Chapter 11 Quasi-Newton Methods

1. Approximating the Inverse Hessian
2. Rank One Correction Formula
3. DFP Algorithm
4. BFGS Algorithm
5. BB Algorithm

Newton and Secant Methods

1D case: $\min\limits_{x\in\mathbb{R}} f(x)$

- Newton's method:

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}.$$

- secant method:

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{F(x^{(k)})},$$

where $F(x^{(k)}) = \frac{f'(x^{(k)}) - f'(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$ is an approximation of $f''(x^{(k)})$.

nD case:

- Newton's method: $\min\limits_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)},$$

where $\boldsymbol{F}(\boldsymbol{x}^{(k)}) = \nabla^2 f(\boldsymbol{x}^{(k)})$, $\boldsymbol{g}^{(k)} = \nabla f(\boldsymbol{x}^{(k)})$.

- quasi-Newton/secant methods: approximates $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ or $[\boldsymbol{F}(\boldsymbol{x}^{(k)})]^{-1}$, i.e.,

$$\begin{cases} \boldsymbol{H}_k \approx [\boldsymbol{F}(\boldsymbol{x}^{(k)})]^{-1}, \\ \boldsymbol{B}_k \approx \boldsymbol{F}(\boldsymbol{x}^{(k)}), \end{cases}$$

version 1: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{H}_k\boldsymbol{g}^{(k)}$,

version 2: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - (\boldsymbol{B}_k)^{-1}\boldsymbol{g}^{(k)}$.

Version 1: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{H}_k \boldsymbol{g}^{(k)}$

Premises on $\boldsymbol{H}_k$

1. $\boldsymbol{H}_k \approx [\boldsymbol{F}(\boldsymbol{x}^{(k)})]^{-1}$ and $\boldsymbol{H}_k$ is symmetric positive definite;
2. trivial computation from $\boldsymbol{H}_k$ to $\boldsymbol{H}_{k+1}$, e.g., $\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \Delta_k$, where Δ_k is a "small" matrix.

Analysis: properties of the Hessian matrix $\boldsymbol{F}(\boldsymbol{x}^{(k+1)})$:

- 1D case: $\because f'(x^{(k)}) \approx f'(x^{(k+1)}) + f''(x^{(k+1)})(x^{(k)} - x^{(k+1)})$,

$$\therefore \boxed{f''(x^{(k+1)}) \approx \frac{f'(x^{(k+1)}) - f'(x^{(k)})}{x^{(k+1)} - x^{(k)}}}$$

- nD case: $\because \nabla f(\boldsymbol{x}^{(k)}) \approx \nabla f(\boldsymbol{x}^{(k+1)}) + \nabla^2 f(\boldsymbol{x}^{(k+1)})(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)})$,

$$\therefore \underbrace{\nabla f(\boldsymbol{x}^{(k)}) - \nabla f(\boldsymbol{x}^{(k+1)})}_{\boldsymbol{y}^{(k)}} \approx \underbrace{\nabla^2 f(\boldsymbol{x}^{(k+1)})}_{\boldsymbol{F}(\boldsymbol{x}^{(k+1)})} \underbrace{(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)})}_{\boldsymbol{s}^{(k)}}.$$

$$\iff \boxed{\boldsymbol{y}^{(k)} \approx \boldsymbol{F}(\boldsymbol{x}^{(k+1)})\boldsymbol{s}^{(k)}} \iff \boxed{[\boldsymbol{F}(\boldsymbol{x}^{(k+1)})]^{-1}\boldsymbol{y}^{(k)} \approx \boldsymbol{s}^{(k)}}.$$

Replacing $[\boldsymbol{F}(\boldsymbol{x}^{(k+1)})]^{-1}$ with $\boldsymbol{H}_{k+1}$ and oblige equation to be true, we have the quasi-Newton equation, also called secant equation.

Rank-One Correction

quasi-Newton equation: $\boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)}$. (1)

★ solution of (1) is nonunique because there are $\frac{n(n+1)}{2}$ degrees of freedom in symmetric matrix, and the secant equation represents n conditions. Thus, there are many choices on $\boldsymbol{H}_{k+1}$.

