10-725/36-725: Convex Optimization **Fall 2018**

Lecture 14: October 17

Lecturer: Lecturer: Ryan Tibshirani *Scribes: Pengsheng Guo, Xian Zhou*

14.1 Newton's method

To start with, we consider an unconstrained, smooth convex optimization

$$\min_x f(x)$$

where f is convex, twice differentiable, and $\text{dom}(f) = \mathbb{R}^n$.

Recall that the gradient descent chooses initial $x^{(0)} \in \mathbb{R}$ and repeats

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

In comparison, Newton's method repeats

$$x^{(k)} = x^{(k-1)} - (\nabla^2 f(x^{(k-1)}))^{-1} \nabla f(x^{(k-1)}), \quad k = 1, 2, 3, \ldots$$

where $\nabla^2 f(x^{(k-1)})$ is the Hessian matrix of f at $x^{(k-1)}$.

14.1.1 Newton's method interpretation

In the case of gradient descent, the update $x^+ = x - t\nabla f(x)$ minimizes the quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$

over y. While in Newton's method, we minimize a better quadratic approximation

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

and the update is thus given by

$$x^+ = x - (\nabla^2 f(x))^{-1} \nabla f(x)$$

14.1.1.1 Example

Consider minimizing $f(x) = \frac{1}{2}(10x_1^2 + x_2^2) + 5\log(1 + e^{-x_1 - x_2})$. Figure 14.1 shows a comparison between gradient descent and Newton's method.

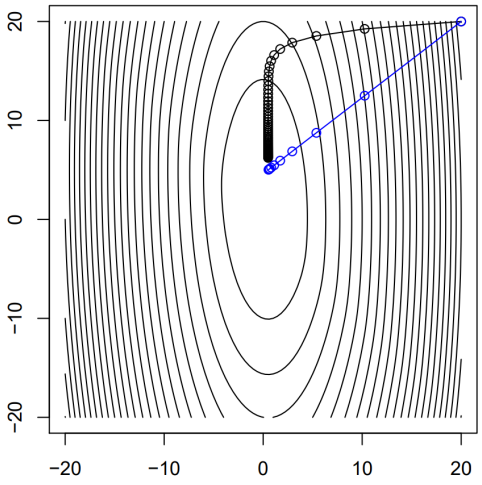

Figure 14.1: Comparison of gradient descent (black) and Newton's method (blue), where both take steps of roughly same length.

14.1.2 Linearized optimality condition

Here is another interpretation of Newton's method. Let $F(x) = \nabla f(x)$. We can linearize F around x via approximation $F(y) \approx F(x) + DF(x)(y - x)$, where $DF(x)$ is the Jacobian matrix of F.

We seek a optimization step v so that $\nabla f(x + v) = 0$, which is

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

Solving this yields the same Newton's step $v = -(\nabla^2 f(x))^{-1} \nabla f(x)$.

14.1.3 Affine invariance of Newton's method

Affine invariance is an important property of Newton's method. Given f and nonsingular $A \in \mathbb{R}^{n \times n}$, let $x = Ay$ and $g(y) = f(Ay)$. The Newton step on g is given by

$$\begin{aligned} y^+ &= y - (\nabla^2 g(y))^{-1} \nabla g(y) \\ &= y - (A^T \nabla^2 f(Ay) A)^{-1} A^T \nabla f(Ay) \\ &= y - A^{-1} (\nabla^2 f(Ay))^{-1} \nabla f(Ay) \end{aligned}$$

which then gives

$$Ay^+ = Ay - (\nabla^2 f(Ay))^{-1} \nabla f(Ay)$$

This essentially shows $x^+ = x - (\nabla^2 f(x))^{-1} f(x)$. The affine invariance property shows the progress of Newton's method is independent of problem scaling. We do not achieve this property with gradient descent.

14.1.4 Newton decrement

The difference between $f(x)$ and the minimum of its quadratic approximation is

$$\begin{aligned} & f(x) - \min_y \left(f(x) + \nabla f(x)^T (y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(x)(y-x) \right) \\ &= f(x) - \left(f(x) - \frac{1}{2} \nabla f(x)^T (\nabla^2 f(x))^{-1} \nabla f(x) \right) \\ &= \frac{1}{2} \lambda(x)^2 \end{aligned}$$

Where $\lambda(x)$ is defined as the Newton decrement and given by

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

The Newton decrement can be seen as an approximate upper bound on the suboptimality gap $f(x) - f^\star$.

There is another way to interpret Newton decrement. Note that the Newton direction is given by $v = -(\nabla^2 f(x))^{-1} \nabla f(x)$, then we can write Newton decrement as

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

This shows $\lambda(x)$ is the length of the Newton step in the norm defined by the Hessian $\nabla^2 f(x)$.

