Computational Method of Optimization

Unconstrained Optimization- Lecture IX

Unconstrained Optimization

- Agenda
- terminology and assumptions
- gradient descent method
- steepest descent method
- Newton’s method

Source: Convex Optimization — Boyd & Vandenberghe

Unconstrained minimization

$$\text{minimize} \quad f(x)$$

- f convex, twice continuously differentiable (hence $\mathbf{dom}\, f$ open)
- we assume optimal value $p^\star = \inf_x f(x)$ is attained (and finite)

unconstrained minimization methods

- produce sequence of points $x^{(k)} \in \mathbf{dom}\, f$, $k = 0, 1, \ldots$ with

$$f(x^{(k)}) \to p^\star$$

- can be interpreted as iterative methods for solving optimality condition

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

Let's look at unconstrained minimization. So here, the problem is simple. Just want to minimize a non quadratic convex function smooth, that's minimized f(x). Minimizing an quadratic you can do by solving a linear equation, because the gradient of f is f-line, and so you set that equal to zero, that's a linear equation. So we're interested in minimizing non quadratic convex functions smooth.

we're going to look at methods that produce a sequence of points, there're all in a domain, and the function values are going to go to this algorithm value p^*. By the way, in the case when p^* is $-\infty$, that means this is unbounded below, you also call this a minimizing sequence. So here, in that case, the x's don't converge.

Unconstrained minimization

$$\text{minimize} \quad f(x)$$

- f convex, twice continuously differentiable (hence $\mathbf{dom}\, f$ open)
- we assume optimal value $p^\star = \inf_x f(x)$ is attained (and finite)

unconstrained minimization methods

- produce sequence of points $x^{(k)} \in \mathbf{dom}\, f$, $k = 0, 1, \ldots$ with

$$f(x^{(k)}) \to p^\star$$

- can be interpreted as iterative methods for solving optimality condition

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

And of course, to minimize a smooth convex function is the same as solving the equation, which is the degrading sequel to zero. So, another way to think of this is you want to solve this set of N, non-linear equations and in variables. These equations would be linear equations f were quadratic. So, this is a method for solving these systems. That's another way to think of it.

Initial point and sublevel set

algorithms in this chapter require a starting point $x^{(0)}$ such that

- $x^{(0)} \in \mathbf{dom}\, f$
- sublevel set $S = \{x \mid f(x) \leq f(x^{(0)})\}$ is closed

2nd condition is hard to verify, except when *all* sublevel sets are closed:

- equivalent to condition that $\mathbf{epi}\, f$ is closed
- true if $\mathbf{dom}\, f = \mathbf{R}^n$
- true if $f(x) \to \infty$ as $x \to \mathbf{bd}\, \mathbf{dom}\, f$

examples of differentiable functions with closed sublevel sets:

$$f(x) = \log(\sum_{i=1}^{m} \exp(a_i^T x + b_i)), \qquad f(x) = -\sum_{i=1}^{m} \log(b_i - a_i^T x)$$

We'll assume the following, that we have a starting point because you have a point in the domain of the function. And we're going to assume that the sublevel set is closed.
Now, when all sublevel sets are easy to work out, but that's basically the same as saying that the epigraph is closed. That's standard notation in convex analysis to call a closed function if its epigraph is closed. And this would be true, for example, the domain is everything, so that's one case.

And another case is this, if f goes to ∞, as you approach the boundary of the domain, this has another name, it's called a barrier. So for example, one over x is going to be closed, defined on x positive, that's closed because as you approach the boundary of the domain, which is zero, from, +x, the function goes to ∞, so that's one.

Initial point and sublevel set

algorithms in this chapter require a starting point $x^{(0)}$ such that

- $x^{(0)} \in \mathbf{dom}\, f$
- sublevel set $S = \{x \mid f(x) \leq f(x^{(0)})\}$ is closed

2nd condition is hard to verify, except when *all* sublevel sets are closed:

- equivalent to condition that $\mathbf{epi}\, f$ is closed
- true if $\mathbf{dom}\, f = \mathbf{R}^n$
- true if $f(x) \to \infty$ as $x \to \mathbf{bd}\, \mathbf{dom}\, f$

examples of differentiable functions with closed sublevel sets:

$$f(x) = \log(\sum_{i=1}^{m} \exp(a_i^T x + b_i)), \qquad f(x) = -\sum_{i=1}^{m} \log(b_i - a_i^T x)$$

So here's some examples. This function is to understand why this has a closed sublevel sets. The domain is an open polyhedron here. So if the sets of point x for which $a_i^T x < b_i$. So the domain is an open polyhedron here. The condition that we know x zero in dom f, is not trivial. Because, to find such a point, you'd have to determine if a certain polyhedron has nonempty interior and produce a point in it. Now, in this case, if you approach the boundary, it means you approached the boundary of the polyhedron. It means that one of these terms $b_i - a_i^T x$ goes to zero. If that happens, the associated log term goes to - ∞ and with a minus sign it goes to up + ∞ . So, this satisfies the second condition for sure and, therefore, this has closed sublevel sets.

Strong convexity and implications

f is strongly convex on S if there exists an $m > 0$ such that

$$\nabla^2 f(x) \succeq mI \qquad \text{for all } x \in S$$

implications

- for $x, y \in S$,
$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$
$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2}\|x - y\|_2^2$$

  hence, S is bounded

- $p^\star > -\infty$, and for $x \in S$,

$$f(x) - p^\star \leq \frac{1}{2m}\|\nabla f(x)\|_2^2$$

  useful as stopping criterion (if you know m)

We're going to take strong convexity. Now, strong convexity means this, it means that there's a minimal curvature, a positive minimum curvature. So we'll assume that in this case. This means it's a minimum curvature.
In this case, this is what it means to be convex. This says, if you look at the function at another point it's bigger or equal to, this thing over here is nothing but the first order Taylor approximation and it says, basically, the function is above that.

You write up the second order term here, and use the mean value theorem, or whatever they call it, and plug in some value in this and you'll get this. And this says, you're larger than or equal to the first-order Taylor series, this has a minimum curvature away from it.

Strong convexity and implications

f is strongly convex on S if there exists an $m > 0$ such that

$$\nabla^2 f(x) \succeq mI \qquad \text{for all } x \in S$$

implications

- for $x, y \in S$,
$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$
$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2}\|x - y\|_2^2$$

  hence, S is bounded

- $p^\star > -\infty$, and for $x \in S$,

$$f(x) - p^\star \leq \frac{1}{2m}\|\nabla f(x)\|_2^2$$

  useful as stopping criterion (if you know m)

This says, when you move away, it's not that there's a linear thing and you're above that linear thing, it says there's a quadratic. This is a quadratic here in y. And it says, there's a quadratic that you're above. So it's a strong assumption.

