Quasi-Newton methods

Pierre Ablin

Framework

Objective: solve

$$\min_{x \in \mathbb{R}^p} f(x)$$

where f is **twice differentiable** (the Hessian matrix exists).

Not seen in the course:

- Prox-Newton methods for the twice differentiable + proximable penalty case
- Constrained methods (x is constrained to a subset of $\mathbb{R}^p$)
- Stochastic quasi-Newton methods (when f is a sum)

These slides are mostly based on the excellent and very well written ***Numerical Optimization*** by Nocedal and Wright.

Overview

Today: second order methods !

- Newton's method
- DFP and BFGS

These methods are widely used and are state-of-the-art for some large scale smooth problems (e.g. ℓ_2 logistic regression).

Differential calculus in $\mathbb{R}^p$ 101

$$f : \mathbb{R}^p \to \mathbb{R} \quad \text{twice differentiable}$$

Gradient: $\nabla f(x) = [\frac{\partial f}{\partial x_1}(x), \cdots, \frac{\partial f}{\partial x_p}(x)] \in \mathbb{R}^p$

Hessian: $\nabla^2 f(x) = [\frac{\partial^2 f}{\partial x^i \partial x^j}(x)]_{ij} \in \mathbb{R}^{p \times p}$

Second order Taylor expansion ($\langle a, b\rangle = a^\top b$):

$$f(x+\varepsilon) = f(x) + \quad \langle \varepsilon, \nabla f(x)\rangle \quad + \quad \frac{1}{2}\langle \varepsilon, \nabla^2 f(x)\varepsilon\rangle \qquad + o(||\varepsilon||^2)$$

$$f(x+\varepsilon) = f(x) + \sum_{i=1}^{p} \varepsilon_i \frac{\partial f}{\partial x^i}(x) + \frac{1}{2}\sum_{i,j=1}^{p} \varepsilon_i \varepsilon_j \frac{\partial^2 f}{\partial x^i \partial x^j}(x) + o(||\varepsilon||^2)$$

Hessian 101

$f : \mathbb{R}^p \to \mathbb{R}$ twice differentiable

$\nabla^2 f(x) = [\frac{\partial^2 f}{\partial x^i \partial x^j}(x)]_{ij} \in \mathbb{R}^{p \times p}$

- f is convex iff $\nabla^2 f(x)$ is positive for all x ($\nabla^2 f(x) \succeq 0$)
- f is μ strongly convex iff $\forall x,\ \nabla^2 f(x) \succeq \mu I_p$
- f is $L-$smooth iff $\forall x,\ \nabla^2 f(x) \preceq L \cdot I_p$

Newton's method

Start at a point x_0.
For an iterate x_t, second order Taylor expansion:

$$f(x_t + \varepsilon) \simeq f(x_t) + \langle \varepsilon, \nabla f(x_t) \rangle + \frac{1}{2} \langle \varepsilon, \nabla^2 f(x_t) \varepsilon \rangle = Q_{x_t}(\varepsilon)$$

Exercise:
Can you minimize the right hand side with respect to ε?

Answer

Minimize $Q_{x_t}(\varepsilon) = f(x_t) + \langle \varepsilon, \nabla f(x_t) \rangle + \frac{1}{2}\langle \varepsilon, \nabla^2 f(x_t)\varepsilon \rangle$ w.r.t ε

It all depends on $\nabla^2 f(x_t)$!

- If $\nabla^2 f(x_t) \succ 0$, f is locally strongly convex ☺

$$\rightarrow \varepsilon^* = -[\nabla^2 f(x_t)]^{-1} \nabla f(x_t)$$

- Otherwise, x_t is a saddle / concave point ☹ (impossible if convex)

$\rightarrow$ No minimum

Newton's Method (preliminary version.)

Newton's method iterates:

$$\boxed{\begin{aligned} p_t &= -[\nabla^2 f(x_t)]^{-1} \nabla f(x_t) \\ x_{t+1} &= x_t + p_t \end{aligned}}$$

Exercises:

- Show that it converges in one step on a quadratic problem.
- Does it always converge?
- Is it a guaranteed descent method? What if the problem is convex?

Newton's Method converges in one step on a quadratic problem...

because the second order Taylor expansion is exact in this case.
Does not depend on the conditionning (unlike gradient descent).

It does not require the problem to be convex, Newton's method finds the **stationnary points** ($\nabla f = 0$).