The property of *affine invariance* also holds for the Newton decrement; i.e. given $g(y) = f(Ay)$ for nonsingular A, $\lambda_g(y)$ would match $\lambda_f(x)$ at $x = Ay$.

14.2 Backtracking line search

Pure Newton's method does not always converge. Depending on where we start, Newton's method can either converge or diverge rapidly. Instead, a damped Newton's method is utilized which repeats

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

Step sizes t are found by backtracking search. At each iteration, we start with $t = 1$ and while

$$f(x + tv) > f(x) + \alpha t \nabla f(x)^T v = f(x) + \alpha t \nabla f(x)^T \left(-(\nabla^2 f(x))^{-1} \nabla f(x) \right) = f(x) - \alpha t \lambda^2(x)$$

we shrink $t = \beta t$, else we find the step size t and perform the Newton update.

An comparison of Newton's method and gradient descent in terms of iteration number is shown in Figure 14.2, both using backtracking line search. One thing to notice is that such comparison is not fair since Newton's method requires the calculation of Hessian matrix which has a higher computational cost.

14.3 Convergence analysis

Assume that f is convex, twice differential, and has a domain $\text{dom}(f) = \mathbb{R}^n$. Additionally,

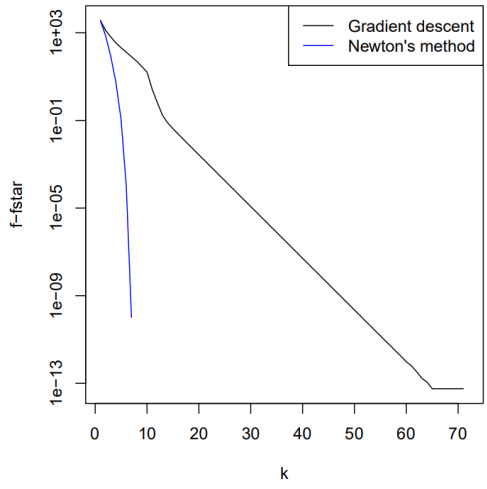

Figure 14.2: Comparison of gradient descent and Newton's method in terms of iteration number, both with backtracking

- ∇f is Lipschitz with parameter L
- f is strongly convex with parameter m
- $\nabla^2 f$ is Lipschitz with parameter M

Then we would have the following theorem:

Theorem 14.1 *Newton's method with backtracking line search satisfies the following two-stage convergence bounds*

$$f(x^{(k)}) - f^\star \leq \begin{cases} (f(x^{(0)}) - f^\star) - \gamma k & \text{if } k \leq k_0 \\ \frac{2m^3}{M^2}(\frac{1}{2})^{2^{k-k_0+1}} & k > k_0 \end{cases}$$

where $\gamma = \alpha\beta^2\eta^2 m/L^2$, $\eta = \min\{1, 3(1-2\alpha)\}m^2/M$, *and* k_0 *is the number of steps until* $\|\nabla f(x^{(k_0+1)})\| < \eta$.

In more detail, Newton's method converges in two stages, namely damped phase and pure phase.

- Damped phase: $\|\nabla f(x^{(k)})\|_2 \geq \eta$ and

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma$$

- Pure phase: $\|\nabla f(x^{(k)})\|_2 < \eta$, backtracking selects $t = 1$, and

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

Assume that we're in pure phase and backtracking gives $t = 1$. We would have 3 facts

- Fact 1. f is m-strongly convex: $f(x) - f^\star \leq \frac{1}{2m}\|\nabla f(x)\|_2^2$
- Fact 2. $\frac{M}{2m^2}\|\nabla f(x^\star)\|_2 \leq (\frac{M}{2m^2}\|\nabla f(x)\|_2)^2$

- Fact 3. $f(x^{(k)}) - f^\star \leq \frac{2m^3}{M^2}(\frac{1}{2})^{2^{k-k_0}}$

The proof is shown below.