★ To guarantee $\boldsymbol{H}_{k+1} \succ 0$, it suffices the curvature condition ${\boldsymbol{s}^{(k)}}^\top \boldsymbol{y}^{(k)} > 0$ (which holds if the strong Wolfe conditions is used in line search).

rank-one correction of $\boldsymbol{H}_{k+1}$

Given $\boldsymbol{H}_k$, $\boldsymbol{y}^{(k)}$, $\boldsymbol{s}^{(k)}$, update $\boldsymbol{H}_{k+1}$ by solving the systems

$$\begin{cases} \boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)} \\ \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + a\boldsymbol{u}\boldsymbol{u}^\top \end{cases} \Longrightarrow \cdots \xRightarrow{\text{details}} \cdots \Longrightarrow \cdots$$

$$\Longrightarrow \begin{cases} \boldsymbol{u} = \boldsymbol{s}^{(k)} - \boldsymbol{H}_k\boldsymbol{y}^{(k)} \\ a = \frac{1}{\boldsymbol{y}^{(k)\top}(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k\boldsymbol{y}^{(k)})} \end{cases} \Longrightarrow \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k\boldsymbol{y}^{(k)})(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k\boldsymbol{y}^{(k)})^\top}{\boldsymbol{y}^{(k)\top}(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k\boldsymbol{y}^{(k)})}.$$

★ all the premises for $\boldsymbol{H}_{k+1}$ are fulfilled (check them).

Conjugate Property

Theorem (conjugate property)

If quasi-Newton method is applied to minimizing a quadratic function with Hessian $\boldsymbol{Q} = \boldsymbol{Q}^\top$ such that for $0 \le k < n-1$, $\boldsymbol{H}_{k+1}\boldsymbol{y}^{(i)} = \boldsymbol{s}^{(i)}$ for all $0 \le i \le k$, If $\alpha_i \neq 0$, $0 \le i \le k$, then $\{\boldsymbol{d}^{(i)} = \boldsymbol{H}_i \boldsymbol{g}^{(i)}\}_{i=0}^{k+1}$ are $\boldsymbol{Q}$-conjugate.

proof. (induction): recall that $\boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}$,

$$\therefore\ \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)} \overset{\text{quadratic function}}{=\!=\!=} \boldsymbol{Q}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}) = \boldsymbol{Q}\boldsymbol{s}^{(k)},$$

Recall that $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{H}_k \boldsymbol{g}^{(k)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)}$.

(1) case of $k = 0$. $\because\ \alpha_0 \neq 0$, $\therefore\ \boldsymbol{d}^{(0)} = -\frac{\boldsymbol{s}^{(0)}}{\alpha_0}$.

$$\because (\boldsymbol{d}^{(1)})^\top \boldsymbol{Q}\boldsymbol{d}^{(0)} = -(\boldsymbol{g}^{(1)})^\top \boldsymbol{H}_1 \boldsymbol{Q}\boldsymbol{d}^{(0)} = (\boldsymbol{g}^{(1)})^\top \boldsymbol{H}_1 \frac{\boldsymbol{Q}\boldsymbol{s}^{(0)}}{\alpha_0}$$

$$= (\boldsymbol{g}^{(1)})^\top \frac{\boldsymbol{H}_1 \boldsymbol{y}^{(0)}}{\alpha_0} \overset{\text{secant equation}}{=\!=\!=} (\boldsymbol{g}^{(1)})^\top \frac{\boldsymbol{s}^{(0)}}{\alpha_0} = (\boldsymbol{g}^{(1)})^\top \boldsymbol{d}^{(0)} \overset{\text{exact } \alpha_k}{=\!=\!=} 0,$$

$\therefore\ \boldsymbol{d}^{(0)}$ and $\boldsymbol{d}^{(1)}$ are $\boldsymbol{Q}$-conjugate.

