IE 539: Convex Optimization **KAIST, Fall 2022**
Lecture #24: Convergence of Newton's method November 29, 2022
Lecturer: Dabeen Lee

1 Outline

In this lecture, we study

- Convergence of Newton's method.

2 Newton's method

2.1 Affine transformation

Let us get back to the quadratic minimization example. Let us take

$$Q = \begin{bmatrix} \sqrt{M} & 0 \\ 0 & \sqrt{m} \end{bmatrix}$$

and consider the linear transformation defined by $y = Qx$. Then

$$f(x) = \frac{1}{2}x^\top \begin{bmatrix} M & 0 \\ 0 & m \end{bmatrix} x = \frac{1}{2}x^\top Q^\top Qx = \frac{1}{2}y^\top y.$$

Therefore, after the linear transformation, the function becomes 1-smooth and 1-strongly convex in the ℓ_2 norm. Moreover, $g(y) = (1/2)y^\top y$ satisfies

$$\nabla^2 g(y) = I.$$

Hence, gradient descent converges after $O(\log(1/\epsilon))$ iterations. This implies that gradient descent can be improved by taking a proper affine transformation.

Newton's method can be interpreted as finding an affine transformation that is locally optimal at each iteration. To be more precise, at x_t, we want to find an affine transformation $y = Q_t x$ so that $g(y) = f(Q_t^{-1}y) = f(x)$ and $\nabla^2 g(y_t) = I$. Note that

$$\nabla^2 g(y_t) = (Q_t^{-1})^\top \nabla^2 f(Q_t^{-1}y_t)Q_t^{-1}.$$

Hence, $\nabla^2 g(y_t) = I$ if and only if

$$Q_t = (\nabla^2 f(Q_t^{-1}y_t))^{-1/2} = (\nabla^2 f(x_t))^{-1/2}.$$

In this case, g becomes 1-smooth and 1-strongly convex (we will see this later), and thus gradient descent on g proceeds with

$$y_{t+1} = y_t - \nabla g(y_t).$$

Here, we have $y_t = Q_t x_t$ and $\nabla g(y_t) = Q_t^{-1}\nabla f(Q_t^{-1}y_t) = Q_t^{-1}\nabla f(x_t)$. This implies that

$$y_{t+1} = Q_t x_t - Q_t^{-1}\nabla f(x_t).$$

Multiplying both sides by Q_t^{-1}, it follows that

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

which is prrecisely the update rule of Newton's method.

2.2 Netwon's method with backtracking line search

We have built up intuitions for the update rule of Newton's method and the corresponding descent direction. However, the method as it is does not necessarily converge. Let us consider the following example. Let us consider

$$\text{minimize}_{x\in\mathbb{R}} \quad f(x) = \sqrt{1+x^2}.$$

Note that

$$f'(x) = \frac{x}{\sqrt{1+x^2}},$$
$$f''(x) = \frac{1}{(1+x^2)^{3/2}}.$$

Hence, Newton's method runs with

$$x_{t+1} = x_t - f''(x_t)^{-1}f'(x_t) = x_t - x_t(1+x_t^2) = -x_t^3.$$

Then if $|x_1| \geq 1$, the method diverges, while it converges when $|x_1| < 1$.

To remedy this, we combine Newton's method and backtracking line search.

Algorithm 1 Newton's method with backtracking line search

Initialize x_1.
for $t = 1, \ldots, T-1$ **do**
  Compute $d_t = -\nabla^2 f(x_t)^{-1}\nabla f(x_t)$.
  Compute a step size η_t by backtracking line search.
  Update $x_{t+1} = x_t - \eta_t d_t$.
end for
Return x_T.

What is backtracking line search for Newton's method?

1. Fix parameters $0 < \beta < 1$ and $0 < \alpha < \frac{1}{2}$.
2. Start with an initial step size $\eta = 1$.
3. Until the following condition is satisfied, we shrink $\eta \leftarrow \beta\eta$.

$$f(x+\eta d) < f(x) + \alpha\eta\nabla f(x)^\top d$$

   where $d = -\nabla^2 f(x)^{-1}\nabla f(x)$.

4. We take the final η.

In fact, it is proved that Newton's method runs with two phases. The first phase applies the backtracking line search and ends up with step sizes less than 1. For the second phase, backtracking line search returns step size $\eta = 1$, which means the sufficient descent condition is satisfied with $\eta = 1$ at each iteration of the second phase. For this reason, Newton's method is often referred to as the combination of "damped" Newton phase and "undamped" Newton phase where the damped and undamped versions are Newton's method with and without backtracking line search, respectively.

Algorithm 2 Undamped Newton's method

Initialize x_1.
for $t = 1, \ldots, T-1$ **do**
  Compute $d_t = -\nabla^2 f(x_t)^{-1} \nabla f(x_t)$.
  Update $x_{t+1} = x_t - d_t$.
end for
Return x_T.

2.3 Convergence of Newton's method

Suppose that the objective function f satisfies the following conditions.

- f is twice continuously differentiable.
- f is m-strongly convex in the ℓ_2 norm, i.e.,

$$\nabla^2 f(x) \succeq mI.$$

- f is M-smooth in the ℓ_2 norm, i.e.,

$$\nabla^2 f(x) \preceq MI.$$

- The Hessian of f is L-Lipschitz continuous in the ℓ_2 norm, i.e.,

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

What we can show about Algorithm 1 is what follows. There exist numbers δ and γ such that the following is satisfied.

