarrow$ $\underbrace{\text{exact line search}}$ possible $\rightarrow$ $\left.\dfrac{d\phi(\alpha)}{d\alpha}\right|_{\alpha=\alpha_k} = 0$, given $p_k$.

$f(x) = \frac{1}{2} x^T Q x - b^T x$ $\rightarrow$ $Q$ is sym, pos. def.

Except if somehow you got lucky and the first starting point brought you straight to the solution, right? In case your initial direction was pointing straight at the solution, and the gradient, the function value is always reducing, then you will reach in one shot. In that case, we do not have to talk about perpendicularity between steps because there is no step; there is only one step. But I take any other point; this is a property of gradient descent: the steps are always orthogonal to each other.

Just like the tricks mentioned for proving this Zoutendijk condition, can you suggest common tricks used in bounds of different norms? I think one of the most common tricks is the triangle inequality, Cauchy-Schwarz inequality; these are the other tricks which are used. Again, with the elliptical contours, was your implication that apart from that one point which gave a single descent direction, every other starting point will give descent directions perpendicular to previous steps? Yes, it does.

Okay, then there is the weightage of the different components of the course; I will post it on the class website. We will continue our discussion of the convergence of the steepest descent, gradient descent, or any descent method in general. Remember the proof that we gave for convergence did not rely on $p_k$ being a gradient descent direction; $p_k$ just had to be what? A legitimate descent direction, and that is all that was required. So, it is applicable to a broader family.

Now, we proved that convergence happens; what we have not figured out yet is at what rate does convergence happen, right? This discussion is going to be about the rate. Now as it turns out, this rate analysis is a little bit more tricky to do. So I will make some simplifying assumptions, and with that simplifying assumption, we will arrive at a certain rate, and then I will point you to some other resources where people have worked it out in a more general case. It is much longer, the proof. So we will prove it for a simple case, and that simple case is assuming the objective function to be convex and quadratic.

So, for simplicity, let $Cf$ be the cost function, which is convex and quadratic. One of the advantages of having a quadratic cost function is that I can actually do line search exactly. This is, I believe, one of the tutorial questions as well. So, exact line search is possible. It's a simple matter of calculus to show this.

Exact line search means I actually have a closed-form expression for the step length. So that's great because it helps me with the analysis. I don't have to do backtracking line search. I don't have to check for Wolfe conditions. I can just get an expression at each $k$.

This is the step length. Walk along it. Okay. Remember what was the definition of exact line search? How did I define exact line search? Right. Rate $\frac{d\phi}{d\alpha} = 0$ at $\alpha = \alpha_k$. That is the meaning. I cannot do any better than this step length. Given what? What is assumed fixed over here? What am I assuming fixed over here? The direction.

$p_k$ is fixed. So I have been given the descent direction, and I cannot change that. I am going along that. What is the best I could do? It is $\alpha_k$. Ok. Let us just note that over here.

Given $p_k$. So, here is my cost function. I want it to be convex and quadratic. We have already seen a quadratic form before, right? So, this is the form of a quadratic cost function. Are there any constraints on $Q$? For it to be convex, it is already quadratic because there is $x$ multiplied by $x$.

For it to be convex, do I need something else? $Q$ should be positive definite. Obviously, it's also symmetric and positive definite. And remember the geometric intuition for that. If it is not positive definite, I don't have an upward opening cup; I can have maybe a downward cup.

I can have a saddle point, in which case it's meaningless to optimize a function that is unbounded from below. So this is guaranteeing me that it's at least a sensible thing to optimize. Okay, so this was our expression. What is, so now to do calculus with this, the things I will need is $\nabla f$, which is going to be needed; that is going to be used everywhere.

