CSED490Y: Optimization for Machine Learning

Week 13: Second order methods

Namhoon Lee

POSTECH

Spring 2022

Admin

Online lectures:

- This week: May 16 (Mon) and May 18 (Wed)
- Next week: May 25 (Wed)

Midway group presentation:

- Schedule:
  - ~~May 11: groups 6, 4, 3, 7, 10~~
  - May 18: groups 1, 5, 9, 11, 8
  - May 25: groups 2, 12, 13
- Upload your slides on PLMS by 11am of the presentation day.

Newton's method

For unconstrained optimization problem

$$\min_x f(x)$$

gradient descent (GD) can be interpreted as minimizing a quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^\top (y - x) + \frac{1}{2\eta} \|y - x\|_2^2 \ .$$

Finding the minimum of it yields the GD update rule

$$x_{t+1} = x_t - \eta \nabla f(x_t) \ .$$

Newton's method

The idea is to approximate f better with the second order Taylor approximation

$$f(y) \approx f(x) + \nabla f(x)^\top (y-x) + \frac{1}{2}(y-x)^\top \nabla^2 f(x)(y-x) \ ,$$

and minimizing it gives Newton's method:

$$x_{t+1} = x_t - \left(\nabla^2 f(x_t)\right)^{-1} \nabla f(x_t) \ ,$$

where the Hessian $\nabla^2 f(x_t)$ is positive definite and thus invertible.

It turns out that for smooth and strongly convex function f, Newton's method achieves quadratic convergence rate of $\mathcal{O}(\log\log(1/\epsilon))$. $\longleftrightarrow$ (GD) $O(\lg(1/\epsilon))$

(superlinear) # iterations required to achieve ϵ-accuracy.

Newton's method

(finding an optimal η, η^*)

In practice

Start with some large step size and decrease it until it satisfies "some" condition.

Two phase Newton's method:

x_t

Phase 1: damped Newton (when not close to x^*)

$$x_{t+1} = x_t - \eta\left(\nabla^2 f(x_t)\right)^{-1}\nabla f(x_t)$$

where the step size η is found by a back-tracking line search (*e.g.*, Armijo condition).

Phase 2: undamped Newton (when close to x^*)

$$x_{t+1} = x_t - \left(\nabla^2 f(x_t)\right)^{-1}\nabla f(x_t)$$

which achieves quadratic convergence rate.

Newton's method

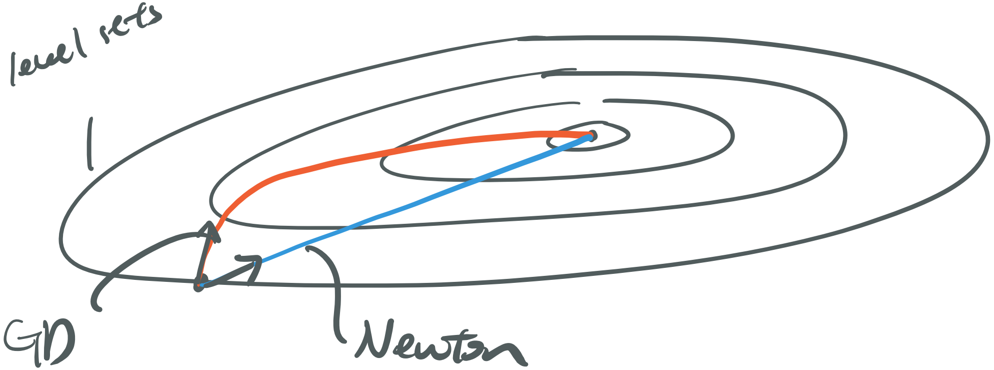
$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

(say η^*)

GD vs Newton's method – illustration

If f quadratic, x near x^*

Newton's method converges in just 1 step.



Why would we ever use GD if Newton is so fast?

Newton's method

β α

f – smooth & s.c

$\kappa = \beta/\alpha$

GD vs Newton's method – computations

Gradient descent:

$$x_{t+1} = x_t - \eta \nabla f(x_t)$$

iterations

- The total number of computations: $\mathcal{O}(n\kappa \log(1/\epsilon))$.