Conjugate Property

(2) Assume the conjugacy holds for $k-1$ (where $k < n-1$). We now prove the the conjugate statement hold for k, i.e., $\{\boldsymbol{d}^{(k+1)}\}_{i=0}^{k+1}$ are $\boldsymbol{Q}$-conjugate. It suffices to show that

$$(\boldsymbol{d}^{(k+1)})^\top \boldsymbol{Q}\boldsymbol{d}^{(i)} = 0, \;\; 0 \le i \le k.$$

Given i $(0 \le i \le k)$, using the same algebraic steps as $k = 0$, and assumption that $\alpha_i \neq 0$, we have

$$(\boldsymbol{d}^{(k+1)})^\top \boldsymbol{Q}\boldsymbol{d}^{(i)} = -(\boldsymbol{g}^{(k+1)})^\top \boldsymbol{H}_{k+1}\boldsymbol{Q}\boldsymbol{d}^{(i)} = \cdots = -(\boldsymbol{g}^{(k+1)})^\top \boldsymbol{d}^{(i)}$$

$\because$ $\boldsymbol{d}^{(0)}, \ldots, \boldsymbol{d}^{(k)}$ are $\boldsymbol{Q}$-conjugate by assumption, we have

$$(\boldsymbol{g}^{(k+1)})^\top \boldsymbol{d}^{(i)} = 0.$$

$\therefore$ $(\boldsymbol{d}^{(k+1)})^\top \boldsymbol{Q}\boldsymbol{d}^{(i)} = 0.$

★ quasi-Newton method solves a quadratic of n variables in at most n steps.

Rank-One Quasi-Newton Method

Require: $\boldsymbol{x}^{(0)} \in \mathbb{R}^n$ and $\boldsymbol{H}_0 \succ 0$. set $k := 0$;

1: **repeat**

2: $\boldsymbol{d}^{(k)} = -\boldsymbol{H}_k \boldsymbol{g}^{(k)} \Longrightarrow$ quasi-Newton direction

3: $\alpha_k = \arg\min_{\alpha \geq 0} f(\boldsymbol{x}^{(k)} + \alpha \boldsymbol{d}^{(k)}) \Longrightarrow$ compute step size

4: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{d}^{(k)} \Longrightarrow$ updating iterative point.

5: $\boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}, \quad \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)},$

6: $\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k \boldsymbol{y}^{(k)})(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k \boldsymbol{y}^{(k)})^\top}{\boldsymbol{y}^{(k)\top}(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k \boldsymbol{y}^{(k)})} \Longrightarrow$ updating $\boldsymbol{H}_k$;

7: **until** $\|\boldsymbol{g}^{(k)}\| \leq \epsilon$.

★ $\boldsymbol{H}_{k+1}$ generated by rank-one algorithm may be $\boldsymbol{H}_{k+1} \not\succ 0$ and thus $\boldsymbol{d}^{(k+1)}$ may not be a descent direction. This happens even in the quadratic case.

★ if $(\boldsymbol{y}^{(k)})^\top(\boldsymbol{s}^{(k)} - \boldsymbol{H}_k \boldsymbol{y}^{(k)})$ is close to zero, then there may be numerical problems in evaluating $\boldsymbol{H}_{k+1}$.

An example of $\boldsymbol{H}_{k+1} \not\succ 0$: see 11.2 in textbook.

Rank-One Algorithm

Example (apply rank-one algorithm to minimize f)

$f(x_1, x_2) = x_1^2 + \frac{1}{2}x_2^2 + 3$, $\boldsymbol{x}^{(0)} = [1, 2]^\top$ and $\boldsymbol{H}_0 = \boldsymbol{I}_2$.

Ans: $\boldsymbol{g}^{(k)} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{x}^{(k)}$.

S1: $\boldsymbol{d}^{(0)} = -\boldsymbol{g}^{(0)} = [-2, -2]^\top$,

$\alpha_0 = \arg\min_{\alpha \geqslant 0} f(\boldsymbol{x}^{(0)} + \alpha \boldsymbol{d}^{(0)}) = -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = \frac{2}{3}$,

$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0 \boldsymbol{d}^{(0)} = \left[-\frac{1}{3}, \frac{2}{3}\right]^\top$.