It implies that the sublevel set it bounded.
It also implies, although it's a couple lines to derive at the following, it says that if you have a point x then f(x)- p^*, that's the minimum value of f, it's $\frac{1}{2m}\|\nabla f(x)\|_2^2$

Stopping criteria. And it says, when the norm of the gradient is small you can stop.
How small does it have to be? It depends on m. if the norm of the gradient is less than some number, they say, how do you know you're close to the optimum?

Descent methods

$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)} \quad \text{with } f(x^{(k+1)}) < f(x^{(k)})$$

- other notations: $x^+ = x + t\Delta x$, $x := x + t\Delta x$
- Δx is the *step*, or *search direction*; t is the *step size*, or *step length*
- from convexity, $f(x^+) < f(x)$ implies $\nabla f(x)^T \Delta x < 0$ ($i.e.$, Δx is a *descent direction*)

General descent method.

given a starting point $x \in \mathbf{dom}\, f$.
repeat
1. Determine a descent direction Δx.
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.

We'll look at descent methods after this. It goes like this, you're next iterate is going to be the current point plus a direction, a vector, a displacement times the scale factor. Now, these are universally come to be called, the search direction

The general step goes like this. You have a point x, you calculate at that point a direction to move in, and then you calculate, it's called a line search but, in fact, it's more a ray search, because you only look in positive x. You find a positive step size and then you update like this. usually in finding a descent direction, if the gradient of f is zero, then you terminate, and you don't generate.

So when you determine this end direction part of that usually is something that checks the stopping criteria. So this is the general method. Now, these search directions have to make a negative enter product with the gradient.

Descent methods

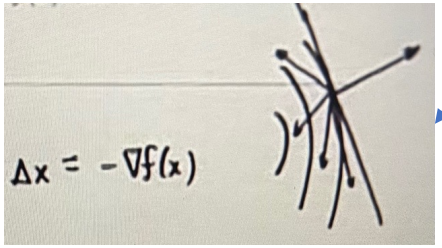
$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)} \quad \text{with } f(x^{(k+1)}) < f(x^{(k)})$$

- other notations: $x^+ = x + t\Delta x$, $x := x + t\Delta x$
- Δx is the *step*, or *search direction*; t is the *step size*, or *step length*
- from convexity, $f(x^+) < f(x)$ implies $\nabla f(x)^T \Delta x < 0$ ($i.e.$, Δx is a *descent direction*)

General descent method.

given a starting point $x \in \mathbf{dom}\, f$.
repeat
1. Determine a descent direction Δx.
2. *Line search.* Choose a step size $t > 0$.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.



Now, these search directions have to make a negative enter product with the gradient. So, in another works, if you're function look like this, and the gradient looks like that, then it says that you're search direction has to be sort of in this open as space, over here. And that's the most obvious search direction there is, which is the negative gradient.

Line search types

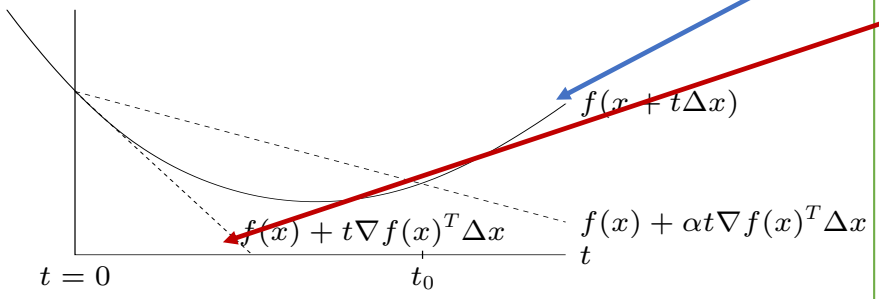
exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

backtracking line search (with parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$)

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

- graphical interpretation: backtrack until $t \leq t_0$



On an exact line search says, once you point on the direction and you sort of plot, you work out what f is along that way. In fact, for all practical purposes, you've reduced your problem to a one-dimensional problem. You plot it and, that's clear, right, it just goes like that. If it just keeps going down, you found a direction of infinite descent.

If it points up, when t is positive it means you need to fire whoever wrote the code that generated the search direction. So, this is what an exact search would give you this here. And you'd think, well, look that's the point to minimizing f that's got to work well. But the other one – there are lots, and lots, and lots of line searches. But at the other extreme, there is something that is simple, and it goes like this. You start with a step of one. And then, what happens is, you repeatedly multiple by a constant **β**

Line search types

exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

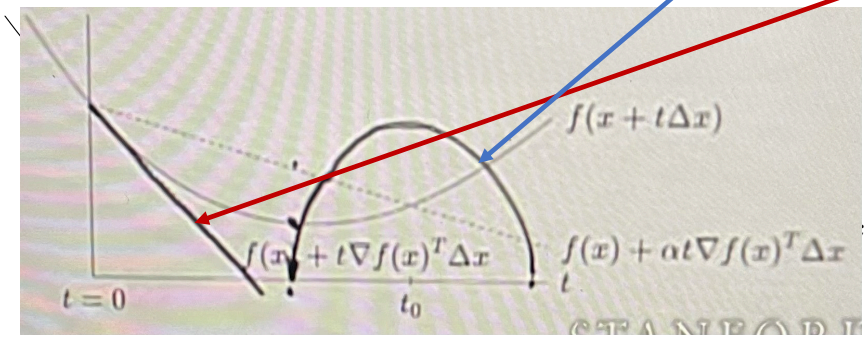
backtracking line search (with parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$)

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

(f(x + tΔx) < f(x) + αt∇f(xΔx)^T)

- graphical interpretation: backtrack until $t \leq t_0$



β's typically a half or something like that So this condition says this, it says when you evaluate (f(x)+ t Δx), and let's look at a few things.

Let's put α=1. This is the derivative, is a directional derivative of the function along, you see this function right here where I erased the α? This is this. And by convexity, this condition here will never happen. It cannot. That's the point. This thing here is a lower bound on that. So you'll never get a decrease as good as that.

So for example, if this number here is like, you know, .6 and you try a step size of t =1, this predicts that f will go down by .6. But it cannot go down by .6. So you have to degrade how much reduction you want. what this does, is you multiple by α which is between 0 and a 1/2.