Newton's method:

$$x_{t+1} = x_t - \left(\nabla^2 f(x_t)\right)^{-1} \nabla f(x_t)$$

at t

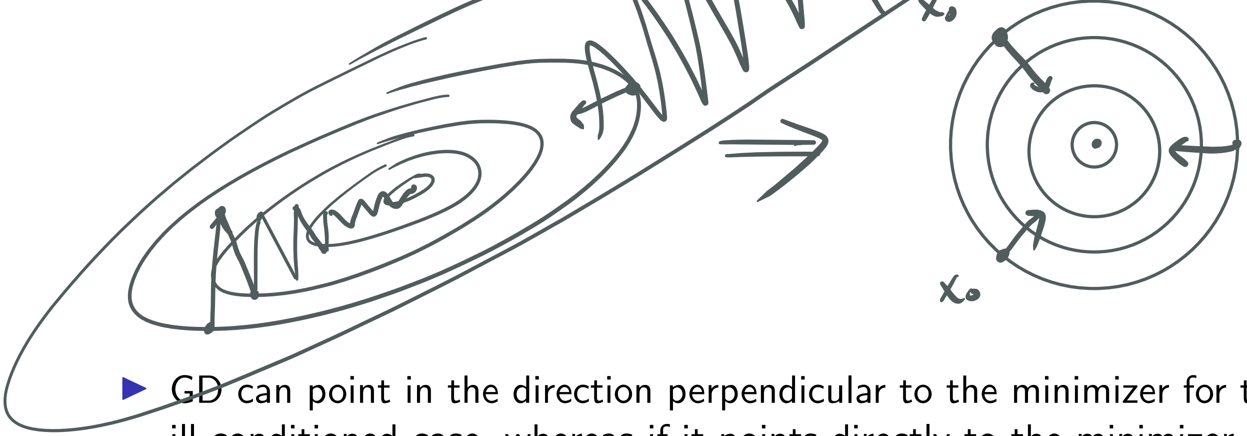
- The total number of computations: $\mathcal{O}(n^3 \log\log(1/\epsilon))$.

$1/\epsilon$

I.e., GD if n is large (or moderate ϵ) and Newton if ϵ is tiny (or smal n).

Newton's method

Now let's see GD for an ill-conditioned quadratic and compare it to the perfectly-conditioned case:



- GD can point in the direction perpendicular to the minimizer for the ill-conditioned case, whereas if it points directly to the minimizer for the perfectly-conditioned case.
- This affects GD performance, and in fact, it is reflected on the convergence rate.

Newton's method

$x_{t+1} = x_t - (\nabla^2 f(x_t))^{-1} \nabla f(x_t)$

We can interpret Newton's method as a problem of conditioning.

The idea is to rescale space using affine transformation and then perform GD.

Rescale $x = Ay$ $\iff$ $y = A^{-1}x$

$\min_y g(y) \iff \min_x f(x)$

Define $g(y) = f(x) = f(Ay)$

$g(y^*) = f(x^*)$

- Minimizing $g(y)$ and $f(x)$ is equivalent, *i.e.*, $g(y^*) = f(x^*)$.

GD after affine transformation: example

$$f(x) = \frac{1}{2}x^\top Qx, \quad Q = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}$$

$\Rightarrow \lambda_{max} = 100$

$\lambda_{min} = 1$

$\kappa = 100$

Let $A = \begin{bmatrix} 1/10 & 0 \\ 0 & 1 \end{bmatrix}$ and $x = Ay$. Then

$$\underline{g(y)} = f(Ay) = \frac{1}{2}y^\top A^\top QAy = \frac{1}{2}\,y^T\begin{bmatrix} 1/10 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1/10 & 0 \\ 0 & 1 \end{bmatrix}y = \frac{1}{2}y^\top y$$

$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \Rightarrow \kappa = 1$

Newton's method

$g(y) = \frac{1}{2} y^T y$

How does GD work for g and f?

GD on g:

$$y_1 = y_0 - \eta \nabla (\frac{1}{2} y_0^\top y_0) = y_0 - y_0 = 0$$

- GD finds the minimum in just 1 step regardless of $\cancel{x_0}$ y_0.