S2: $\boldsymbol{g}^{(1)} = \boldsymbol{Q}\boldsymbol{x}^{(1)} = \left[-\frac{2}{3}, \frac{2}{3}\right]^\top$,

$\boldsymbol{s}^{(0)} = \boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)} = \left[-\frac{4}{3}, -\frac{4}{3}\right]^\top$, $\boldsymbol{y}^{(0)} = \boldsymbol{g}^{(1)} - \boldsymbol{g}^{(0)} = \left[-\frac{8}{3}, -\frac{4}{3}\right]^\top$,

$\boldsymbol{H}_1 = \boldsymbol{H}_0 + \frac{(\boldsymbol{s}^{(0)} - \boldsymbol{H}_0\boldsymbol{y}^{(0)})(\boldsymbol{s}^{(0)} - \boldsymbol{H}_0\boldsymbol{y}^{(0)})^\top}{\boldsymbol{y}^{(0)\top}(\boldsymbol{s}^{(0)} - \boldsymbol{H}_0\boldsymbol{y}^{(0)})} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix}$,

$\boldsymbol{d}^{(1)} = -\boldsymbol{H}_1\boldsymbol{g}^{(1)} = \left[\frac{1}{3}, -\frac{2}{3}\right]^\top$,

$\alpha_1 = -\frac{\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = 1$, $\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1 \boldsymbol{d}^{(1)} = [0, 0]^\top$,

$\because \boldsymbol{g}^{(2)} = \boldsymbol{0}$, $\therefore \boldsymbol{x}^* = \boldsymbol{x}^{(2)}$.

DFP Algorithm

quasi-Newton equation

$$\boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)}, \quad \text{where} \quad \boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}, \quad \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)}$$

rank-two correction of $\boldsymbol{H}_{k+1}$: DFP formula

Given $\boldsymbol{H}_k$, $\boldsymbol{y}^{(k)}$, $\boldsymbol{s}^{(k)}$, update $\boldsymbol{H}_{k+1}$ by solving the systems

$$\begin{cases} \boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)} \\ \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + a\boldsymbol{u}\boldsymbol{u}^\top + b\boldsymbol{v}\boldsymbol{v}^\top \end{cases} \xrightarrow{\text{details}} \cdots \Longrightarrow$$

DFP Algorithm

quasi-Newton equation

$$\boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)}, \quad \text{where} \quad \boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}, \quad \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)}$$

rank-two correction of $\boldsymbol{H}_{k+1}$: DFP formula

Given $\boldsymbol{H}_k$, $\boldsymbol{y}^{(k)}$, $\boldsymbol{s}^{(k)}$, update $\boldsymbol{H}_{k+1}$ by solving the systems

$$\begin{cases} \boldsymbol{H}_{k+1}\boldsymbol{y}^{(k)} = \boldsymbol{s}^{(k)} \\ \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + a\boldsymbol{u}\boldsymbol{u}^\top + b\boldsymbol{v}\boldsymbol{v}^\top \end{cases} \xrightarrow{\text{details}} \cdots \Longrightarrow \begin{cases} \boldsymbol{u} = \boldsymbol{s}^{(k)}, \\ \boldsymbol{v} = \boldsymbol{H}_k\boldsymbol{y}^{(k)}, \\ a = \frac{1}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}}, \\ b = -\frac{1}{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}. \end{cases}$$

$$\Longrightarrow \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}} - \frac{\left[\boldsymbol{H}_k\boldsymbol{y}^{(k)}\right]\left[\boldsymbol{H}_k\boldsymbol{y}^{(k)}\right]^\top}{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}.$$

DFP Algorithm

Example (apply DFP algorithm to minimize f)

$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \boldsymbol{x} - \boldsymbol{x}^\top \begin{bmatrix} -1 \\ 1 \end{bmatrix}$. The initial point $\boldsymbol{x}^{(0)} = [0, 0]^\top$ and $\boldsymbol{H}_0 = \boldsymbol{I}_2$.

Ans: $\boldsymbol{g}^{(k)} = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \boldsymbol{x}^{(k)} - \begin{bmatrix} -1 \\ 1 \end{bmatrix}$.

