Convex Optimization

(EE227A: UC Berkeley)

Lecture 25
(Newton, quasi-Newton)

23 Apr, 2013

Suvrit Sra

Admin

- Project **poster** presentations:

Soda 306 HP Auditorium Fri May 10, 2013 4pm – 8pm

- HW5 due on **May 02, 2013**
  Will be released today.

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

$$g(x + \Delta x) = g(x) + g'(x)\Delta x + o(|\Delta x|).$$

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

$$g(x + \Delta x) = g(x) + g'(x)\Delta x + o(|\Delta x|).$$

- Equation $g(x + \Delta x) = 0$ **approximated** by

$$g(x) + g'(x)\Delta x = 0 \qquad \Longrightarrow \Delta x = -g(x)/g'(x).$$

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

$$g(x + \Delta x) = g(x) + g'(x)\Delta x + o(|\Delta x|).$$

- Equation $g(x + \Delta x) = 0$ **approximated** by

$$g(x) + g'(x)\Delta x = 0 \qquad \Longrightarrow \Delta x = -g(x)/g'(x).$$

- If x is close to x^*, we can expect $\Delta x \approx \Delta x^* = x^* - x$

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

$$g(x + \Delta x) = g(x) + g'(x)\Delta x + o(|\Delta x|).$$

- Equation $g(x + \Delta x) = 0$ **approximated** by

$$g(x) + g'(x)\Delta x = 0 \qquad \Longrightarrow \; \Delta x = -g(x)/g'(x).$$

- If x is close to x^*, we can expect $\Delta x \approx \Delta x^* = x^* - x$
- Thus, we may write

$$x^* \approx x - \frac{g(x)}{g'(x)}$$

Newton method

- Recall numerical analysis: Newton method for solving equations

$$g(x) = 0 \quad x \in \mathbb{R}.$$

- Key idea: *linear approximation*.
- Suppose we are at some x close to x^* (the **root**)

$$g(x + \Delta x) = g(x) + g'(x)\Delta x + o(|\Delta x|).$$

- Equation $g(x + \Delta x) = 0$ **approximated** by

$$g(x) + g'(x)\Delta x = 0 \qquad \Longrightarrow \; \Delta x = -g(x)/g'(x).$$

- If x is close to x^*, we can expect $\Delta x \approx \Delta x^* = x^* - x$
- Thus, we may write

$$x^* \approx x - \frac{g(x)}{g'(x)}$$

- Which suggests the iterative process

$$x_{k+1} \leftarrow x_k - \frac{g(x_k)}{g'(x_k)}$$

Newton method

- Suppose we have a system of nonlinear equations

$$G(x) = 0 \qquad G : \mathbb{R}^n \to \mathbb{R}^n.$$

Newton method

- Suppose we have a system of nonlinear equations

$$G(x) = 0 \qquad G : \mathbb{R}^n \to \mathbb{R}^n.$$

- Again, arguing as above we arrive at the **Newton system**

$$G(x) + G'(x)\Delta x = 0,$$

where $G'(x)$ is the **Jacobian**.

Newton method

- Suppose we have a system of nonlinear equations

$$G(x) = 0 \qquad G : \mathbb{R}^n \to \mathbb{R}^n.$$

- Again, arguing as above we arrive at the **Newton system**

$$G(x) + G'(x)\Delta x = 0,$$

  where $G'(x)$ is the **Jacobian**.
- Assume $G'(x)$ is non-degenerate (invertible), we obtain

$$x_{k+1} = x_k - [G'(x_k)]^{-1} G(x_k).$$

Newton method

- Suppose we have a system of nonlinear equations

$$G(x) = 0 \qquad G : \mathbb{R}^n \to \mathbb{R}^n.$$

- Again, arguing as above we arrive at the **Newton system**

$$G(x) + G'(x)\Delta x = 0,$$

  where $G'(x)$ is the **Jacobian**.
- Assume $G'(x)$ is non-degenerate (invertible), we obtain

$$x_{k+1} = x_k - [G'(x_k)]^{-1} G(x_k).$$

- This is Newton's method for solving nonlinear equations

Newton method

$$\boxed{\min \quad f(x) \text{ such that } x \in \mathbb{R}^n}$$

Newton method

$$\boxed{\min \quad f(x) \text{ such that } x \in \mathbb{R}^n}$$

$$\boxed{\nabla f(x) = 0 \text{ is necessary for optimality}}$$

Newton method

$$\boxed{\min \quad f(x) \text{ such that } x \in \mathbb{R}^n}$$

$$\boxed{\nabla f(x) = 0 \text{ is necessary for optimality}}$$

Newton system

$$\nabla f(x) + \nabla^2 f(x) \Delta x = 0,$$

which leads to

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

the **Newton method** for optimization

Newton method – remarks

- Newton method for equations is **more general** than minimizing $f(x)$ by finding roots of $\nabla f(x) = 0$