- Fact 1. **Proof:**

$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{m}{2}\|y-x\|_2^2$$

We minimized both sides over y

$$f^\star \geq \min_y (f(x) + \nabla f(x)^T(y-x) + \frac{m}{2}\|y-x\|_2^2)$$

Since we know $0 = \nabla f(x) + m(y-x)$ and $y = -\frac{1}{m}\nabla f(x) + x$, thus

$$\begin{aligned} f^\star &\geq f(x) - \frac{1}{m}\|\nabla f(x)\|_2^2 + \frac{1}{2m}\|\nabla f(x)\|_2^2 \\ &= f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \end{aligned}$$

■

- Fact 2. **Proof:**

With $v = -(\nabla^2 f(x))^{-1}\nabla f(x)$, we can rewrite $\|\nabla f(x^+)\|_2^2$ as

$$\begin{aligned} \|\nabla f(x^+)\|_2 &= \|\nabla f(x+v)\|_2 \\ &= \|\nabla f(x+v) - \nabla f(x) - \nabla^2 f(x)v\|_2 \\ &= \|\int_0^1 \nabla^2 f(x+tv)vdt - \nabla^2 f(x)v\|_2 \\ &= \|\int_0^1 (\nabla^2(f(x+tv) - \nabla^2 f(x))vdt\|_2 \\ &\leq \int_0^1 \|(\nabla^2(f(x+tv) - \nabla^2 f(x))v\|_2 dt \\ &\leq \int_0^1 \|(\nabla^2 f(x+tv) - \nabla^2 f(x)\|_{op}\|v\|_2 dt \\ &\leq \int_0^1 Mt\|v\|_2^2 dt \\ &= M\|v\|_2^2 \int_0^1 tdt \\ &= \frac{1}{2}M\|(\nabla^2 f(x))^{-1}\nabla f(x)\|_2^2 \\ &\leq \frac{1}{2}M\|(\nabla^2 f(x))^{-1}\|_{op}^2\|\nabla f(x)\|_2^2 \\ &\leq \frac{M}{2m^2}\|\nabla f(x)\|_2^2 \end{aligned}$$

So we have

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

■

- Fact 3. **Proof:** We know that

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

By having $a_k = \frac{M}{2m^2}\|\nabla f(x^{(k)})\|_2$, we know

$$\begin{aligned} a_k &\leq a_{k-1}^2 \\ &\leq a_{k-2}^4 \\ &\vdots \\ &\leq a_{k_0}^{2^{k-k_0}} \end{aligned}$$

Thus,

$$\frac{M}{2m^2}\|\nabla f(x^{(k)})\|_2 \leq (\frac{M}{2m^2}\|\nabla f(x^{(k_0)})\|_2)^{2^{k-k_0}}$$

At k_0, we know that $\|\nabla f(x^{(k_0)})\|_2 < \eta \leq \frac{m^2}{M}$

So we have

$$\frac{M}{2m^2}\|\nabla f(x^{(k)})\|_2 \leq (\frac{1}{2})^{2^{k-k_0}}$$

Finally,

$$\begin{aligned} f(x^{(k)}) - f^\star &\leq \frac{1}{2m}\|\nabla f(x^{(k)})\|_2^2 \\ &\leq \frac{1}{2m}(\frac{2m^2}{M})^2(\frac{1}{2})^{2^{k-k_0}} \\ &= \frac{2m^3}{M^2}(\frac{1}{2})^{2^{k-k_0}} \end{aligned}$$

■

Thus, in order to get $f(x^{(k)}) - f^\star \leq \epsilon$, we set

$$\epsilon_0(\frac{1}{2})^{2^{k-k_0}} = \epsilon$$

$$k - k_0 = \log\log(\epsilon_0/\epsilon)$$

As such, we need at most

$$\frac{f(x^{(0)}) - f^\star}{\gamma} + \log\log(\epsilon_0/\epsilon)$$

iterations, where $\epsilon_0 = 2m^3/M^2$ This gives us quadratic convergence and the above result is also a local convergence rate. We are guaranteed quadratic convergence after some number of steps $k_0 \leq \frac{f(x^{(0)})-f^\star}{\gamma}$ to enter pure phase.

14.3.1 Self-concordance

The bound derived above is somewhat bothersome as it depends on L, m, M although the algorithm itself is affine invariant. A scale-free analysis has been proposed by Nesterov and Nemirovskii named self-concordant functions, f to bridge this gap between theory and practice.

A function is self-concordant if it satisfies

$$|f'''(x)| \leq 2f''(x)^{3/2} \forall x$$

Theorem 14.2 *If a function is self-concordant, then Newton's method with backtracking line search requires at most*

$$C(\alpha, \beta)(f(x^{(0)}) - f^\star + \log\log(1/\epsilon)$$

iterations to reach $f(x^{(0)}) - f^\star \leq \epsilon$*, where* $C(\alpha, \beta)$ *is a constant only depends on backtracking parameters* α, β*.*

This result is useful as it is not related the Lipschitz or strong convexity constants anymore thus is a better way of characterizing Newton's method.

