Optimization from Fundamentals

Prof. Ankur Kulkarni
Department of Systems and Control Engineering
Indian Institute of Technology, Bombay

Lecture - 19A
Line search methods for unconstrained optimization

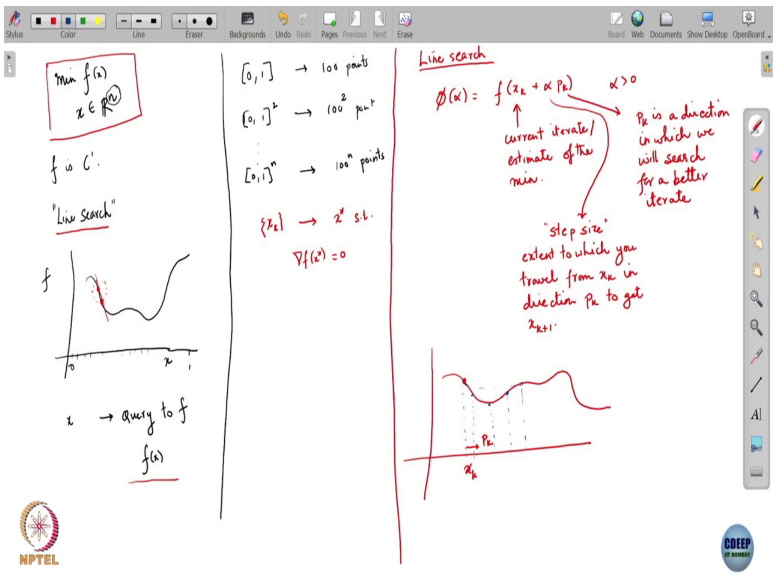
Welcome everyone. So, we have so, far studied Optimization from a perspective of characterizing the solution of an optimization problem, but we have not considered the question of how does one compute such a solution? So, today what I will talk about and is this is the basic ingredients of algorithms for computing solutions of optimization problems. Now, algorithms comes algorithmic development in optimization has happened ever since the history is ever since the subject itself was studied.

So, there is a very long and varied history of the subject. So, we will not have the time to cover all such algorithms, but what I want to get a sense of are the various types of algorithms, or the various flavors of algorithms that are classes of algorithms, that are out there today.

And, the kind of ideas that underpin such classes of algorithms. The other thing I want you to get a flavor of is the kind of issues that arise in practical optimization algorithms. It is often very easy to think of algorithms as simply an iteration and analyze them simply on paper, but usually when one practically implements them the kind of corner cases that come up require some special attention. So, I will shed some light on a few such practical issues also, as per as part of this.

So, this is in this module of our course, we will that is what we will be focusing on. So, to begin with what I want you to, what I want to talk about is simply I will optimization without any constraints. So, we will be talking about, what we what you would call unconstrained optimization or optimization over open sets, but for us for simplicity we will take the open set to be R n.

(Refer Slide Time: 02:27)



$\boxed{\begin{array}{l}\min f(x)\\ x \in \mathbb{R}^{n}\end{array}}$

f is C^1.

"Line Search"

$x \rightarrow$ query to f

$f(x)$

$[0,1] \rightarrow 100$ points

$[0,1]^2 \rightarrow 100^2$ point

$[0,1]^n \rightarrow 100^n$ points

$\{x_k\} \rightarrow x^*$ s.t.

$\nabla f(x^*) = 0$

Line search

$\phi(\alpha) = f(x_k + \alpha p_k) \quad \alpha > 0$

current iterate/ estimate of the min.

p_k is a direction in which we will search for a better iterate

"step size" extent to which you travel from x_k in direction p_k to get x_{k+1}.

So, we will so, there is so, we would have, we would be looking at basically a function f, that you would be looking to minimize over the entire space R n. And, we will be so; we will assume that f is c 1. So, it is continuously differentiable ok. Now, the simplest idea that you can potentially use is in optimization is what is called “Line Search”. Now what is so, before I tell you this particular idea? I should actually let us first get a sense of why optimization, algorithms are a subtle matter.

And, why is it that these algorithms require, why do we even need to be careful about how we think of algorithms? So, when I, when we do analysis we or when I draw a picture of the function like this on the board something like this. I am effectively giving you the graph of, I am giving you the graph of f and I am giving you at one glance how the function f looks over the entire domain. Now, this view is usually not present in an for an optimization algorithm.