Newton method – remarks

- Newton method for equations is **more general** than minimizing $f(x)$ by finding roots of $\nabla f(x) = 0$
- **Reason:** Not every function $G : \mathbb{R}^n \to \mathbb{R}^n$ is a derivative!

Example Consider the linear system

$$Ax - b = 0.$$

Unless A is symmetric, does not correspond to a derivative (Why?)

Newton method – remarks

- Newton method for equations is **more general** than minimizing $f(x)$ by finding roots of $\nabla f(x) = 0$
- **Reason:** Not every function $G : \mathbb{R}^n \to \mathbb{R}^n$ is a derivative!

Example Consider the linear system

$$Ax - b = 0.$$

Unless A is symmetric, does not correspond to a derivative (Why?)

- If it were a derivative, then its own derivative is a Hessian, and we know that Hessians must be symmetric, QED.

Newton method – remarks

- In general, Newton method highly nontrivial to analyze

Example Consider the iteration

$$x_{k+1} = x_k - \frac{1}{x_k}, \qquad x_0 = 2.$$

May be viewed as iter for $e^{x^2/2} = 0$ (which has **no real solution**)

Newton method – remarks

- In general, Newton method highly nontrivial to analyze

Example Consider the iteration

$$x_{k+1} = x_k - \frac{1}{x_k}, \qquad x_0 = 2.$$

May be viewed as iter for $e^{x^2/2} = 0$ (which has **no real solution**)

Unknown whether this iteration generates a bounded sequence!

Newton method – remarks

- In general, Newton method highly nontrivial to analyze

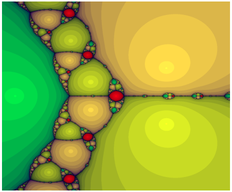
Example Consider the iteration

$$x_{k+1} = x_k - \frac{1}{x_k}, \qquad x_0 = 2.$$

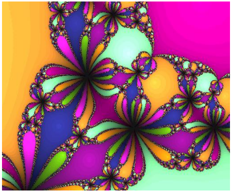
May be viewed as iter for $e^{x^2/2} = 0$ (which has **no real solution**)

Unknown whether this iteration generates a bounded sequence!

Newton fractals (Complex dynamics)



$z^3 - 2z + 2$



$x^8 + 15x^4 - 16$

Newton method – alternative view

Quadratic approximation

$$\phi(x) := f(x) + \langle \nabla f(x_k),\, x - x_k \rangle + \tfrac{1}{2}\langle \nabla^2 f(x_k)(x - x_k),\, x - x_k \rangle.$$

Newton method – alternative view

Quadratic approximation

$$\phi(x) := f(x) + \langle \nabla f(x_k), x - x_k \rangle + \tfrac{1}{2}\langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

Assuming $\nabla^2 f(x_k) \succ 0$, choose x_{k+1} as argmin of $\phi(x)$

Newton method – alternative view

Quadratic approximation

$$\phi(x) := f(x) + \langle \nabla f(x_k),\, x - x_k \rangle + \tfrac{1}{2}\langle \nabla^2 f(x_k)(x - x_k),\, x - x_k \rangle.$$

Assuming $\nabla^2 f(x_k) \succ 0$, choose x_{k+1} as argmin of $\phi(x)$

$$\phi'(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0.$$

Newton method – convergence

- Method breaks down if $\nabla^2 f(x_k) \not\succ 0$
- Only **locally convergent**

Example Find the root of

$$g(x) = \frac{x}{\sqrt{1+x^2}}.$$

Clearly, $x^* = 0$.

Newton method – convergence

- Method breaks down if $\nabla^2 f(x_k) \not\succ 0$
- Only **locally convergent**

Example Find the root of

$$g(x) = \frac{x}{\sqrt{1+x^2}}.$$

Clearly, $x^* = 0$.

Exercise: Analyze behavior of Newton method for this problem.
Hint: Consider the cases: $|x_0| < 1$, $x_0 = \pm 1$ and $|x_0| > 1$.