Attracted to saddle points !

Does it always converge? Guaranteed descent?

No !

Cf code example...

Q_{x_t} is not a majorizing function, for some f (even convex):

$$f(x_t + \varepsilon) \not\leq f(x_t) + \langle \varepsilon, \nabla f(x_t) \rangle + \frac{1}{2} \langle \varepsilon, \nabla^2 f(x_t) \varepsilon \rangle$$

How to fix it?

Guaranteed descent: for α small:

$$f(x_t + \alpha p_t) \simeq f(x_t) + \alpha \langle \nabla f(x_t), p_t \rangle \ .$$

p_t is a descent direction if and only if $\boxed{\langle \nabla f(x_t), p_t \rangle < 0}$.
Recall that $p_t = -[\nabla^2 f(x_t)]^{-1} \nabla f(x_t)$. Safe condition for guaranteed descent: $\nabla^2 f(x_t) \succ 0$ (sufficient but not necessary).
Regularization: $\nabla^2 f(x_t) \mathrel{+}= \lambda I_p$ for λ large enough.

Use a line-search to guarantee convergence

Newton's Method (final version.)

Note: unless there is an ambiguity, $g_t = \nabla f(x_t)$.

- Compute $H_t = \nabla^2 f(x_t)$ and regularize it if it is not positive.
- Set $p_t = -H_t^{-1} g_t$
- Find α_t using line-search
- $x_{t+1} = x_t + \alpha_t p_t$

Theorem[Convergence]: If the set $\{x \in \mathbb{R}^p | f(x) \leq f(x_0)\}$ is compact, and $||H_t|| \times ||H_t^{-1}||$ is bounded, $\lim_{t \to +\infty} \nabla f(x_t) = 0$.

Theorem[Quadratic rate]: Assume that $x^* \in \mathbb{R}^p$ is such that $\nabla f(x) = 0$ and $\nabla^2 f(x) \succ 0$. Then, Newton's method starting close enough from x^* will converge to x^* at a quadratic rate:

$$||x_{t+1} - x^*|| = O(||x_t - x^*||^{\mathbf{2}})$$

Drawbacks of Newton's method

Quadratic convergence is an interesting property, but most of the time, Newton's method is **too costly**!

- Computing the Hessian is p times more costly in time and memory than the gradient ☹
- If the problem is non-convex, regularization is hard and costly
- Then, need to compute $H_t^{-1}\nabla f(x_t) \ \rightarrow O(p^3)$
- What if $p = 10000$?

Quasi-Newton methods: try to mimic Newton's direction without the computational load.

Quasi-Newton's methods

Uses an approximation of the Hessian:

$$\begin{cases} \text{Compute } H_t \\ p_t = -H_t^{-1} g_t \\ x_{t+1} = x_t + \alpha_t p_t \end{cases} \tag{1}$$

Or of the inverse of the Hessian:

$$\begin{cases} \text{Compute } B_t \\ p_t = -B_t g_t \\ x_{t+1} = x_t + \alpha_t p_t \end{cases} \tag{2}$$

Rest of the class: How do you find good Hessian / Inverse Hessian approximations?

Important note: In practice B_t/H_t are never stored as $p \times p$ matrices, but in an intermediate form that takes less memory and simplifies the computation of $H_t^{-1} g_t$.

Exercise

$$x_{t+1} = x_t - \alpha_t B_t \nabla f(x_t) \tag{3}$$

For a $p \times p$ matrix C, define $y = C^{-1}x$, and $\tilde{f}(y) = f(Cy)$.

- What are the gradient, Hessian of $\tilde{f}$?
- Show that the update (3) corresponds to a *gradient* descent move on $\tilde{f}$ for a specific C.

So: Quasi-Newton methods can be seen as **gradient descent + variable metric**.

Secant condition

B_t or H_t are **updated** after each step, using the knowledge gained after a step.

Key idea: The change in ∇f provides information about $\nabla^2 f$ along the search direction!

Exercise

Show that:

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

notation: $y_t = g_{t+1} - g_t$, $s_t = x_{t+1} - x_t$

Secant condition: Impose $\begin{cases} H_{t+1} s_t = y_t \\ or \\ B_{t+1} y_t = s_t \end{cases}$

$\rightarrow$ Constrains H_{t+1} in the search direction.