GD on f:

$$x_1 = x_0 - \eta \nabla (\frac{1}{2} x_0^\top \cancel{1} x_0) = x_0 - \frac{1}{100} (Q x_0) = x_0 - \begin{bmatrix} 1 & 0 \\ 0 & 1/100 \end{bmatrix} x_0 \neq 0$$

Q

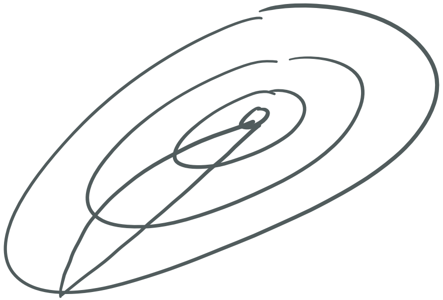
- For say $x_0 = [0, 1]^\top$, x_1 is not the minimum; GD needs to run many more steps.

$x_1 = \begin{pmatrix} 0 \\ 0.99 \end{pmatrix}$

Newton's method

GD takes different trajectories on $f(x)$ and $g(y)$:

$$x_0 = y_0$$
$$x_1 \quad y_1$$
$$x_2 \quad y_2$$
$$\vdots$$
$$x^* = y^*$$



- GD is not invariant to affine transformation.

Newton's method

We have seen that affine transformation can speed up GD by rescaling space.

What is a good transformation?

- Ideally affine transformation A such that $x = Ay$ and for $g(y) = f(x)$ the local condition number κ becomes 1, which requires $\nabla^2 g(y) = I$.

Hence the following has to satisfy

$$\nabla^2 g(y) = A^\top \nabla^2 f(Ay) A = I \ .$$

- $A = \left(\nabla^2 f(Ay)\right)^{-1/2}$ will satisfy.

Newton's method

Newton's method can be interpreted in this way, *i.e.*, perform GD with the best affine transformation at every step.

$$
\begin{aligned}
y_{t+1} &= y_t - \left(\nabla^2 g(y_t)\right)^{-1} \nabla g(y_t) \\
&= y_t - \left(A^\top \nabla^2 f(Ay_t) A\right)^{-1} A^\top \nabla f(Ay_t) \\
&= y_t - A^{-1} \left(\nabla^2 f(Ay_t)\right)^{-1} \nabla f(Ay_t)
\end{aligned}
$$

Muliplying A both sides gives

$$
Ay_{t+1} = Ay_t - (\nabla^2 f(Ay_t))^{-1} \nabla f(Ay_t)
$$

or

$$
x_{t+1} = x_t - (\nabla^2 f(x_t))^{-1} \nabla f(x_t)
$$

Convergence analysis

$\|\nabla^2 f(x) - \nabla^2 f(y)\|_2 \le L\|x-y\|_2$

matrix 2-norm

Theorem (as in Nocedal and Wright 1999)

Suppose that f is twice differentiable and that the Hessian $\nabla^2 f(x)$ is Lipschitz continuous in a neighborhood of a solution x^* at which the sufficient condition are satisfied. Consider the iteration $x_{k+1} = x_k + p_k$, where p_k is given by $-\nabla^2 f(x_k)^{-1}\nabla f(x_k)$. Then

1. *if the starting point x_0 is sufficiently close to x^*, the sequence of iterates converges to x^*;*
2. *the rate of convergence of $\{x_k\}$ is quadratic; and*
3. *the sequence of gradient norms $\{\|\nabla f(x_k)\|\}$ converges quadratically to zero.*

Convergence analysis

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

Proof (of 1 and 2). From the definition of the Newton step and the optimality condition $\nabla f(x^*) = 0$ we have that

$$\underline{x_k + p_k} - x^* = x_k - x^* - \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)$$
$$= \left(\nabla^2 f(x_k)\right)^{-1} \left(\nabla^2 f(x_k)(x_k - x^*) - (\nabla f(x_k) - \nabla f(x^*)\right) .$$

We are going to take $\|\cdot\|$ both sides.

Since the Taylor's Theorem (or the fundamental theorem of calculus) tells us that

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x_k + t(x^* - x_k))(x_k - x^*) dt \ ,$$

Convergence analysis

we have $\left\|\nabla^2 f(x_k)(x_k - x^*) - (\nabla f(x_k) - \nabla f(x^*)\right\|$

$$
\begin{aligned}
&= \left\| \int_0^1 \left(\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k))\right)(x_k - x^*) dt \right\| \\
&\leq \int_0^1 \left\| \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x^* - x_k)) \right\| \|x_k - x^*\| dt \\
&\leq \|x_k - x^*\|^2 \int_0^1 Ltdt = \frac{1}{2} L \|x_k - x^*\|^2 \ ,
\end{aligned}
$$

where L is the Lipschitz constant for $\nabla^2 f(x)$ for x near x^*.