Newton method – convergence

- Method breaks down if $\nabla^2 f(x_k) \not\succ 0$
- Only **locally convergent**

Example Find the root of

$$g(x) = \frac{x}{\sqrt{1+x^2}}.$$

Clearly, $x^* = 0$.

Exercise: Analyze behavior of Newton method for this problem.
Hint: Consider the cases: $|x_0| < 1$, $x_0 = \pm 1$ and $|x_0| > 1$.

Damped Newton method

$$\boxed{x_{k+1} = x_k - \alpha_k [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)}$$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$
- where x^* is a local min, i.e., $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$
- where x^* is a local min, i.e., $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$
- Let $g(x_k) \equiv \nabla f(x_k)$; Taylor's theorem:

$$0 = g(x^*) = g(x_k) + \langle \nabla g(x_k),\, x^* - x_k \rangle + o(\|x_k - x^*\|)$$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$
- where x^* is a local min, i.e., $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$
- Let $g(x_k) \equiv \nabla f(x_k)$; Taylor's theorem:

$$0 = g(x^*) = g(x_k) + \langle \nabla g(x_k), x^* - x_k \rangle + o(\|x_k - x^*\|)$$

- Multiply by $[\nabla g(x_k)]^{-1}$ to obtain

$$x_k - x^* - [\nabla g(x_k)]^{-1} g(x_k) = o(\|x_k - x^*\|)$$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$
- where x^* is a local min, i.e., $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$
- Let $g(x_k) \equiv \nabla f(x_k)$; Taylor's theorem:
$$0 = g(x^*) = g(x_k) + \langle \nabla g(x_k), x^* - x_k \rangle + o(\|x_k - x^*\|)$$
- Multiply by $[\nabla g(x_k)]^{-1}$ to obtain
$$x_k - x^* - [\nabla g(x_k)]^{-1} g(x_k) = o(\|x_k - x^*\|)$$
- Newton iteration is: $x_{k+1} = x_k - [\nabla g(x_k)]^{-1} g(x_k)$, so
$$x_{k+1} - x^* = o(\|x_k - x^*\|),$$

Newton – local convergence rate

- Suppose method generates sequence $\{x_k\} \to x^*$
- where x^* is a local min, i.e., $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$
- Let $g(x_k) \equiv \nabla f(x_k)$; Taylor's theorem:
$$0 = g(x^*) = g(x_k) + \langle \nabla g(x_k), x^* - x_k \rangle + o(\|x_k - x^*\|)$$
- Multiply by $[\nabla g(x_k)]^{-1}$ to obtain
$$x_k - x^* - [\nabla g(x_k)]^{-1} g(x_k) = o(\|x_k - x^*\|)$$
- Newton iteration is: $x_{k+1} = x_k - [\nabla g(x_k)]^{-1} g(x_k)$, so
$$x_{k+1} - x^* = o(\|x_k - x^*\|),$$
- So for $x_k \neq x^*$ we get
$$\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \lim_{k\to\infty} \frac{o(\|x_{k+1} - x^*\|)}{\|x_k - x^*\|} = 0.$$

Local superlinear convergence rate

Newton method – local convergence

Assumptions

- **Lipschitz Hessian**: $\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\|$
- **Local strong convexity**: There exists a local minimum x^* with

$$\nabla^2 f(x^*) \succeq \mu I, \qquad \mu > 0.$$

- **Locality**: Starting point x_0 "close enough" to x^*

Newton method – local convergence

Assumptions

- **Lipschitz Hessian**: $\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\|$
- **Local strong convexity**: There exists a local minimum x^* with

$$\nabla^2 f(x^*) \succeq \mu I, \qquad \mu > 0.$$

- **Locality**: Starting point x_0 "close enough" to x^*

Theorem Suppose x_0 satisfies

$$\|x_0 - x^*\| < r := \frac{2\mu}{3M}.$$

Then, $\|x_k - x^*\| < r$, $\forall k$ and the NM converges quadratically

$$\|x_{k+1} - x^*\| \leq \frac{M\|x_k - x^*\|^2}{2(\mu - M\|x_k - x^*\|)}$$

Newton method – local convergence

Assumptions

- **Lipschitz Hessian**: $\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\|$
- **Local strong convexity**: There exists a local minimum x^* with