Line search types

exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

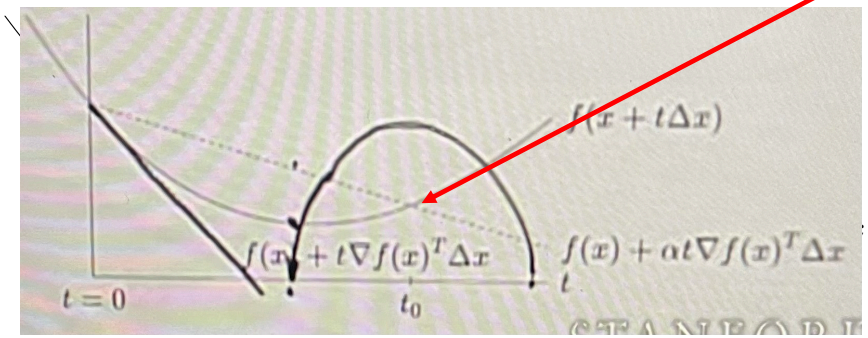
backtracking line search (with parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$)

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

(f(x + tΔx) < f(x) + αt∇f(xΔx)^T)

- graphical interpretation: backtrack until $t \leq t_0$



This is called backtracking, this method, and that's the idea. And what happens is, you get the first point, when you multiple by **β**, that lies below this critical value t_0. t_0 is the first point where you get equality in this thing. It's where this line intersects the function there.

Line search types

exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

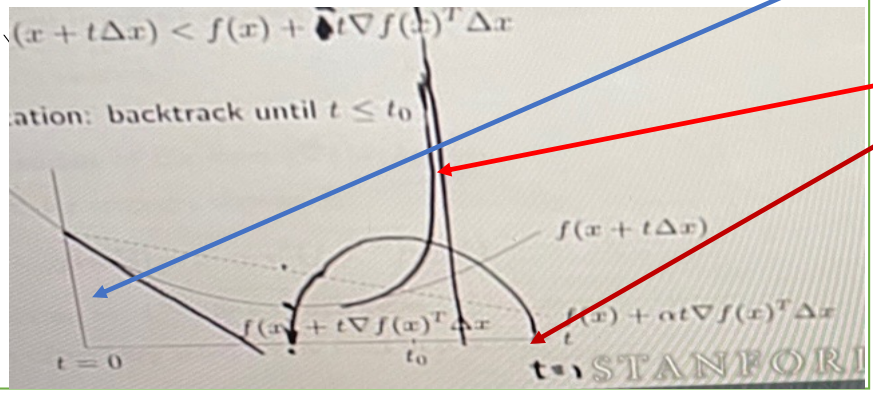
backtracking line search (with parameters $\alpha \in (0, 1/2)$, $\beta \in (0, 1)$)

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

(f(x + tΔx) < f(x) + αt∇f(xΔx)^T)

- graphical interpretation: backtrack until $t \leq t_0$



So, this is called backtracking, this method, and that's the idea. And what happens is, you get the first point, when you multiple by **β**, that lies below this critical value t=0 . T=0 is the first point where you get equality in this thing. It's where this line intersects the function there. So that's the picture. These are line search types.

One thing very important is this, is f can have a domain that looks like that. Here. And if this is t=1 here, I mean, this is kind of obvious. This thing works perfectly provided your function returns the right thing. If you evaluate this thing and you're out of domain, if it returns plus infinity then this in equality is interpreted as false, because there's no way infinity is going to be less than anything.

Gradient descent method

general descent method with $\Delta x = -\nabla f(x)$

---

given a starting point $x \in \mathbf{dom}\, f$.
repeat
1. $\Delta x := -\nabla f(x)$.
2. *Line search.* Choose step size t via exact or backtracking line search.
3. *Update.* $x := x + t\Delta x$.

until stopping criterion is satisfied.

- stopping criterion usually of the form $\|\nabla f(x)\|_2 \leq \epsilon$
- convergence result: for strongly convex f,

$$f(x^{(k)}) - p^\star \leq c^k (f(x^{(0)}) - p^\star)$$

  $c \in (0, 1)$ depends on m, $x^{(0)}$, line search type
- very simple, but often very slow; rarely used in practice

We get to gradient descent method. And it takes the negative gradient direction.
By the way, this is a classic, greedy algorithm, which is to minimize f, and you simply look at that iteration and make the maximum progress towards goal. This is the greedy algorithm for minimization.

There's what you do, you chose a negative gradient, you do a line search, and the stopping criterion would typically be that the gradient is small. So the exit criterion would be checked in step one.

So that means that your sub optimality, that's the difference between f and the optimal value of f, goes down by a factor less than one at each step. Now, that's a pretty good convergence.

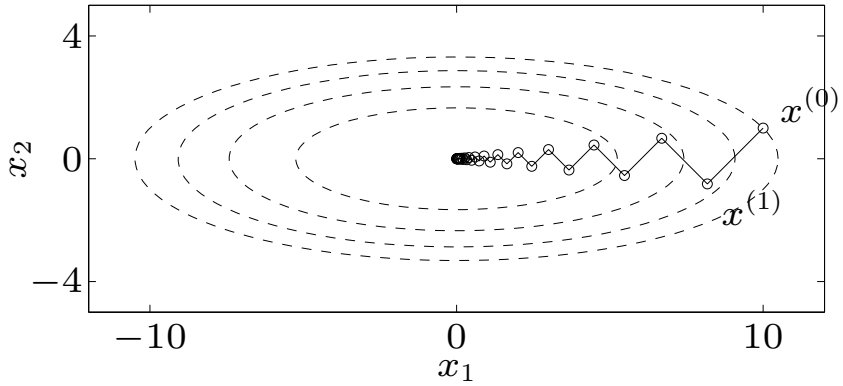
Quadratic problem in R^2

$$f(x) = (1/2)(x_1^2 + \gamma x_2^2) \qquad (\gamma > 0)$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

$$x_1^{(k)} = \gamma \left(\frac{\gamma - 1}{\gamma + 1}\right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma - 1}{\gamma + 1}\right)^k$$

- very slow if $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$:



So here's just a quadratic problem. Obviously, we know what the minimum is. The minimum is zero, that's clear. But let's just run the algorithm on it. Well, when I minimize this, I can work out the gradient, I can work out exact line search, and so on, and you can, just analytically work out what the points are, what the iterates are, so they look like this. They converge to zero geometrically, Exponentially.

Then, it says that you go like, actually, the γ-1 over γ + 1. Now, if you're condition number is 10, You multiply that .90 step. If you're condition number's 100, it means that you're only making 1% progress and if it's 10,000 or a million, it means you're making very, very poor progress at each step.