1. If $\|\nabla f(x_t)\|_2 \geq \delta$, then

$$f(x_{t+1}) - f(x_t) \leq -\gamma.$$

2. If $\|\nabla f(x_t)\|_2 < \delta$, then the backtracking line search selects $\eta_t = 1$ and

$$\frac{L}{2m^2}\|\nabla f(x_{t+1})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x_t)\|_2\right)^2.$$

This implies that Newton's method consists of two phases. When $\|\nabla f(x_t)\|_2$ is large, the algorithm is in the damped phase where backtracking line search is used to find a step size. On the other hand, when $\|\nabla f(x_t)\|_2 < \delta$, the backtracking line search step sets $\eta_t = 1$, which means that the step is in the undamped phase. Moreover, we can argue that $\|\nabla f(x_{t+1})\|_2 < \delta$ and therefore the undamped phase continues.

Note that the moment of transition from the damped Newton phase to the undamped phase can be determined if we know the value of δ. However, the algorithm does not assume the knowledge of its precise value.

In fact, we can argue that the statements hold with

$$\delta = \min\{1,\ 3(1-2\alpha)\}\,\frac{m^2}{L} \quad \text{and} \quad \gamma = \alpha\beta\delta^2\frac{m}{M^2}.$$

To compute the value of δ, we need the values of the Lipschitz continuity parameter L and the strongly convexity parameter m. Assuming the knowledge of these parameters, one can artificially set the moment of transition from the damped phase to the undamped phase. We refer the reader convex optimization textbook of Boyd and Vandenberghe [BV04] for more details of the convergence result and proof.

Under 1, the objective value decreases by at least γ. Hence, we can bound the number of iterations under the damped phase. It is

$$\frac{f(x_1) - f(x^*)}{\gamma}$$

where x^* is a minimizer of f.

Suppose that $\|\nabla f(x_k)\|_2 < \delta$. Then the inequality under 2 holds, and thus

$$\|\nabla f(x_{k+1})\|_2 \leq \frac{L}{2m^2}\delta^2 = \frac{\delta}{2} \cdot \min\{1,\ 3(1-2\alpha)\} \leq \frac{\delta}{2}.$$

This implies that $\|\nabla f(x_t)\|_2 < \delta$ for any $t \geq k$. Moreover, for any $t \geq k$,

$$\frac{L}{2m^2}\|\nabla f(x_{t+1})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x_t)\|_2\right)^2 \leq \left(\frac{L}{2m^2}\|\nabla f(x_k)\|_2\right)^{2^{t+1-k}} \leq \left(\frac{1}{2}\right)^{2^{t+1-k}}$$

Therefore, it follows that

$$\|\nabla f(x_t)\|_2 \leq \frac{2m^2}{L}\left(\frac{1}{2}\right)^{2^{t-k}}.$$

By the m-strong convexity of f, we have

$$\begin{aligned} f(x^*) &\geq f(x_t) + \nabla f(x_t)^\top (x^* - x_t) + \frac{m}{2}\|x^* - x_t\|_2^2 \\ &\geq \min_y \left\{ f(x_t) + \nabla f(x_t)^\top (y - x_t) + \frac{m}{2}\|y - x_t\|_2^2 \right\} \\ &= f(x_t) - \frac{1}{2m}\|\nabla f(x_t)\|_2^2. \end{aligned}$$

Hence,

$$f(x_t) - f(x^*) \leq \frac{1}{2m}\|\nabla f(x_t)\|_2^2 \leq \frac{2m^3}{L^2}\left(\frac{1}{2}\right)^{2^{t-k+1}}.$$

In summary, the number of iterations to obtain an ϵ-optimal solution is bounded above by

$$\frac{f(x_1) - f(x^*)}{\gamma} + \log\log\left(\frac{2m^3}{L^2} \cdot \frac{1}{\epsilon}\right) \leq \alpha\beta(f(x_1) - f(x^*))\frac{L^2M^2}{\min\{1,\ 3(1-2\alpha)\}m^5} + \log\log\left(\frac{2m^3}{L^2} \cdot \frac{1}{\epsilon}\right).$$

2.4 Time complexity

Remember that gradient descent for a m-strongly convex and M-smooth function converges to an ϵ-optimal solution after $O(M/m \log(1/\epsilon))$ iterations. For each iteration, we compute the gradient and excutes vector arithmetics, which costs $O(d)$ time where d is the ambient dimension. Hence, the time complexity of gradient descent is

$$O\left(d\frac{M}{m}\log\frac{1}{\epsilon}\right).$$

In contrast, Newton's method requires $O(\log\log(1/\epsilon))$ iterations, while each step requires computing the Hessian and its inverse. Computing the Hessian takes $O(d^2)$ time steps while computing the inverse takes $O(d^\omega)$ where ω is the exponent for matrix multiplication. The current best known bound for ω is 2.373 [AW][1]. Hence, the time complexity of Newton's method is

$$O\left(d^\omega \log\log\frac{1}{\epsilon}\right).$$

However, algorithms that achieve the best time compddddddlexity for matrix multiplication are not necessarily practical. We often use the Gaussian elimination based methods, which costs $O(d^3)$ time steps, in which case, the time complexity is

$$O\left(d^3 \log\log\frac{1}{\epsilon}\right).$$

Although the dependence on the error tolerance ϵ is small, Newton's method suffers from high-dimensional problems.

References

[AW] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539. 2.4

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 2.3

[DWZ22] Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. Technical report, 2022. 1

[1] Very recently, a better time complexity for matrix multiplication has been announced in ArXiv [DWZ22].