$$\nabla^2 f(x^*) \succeq \mu I, \qquad \mu > 0.$$

- **Locality**: Starting point x_0 “close enough” to x^*

Theorem Suppose x_0 satisfies

$$\|x_0 - x^*\| < r := \frac{2\mu}{3M}.$$

Then, $\|x_k - x^*\| < r$, $\forall k$ and the NM converges quadratically

$$\|x_{k+1} - x^*\| \leq \frac{M\|x_k - x^*\|^2}{2(\mu - M\|x_k - x^*\|)}$$

Reading assignment: Read §9.5.3 of Boyd-Vandenberghe

Quasi-Newton

Gradient and Newton

$$(\text{Grad}) \qquad x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0$$

$$(\text{Newton}) \qquad x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Gradient and Newton

$$\text{(Grad)} \qquad x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0$$
$$\text{(Newton)} \qquad x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Viewpoint for the **gradient method**.

Gradient and Newton

$$\text{(Grad)} \qquad x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0$$
$$\text{(Newton)} \qquad x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Viewpoint for the **gradient method**. Consider approximation

$$\phi_1(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha}\|x - x_k\|^2$$

Gradient and Newton

$$\text{(Grad)} \qquad x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0$$
$$\text{(Newton)} \qquad x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Viewpoint for the **gradient method**. Consider approximation

$$\phi_1(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha} \|x - x_k\|^2$$

Optimality condition yields

$$\phi'(x^*) \quad = \quad \nabla f(x_k) + \tfrac{1}{\alpha}(x^* - x_k) = 0$$

Gradient and Newton

$$
\begin{aligned}
&\text{(Grad)} \qquad && x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0 \\
&\text{(Newton)} \qquad && x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).
\end{aligned}
$$

Viewpoint for the **gradient method**. Consider approximation

$$
\phi_1(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha} \|x - x_k\|^2
$$

Optimality condition yields

$$
\begin{aligned}
\phi'(x^*) &= \nabla f(x_k) + \tfrac{1}{\alpha}(x^* - x_k) = 0 \\
x^* &= x_k - \alpha \nabla f(x_k)
\end{aligned}
$$

Gradient and Newton

$$\text{(Grad)} \qquad x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad \alpha_k > 0$$
$$\text{(Newton)} \qquad x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Viewpoint for the **gradient method**. Consider approximation

$$\phi_1(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\alpha} \|x - x_k\|^2$$

Optimality condition yields

$$\begin{aligned} \phi'(x^*) &= \nabla f(x_k) + \tfrac{1}{\alpha}(x^* - x_k) = 0 \\ x^* &= x_k - \alpha \nabla f(x_k) \end{aligned}$$

If $\alpha \in (0, \frac{1}{L}]$, $\phi_1(x)$ is **global overestimator**

$$f(x) \leq \phi_1(x), \qquad \forall x \in \mathbb{R}^n.$$

Gradient and Newton

Viewpoint for **Newton method**. Consider quadratic approx

$$\phi_2(x) := f(x_k) + \langle \nabla f(x_k),\, x - x_k\rangle + \frac{1}{2}\langle \nabla^2 f(x_k)(x - x_k),\, x - x_k\rangle.$$

Gradient and Newton

Viewpoint for **Newton method**. Consider quadratic approx

$$\phi_2(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}\langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

Minimum of this function is

$$x^* = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Gradient and Newton

Viewpoint for **Newton method**. Consider quadratic approx

$$\phi_2(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}\langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

Minimum of this function is

$$x^* = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Something better than ϕ_1, less expensive than ϕ_2?

Quasi-Newton methods

Generic Quadratic Model

$$\phi_D(x) := f(x_k) + \langle \nabla f(x_k),\, x - x_k \rangle + \tfrac{1}{2}\langle H_k(x - x_k),\, x - x_k \rangle.$$

Quasi-Newton methods

Generic Quadratic Model

$$\phi_D(x) := f(x_k) + \langle \nabla f(x_k),\, x - x_k \rangle + \tfrac{1}{2}\langle H_k(x - x_k),\, x - x_k \rangle.$$

- Matrix $H_k \succ 0$, some posdef matrix
- Leads to optimum

$$x^* = x_k - H_k^{-1}\nabla f(x_k)$$
$$x^* = x_k - S_k \nabla f(x_k).$$

Quasi-Newton methods

Generic Quadratic Model

$$\phi_D(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \tfrac{1}{2}\langle H_k(x - x_k), x - x_k \rangle.$$