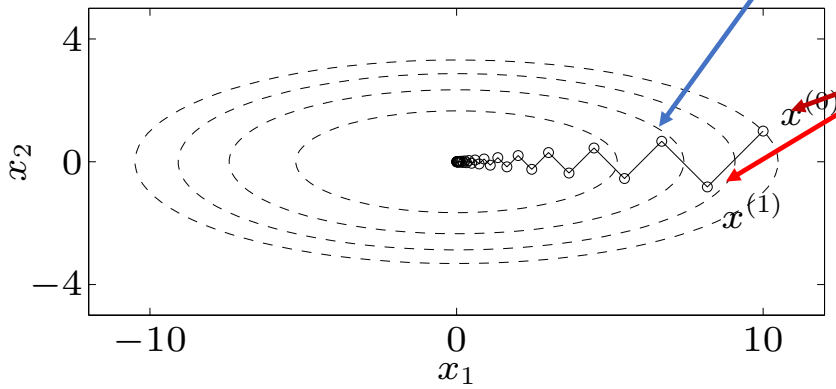
Quadratic problem in R^2

$$f(x) = (1/2)(x_1^2 + \gamma x_2^2) \qquad (\gamma > 0)$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

$$x_1^{(k)} = \gamma \left(\frac{\gamma - 1}{\gamma + 1}\right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma - 1}{\gamma + 1}\right)^k$$

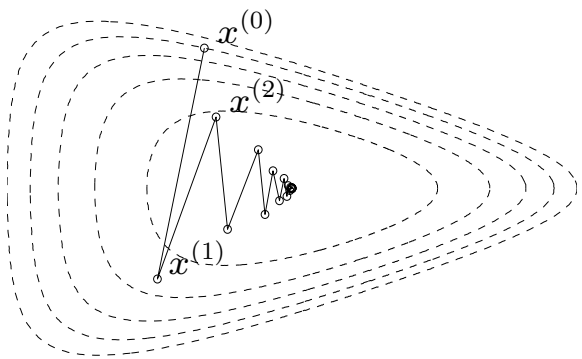
- very slow if $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$:



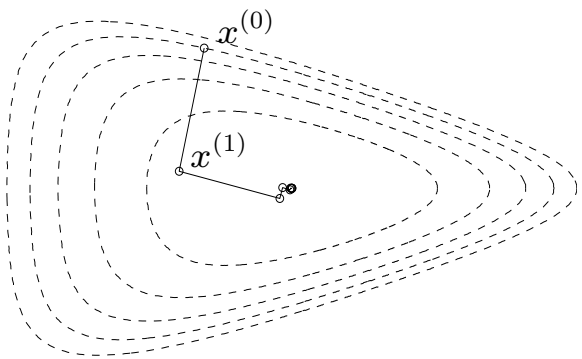
What happens is this, you're here – by the way, this is for γ=10. That's a very small condition number. You're here, and so the direction, that's the gradient and it gives you the direction of most rapid local increase. It's the most uphill direction. You go in the most downhill direction. So you go along here and then, of course, you minimize along that line, that when you minimize you're orthogonal to the gradient here, and that's here.

Non-quadratic Example

$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$



backtracking line search

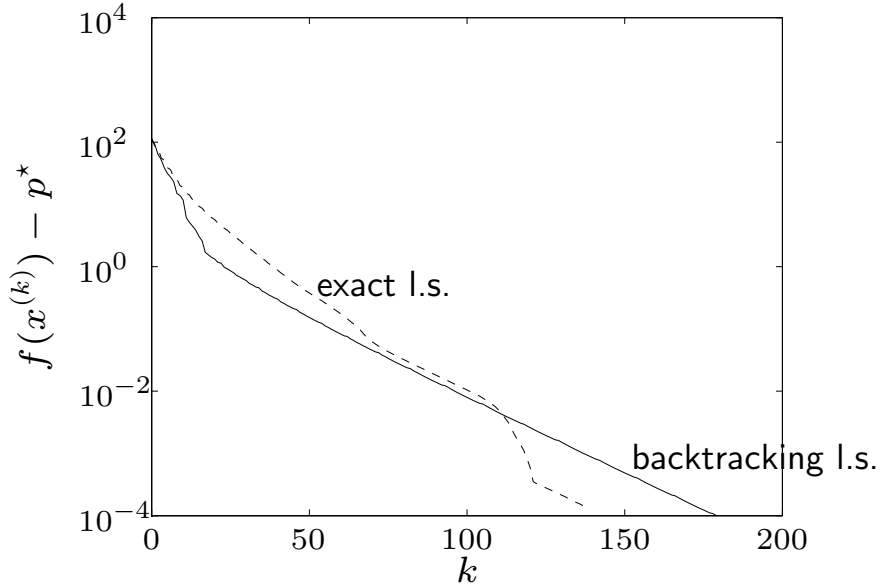


exact line search

Here's a non quadratic example and you can see here – the backtracking lines search worked quite, the exactly line search worked quite well, but just to show you a rough idea how this looks. Let's look at some bigger problems to see how this scales. Here's a problem that looks like that, and you can look at it, and these are the two things.

A problem in R^{100}

$$f(x) = c^T x - \sum_{i=1}^{500} \log(b_i - a_i^T x)$$



'linear' convergence, $i.e.$, a straight line on a semilog plot

So the first thing here is you see that there's not a huge difference between the backtracking and the exact line search. The other thing is you can see it's taking, you know, it's taking hundreds of iterations. it's getting a reasonable accuracy. And you can estimate c from this. And the other thing you see is that's like a line. And a line here, since that's a log access here, means that you're reducing your error, or you sub optimality, very roughly speaking, a fix fraction each step. Let's say you do 100 steps and you reduce it by a factor of 10^4, so a 10^4 by, you have to take logs or something like that to get the right number in 100 steps? You can figure it out.

Steepest descent method

normalized steepest descent direction (at x, for norm $\|\cdot\|$):

$$\Delta x_{\text{nsd}} = \text{argmin}\{\nabla f(x)^T v \mid \|v\| = 1\}$$

interpretation: for small v, $f(x+v) \approx f(x) + \nabla f(x)^T v$;
direction Δx_{nsd} is unit-norm step with most negative directional derivative

(unnormalized) steepest descent direction

$$\Delta x_{\text{sd}} = \|\nabla f(x)\|_* \Delta x_{\text{nsd}}$$

satisfies $\nabla f(x)^T \Delta x_{\text{sd}} = -\|\nabla f(x)\|_*^2$

steepest descent method

- general descent method with $\Delta x = \Delta x_{\text{sd}}$
- convergence properties similar to gradient descent

The steepest descent method says, we have a norm and we want to minimize, we want to find the steepest direction or descent. So what you do is you simply to minimize the $\nabla f(x)^T v$. So v is the direction. That's what $\nabla f(x)^T = 0$. But obviously, this has to be normalized so $\|v\| = 1$. So this would be the search direction. Now, it's usually un normalized. So the steepest descent is just basically so you know what it is and all that. But it's not going to be key because we're going to focus on something else.