S1: $\boldsymbol{g}^{(0)} = [1, -1]^\top$, $\boldsymbol{d}^{(0)} = -\boldsymbol{H}_0\boldsymbol{g}^{(0)} = [-1, 1]^\top$,
$\alpha_0 = \arg\min_{\alpha \geqslant 0} f(\boldsymbol{x}^{(0)} + \alpha \boldsymbol{d}^{(0)}) = 1$, $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0 \boldsymbol{d}^{(0)} = [-1, 1]^\top$.

S2: $\boldsymbol{g}^{(1)} = [-1, -1]^\top$,
$\boldsymbol{y}^{(0)} = \boldsymbol{g}^{(1)} - \boldsymbol{g}^{(0)} = [-2, 0]^\top$, $\boldsymbol{s}^{(0)} = \boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)} = [-1, 1]^\top$,
$\boldsymbol{H}_1 = \boldsymbol{H}_0 + \frac{\boldsymbol{s}^{(0)}\boldsymbol{s}^{(0)\top}}{\boldsymbol{s}^{(0)\top}\boldsymbol{y}^{(0)}} - \frac{(\boldsymbol{H}_0\boldsymbol{y}^{(0)})(\boldsymbol{H}_0\boldsymbol{y}^{(0)})^\top}{\boldsymbol{y}^{(0)\top}\boldsymbol{H}_0\boldsymbol{y}^{(0)}} = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{3}{2} \end{bmatrix}$,
$\boldsymbol{d}^{(1)} = -\boldsymbol{H}_1\boldsymbol{g}^{(1)} = [0, 1]^\top$,
$\alpha_1 = \arg\min_{\alpha \geqslant 0} f(\boldsymbol{x}^{(1)} + \alpha \boldsymbol{d}^{(1)}) = \frac{1}{2}$, $\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1 \boldsymbol{d}^{(1)} = \left[-1, \frac{3}{2}\right]^\top$.

$\because\ \boldsymbol{g}^{(2)} = [0, 0]^\top$, $\therefore\ \boldsymbol{x}^{(2)}$ is the minimizer.

★ $\boldsymbol{d}^{(0)}$ and $\boldsymbol{d}^{(1)}$ are $\boldsymbol{Q}$-conjugate (check it).

DFP Algorithm

Example (apply DFP algorithm to minimize f)

$f(\boldsymbol{x}) = 2x_1^2 + x_2^2 - 4x_1 + 2$ with $\boldsymbol{H}_0 = \boldsymbol{I}_2$ and $\boldsymbol{x}^{(0)} = [2, 1]^\top$.

Ans: $\nabla f(\boldsymbol{x}) = \begin{bmatrix} 4x_1 - 4 \\ 2x_2 \end{bmatrix}$, $\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} =: \boldsymbol{Q}$,

S1: $\boldsymbol{g}^{(0)} = [4, 2]^\top$, $\boldsymbol{d}_0 = -\boldsymbol{H}_0\boldsymbol{g}^{(0)} = [-4, -2]^\top$,

$\alpha_0 = -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = \frac{5}{18}$, $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0\boldsymbol{d}^{(0)} = \left[\frac{8}{9}, \frac{4}{9}\right]^\top$.

S2: $\boldsymbol{s}^{(0)} = \boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)} = [-\frac{10}{9}0, -\frac{5}{9}]^\top$, $\boldsymbol{g}^{(1)} = [-\frac{4}{9}, \frac{8}{9}]^\top$,

$\boldsymbol{y}^{(0)} = \boldsymbol{g}^{(1)} - \boldsymbol{g}^{(0)} = [-\frac{40}{9}, -\frac{10}{9}]^\top$,

$\boldsymbol{H}_1 = \boldsymbol{H}_0 + \frac{\boldsymbol{s}^{(0)}\boldsymbol{s}^{(0)\top}}{\boldsymbol{s}^{(0)\top}\boldsymbol{y}^{(0)}} - \frac{\boldsymbol{H}_0\boldsymbol{y}^{(0)}\boldsymbol{y}^{(0)\top}\boldsymbol{H}_0}{\boldsymbol{y}^{(0)\top}\boldsymbol{H}_0\boldsymbol{y}^{(0)}} = \frac{1}{306}\begin{bmatrix} 86 & -38 \\ -38 & 305 \end{bmatrix}$,