- Matrix $H_k \succ 0$, some posdef matrix
- Leads to optimum

$$x^* = x_k - H_k^{-1} \nabla f(x_k)$$
$$x^* = x_k - S_k \nabla f(x_k).$$

- The first-order methods that form a sequence of matrices

$$\{H_k\} \;:\; H_k \to \nabla^2 f(x^*)$$

where H_k is constructed using **only gradient information**,

Quasi-Newton methods

Generic Quadratic Model

$$\phi_D(x) := f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \tfrac{1}{2}\langle H_k(x - x_k), x - x_k \rangle.$$

- Matrix $H_k \succ 0$, some posdef matrix
- Leads to optimum

$$x^* = x_k - H_k^{-1}\nabla f(x_k)$$
$$x^* = x_k - S_k \nabla f(x_k).$$

- The first-order methods that form a sequence of matrices

$$\{H_k\} \;:\; H_k \to \nabla^2 f(x^*)$$

where H_k is constructed using **only gradient information**,are called **variable metric** or **quasi-Newton** methods.

$$x_{k+1} = x_k - H_k^{-1}\nabla f(x_k) \qquad k = 0, 1, \ldots$$
$$x_{k+1} = x_k - S_k \nabla f(x_k) \qquad k = 0, 1, \ldots$$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$
  2. **stepsize**: search for good $\alpha_k > 0$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$
  2. **stepsize**: search for good $\alpha_k > 0$
  3. **update**: $x_{k+1} = x_k - \alpha_k d_k$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$
  2. **stepsize**: search for good $\alpha_k > 0$
  3. **update**: $x_{k+1} = x_k - \alpha_k d_k$
  4. compute $f(x_{k+1})$ and $\nabla f(x_{k+1})$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$
  2. **stepsize**: search for good $\alpha_k > 0$
  3. **update**: $x_{k+1} = x_k - \alpha_k d_k$
  4. compute $f(x_{k+1})$ and $\nabla f(x_{k+1})$
  5. **QN update:** $S_k \to S_{k+1}$

Quasi-Newton method

- Choose $x_0 \in \mathbb{R}^n$. Let $H_0 = I$.
  Compute $f(x_0)$ and $\nabla f(x_0)$
- For $k \geq 0$:
  1. **descent direction**: $d_k \leftarrow S_k \nabla f(x_k)$
  2. **stepsize**: search for good $\alpha_k > 0$
  3. **update**: $x_{k+1} = x_k - \alpha_k d_k$
  4. compute $f(x_{k+1})$ and $\nabla f(x_{k+1})$
  5. **QN update:** $S_k \to S_{k+1}$

QN schemes differ in how $S_k \equiv H_k^{-1}$ are updated!

Quasi-Newton methods

Secant equation / QN rule

$$S_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = x_{k+1} - x_k.$$

Quasi-Newton methods

Secant equation / QN rule

$$S_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = x_{k+1} - x_k.$$

- Quadratic models from iteration $k \to k+1$

$$\phi_k(x) = a_k + \langle g_k,\, x - x_k\rangle + \tfrac{1}{2}\langle H(x - x_k),\, x - x_k\rangle$$
$$\phi_{k+1}(x) = a_{k+1} + \langle g_{k+1},\, x - x_{k+1}\rangle + \tfrac{1}{2}\langle H(x - x_{k+1}),\, x - x_{k+1}\rangle$$

Quasi-Newton methods

Secant equation / QN rule

$$S_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = x_{k+1} - x_k.$$

- Quadratic models from iteration $k \to k+1$

$$\begin{aligned} \phi_k(x) &= a_k + \langle g_k,\, x - x_k\rangle + \tfrac{1}{2}\langle H(x - x_k),\, x - x_k\rangle \\ \phi_{k+1}(x) &= a_{k+1} + \langle g_{k+1},\, x - x_{k+1}\rangle + \tfrac{1}{2}\langle H(x - x_{k+1}),\, x - x_{k+1}\rangle \end{aligned}$$

- $\phi_k'(x) - \phi_{k+1}'(x) = g_k - g_{k+1} + H(x_{k+1} - x_k)$

Quasi-Newton methods

Secant equation / QN rule

$$S_{k+1}(\nabla f(x_{k+1}) - \nabla f(x_k)) = x_{k+1} - x_k.$$

- Quadratic models from iteration $k \to k+1$

$$\begin{aligned} \phi_k(x) &= a_k + \langle g_k,\, x - x_k\rangle + \tfrac{1}{2}\langle H(x - x_k),\, x - x_k\rangle \\ \phi_{k+1}(x) &= a_{k+1} + \langle g_{k+1},\, x - x_{k+1}\rangle + \tfrac{1}{2}\langle H(x - x_{k+1}),\, x - x_{k+1}\rangle \end{aligned}$$

- $\phi_k'(x) - \phi_{k+1}'(x) = g_k - g_{k+1} + H(x_{k+1} - x_k)$
- Setting this to zero, we get

$$\begin{aligned} g_{k+1} - g_k &= H(x_{k+1} - x_k) \\ S(g_{k+1} - g_k) &= x_{k+1} - x_k. \end{aligned}$$

- So we construct $H_k \to H_{k+1}$ or $S_k \to S_{k+1}$ to respect this.