Steepest descent method

normalized steepest descent direction (at x, for norm $\|\cdot\|$):

$$\Delta x_{\text{nsd}} = \text{argmin}\{\nabla f(x)^T v \mid \|v\| = 1\}$$

interpretation: for small v, $f(x+v) \approx f(x) + \nabla f(x)^T v$;
direction Δx_{nsd} is unit-norm step with most negative directional derivative

(unnormalized) steepest descent direction

$$\Delta x_{\text{sd}} = \|\nabla f(x)\|_* \Delta x_{\text{nsd}}$$

satisfies $\nabla f(x)^T \Delta x_{\text{sd}} = -\|\nabla f(x)\|_*^2$

steepest descent method

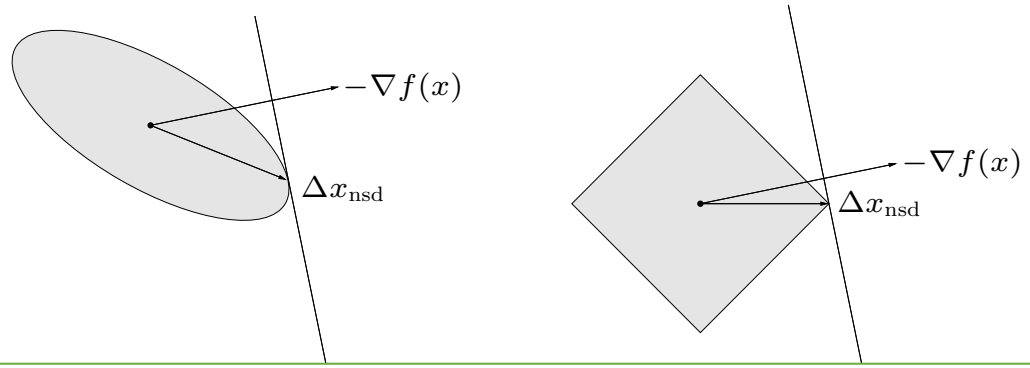
- general descent method with $\Delta x = \Delta x_{\text{sd}}$
- convergence properties similar to gradient descent

Now, the interpretation here is that in this norm you get this tells you sort of the direction where for a unit step in that direction the first order prediction of decrease is maximized. And it depends on the norm. You would imagine if you're on some sort of surface and you asked someone which way is the fast way to go downhill. Everyone would point in the same direction. It depends on the norm.
Now, the convergence is something like the gradient descent method. In fact, gradient descent is steepest descent in the Euclidean norm.

Examples

- Euclidean norm: $\Delta x_{\text{sd}} = -\nabla f(x)$
- quadratic norm $\|x\|_P = (x^T P x)^{1/2}$ ($P \in \mathbf{S}^n_{++}$): $\Delta x_{\text{sd}} = -P^{-1}\nabla f(x)$
- ℓ_1-norm: $\Delta x_{\text{sd}} = -(\partial f(x)/\partial x_i)e_i$, where $|\partial f(x)/\partial x_i| = \|\nabla f(x)\|_\infty$

unit balls and normalized steepest descent directions for a quadratic norm and the ℓ_1-norm:

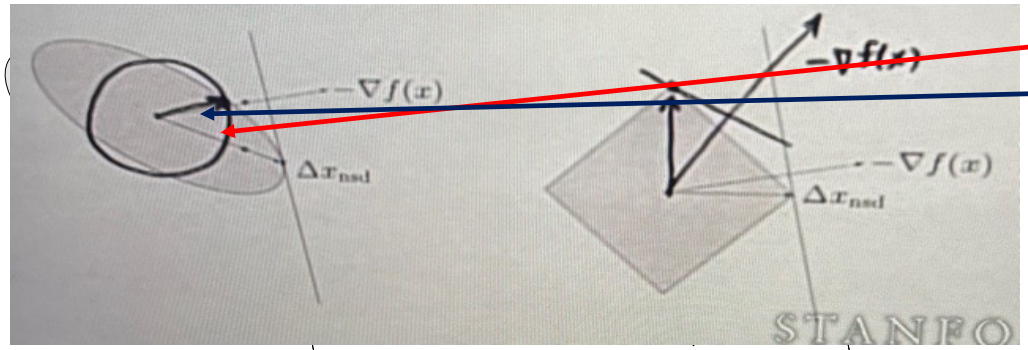


So let's look at an example here. And once you visualize what steepest descent is you'll realize why it is when you ask two people at the same place what the fastest way to go downhill, they can point in different directions. By the way, they can't point – they have to point in directions that – They can only differ by less than 180 degrees, that kind of obvious. If they're pointing completely in opposite directions, there's some trouble. Or it's very good or you have to minimize. But they can point different directions.

Examples

- Euclidean norm: $\Delta x_{\text{sd}} = -\nabla f(x)$
- quadratic norm $\|x\|_P = (x^T P x)^{1/2}$ ($P \in \mathbf{S}^n_{++}$): $\Delta x_{\text{sd}} = -P^{-1}\nabla f(x)$
- ℓ_1-norm: $\Delta x_{\text{sd}} = -(\partial f(x)/\partial x_i)e_i$, where $|\partial f(x)/\partial x_i| = \|\nabla f(x)\|_\infty$

unit balls and normalized steepest descent directions for a quadratic norm and the ℓ_1-norm:



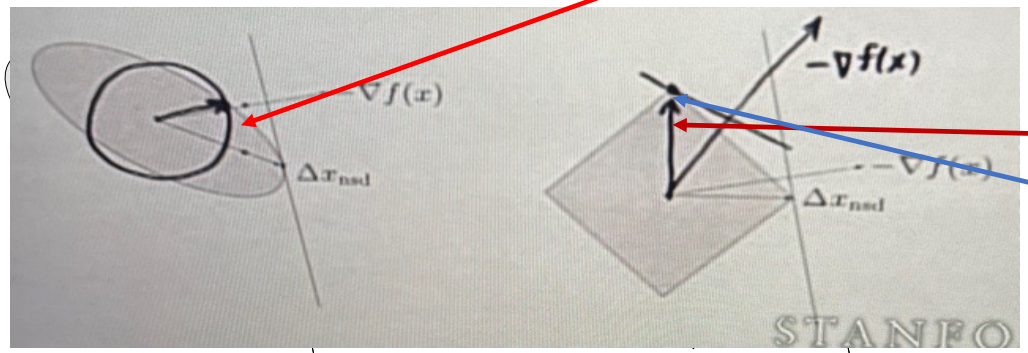
If I take just a quadratic norm then it turns out the steepest descent direction is $-P^{-1}$ descent. We can work out what this is. You're here, first you can do the following. Let's, workout what the steepest descent is in Euclidean norm.So use the lineralized model, that's a negative gradient. And you say I'm going to move ∇f(x) in this ball.. It's says go in that direction., which is the negative gradient.