Also, since $\nabla^2 f(x^*)$ is nonsingular, there is a radius $r > 0$ such that $\|(\nabla^2 f(x_k))^{-1}\| \leq 2\|\nabla^2 f(x^*)^{-1}\|$ for all x_k with $\|x_k - x^*\| \leq r$.

Convergence analysis

Now putting all together gives

$$\|x_k + p_k - x^*\| \leq L\|\nabla^2 f(x^*)^{-1}\|\|x_k - x^*\|^2 = \tilde{L}\|x_k - x^*\|^2 \ ,$$

where $\tilde{L} = L\|\nabla^2 f(x^*)^{-1}\|$.

Choosing x_0 so that $\|x_0 - x^*\| \leq \min(r, 1/(2\tilde{L}))$ (*i.e.*, initial point is close to the minimum), we can use this to deduce that the sequence converges to x^* quadratically.

Convergence analysis

(minimizing the second order Taylor

Proof (of 3). By using the relations $x_{k+1} - x_k = p_k$ and $\nabla f(x_k) + \nabla^2 f(x_k) p_k = 0$, we obtain that

$$
\begin{aligned}
\|\nabla f(x_{k+1})\| &= \|\nabla f(x_{k+1}) - \nabla f(x_k) - \nabla^2 f(x_k) p_k\| \\
&= \left\| \int_0^1 \nabla^2 f(x_k + t p_k)(x_{k+1} - x_k) dt - \nabla^2 f(x_k) p_k \right\| \\
&\le \int_0^1 \|\nabla^2 f(x_k + t p_k) - \nabla^2 f(x_k)\| \|p_k\| \\
&\le \frac{1}{2} L \|p_k\|^2 \\
&\le \frac{1}{2} L \|(\nabla^2 f(x_k))^{-1}\|^2 \|\nabla f_k\|^2 \\
&\le 2L \|(\nabla^2 f(x^*))^{-1}\|^2 \|\nabla f_k\|^2,
\end{aligned}
$$

proving that the gradient norms converge to zero quadratically.

Quasi Newton methods

Newton direction

Newton's method is fast, but very expensive.

Inverse

- Compute Hessian $\nabla^2 f(x)$
- Solve the system $\nabla f(x) + \nabla^2 f(x) d = 0$

Quasi-Newton methods take the following form

$$x_{t+1} = x_t - \eta B_t^{-1} \nabla f(x_t)$$

where B_t is some approximation of the Hessian.

- The idea is to attempt to replace the Hessian with some approximation that is less expensive but more useful than simple identity (*e.g.*, diagonal Hessian).
- B_t^{-1} is updated iteratively.

Quasi Newton methods

Quasi Newton methods compute B_t iteratively.

- The idea is that since B_{t-1} already contains some information about the Hessian, make some update to form B_t.

Quasi Newton methods differ by how to compute B_t.

- SR1, BFGS, DFP, etc.

The key idea behind quasi Newton methods is to match the gradients of f at the last two iterations, *i.e.*,

$$\nabla f(x_{t+1}) = \nabla f(x_t) + B_{t+1}(x_{t+1} - x_t)$$

- By rewriting it $B_{t+1}s_t = y_k$ with $s_t = x_{t+1} - x_t$ and $y_t = \nabla f(x_{t+1}) - \nabla f(x_t)\cancel{)}$, it is called the secant equation.

Thank you

Any questions?

Credits

A lot of material in this course is borrowed or derived from the following:

- Numerical Optimization, Jorge Nocedal and Stephen J. Wright.
- Convex Optimization, Stephen Boyd and Lieven Vandenberghe.
- Convex Optimization, Ryan Tibshirani.
- Optimization for Machine Learning, Martin Jaggi and Nicolas Flammarion.
- Optimization Algorithms, Constantine Caramanis.
- Advanced Machine Learning, Mark Schmidt.

References I

Nocedal, Jorge and Stephen J Wright (1999). *Numerical optimization*. Springer.