Hessian updates

- **Barzilai-Borwein** stepsize. Let $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$:

$$\min_H \|Hs_k - y_k\|, \qquad H = \alpha I.$$

Hessian updates

- **Barzilai-Borwein** stepsize. Let $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$:

$$\min_H \|Hs_k - y_k\|, \qquad H = \alpha I.$$

- **Davidon-Fletcher-Powell** (DFP): $\beta := 1/\langle y_k,\, s_k\rangle$

$$H_{k+1} = (I - \beta y_k s_k^T) H_k (I - \beta s_k y_k^T) + \beta y_k y_k^T$$
$$S_{k+1} = S_k - \frac{S_k s_k s_k^T S_k}{\langle S_k s_k,\, s_k\rangle} + \beta y_k y_k^T.$$

Hessian updates

- **Barzilai-Borwein** stepsize. Let $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$:

$$\min_H \|Hs_k - y_k\|, \qquad H = \alpha I.$$

- **Davidon-Fletcher-Powell** (DFP): $\beta := 1/\langle y_k,\, s_k\rangle$

$$H_{k+1} = (I - \beta y_k s_k^T) H_k (I - \beta s_k y_k^T) + \beta y_k y_k^T$$

$$S_{k+1} = S_k - \frac{S_k s_k s_k^T S_k}{\langle S_k s_k,\, s_k\rangle} + \beta y_k y_k^T.$$

- **Broyden-Fletcher-Goldfarb-Shanno** (BFGS)

$$S_{k+1} = (I - \beta s_k y_k^T) S_k (I - \beta y_k s_k^T) + \beta s_k s_k^T$$

$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{\langle H_k s_k,\, s_k\rangle} + \beta y_k y_k T.$$

Hessian updates

- **Barzilai-Borwein** stepsize. Let $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$:

$$\min_H \|H s_k - y_k\|, \qquad H = \alpha I.$$

- **Davidon-Fletcher-Powell** (DFP): $\beta := 1/\langle y_k,\, s_k\rangle$

$$H_{k+1} = (I - \beta y_k s_k^T) H_k (I - \beta s_k y_k^T) + \beta y_k y_k^T$$
$$S_{k+1} = S_k - \frac{S_k s_k s_k^T S_k}{\langle S_k s_k,\, s_k\rangle} + \beta y_k y_k^T.$$

- **Broyden-Fletcher-Goldfarb-Shanno** (BFGS)

$$S_{k+1} = (I - \beta s_k y_k^T) S_k (I - \beta y_k s_k^T) + \beta s_k s_k^T$$
$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{\langle H_k s_k,\, s_k\rangle} + \beta y_k y_k T.$$

BFGS believed to be most stable, best scheme.

Hessian updates

- **Barzilai-Borwein** stepsize. Let $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$:

$$\min_H \|Hs_k - y_k\|, \qquad H = \alpha I.$$

- **Davidon-Fletcher-Powell** (DFP): $\beta := 1/\langle y_k, s_k \rangle$

$$H_{k+1} = (I - \beta y_k s_k^T) H_k (I - \beta s_k y_k^T) + \beta y_k y_k^T$$
$$S_{k+1} = S_k - \frac{S_k s_k s_k^T S_k}{\langle S_k s_k, s_k \rangle} + \beta y_k y_k^T.$$

- **Broyden-Fletcher-Goldfarb-Shanno** (BFGS)

$$S_{k+1} = (I - \beta s_k y_k^T) S_k (I - \beta y_k s_k^T) + \beta s_k s_k^T$$
$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{\langle H_k s_k, s_k \rangle} + \beta y_k y_k T.$$

  BFGS believed to be most stable, best scheme.

