Quasi-Newton Methods

Friday, March 26, 2021 9:56 AM

following Nocedal + Wright textbook

gold-standard for largescale smooth, unconstrained optimization

Quadratic approximation at the current iterate x_k (let $p = x - x_k$)

$$m_k(p) = \underbrace{f_k}_{f(x_k)} + \langle \underbrace{\nabla f_k}_{\nabla f(x_k)}, p \rangle + \tfrac{1}{2}\langle p | B_k | p \rangle$$

$\underline{B_k \succ 0}$

$\Rightarrow m$ is strongly convex

$\tilde{x}_{k+1} = x_k + p_k, \quad p_k = \operatorname{argmin}_p m_k(p)$

Ex: Gradient Descent $\quad B_k = L \cdot I$, $\quad L$ is Lipsch. constant of ∇f (ie., $\nabla^2 f(x_k) \preceq L \cdot I$)

Ex: Newton's Method $\quad B_k = \nabla^2 f(x_k)$

$$\boxed{p = -B_k^{-1} \cdot \nabla f_k}$$

Sometimes do linesearch, $x_{k+1} = x_k + \alpha \cdot p^*$ $\quad \alpha \leq 1$

Quasi-Newton method

means use the above framework w/ B_k such that

1. B_k is more accurate than $L \cdot I$
2. B_k is cheaper than Newton
   both forming B_k and inverting B_k

Main trick is to construct B_{k+1} by updating B_k

To find x_{k+2}, minimize $\quad m_{k+1}(p) := f_{k+1} + \langle \nabla f_{k+1}, p \rangle + \tfrac{1}{2} \langle p | B_{k+1} | p \rangle$

$$\nabla m_{k+1}(p) = \nabla f_{k+1} + B_{k+1} \cdot p$$

so $\nabla m_{k+1}(0) = \nabla f_{k+1}$

we've automatically enforced: $\left.\begin{array}{l} ① \ m_{k+1}(0) = f_{k+1} \\ ② \ \nabla m_{k+1}(0) = \nabla f_{k+1} \end{array}\right\} x_{k+1}$

Freedom in B_{k+1}

let's impose $\left.③ \ \nabla m_{k+1}(\text{previous iterate}) = \nabla f_k \right\} x_k$

$x_{k+1} = x_k + p_k$

Notation

$s_k = x_{k+1} - x_k$

$y_k = \nabla f_{k+1} - \nabla f_k$

$p = x - x_{k+1}$, x_k previous iterate, $p = x_k - x_{k+1} = -s_k$

③ $\nabla m_{k+1}(-s_k) = \nabla f_{k+1} - B_{k+1} \cdot s_k \overset{\text{desired}}{=} \nabla f_k$

$$\boxed{B_{k+1} \cdot s_k = y_k}$$ "secant equation"

Most/all quasi-Newton methods choose B_{k+1} to satisfy the secant equation.

Observe: want $B_{k+1} \succ 0$

$$\underbrace{\langle s_k | B_{k+1} | s_k \rangle}_{>0} = \langle s_k | y_k \rangle$$

$\langle s_k | y_k \rangle > 0$ is necessary. — curvature condition

i.e., $\underbrace{\langle x_{k+1} - x_k, \nabla f_{k+1} - \nabla f_k \rangle}_{\text{strictly monotone}} \overset{?}{>} 0$

happens automatically if f is strictly cvx.

So if f isn't strictly cvx, that complicates quasi-Newton method.
eg. add a linesearch

B_{k+1} chosen to solve secant equations. $x \in \mathbb{R}^n$ so $B_{k+1} \in S^{n \times n}$

deg. of freedom of B_{k+1} is $\frac{n \cdot (n+1)}{2}$ — n constraints

Ex: $n=1$, 1 d.o.f., 1 constraint. B_{k+1} is completely determined

Secant method

Ex: $n=2$, 3 d.o.f., 2 constraints.

$\Rightarrow$ $n>1$, there's a whole family of reasonable quasi-Newton methods

How to choose B_{k+1}?

Standard ways: either solve $B_{k+1} = \underset{\substack{B \succ 0 \\ B \cdot s_k = y_k}}{\operatorname{argmin}} \|B - B_k\|_w^2$

or $\underset{H_{k+1}}{B_{k+1}^{-1}} = \underset{\substack{B^{-1} \succ 0 \\ B^{-1} y_k = s_k}}{\operatorname{argmin}} \underset{\|B^{-1} - B_k^{-1}\|_w^2}{\|H - H_k\|_w^2}$

Notation

B approximates $\nabla^2 f$

H approximates $(\nabla^2 f)^{-1}$

$\underbrace{\|B\|_w \text{ is some norm, chosen so}}_{\text{Broyden class}}$ problem ↑ has a closed-form solution.

$B_k^{-1} = H_k$

Issue of inverting B_k : not an issue,

1) use H_k not B_k

and/or 2) B_{k+1} is a low-rank update of B_k $\qquad B_{k+1} = B_k + u \cdot u^T$

so use Sherman-Morrison-Woodbury formula

BFGS

Works w/ H update above, special choice of weighted norm

$$H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \quad \rho_k = \frac{1}{\langle y_k, s_k \rangle}$$

$$H_0 = \frac{\langle y_k, s_k \rangle}{\langle y_k, y_k \rangle} \cdot I$$ (aka Barzilai-Borwein or "spectral" stepsize)

Benefits:

- more accurate than gradient descent
- cheaper than Newton ($O(n^2)$ to update H_{k+1}, rather than $O(n^3)$ for Newton to invert $\nabla^2 f_k$)

Disadvantages

- still needs $O(n^2)$ memory

fix

Theory: iffy.

If f is nonconvex, does BFGS always converge to a stationary pt.?

unknown!

Thm: Nocedal + Wright, Thm 6.5

If $0 < \mu I \preceq \nabla^2 f(x) \preceq L \cdot I \quad \forall x$, then BFGS sequence converges to the global minimizer.

Thm: local convergence is often super-linear

(see book for details)

Limited-memory BFGS "L-BFGS"

Recall BFGS update $\rho_k = \frac{1}{\langle y_k, s_k \rangle}$, let $V_k = I - \rho_k y_k s_k^T$

$s_k = x_{k+1} - x_k$

$y_k = \nabla f_{k+1} - \nabla f_k$

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T$$

so to compute matrix-vector multiply (eg., w/ $x = \nabla f(x_{k+1})$)

$$H_{k+1} \cdot x = V_k^T H_k \underbrace{V_k x}_{z} + \underbrace{\rho_k s_k s_k^T x}_{\text{easy}}$$

$z = V_k x$ easy

What about $H_k \cdot z$?

compute recursively! $H_{k+1} \cdot x$ computed using $\{s_k, y_k, H_k\}$

For BFGS, recurse until $k=0$, $H_0 = \text{const} \cdot I$.

For L-BFGS, only recurse back $m \ll n$ steps,

and set "base case" to $H_0^{(k)} = \gamma_k \cdot I$, $\gamma_k = \dfrac{\langle s_{k-1}, y_{k-1} \rangle}{\|y_{k-1}\|^2}$

Only need to store $O(mn)$ data

$\{s_i, y_i\}_{i=k-m+1}^{i=m}$ and flop count for $H_{k+1} \cdot x$ is $O(mn)$

Typically $m \in \{3, 20\}$

Cool fact: $m=0$ "memoryless" BFGS, w/ exact linesearch, is the HS variant of nonlinear CG.