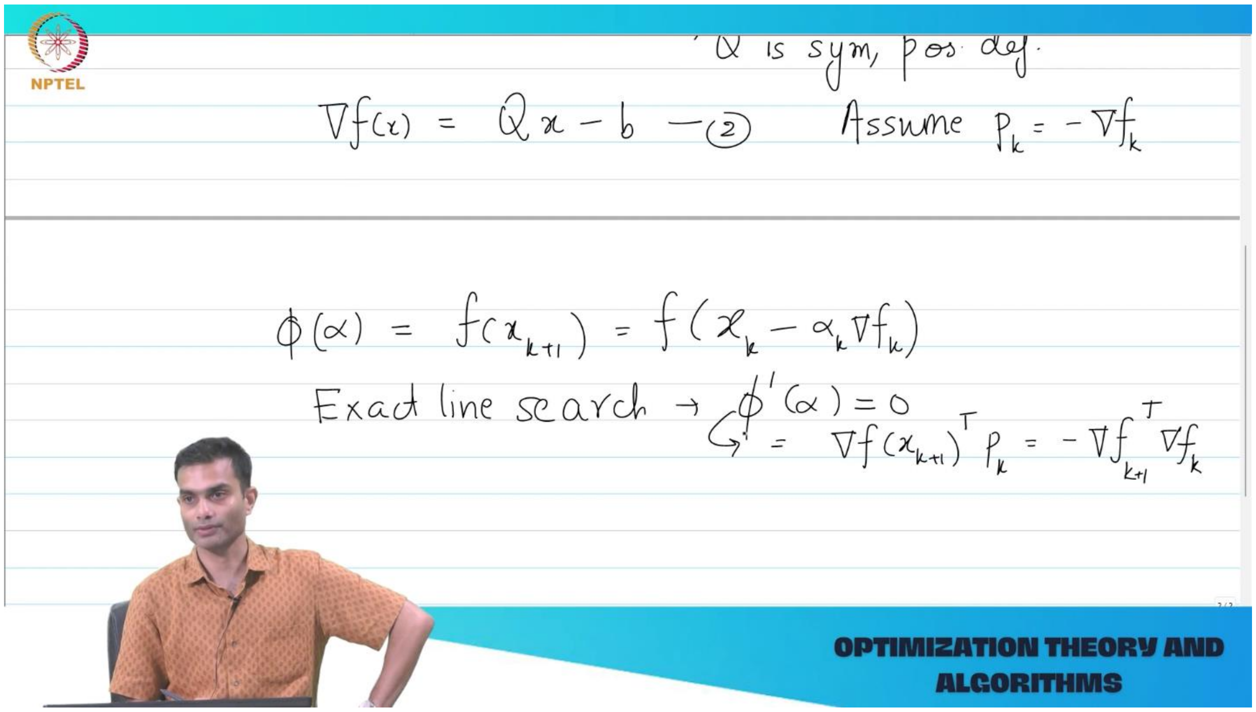
So we have derived this in great detail. What is $\nabla f$ in this case? First term I can do by product rule. What will I get? $Q + Q^\top$ multiplied by, the whole thing multiplied by $x$. It is symmetric $Q + Q^\top$ will give me $2Q$, right, and half cancels off; I am left with $Qx - b^\top$, okay. So, let us call this (1), this is (2), okay.

For this proof, I am going to assume that we are doing steepest descent, okay? Not any general descent direction; we will assume steepest descent. So, assume $p_k = -\nabla f_k$, okay, this is it. So, our old friend $\phi(\alpha)$, how did I define $\phi(\alpha)$? So, it is $f(x_{k+1})$, $f(x_k + \alpha_k p_k)$, but $p_k$ is already assumed to be $-\nabla f$. So, $-\alpha_k \nabla f_k$. Now, exact line search implies that $\phi'(\alpha) = 0$, right?

You can see that I already have in the past computed $\phi'(\alpha)$ using which rule? The chain rule, right? Can someone remind me what was the expression for $\phi'(\alpha)$? $\nabla f(x_{k+1})^\top p_k$, right? Which is going to be negative $\nabla f_k^\top$, let us call this $f_k$. This is just simple substitution; this is what $\phi'(\alpha)$ is going to give me, yeah.

You can see that this is going to be, is this an expression of a scalar, a vector, or a matrix? It is a scalar; it is an inner product of two vectors, and that is what I expect because $\phi$ is a function of a single variable, and the output is a single variable. So everything is fine, right?

Now, if you assume exact line search, which gives you this, we can actually get a closed form expression for $\alpha$ which we will derive in the tutorial. So let us assume it for now. I will give you the final result.


$Q$ is sym, pos def.

$\nabla f(x) = Qx - b$ — (2) Assume $p_k = -\nabla f_k$

$\phi(\alpha) = f(x_{k+1}) = f(x_k - \alpha_k \nabla f_k)$

Exact line search $\rightarrow \phi'(\alpha) = 0$

$\hookrightarrow \phi' = \nabla f(x_{k+1})^\top p_k = -\nabla f_{k+1}^\top \nabla f_k$

So I will be able to write my $x_{k+1}$ in terms of $x_k$ and this expression for $\alpha$. Quite a simple derivation. Now, when we talk about convergence, okay, so this is just sort of building up all the nuts and bolts needed. Let's get back to what we wanted to talk about.

We wanted to talk about convergence. Now convergence means the distance between what and what are we interested in? Is it that, or is it the solution? Just look back at the definition of convergence. Is it the difference between iterates, or is it the difference between the iterate and the true solution, the stationary point between iterates, the convergence points, right?

So, I mean that kind of makes sense. When I am talking about convergence, I am interested in something like where $x^*$ is the, what is $x^*$? Stationary point. This is what I am interested in, right. When I say convergence rate means how quickly am I arriving at the solution; this is finally what I am interested in, ok.