An optimization algorithm would work on in a query mode. What is what does that mean, it would, it would query the function. For f if you it takes an x and queries sends this as a query to f and then what you have is a routine, which will return you the value of f of x.

So, it is possible to query the value of f of x and get it for very any particular value, any candidate x that you pick. What you do not have is this kind of birds eye view, where the entire graph is listed for you. Even if you, even if you wanted to construct it, what you would need is the value of f for every possible x and there are infinitely many such x s right. So, the naive way for optimization is to approach for optimization is that you try and graph out the entire function to some degree of approximation.

You cannot plot you cannot evaluate it for every point. So, you say well let me evaluate it over a grid of points say there are. So, you suppose you are evaluating it from 0 to 1, you make let us say this is 0 this is 1. You are evaluating it from 0 to 1, you make a fine grid say of 100 points suppose and evaluate the function at each of these points, assume that there is that you can interpolate between the end points of the function or some such way, you create an approximation.

And, then search over these points over that approximation or search over these points to get an approximate minimum answer. The trouble with doing this is that this works ok when the number of variables or the dimension n here is small right. If the dimension n here becomes larger, then the number of points that you would have to make starts growing exponentially with n right. So, if you have, if you are in 0 1 and you decide to make a grid of 100 points, you would need 100 points.

You grid with where you need where you sub divide 0 to 1 into if your accuracy level is 10 to the minus 2, then you would need 100 points. But, if you are, if you are in 0 1 square, suddenly you would need 100 square 0 1, cross 0 1 then suddenly you would need 100 square points.

Or and more generally if you are in 0 1 to the n you would need 100 to the n points. So, you can see this sort of approach to optimization. Where you do what you would have what you are attempting to do on paper which is simply graph out the entire function over the graph out the function over the entire space.

And, and try to compute try to compute the value of the function on the, on a grid of points and then from there evaluate basically just list out the values and pick out the least possible value. This kind of approach is not feasible computationally at all because when the dimension becomes larger.

It is the, it is the it is today considered the approach of last result. Essentially, you would not want to do this as and if you have any better option right. Because, the number of function evaluations you need is ridiculously large. Second, so this is one so, in addition to this there is also the fact that you have not made any use of any of the theory that we have developed about optimization right, we have developed kkd conditions. We have developed conditions for a function to be a minimum and so on.

It this way of graphing the function and then simply selecting the least point makes no use of any of that. So, it is actually, it shows you how generic this is and how and as a result, how you know potentially inefficient that it could be. Because, there are many more things that you can make use of which you have ignored in this approach. So, what we would the approach that we will take is that of search. So, what is search mean by search I mean that we will be we start out at some point.

And then we query the function. The query reports back to us the value of the function. Some in some cases it may report back to us more, you could ask for you could ask not just for the value of the function, but you could also ask for say the derivative of the function, the hessian of the function, or you know some or any other such information about the function. Depends on what kind of properties you have and what are you allowed to query.

Now, in suppose you are suppose so, what we will do is you would you start out at a point, you query the function to get these vital stats about the function, it is value, it is derivative etcetera. Then from there make an educated guess about where you should be going ok. You started out at this point which is kind of arbitrary to begin with, but then based on based on the information that you have got about the function and it is derivative you get a sense of where you, where you could, where you should next go.

What does that mean? You try to create in using this information a model of the function around this point. So, you could try to create a model of the function in a neighborhood of this point. And, then we say ok well now based on this model, where what should I be doing. And, that based on that you select the next point. And, the next point and then having selected the next point you again query the function at that point, you get values of the function and it is derivative at that point and you create a new model.

You could you create a new model and then decide, where you should next go ok. Now, by when I say we create a model and so on. These are not very complicated models, usually what we are saying is we are using some sort of a heuristic, which says that we will be moving say for example, along the linear approximation to the function. We take the function to be equal to it is linear approximation that is our model.

And, then we move in that direction and then and then we move step further in that direction, revise our model move a step further and so on. Now, every such approach will be only as good as the model. So, we cannot; obviously, trust our model too much to we so, we would wanted to we so, therefore, this kind of an approach works in baby steps. You start from a point, you create you have a model say for example, you have you take a linear approximation.