So you can see, these are, obviously, not the same direction, and you've twisted the thing. Now, in fact, you can't twist it more than 180 degrees because that's a positive definite matrix and you multiple a gradient of one vector by a positive definite matrix, you don't rotate it more than 180 degrees.

Examples

- Euclidean norm: $\Delta x_{\text{sd}} = -\nabla f(x)$
- quadratic norm $\|x\|_P = (x^T P x)^{1/2}$ $(P \in \mathbf{S}^n_{++})$: $\Delta x_{\text{sd}} = -P^{-1}\nabla f(x)$
- ℓ_1-norm: $\Delta x_{\text{sd}} = -(\partial f(x)/\partial x_i)e_i$, where $|\partial f(x)/\partial x_i| = \|\nabla f(x)\|_\infty$

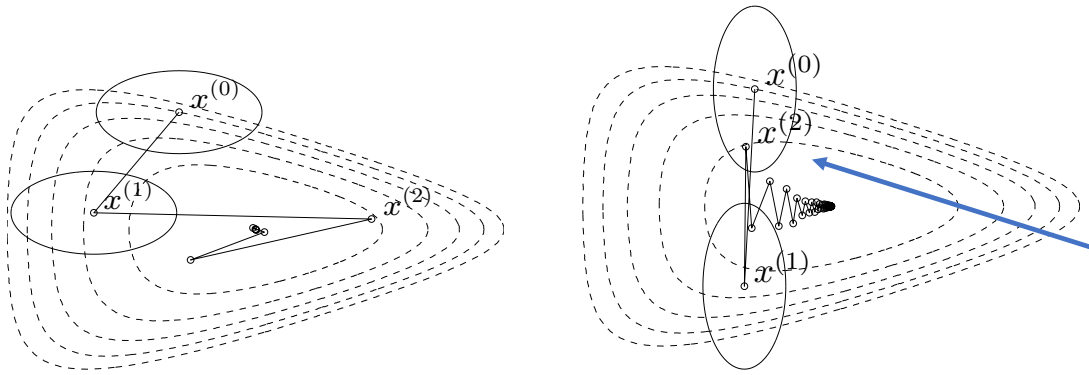
unit balls and normalized steepest descent directions for a quadratic norm and the ℓ_1-norm:



Now, in fact, you can't twist it more than 180 degrees because that's a positive definite matrix and you multiple a gradient of one vector by a positive definite matrix, you don't rotate it more than 180 degrees. In fact, that's 90. That is exactly what a positive definite matrix is. if you want to do steepest descent in the L_1 norm because you'd draw a ball like this and take the gradient like this

Imagining that I had drawn that arrow straight, what would be the steepest descent direction in L_1? you go as far as you can in the direction and so it would be this point here, and that would be your direction. And you can kind of make a guess about steepest descent in L_1. Go ahead and make a guess as to what it –it's always a long unit vector. you're updating one component of the variable. So the deepest descent in L_1 at each step you optimize over one component of the variable.

Choice of norm for steepest descent



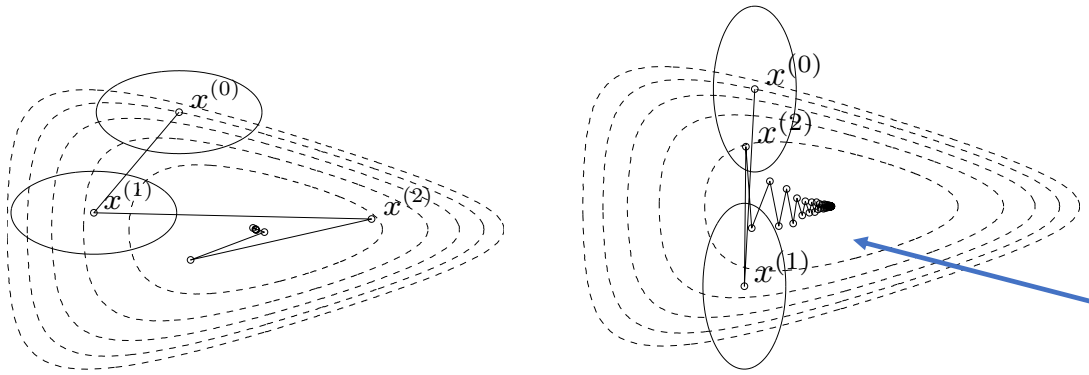
- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent interpretation of steepest descent with quadratic norm $\|\cdot\|_P$: gradient descent after change of variables $\bar{x} = P^{1/2}x$

shows choice of P has strong effect on speed of convergence

It has to do with the choice of the norm for steepest descent and so this will illustrate this here. These are sublevels of some function here, some function we just looked at, like at the **left side of the foil**.
And the point is if you look at his function what you see, but this is the idea. The function is sort of elongated, it's not spherical. It's kind of fatter than it is tall. That's the main thing here.

If you chose a norm that is sort of aligned with the gross geometry of the sublevel set, you end up making very fast progress towards the center, because you end up going in the right direction. If you take a norm, which is whose unit ball is sort of not aligned the same way the gross geometry of the sublevel set is, it amplifies this effect of osculation. So the rough idea is something like this.

Choice of norm for steepest descent



- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent interpretation of steepest descent with quadratic norm $\|\cdot\|_P$: gradient descent after change of variables $\bar{x} = P^{1/2}x$

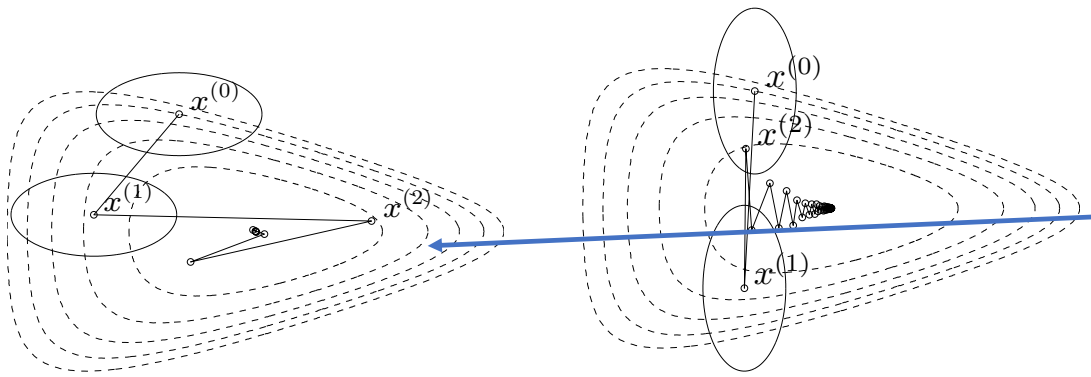
shows choice of P has strong effect on speed of convergence

Actually, if it were perfectly spherical it would work flawlessly. It would say that's the negative gradient, it would go in that direction. Negative gradient would point, if not to the center, to the optimizer, it would point very near it.