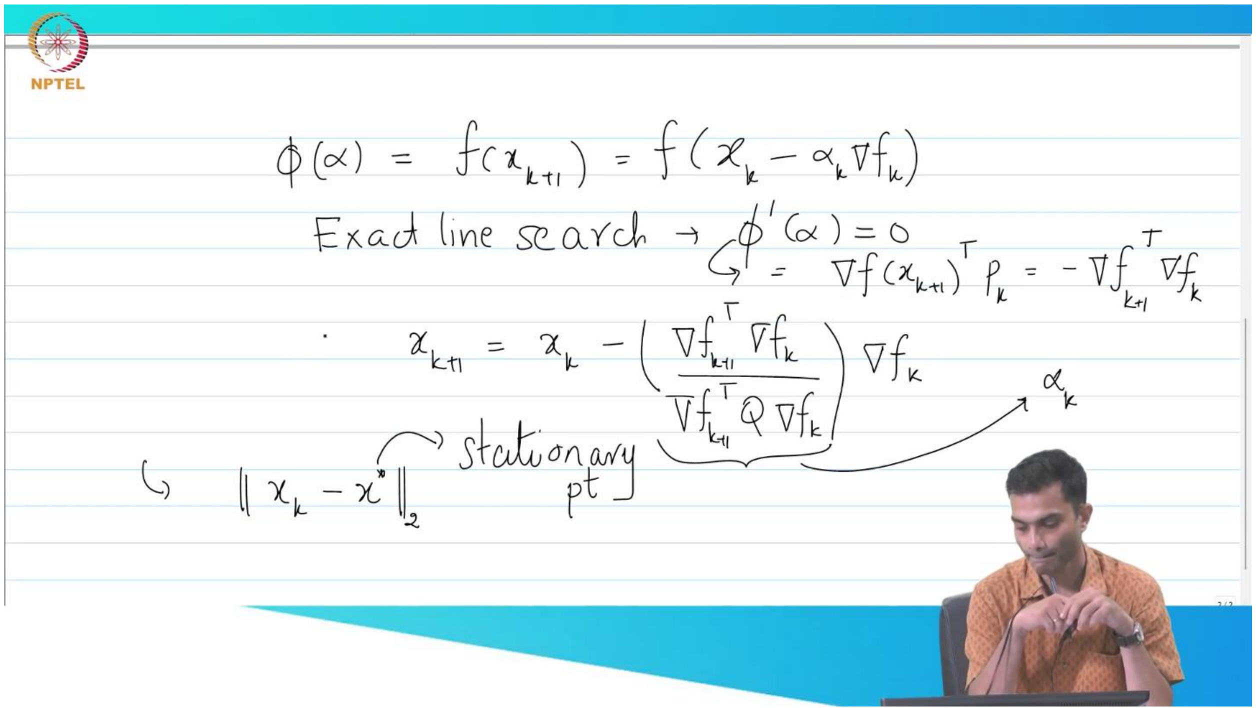
And typically, the way it is defined is with the two norm; that is the definition that we had seen, ok. As it turns out, if I were to use the two norm, the math gets quite grungy; it does, but we will define a simpler version of it. What if we define convergence rates using the 2-norm?

We will define a quantity like this; it is the distance, the current iterate and the stationary point, ok. So I will just write this as a simple notation; let us say the iterate at $k$ is denoted by $x_k$ and stationary point is denoted by $x^*$. And I want to see how far I am from this. And this is defined by something like this. So at each iteration, let me say what happens to this distance at every iteration?

So we are interested in how it evolves as $k$ increases. So, as we keep on going up, we will have an expression that depends on how far we were previously to where we want to be. This distance becomes smaller and smaller as the iterations go on, right? So as I do more and more iterations, this should be a decreasing function.

So in this case, this is the quantity we want to analyze for convergence.

This is my convergence; distance goes to zero. If I can establish a bound on this, it will help me get convergence rates, ok?



$$\phi(\alpha) = f(x_{k+1}) = f(x_k - \alpha_k \nabla f_k)$$

Exact line search $\rightarrow \phi'(\alpha) = 0$

$$\hookrightarrow = \nabla f(x_{k+1})^T p_k = -\nabla f_{k+1}^T \nabla f_k$$

$$x_{k+1} = x_k - \underbrace{\left(\frac{\nabla f_{k+1}^T \nabla f_k}{\nabla f_{k+1}^T Q \nabla f_k}\right)}_{\alpha_k} \nabla f_k$$

$\hookrightarrow \| x_k - x^* \|_2$ stationary pt

In general, let's say the relationship is like this, alright? So this is the distance to the stationary point at step $k+1$ is less than some constant $\alpha$ times the distance from the previous iterate at $k$, ok? And typically this $\alpha$ is a constant less than one.

Let's now relate this to the quadratic case, ok? So, if I take the 2-norm and I assume the quadratic structure of the cost function, I will arrive at something like this:

$$\| x_{k+1} - x^* \|_2^2 \leq \alpha \| x_k - x^* \|_2^2$$

With some constant $\alpha < 1$. This is the kind of convergence we are interested in. If $\alpha$ is sufficiently small, the convergence will be quick, okay?

To summarize, we need to establish the relationship above in terms of the distance between current iterates and stationary points and analyze how quickly we approach the optimal point, given a descent algorithm and convex quadratic structure of the objective function.