- Notice, updates computationally "cheap"

Limited memory methods

$$\boxed{\text{Hessian storage and update has } O(n^2) \text{ cost}}$$

Limited memory methods

Hessian storage and update has $O(n^2)$ cost

Estimate H_k or S_k using only previous few iterations; so essentially, use only $O(mn)$ storage, where $m \approx$ 5-17

- Each step of BFGS is: $x_{k+1} = x_k - \alpha_k S_k \nabla f(x_k)$

Limited memory methods

Hessian storage and update has $O(n^2)$ cost

Estimate H_k or S_k using only previous few iterations; so essentially, use only $O(mn)$ storage, where $m \approx$ 5-17

- Each step of BFGS is: $x_{k+1} = x_k - \alpha_k S_k \nabla f(x_k)$
- S_k is updated at every iteration using

$$S_{k+1} = V_k^T S_k V_k + \beta_k s_k s_k^T$$

Limited memory methods

Hessian storage and update has $O(n^2)$ cost

Estimate H_k or S_k using only previous few iterations; so essentially, use only $O(mn)$ storage, where $m \approx$ 5-17

- Each step of BFGS is: $x_{k+1} = x_k - \alpha_k S_k \nabla f(x_k)$
- S_k is updated at every iteration using

$$S_{k+1} = V_k^T S_k V_k + \beta_k s_k s_k^T$$

where, with $s_k := x_{k+1} - x_k$ and $y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$,

$$\beta_k = \frac{1}{y_k^T s_k}, \quad V_k = I - \beta_k y_k s_k^T,$$

Limited memory methods

Hessian storage and update has $O(n^2)$ cost

Estimate H_k or S_k using only previous few iterations; so essentially, use only $O(mn)$ storage, where $m \approx$ 5-17

- Each step of BFGS is: $x_{k+1} = x_k - \alpha_k S_k \nabla f(x_k)$
- S_k is updated at every iteration using

$$S_{k+1} = V_k^T S_k V_k + \beta_k s_k s_k^T$$

where, with $s_k := x_{k+1} - x_k$ and $y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$,

$$\beta_k = \frac{1}{y_k^T s_k}, \quad V_k = I - \beta_k y_k s_k^T,$$

- We use m vector pairs (s_i, y_i), for $i = k-m, \ldots, k-1$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$S_k = (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1})$$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$\begin{aligned} S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\ &+ \beta_{k-m} (V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \end{aligned}$$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$\begin{aligned}
S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\
&+ \beta_{k-m} (V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \\
&+ \beta_{k-m+1} (V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2}^T \cdots V_{k-1}^T)
\end{aligned}$$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$\begin{aligned}
S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\
&+ \beta_{k-m}(V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \\
&+ \beta_{k-m+1}(V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2}^T \cdots V_{k-1}^T) \\
&+ \cdots \\
&+ \beta_{k-1} s_{k-1} s_{k-1}^T.
\end{aligned}$$

Ultimate aim is to efficiently compute: $S_k \nabla f(x_k)$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$\begin{aligned} S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\ &+ \beta_{k-m}(V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \\ &+ \beta_{k-m+1}(V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2}^T \cdots V_{k-1}^T) \\ &+ \cdots \\ &+ \beta_{k-1} s_{k-1} s_{k-1}^T. \end{aligned}$$

Ultimate aim is to efficiently compute: $S_k \nabla f(x_k)$

Exercise: Implement procedure to compute $S_k \nabla f(x_k)$ efficiently.

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$
\begin{aligned}
S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\
&+ \beta_{k-m} (V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \\
&+ \beta_{k-m+1} (V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2}^T \cdots V_{k-1}^T) \\
&+ \cdots \\
&+ \beta_{k-1} s_{k-1} s_{k-1}^T .
\end{aligned}
$$

$$\boxed{\text{Ultimate aim is to efficiently compute: } S_k \nabla f(x_k)}$$

Exercise: Implement procedure to compute $S_k \nabla f(x_k)$ efficiently.

- Typical choice for $S_k^0 = \dfrac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}} I$

Limited memory methods

Unroll the S_k update loop for m iterations to obtain

$$\begin{aligned} S_k &= (V_{k-1}^T \cdots V_{k-m}^T) S_k^0 (V_{k-m} \cdots V_{k-1}) \\ &+ \beta_{k-m}(V_{k-1}^T \cdots V_{k-m+1}^T) s_{k-m} s_{k-m}^T (V_{k-m+1}^T \cdots V_{k-1}^T) \\ &+ \beta_{k-m+1}(V_{k-1}^T \cdots V_{k-m+2}^T) s_{k-m+1} s_{k-m+1}^T (V_{k-m+2}^T \cdots V_{k-1}^T) \\ &+ \cdots \\ &+ \beta_{k-1} s_{k-1} s_{k-1}^T. \end{aligned}$$

Ultimate aim is to efficiently compute: $S_k \nabla f(x_k)$

Exercise: Implement procedure to compute $S_k \nabla f(x_k)$ efficiently.