You'd minimize it and you'd end up over here. So the gradient works perfectly when your function is, and when it's isotropic or something like that when it kind of extends, when it looks about the same in all directions.

It is kind of spherical or in all directions it looks about the same. Gradient descent works horribly when the sublevel sets look like this. That's exactly when you get this sort of story.

Choice of norm for steepest descent



- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent interpretation of steepest descent with quadratic norm $\|\cdot\|_P$: gradient descent after change of variables $\bar{x} = P^{1/2}x$

shows choice of P has strong effect on speed of convergence

Now, when you do steepest descent in a norm, it's the same as changing concordance and then applying gradient descent. So what you want to do is you want to change concordance to make things that look like this look like that.

So the whole key, in fact, is choosing the norm. That's the key to everything. So the norm you do steepest descent in is important.

So for example, suppose you know something about the function you're going to minimize. Suppose you know you have samples of it or something like that. You could like fit a quadratic.

Suppose you already evaluated the function like a couple hundred points, you can fit a quadratic to it, or do whatever you like.

Newton step

$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

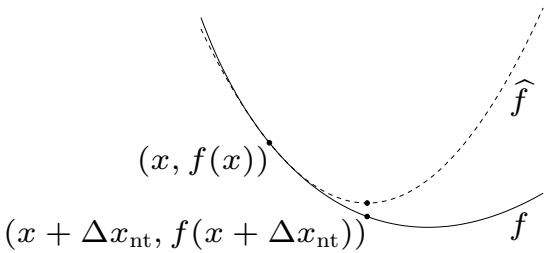
interpretations

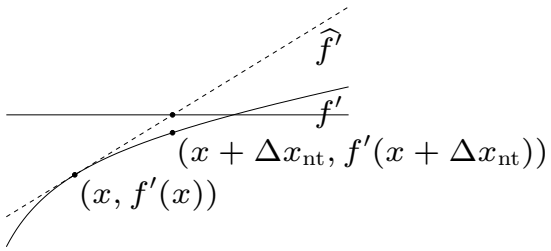
- $x + \Delta x_{\mathrm{nt}}$ minimizes second order approximation

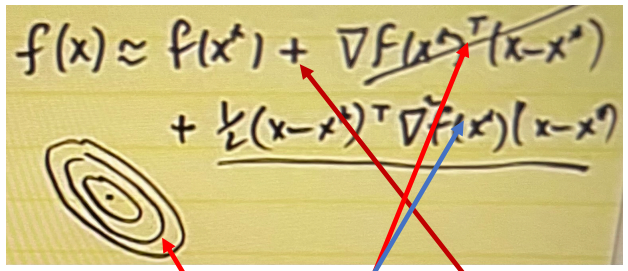
$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$







What does the function look like near the minimizer? What's an approximation of f(x) near the minimizer? So, it's equal to $f(x^*)$ + $\nabla f(x^*)^T (x - x^*)$. That's zero. So we go to our next string, which is $1/2(x - x^*)^T$, you know, the Hessian $\nabla^2 f(x)(x - x^*)$. So that's our next string.

Now, near the optimal point, it looks quadratic. And so what does the sublevels of f look like, approximately, near the minimum? And they're ellipsoids determined by the Hessian. So they look like this. they're ellipsoids. That's exactly what we want

Newton step

$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

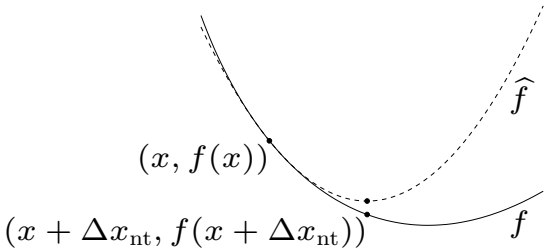
interpretations

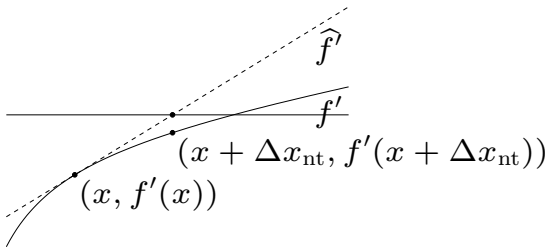
- $x + \Delta x_{\mathrm{nt}}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$





So let's look at the Newton step. So the Newton step is this. In fact, a – it's most way interpreted, it's Hessian, $-\nabla^2 f(x)^{-1}$ times the gradient, $\nabla f(x)$. That's all.
What the Newton step is this, you take function and you develop a second order approximation, that's this.

If you minimize this, you just take the gradient with respect to v equal zero and you'll get v equals the Newton step here. So this minimizes that. Newton's method will use the Newton step.

Newton step

$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

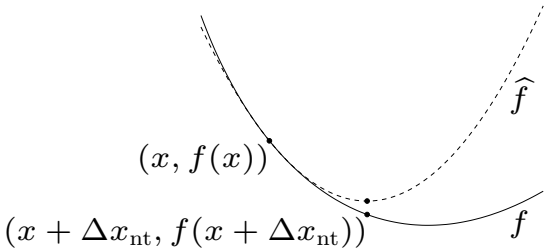
interpretations

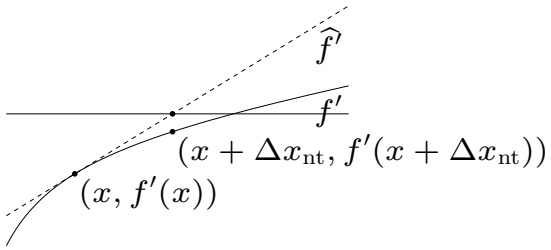
- $x + \Delta x_{\mathrm{nt}}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$





And it basically says, you take a function at each point, you develop, not a first order approximation, but a second order, and you minimize that, and then you update. So that's this. Another way to say it is this, you want to solve, if you're at the point x you would like to find a v for which the gradient of f, evaluated x+v, that's where you're going to step. That should be zero.

But the problem is the gradient nonlinear function, you don't know what gradient of x + v is, this nonlinear function is about equal with this linearized approximation, which is the gradient where you are plus the derivative of the gradient, but that's the Hessian. And then you set that equal to zero. And if you solve this equation, you get v equals minus Hessian inverse gradient. You get that.