You use you trust that model to work in a certain neighborhood. So, you take a step that is not too large within using this particular model, reach a new point and then revise the model, and then take another step and so on. So, this leads us to an iteration, that and what we want is that the limiting point of this iteration.

The limit as n as the number of iterates goes to infinity of this particular iteration, converges to the minimum of the function. So, this is the mathematical property that we are looking for. So, what we usually what one looks for is that you that these iterations, end up verifying at least the necessary conditions of optimality. So, it should satisfy all the necessary conditions of optimality that you can say about this particular function class that you are considering.

So, for example, if the function class is a differentiable function, then we would want that this sequence of iterations say x, say x k we would want that this sequence of iterations converges to an x star such that gradient of f at x star is equal to 0 right. So, this the, this is the flavor of results that we are looking for alright ok. So, the first type of algorithm of this kind of this category is what is called, the what are called lines search, as I mentioned here lines search, now what is line search?

So, line search essentially does the following that it looks at this looks for it looks at an optimization problem of this kind. So, you are minimizing a function f over the entire space x in rn it starts out at an, at a current iterate x k. And, then updates x k by looking at how the function behaves along a, along the direction along a linear, along a linear approximation. Where the linear approximation is obtained, by where the linear approximation is obtained by using the gradient of the function right.

So, let us define. So, let me erase this and let us define the following let us define the following. So, define phi of alpha as f of x k plus alpha times p k, where alpha is some is greater than 0. So, let me explain what each of these terms are. So, x k here is your current iteration, current iterate ok, current iterate or estimate of the minimum, p k is a is pk is a direction is a direction in which is a direction in which we will search for a better iterate. Now, alpha here this is what we call step size ok.

What is this step size? Step size tells you the extent or the distance or the you would like to travel along this direction p k to get to the next iterate ok. So, alpha the step size, what does it capture, it captures the extent to which you travel from x k in direction, p k to get x k plus 1 alright ok.

So, what we have done is we have we parameterize the next iterate by this step size. And the entire idea of line search is to keep is to pick the right step size and the right direction ok alright. So, let us take this one step forward now ok. So, a typical line search algorithms what they would do is they would keep trying out values of alpha, so ok. So, before I tell you that let us actually see what we would want to do. So, suppose you are, suppose we have a function this sort of function.

And, suppose you are at a point like this. You and this is the direction, this is the direction p k that we have, we are at a point x k here this is and p k is the direction pointing to the right. Now, what you would like to do is travel is to starting from x k, you would you want to travel in the direction p k.

And, the question you are faced with is how much should I draw, how much should I travel? So, you could say well let me go ok. So, remember p k is simply a direction. So, there is no scale to pk. So, you can you have to actually define the scale and the extent to which you want to travel in that direction, using, the using, the term alpha right. So, suppose you move until here say ok. Suppose you move until here.

Now, until here you are getting, you are getting actually a decrease in the function right. Now, you could, and you could stop here. And, say well ok I have got enough decrease and then revise the model or you could say well let me I have got so, much decrease everything looks great.

Let me move further, you go till here you could taken you could take or iterate all the way till here. And, then you see ok well my decrease has not started has not has not kept up maybe I should now revise my model and get to a new point. Or you might say well no ok let us give it a try let us increase alpha further and see if that work. And, then the function starts increasing and then you end up with somewhere here.

So, you can see what is happening here is that there are the if as you go from in this entire range of alphas, you encounter various types of situations. You encounter situations, where it seems like your decrease is you made a decrease, but it could the function could decrease further.

You get to a you could get to a situation where you have made a decrease, but the function cannot decrease you do not think the function can decrease any further in this direction. You get to have you could also get to a situation where you have over compensated, you have gone to you have taken too larger a step.

And you have not you have not really got the decrease that you think you should have got. For example, if you take a decrease all the if you take a very large step you could end up you could end up here and you end up actually missing what could have potentially been a solution here also right.

So, now, the problem is there are infinitely many alphas to choose from and there we do not we since we do not have the graph of the function, the only way we can go about selecting the alpha is through sort some sort of intelligent trial and error, you might try this you might try that etcetera. And, then you have to you need to get you have to sort of feel your way through and figure out what the correct amount of alpha should be.