$\boldsymbol{d}^{(1)} = -\boldsymbol{H}_1\boldsymbol{g}^{(1)} = [\frac{4}{17}, -\frac{16}{17}]^\top$,

$\alpha_1 = -\frac{\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = \frac{17}{36}$,

$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1\boldsymbol{d}^{(1)} = [1, 0]^\top$,

$\because\ \boldsymbol{g}^{(2)} = \boldsymbol{0}$, $\therefore\ \boldsymbol{x}^{(2)}$ is the minimizer.

★ $\boldsymbol{d}^{(0)}$ and $\boldsymbol{d}^{(1)}$ are $\boldsymbol{Q}$-conjugate (check it).

DFP Algorithm

Theorem (inheritance of DFP)

Suppose that $\boldsymbol{g}^{(k)} \neq \mathbf{0}$. *In the DFP algorithm, if* $\boldsymbol{H}_k \succ 0$, *then so is* $\boldsymbol{H}_{k+1}$.

- ★ DFP was developed by Davidon in 1959 and was modified by Fletcher and Powell in 1963.
- ★ DFP is superior to rank-one algorithm as it can guarantee $\boldsymbol{H}_k \succ 0$.
- ★ DFP may get stuck for large nonlinear optimization because $\boldsymbol{H}_k$ is close to singular.

BFGS Algorithm

History: In 1970, an alternative update formula was suggested independently by Broyden, Fletcher, Goldfarb, and Shanno, which is called BFGS algorithm.

motivation

Newton's method: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{F}(\boldsymbol{x}^{(k)})^{-1}\boldsymbol{g}^{(k)}$.
Instead of seeking $\boldsymbol{H}_k \approx [\boldsymbol{F}(\boldsymbol{x}^{(k)})]^{-1}$, we alternatively find an approximation of Hessian, i.e., $\boldsymbol{B}_k \approx \boldsymbol{F}(\boldsymbol{x}^{(k)})$, thus
quasi-Newton methods: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{B}_k^{-1}\boldsymbol{g}^{(k)}$.

choice of $\boldsymbol{B}_k$ (analogously to $\boldsymbol{H}_k$)

$$\because \nabla f(\boldsymbol{x}^{(k)}) \approx \nabla f(\boldsymbol{x}^{(k+1)}) + \boldsymbol{F}(\boldsymbol{x}^{(k+1)})(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}),$$

$$\therefore \underbrace{\nabla f(\boldsymbol{x}^{(k)}) - \nabla f(\boldsymbol{x}^{(k+1)})}_{\boldsymbol{y}^{(k)}} \approx \boldsymbol{F}(\boldsymbol{x}^{(k+1)}) \underbrace{(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)})}_{\boldsymbol{s}^{(k)}}.$$

$$\therefore \boldsymbol{y}^{(k)} \approx \boldsymbol{F}(\boldsymbol{x}^{(k+1)})\boldsymbol{s}^{(k)}.$$

$$\therefore \text{Quasi-Newton equation: } \boldsymbol{y}^{(k)} = \boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)}.$$

BFGS Algorithm

quasi-Newton equation

$$\boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}, \quad \text{where} \quad \boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}, \quad \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)}$$

rank-two correction of $\boldsymbol{B}_{k+1}$: BFGS formula

Given $\boldsymbol{B}_k$, $\boldsymbol{y}^{(k)}$, $\boldsymbol{s}^{(k)}$, update $\boldsymbol{B}_{k+1}$ by solving the systems

$$\begin{cases} \boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}, \\ \boldsymbol{B}_{k+1} = \boldsymbol{B}_k + a\boldsymbol{u}\boldsymbol{u}^\top + b\boldsymbol{v}\boldsymbol{v}^\top, \end{cases} \Longrightarrow \cdots \xrightarrow{\text{detail}} \cdots$$