Newton step

$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

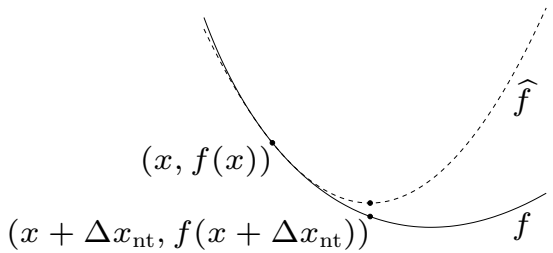
interpretations

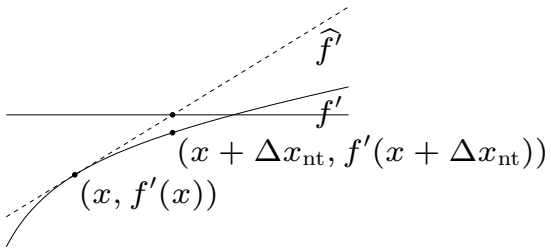
- $x + \Delta x_{\mathrm{nt}}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$





And the two pictures correspond to something like this down here. So in the first case, here's f(x). Notice that you know, at that point already you can see that $\hat{f}$ and f are not the same. If that's our Newton step and we carry out this again, we fit a quadratic to this point and do it again. It's going to be much better this time. And when I start getting near this bottom point, these quadratics are going to good approximations. Another way to see it is this, here's f' and we want to solve f'- 0 , that's our optimally. So it's convex, which means f' is increasing. So f' is increasing. That's F prime. I'm here and evaluate the derivative of prime. That's the second derivative. And I take the zero of that. what happens on the next step?

Newton step

$$\Delta x_{\mathrm{nt}} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

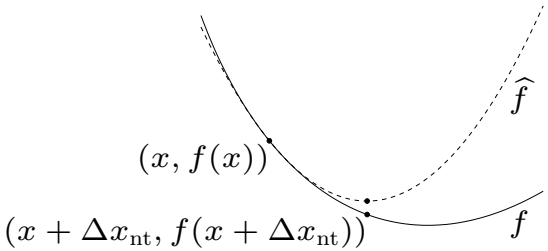
interpretations

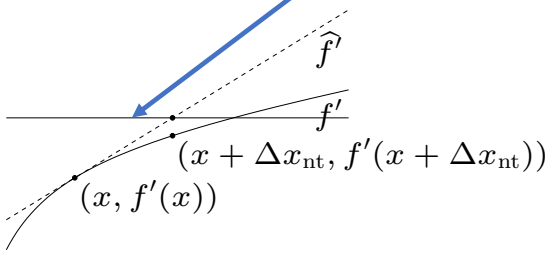
- $x + \Delta x_{\mathrm{nt}}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\mathrm{nt}}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$



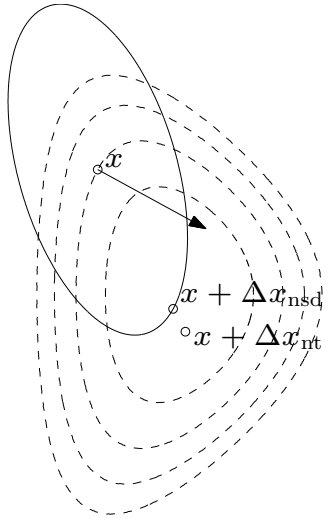


You're going to put a line through here. This is not linear here or F line, but it's very close and the next iterate is going to be somewhere in here. When you get in there, the iterates are going to be unbelievably accurate and you're going to get very fast conversions. So this is sort of the picture. it's a variable metric method. So metric here refers to how you describe distances. So a variable metric method is a method that varies. It's essentially, the norm being used in steepest descent at each step and it adjusts it to the geometry of the problem instance. That's what a variable method metric is.

Newton step

- Δx_{nt} is steepest descent direction at x in local Hessian norm

$$\|u\|_{\nabla^2 f(x)} = \left(u^T \nabla^2 f(x) u\right)^{1/2}$$



dashed lines are contour lines of f; ellipse is $\{x + v \mid v^T \nabla^2 f(x) v = 1\}$

arrow shows $-\nabla f(x)$

So you canthink of the third interpretation is this, is that the Newton step is steepest descent direction in the local Hessian norm. So that's what it is. That's the picture

Newton decrement

$$\lambda(x) = \left(\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x)\right)^{1/2}$$

a measure of the proximity of x to $x^\star$

properties

- gives an estimate of $f(x) - p^\star$, using quadratic approximation $\widehat{f}$:

$$f(x) - \inf_y \widehat{f}(y) = \frac{1}{2}\lambda(x)^2$$

- equal to the norm of the Newton step in the quadratic Hessian norm

$$\lambda(x) = \left(\Delta x_{\mathrm{nt}}^T \nabla^2 f(x) \Delta x_{\mathrm{nt}}\right)^{1/2}$$

- directional derivative in the Newton direction: $\nabla f(x)^T \Delta x_{\mathrm{nt}} = -\lambda(x)^2$
- affine invariant (unlike $\|\nabla f(x)\|_2$)

Let's look at the Newton decrement. Now, the Newton decrement is Newton decrement square, it's defined as this. It's the norm, λ is the norm of the gradient but in the metric induced by the Hessian. So that's what this is. That's that.

And it's a measure of the proximity of the x to x^*. And it's a good one and we'll see some very good properties it. And it allows you to make an approximation of how suboptimal you are. So one half λ of squared is an approximation of how suboptimal you are.

And it's the norm of the Newton step and the quadratic Hessian. It's also the direction of the derivative in the Newton direction so it turns out to calculate the line search.

Newton's method

---

given a starting point $x \in \mathbf{dom}\, f$, tolerance $\epsilon > 0$.
repeat

1. *Compute the Newton step and decrement.*
   $\Delta x_{\rm nt} := -\nabla^2 f(x)^{-1} \nabla f(x)$; $\quad \lambda^2 := \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x)$.
2. *Stopping criterion.* **quit** if $\lambda^2/2 \leq \epsilon$.
3. *Line search.* Choose step size t by backtracking line search.
4. *Update.* $x := x + t\Delta x_{\rm nt}$.

---

affine invariant, *i.e.*, independent of linear changes of coordinates:

Newton iterates for $\tilde{f}(y) = f(Ty)$ with starting point $y^{(0)} = T^{-1}x^{(0)}$ are

$$y^{(k)} = T^{-1}x^{(k)}$$

So Newton's method looks like this. You're given a starting point and a tolerance. You compute the Newton step in decrement. You know, if ∇^2 is small enough you quit. Otherwise, you do a line search and then you update. That's Newton's method.

the best zero order description of why it works is f line invariance. So let me explain what that is.

Min f(x) and let T(y)=x, where T in a nonsingular matrix, and I could, just as well, minimize f(Ty) over y. what happens is in a gradient method when you take, if you call this thing $\tilde{f}$(y)= f(Ty), then $\nabla \tilde{f}$(y)= $T^T \nabla$f(Ty).
In fact, that's the message we just looked at, if you change the metric the gradient method changes.