EE C227C Convex Optimization and Approximation Lecture 23 - 04/12/2022

Lecture 23: Continuing Cubic-Regularized Newton's Method

Lecturer: Jiantao Jiao *Scribe: Druv Pai, Peter Tong*

Today we will wrap up our discussion of second order methods. We will focus on using second order methods to minimize general functions. In the previous lecture, we discussed that in general Newton's method doesn't work, so we have many tricks (e.g. cubic regularization to make it converge).

1 Global Convergence

In general, for second-order convergence methods, we want local information to point us in the direction of the global optima. Our standing assumption for second-order methods is that the Hessian is Lipschitz:

Assumption 1 (Lipschitz Hessian)**.** *The function f to optimize has Lipschitz Hessian:*

$$\left\|\nabla^2 f(x) - \nabla^2 f(y)\right\| \leq L \|x - y\| \qquad \forall x, y \in \mathcal{F}. \tag{1}$$

We also use strong convexity assumption, which means that f has a quadratic lower bound.

Assumption 2 (Strong Convexity)**.** *The function f to optimize is μ-strongly convex:*

$$f(y) - f(x) - \langle \nabla f(x), y - x \rangle \geq \frac{\mu}{2} \|x - y\|^2 \qquad \forall x, y \in \mathcal{F}. \tag{2}$$

We introduce a radius, which is like a sub-level set for $f(x_0)$:

$$D := \max\{\|x - x^\star\| \,|x \in \mathcal{F}, f(x) \leq f(x_0)\}.$$

With the first assumption on second-order information and second assumption on strong convexity, we can easily derive an upper bound of D:

$$D \leq \left[\frac{2}{\mu}(f(x_0) - f^\star)\right]^{\frac{1}{2}}.$$

The parameters L, D, and μ are natural parameters for this class of problems. Define

$$\kappa := \frac{LD}{\mu}.$$

Intuitively this value κ measures the "difficulty" of the problem.

- If L is large then the upper bound on the Lipschitzness of $\nabla^2 f$ is large, meaning that the problem is harder.
- If D is large then the domain is large, meaning that the problem is harder.
- If μ is small then f is not very strongly convex, meaning that the problem is harder.

The correct way to think about convergence of second-order methods is that the time to convergence is two parts:

- The time to reach the region of super-linear convergence, which can be large.
- The time to convergence once in the super-linear regime, which is usually extremely small.

The so-called switching values are

$$\omega_0 := \frac{\mu^3}{18L^2} = \frac{4}{9}\overline{\omega} \qquad \omega_1 := \frac{3}{2}\mu D^2 \qquad \omega_2 := \frac{3}{2}LD^3.$$

If $\kappa \leq 1$, then we take one iteration to get into the super-linear regime, then a short amount of iterations to converge. If $\kappa > 1$: we get

$$\omega_0 \leq \omega_1 \leq \omega_2.$$

We want to see how many iterations of cubic-regularized Newton's method it takes to reduce the objective function value. We break down the analysis into 4 phases:

- (Phase 0) From the initial value $f(x_0)$ to ω_2;
- (Phase 1) From ω_2 to ω_1;
- (Phase 2) From ω_1 to ω_0;
- (Phase 3) From ω_0 to some arbitrarily small $\varepsilon > 0$.

1.1 Phase 0

Theorem 4.1.4 of [1] tells us that it takes a single iteration to get from $f(x_0)$ to ω_2; namely, under the assumptions provided,

$$f(x_1) - f^\star \leq \omega_2.$$

The number of iterations for phase 0 is $k_0 = 1$.

1.2 Phase 1

Moreover, the theorem tells us that

$$\omega_1 \leq \frac{3LD^3}{2\left(1 + \frac{k_1}{3}\right)^2}$$

which implies that $k_1 \leq 3\sqrt{\kappa}$.

1.3 Phase 2

Theorem 4.1.5 of [1] tells us that if

$$f(x_0) - f^\star \geq \omega_0 = \frac{4}{9}\overline{\omega}$$

then

$$f(x_k) - f^\star \leq \left[\left(f(x_0) - f^\star\right)^{1/4} - \frac{k_2}{6}\sqrt{\frac{2}{3}}\overline{\omega}^{1/4}\right]^4.$$

Solving this for ω_0 tells us that

$$\omega_0^{1/4} \leq \left(f(x_{k+1}) - f^\star\right)^{1/4} - \frac{k_2}{6}\omega_0^{1/4}.$$

We solve this for k_2 to get

$$k_2 \leq 3^{3/4}\sqrt{2}\sqrt{\kappa} \leq 3.25\sqrt{\kappa}.$$

1.4 Phase 3

Now we are in the regime of superlinear convergence. Theorem 4.1.5 of [1] tells us that if we define

$$\delta_k \leq \frac{1}{4\omega_0}\left(f(x_k) - f^\star\right)$$

then

$$\delta_{k+1} \leq \delta_k^{3/2}.$$

And

$$\overline{\delta}_{k_0+k_1+k_2} \leq \frac{1}{4}.$$

Thus

$$f(x_{k_3}) - f^\star \leq \varepsilon \Rightarrow \delta_{k_0+k_1+k_2+k_3} \leq \frac{\varepsilon}{4\omega_0}.$$

We have

$$\delta_{k_0+k_1+k_2+k_3} = \delta_{k_0+k_1+k_2}^{(3/2)^{k_3}}$$

which implies that

$$k_3 \leq \log_{3/2}\left(\log_4\left(\frac{2\mu^3}{9\varepsilon L^2}\right)\right).$$

Thus the total number of steps is

$$N \leq 6.25\sqrt{\frac{LD}{\mu}} + \log_{3/2}\left(\log_4\left(\frac{1}{\varepsilon}\right) + \log_4\left(\frac{2\mu^3}{9L^2}\right)\right).$$