Iterative update of B_t or H_t

What else do we want from B_t/H_t?

- Symmetry
- Positivity ($B_t g_t$ or $H_t^{-1} g_t$ should be descent direction)

Idea: Start from $H_0/B_0 = \lambda I_p$, and update:

$$H_{t+1}/B_{t+1} = H_t/B_t + \Delta_t, \text{ such that } H/G \text{ remains positive.}$$

Important efficiency constraint: computing $H_t^{-1} g_t$ or $B_t g_t$ should be quick.
$\rightarrow$ Perform small rank (1 or 2) updates

Broyden / SR1 method

Rank one update on H_t:

$$H_{t+1} = H_t + \sigma v v^\top, \;\; \sigma = \pm 1$$

Exercise

Recall the secant condition : $H_{t+1} s_t = y_t$. Derive the formula for σ, v accordingly.

Broyden / SR1 method

Rank one update on H_t:

$$H_{t+1} = H_t + \sigma v v^\top, \;\; \sigma = \pm 1$$

Exercise

Recall the secant condition : $H_{t+1} s_t = y_t$. Derive the formula for σ, v accordingly.

SR1 updates: $H_{t+1} = H_t + \frac{(y_t - H_t s_t)(y_t - H_t s_t)^\top}{(y_t - H_t s_t)^\top s_t}$

Sherman-Morisson formula: $B_{t+1} = B_t + \frac{(s_t - B_t y_t)(s_t - B_t y_t)^\top}{(s_t - B_t y_t)^\top y_t}$

Important theorem

Let f a **quadratic function** with Hessian $A \succ 0$. Starting from any p.s.d. matrix B_0, the sequence of Hessians produced by the SR1 method with **perfect** line-search (i.e. $\alpha_t = \arg\min_\alpha f(x_t + \alpha p_t)$) verifies:

$$H_p = A$$

Consequently, SR1 converges in at most $p + 1$ iterations.

DFP/BFGS methods

Problems with SR1: not guaranteed positive (we may have $\sigma = -1$) and denominator may vanish $\rightarrow$ not enough to do a rank 1 update...

$\rightarrow$ impose that H_{t+1} or B_{t+1} is the closest to H_t/B_t in some sense.
Davidon-Fletcher-Powell (DFP) method :

$$H_{t+1} = \arg\min_H ||H - H_t|| \text{ s.t. } Hs_t = y_t$$

Broyden-Fletcher-Goldfarb-Shanno (BFGS method):

$$B_{t+1} = \arg\min_B ||B - B_t|| \text{ s.t. } By_t = s_t$$

(Note: the norm is the *weighted* norm $||H|| = ||W^{1/2}HW^{1/2}||_F$, where W is any matrix such that $Wy_t = s_t$)

DFP method

$$H_{t+1} = \arg\min ||H - H_t|| \text{ s.t. } Hs_t = y_t$$

Leads to rank 2 updates on $\mathbf{B_t}$:

$$\boxed{B_{t+1} = B_t + \frac{s_t s_t^\top}{s_t^\top y_t} - \frac{B_t y_t y_t^\top B_t}{y_t^\top B_t y_t}} \tag{4}$$

DFP algorithm:

Start from x_0, $B_0 \succ 0$, and iterate until convergence

- $p_t = -B_t g_t$
- $x_{t+1} = x_t + \alpha_t p_t$ (α_t found by line-search)
- Update B_t using eq.4

Theorem: With **optimal** line-search, B_t remains p.s.d. ☺

Properties of DFP

Theorem[DFP on a quadratic function]:Let f a quadratic function of Hessian $A \succ 0$. DFP on f satisfies $B_p = A^{-1}$, and therefore converges in $p+1$ iterations.

Theorem[Quadratic convergence] For a twice differentiable function f, under mild assumptions, DFP converges to a local minimum x^* of f. Further, $\lim_{t\to\infty} B_t = \nabla^2 f(x^*)^{-1}$. Therefore, the convergence is quadratic.