- Typical choice for $S_k^0 = \dfrac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}} I$
- This is related to the BB stepsize!

Constrained problems

Constrained problems

Two-metric projection method

$$\boxed{x_{k+1} = P_{\mathcal{X}}(x_k - \alpha_k S_k \nabla f(x_k))}$$

Constrained problems

Two-metric projection method

$$\boxed{x_{k+1} = P_{\mathcal{X}}(x_k - \alpha_k S_k \nabla f(x_k))}$$

- Fundamental problem: **not a descent iteration!**

Constrained problems

Two-metric projection method

$$\boxed{x_{k+1} = P_{\mathcal{X}}(x_k - \alpha_k S_k \nabla f(x_k))}$$

- Fundamental problem: **not a descent iteration!**
- We may have $f(x_{k+1}) > f(x_k)$ for all $\alpha_k > 0$

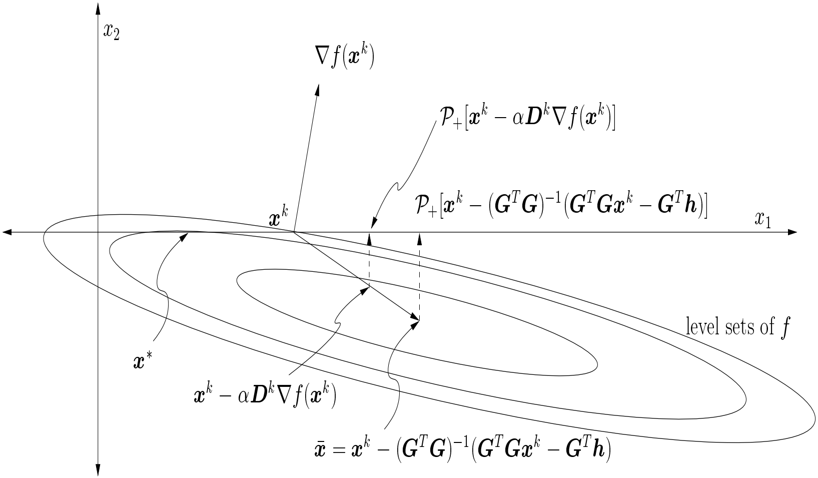
Constrained problems

Two-metric projection method

$$\boxed{x_{k+1} = P_{\mathcal{X}}(x_k - \alpha_k S_k \nabla f(x_k))}$$

- Fundamental problem: **not a descent iteration!**
- We may have $f(x_{k+1}) > f(x_k)$ for all $\alpha_k > 0$
- Method might not even **recognize a stationary point!**

Failure of projected-Newton methods



Constrained problems

- Projected-gradient works! **BUT**

Constrained problems

- Projected-gradient works! **BUT**
- Projected Newton or Quasi-Newton **do not work!**

Constrained problems

- Projected-gradient works! **BUT**
- Projected Newton or Quasi-Newton **do not work!**
- More careful selection of S_k (or H_k) needed

Constrained problems

- Projected-gradient works! **BUT**
- Projected Newton or Quasi-Newton **do not work!**
- More careful selection of S_k (or H_k) needed
- See e.g., Bertsekas and Gafni (Projected QN) (1984)

Constrained problems

- Projected-gradient works! **BUT**
- Projected Newton or Quasi-Newton **do not work!**
- More careful selection of S_k (or H_k) needed
- See e.g., Bertsekas and Gafni (Projected QN) (1984)
- With simple bound constraints: LBFGS-B

Nonsmooth problems

We did not cover many interesting ideas

- ♠ Proximal Newton methods
- ♠ $f(x) + r(x)$ problems (see book chapter)
- ♠ Nonsmooth BFGS – Lewis, Overton
- ♠ Nonsmooth LBFGS

References

♡ Y. Nesterov. *Introductory Lectures on Convex Optimization* (2004).

♡ J. Nocedal, S. J. Wright. *Numerical Optimization* (1999).

♡ M. Schmidt, D. Kim, S. Sra. *Newton-type methods in machine learning*, Chapter 13 in *Optimization for Machine Learning* (2011).