2 Contrasting with First-Order Methods

If we use gradient descent or accelerated gradient descent, then the idea of large eigenvalue of Hessian really kills the first order methods but not the second-order methods. Suppose that

$$D := \max\left\{\|x - x^\star\| \,|x \in \mathcal{F}\right\},$$

and

$$\hat{L} := \lambda_{\max}(\nabla^2 f(x^\star))$$

and also

$$\mu I \leq \nabla^2 f(x) \leq (\hat{L} + LD)I.$$

Then gradient descent makes the total number of iterations to get accuracy ε is

$$N_{\mathrm{GD}} = \frac{\hat{L} + LD}{\mu}\log\left(\frac{\hat{L} + LD}{\epsilon}\right)$$

and accelerated gradient descent gives

$$N_{\mathrm{AGD}} = \sqrt{\frac{\hat{L} + LD}{\mu}}\log\left(\frac{\hat{L} + LD}{\epsilon}\right).$$

This is *multiplicative* in $\log\left(\frac{1}{\epsilon}\right)$, not *additive* as in our evaluation of cubic-regularized Newton's method. So first-order methods are generally worse, with these assumptions. The main bottleneck is, again, the size of the problem; first-order methods are generally much more efficient. But people are still researching fast Hessian approximation, so it is possible to use second-order methods in reasonably sized problems.

3 Acceleration

Vanilla cubic-regularized Newton's method gets a $\frac{1}{k^2}$ convergence rate. Recall that gradient descent gets a $\frac{1}{k}$ convergence rate, and that accelerated gradient descent gets a $\frac{1}{k^2}$ convergence rate. The question is how to accelerate the convergence rate of cubic-regularized Newton's method.

It turns out that there is a simple acceleration to get $\frac{1}{k^3}$ convergence rate (so-called accelerated cubic-regularized Newton's method), which we discuss below. The information-theoretic lower bound is $\frac{1}{k^{3.5}}$ which is widely believed to be tight. However, there is no algorithm – accepted by the community – to achieve this lower bound.

Assume L is known and define $M = 2L$, $C = 6L$. Remember that in Newton's method, we write $T_L(x)$ to be the function

$$T_L(x) := \arg\min_{\widetilde{x} \in \mathcal{F}} \left(f(x) + \langle \nabla f(x), \widetilde{x} - x \rangle + \left\langle \nabla^2 f(x) \cdot (\widetilde{x} - x), \widetilde{x} - x \right\rangle + \frac{L}{6} \left\| \widetilde{x} - x \right\|^3 \right).$$

That is, we approximate the third-order Taylor series and find the minimum of the resulting cubic.

Require: L, the Lipschitz constant of $\nabla^2 f$.
$M := 2L$
$C := 6L$
Define $\psi_1 \colon x \mapsto f(x) + \frac{C}{6} \left\| x - x_0 \right\|^3$
Initialize x_0
for $k \in \{1, 2, \dots, \}$ **do**
  $v_k := \arg\min_x \psi_k(x)$
  $y_k := \frac{k}{k+3} x_k + \frac{3}{k+3} v_k$
  $x_{k+1} := T_M(y_k)$
  Define $\psi_{k+1} \colon x \mapsto \psi_k(x) + \frac{(k+1)(k+2)}{2} \left[f(x_{k+1}) + \langle \nabla f(x_{k+1}), x - x_{k+1} \rangle \right]$

This is easily implementable as each term either uses the zeroth, first, or second order information.

Theorem 4.2.3 of [1] says that for $k \geq 1$,

$$f(x_k) - f^\star \leq \frac{8L}{k(k+1)(k+2)} \left\| x_0 - x^\star \right\|^3.$$

The proof method is very similar to the regular cubic-regularized Newton's method: define estimate sequence and do descent, etc.

The theory is not so important. The idea is that *if* we can compute the Hessian, we definitely *should* use the second-order information, using i.e., Newton's method. If we can only use first-order methods, even accelerated gradient is not always best; there are problem types where algorithms such as *conditional gradient* or *Frank-Wolfe* algorithm are much faster. And of course this whole picture gets upended if we are dealing with nonconvex problems, but even some progress in *composite optimization* has been made there.

Next few lectures:

- Stochastic optimization;
- Conditional gradient;
- Augmented Lagrangian (ADMM);
- Some other ideas of basic optimization, i.e., line search.

4 Stochastic Optimization

We start the topic of stochastic optimization.

Before machine learning, we had the following formulation:

$$\min_x f(x) = \min_x \mathbb{E}_\xi(F(x,\xi))$$

where ξ is a random variable. This literature is not very well explored.

We can solve this by the following method. Suppose we can compute

$$g(x,\xi) := \nabla_x F(x,\xi).$$

Then we claim that

$$\mathbb{E}_\xi(g(x,\xi)) = \nabla f(x).$$

Indeed, this is easy by Fubini's theorem:

$$\begin{aligned}\mathbb{E}_\xi(g(x,\xi)) &= \mathbb{E}_\xi(\nabla_x F(x,\xi)) \\ &= \nabla_x \mathbb{E}_\xi(F(x,\xi)) \\ &= \nabla f(x).\end{aligned}$$

Now, because of modern machine learning, we have the particularly important case where ξ is distributed uniformly over $\{1,2,\ldots,n\}$. Here n can be interpreted as the number of data points we have, and we are minimizing the average loss over all the data points.

By specializing to this case, we can get much better results. However, we will discuss the more general case first.

References

[1] Y. Nesterov, *Lectures on Convex Optimization*, 2nd ed. Springer Publishing Company, Incorporated, 2018.