BFGS method

$$B_{t+1} = \arg\min ||B - B_t|| \text{ s.t. } Bs_t = y_t$$

Leads to rank 2 updates on $\mathbf{H_t}$:

$$\boxed{H_{t+1} = H_t + \frac{y_t y_t^\top}{y_t^\top s_t} - \frac{H_t s_t s_t^\top H_t}{s_t^\top H_t s_t}} \tag{5}$$

BFGS algorithm:

Start from x_0, $H_0 \succ 0$, and iterate until convergence

- $p_t = -H_t^{-1} g_t$
- $x_{t+1} = x_t + \alpha_t p_t$ (α_t found by line-search)
- Update H_t using eq.5

BFGS method

$$H_{t+1} = H_t + \frac{y_t y_t^\top}{y_t^\top s_t} - \frac{H_t s_t s_t^\top H_t}{s_t^\top H_t s_t}$$

Exercise: Use Sherman-Morrison formula $(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1+v^\top A^{-1}u}$ to derive the updates for B_{t+1}.

BFGS method

$$H_{t+1} = H_t + \frac{y_t y_t^\top}{y_t^\top s_t} - \frac{H_t s_t s_t^\top H_t}{s_t^\top H_t s_t}$$

Exercise: Use Sherman-Morrison formula $(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1+v^\top A^{-1}u}$ to derive the updates for B_{t+1}.

$$B_{t+1} = (I_p - \rho_t s_t y_t^\top) B_t (I_p - \rho_t y_t s_t^\top) + \rho_t s_t s_t^\top, \;\; \rho_t = \frac{1}{y_t^\top s_t}$$

BFGS has the same properties as DFP:

- Converges in $p+1$ iterations on a quadratic problem, and perfectly matches the Hessian at iteration p.
- Quadratic convergence in the general case.

BUT:

- Less sensitive than DFP to errors in the line-search $\rightarrow$ more efficient

L-BFGS method

- Limited memory of BFGS, proposed by Liu and Nocedal 89.
- Memory of size m. (usually $m = 10$)
- Does not store the full $p \times p$ Hessian in memory, but the past m values of s_t and y_t.
- Memory loading linear in p.

Idea: Iterate the BFGS formula

$$B_{t+1} = (I_p - \rho_t s_t y_t^\top) B_t (I_p - \rho_t y_t s_t^\top) + \rho_t s_t s_t^\top, \;\; \rho_t = \frac{1}{y_t^\top s_t}$$

only m times.

L-BFGS method

Notation: $V_t = I_p - \rho_t y_t s_t^\top$

$$B_{t+1} = V_t^\top B_t V_t + \rho_t s_t s_t^\top, \;\; \rho_t = \frac{1}{y_t^\top s_t}$$

At each iteration: Start from an initial inverse Hessian B_t^0 (can vary, usually λI_p), and:

$$\begin{aligned}
B_t =& (V_{t-1}^\top \cdots V_{t-m}^\top) B_t^0 (V_{t-m} \cdots V_{t-1}) \\
&+ \rho_{t-m} (V_{t-1}^\top \cdots V_{t-m+1}^\top) s_{t-m} s_{t-m}^\top (V_{t-m+1} \cdots V_{t-1}) \\
&+ \rho_{t-m+1} (V_{t-1}^\top \cdots V_{t-m+2}^\top) s_{t-m+1} s_{t-m+1}^\top (V_{t-m+2} \cdots V_{t-1}) \\
&\cdots \\
&+ \rho_{t-1} s_{t-1} s_{t-1}^\top
\end{aligned}$$

These are just mathematical equations, which lead to an efficient recursive way of computing $B_t g_t$ (the previous matrices are never computed!)

Two loops recursion for L-BFGS

The following algorithm is an efficient recursion to compute $B_t g_t$ witout explicitely computing B_t:

- Set $q = g_t$
- For $i = t-1, \cdots, t-m$:
  - $\alpha_i = \rho_i s_i^\top q$
  - $q = q - \alpha_i y_i$
- $r = B_t^0 q$
- For $i = t-m, \cdots, t-1$:
  - $\beta = \rho_i y_i^\top r$
  - $r = r + (\alpha_i - \beta) s_i$
- Return $r = B_t g_t$

Advantages of L-BFGS

- Low memory cost: only need to store $(y_i)_{i\in[t-m,t-1]}$ and $(s_i)_{i\in[t-m,t-1]} \to 2\times m\times p$.
- Computation of the descent direction is also $O(m\times p)$
- On most problem, the limited memory is actually an advantage because it **forgets the outdated landscape**!
- Can change of initial inverse Hessian guess B_t^0 at each iteration (sometimes there are some very good approximations available)
- In most cases, L-BFGS is the superior Quasi-Newton method.