14.3.2 Comparison to first-order methods

- Memory: each iteration of Newtons method requires $O(n^2)$ storage ($n \times n$ Hessian); each gradient iteration requires $O(n)$
- Computation: each Newton iteration requires $O(n^3)$ flops (solving a dense $n \times n$ linear system); each gradient iteration requires $O(n)$ flops (scaling/adding n-dimensional vectors)
- Backtracking: backtracking line search has roughly the same cost, both use $O(n)$ flops per inner backtracking step
- Conditioning: Newtons method is not affected by a problems conditioning, but gradient descent can seriously degrade
- Fragility: Newtons method may be empirically more sensitive to bugs/numerical errors, gradient descent is more robust storage (n-dimensional gradient)

Moreover, the comparison of Newton's method and gradient descent in terms of time taken per iteration is shown in Figure 14.3.

14.3.3 Sparse and structured problem

Newton's method can perform well when the inner linear systems (in Hessian) can be solved efficiently and reliably. If $\nabla^2 f(x)$ is sparse and structured for all x, for instance, say banded, both memory and computation are $O(n)$ with Newton iterations.

14.4 Equality-constrained Newton's method

We now apply Newton's method to problems with equality constraints.

Consider

$$\min_x f(x) \quad \text{subject to} \quad Ax = b$$

We have several options to address this:

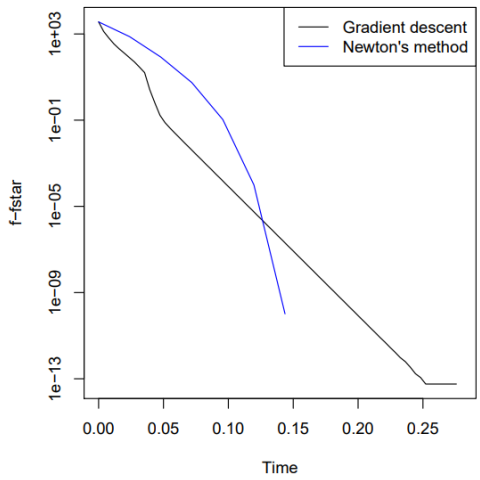

Figure 14.3: Comparison of gradient descent and Newton's method in terms of time, both with backtracking

- **Eliminating equality constraints**
  If we write $x = Fy + x_0$, where F spans null space of A and $Ax_0 = b$. We can then solve in terms of y.

- **Deriving the dual**
  We can check that the Lagrange dual function is $-f^\star(-A^T v) - b^T v$, and strong duality holds, while we need to express $x^\star$ in terms of $v^\star$.

- **Equality-constrained Newton** This is the most straightforward option in many cases and is described in details below.

In euqality-constrained Newton's method, we start with $x^{(0)}$ such that $Ax^{(0)} = b$. Then we repeat the updates

$$x^+ = x + tv$$

where

$$v = \underset{Az=0}{\operatorname{argmin}} \nabla f(x)^T (z - x) + \frac{1}{2}(z - x)^T \nabla^2 f(x)(z - x)$$

This keeps x^+ in feasible set, since $Ax^+ = Ax + tAv = b + 0 = b$.

Furthermore, v is the solution to minimizing a quadratic subject to equality constraints. From KKT consitions v satisfies

$$\begin{bmatrix} \nabla^2 f(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

for some w. Hence Newton direction v is again given by solving a linear system in the Hessian (albeit a bigger one).

14.5 Quasi-Newton methods

If the Hessian is too expensive (or singular), then a quasi-Newton method can be used to approximate $\nabla^2 f(x)$ with $H \succ 0$, and we update according to

$$x^+ = x - tH^{-1}\nabla f(x)$$

Note that

- Approximate Hessian H is recomputed at each step. Goal is to make H^{-1} cheap to apply (possibly, cheap storage too)
- Convergence is fast: superlinear, but not the same as Newton. Roughly n steps of quasi-Newton make same progress as one Newton step
- Very wide variety of quasi-Newton methods; common theme is to "propagate" computation of H across iterations

Davidon-Fletcher-Powell or DFP

- Update H, H^1 via rank 2 updates from previous iterations; cost is $O(n^2)$ for these updates
- Since it is being stored, applying H^1 is simply $O(n^2)$ flops
- Can be motivated by Taylor series expansion

Broyden-Fletcher-Goldfarb-Shanno or BFGS

- Came after DFP, but BFGS is now much more widely used
- Again, updates H, H^{-1} via rank 2 updates, but does so in a "dual" fashion to DFP; cost is still $O(n^2)$
- Also has a limited-memory version, L-BFGS: instead of letting updates propagate over all iterations, only keeps updates from last m iterations; storage is now $O(mn)$ instead of $O(n^2)$