BFGS Algorithm

quasi-Newton equation

$$\boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}, \quad \text{where} \quad \boldsymbol{s}^{(k)} = \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}, \quad \boldsymbol{y}^{(k)} = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)}$$

rank-two correction of $\boldsymbol{B}_{k+1}$: BFGS formula

Given $\boldsymbol{B}_k$, $\boldsymbol{y}^{(k)}$, $\boldsymbol{s}^{(k)}$, update $\boldsymbol{B}_{k+1}$ by solving the systems

$$\begin{cases} \boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}, \\ \boldsymbol{B}_{k+1} = \boldsymbol{B}_k + a\boldsymbol{u}\boldsymbol{u}^\top + b\boldsymbol{v}\boldsymbol{v}^\top, \end{cases} \Longrightarrow \cdots \xrightarrow{\text{detail}} \cdots \begin{cases} \boldsymbol{u} = \boldsymbol{y}^{(k)}, \\ \boldsymbol{v} = \boldsymbol{B}_k\boldsymbol{s}^{(k)}, \\ a = \frac{1}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}}, \\ b = -\frac{1}{\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k\boldsymbol{s}^{(k)}}. \end{cases}$$

$$\Longrightarrow \boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \frac{\boldsymbol{y}^{(k)}\boldsymbol{y}^{(k)\top}}{\boldsymbol{y}^{(k)\top}\boldsymbol{s}^{(k)}} - \frac{\boldsymbol{B}_k\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k}{\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k\boldsymbol{s}^{(k)}}.$$

★ Recall DFP formula $\boldsymbol{H}_{k+1}^{\mathrm{DFP}} = \boldsymbol{H}_k + \frac{\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}} - \frac{\boldsymbol{H}_k\boldsymbol{y}^{(k)}\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k}{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}$, we have $\boldsymbol{H}_{k+1}$ and $\boldsymbol{B}_{k+1}$ are "dual" of each other, i.e., $\boldsymbol{B}_{k+1}$ can be derived by commuting $\boldsymbol{H}_k \leftrightarrow \boldsymbol{B}_k$, $\boldsymbol{y}^{(k)} \leftrightarrow \boldsymbol{s}^{(k)}$.

BFGS Algorithm

quasi-Newton methods with BFGS: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{B}_k^{-1} \boldsymbol{g}^{(k)}$.

Question: how to compute the inverse? any skill to derive $\boldsymbol{B}_{k+1}^{-1}$ from $\boldsymbol{B}_k^{-1}$?

$$\boldsymbol{B}_{k+1}^{-1} = \left(\boldsymbol{B}_k + \frac{\boldsymbol{y}^{(k)} \boldsymbol{y}^{(k)\top}}{\boldsymbol{y}^{(k)T} \boldsymbol{s}^{(k)}} - \frac{\boldsymbol{B}_k \boldsymbol{s}^{(k)} \boldsymbol{s}^{(k)\top} \boldsymbol{B}_k}{\boldsymbol{s}^{(k)\top} \boldsymbol{B}_k \boldsymbol{s}^{(k)}} \right)^{-1} = \boxed{?}$$

Lemma (Sherman-Morrison formula)

Let $\boldsymbol{U} \in \mathbb{R}^{m \times n}$, $\boldsymbol{V} \in \mathbb{R}^{n \times m}$. If $\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V}$ is invertible, then $\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U}$ is also invertible and $(\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U})^{-1} = \boldsymbol{I}_n - \boldsymbol{V}(\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V})^{-1}\boldsymbol{U}$.

proof. $\boldsymbol{V} + \boldsymbol{V}\boldsymbol{U}\boldsymbol{V} = \boldsymbol{V}(\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V}) = (\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U})\boldsymbol{V}$.

$\therefore\ \boldsymbol{V} = (\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U})\boldsymbol{V}(\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V})^{-1}$.

$\because\ \boldsymbol{I}_n = \boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U} - \boldsymbol{V}\boldsymbol{U} = \boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U} - (\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U})\boldsymbol{V}(\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V})^{-1}\boldsymbol{U}$

$= (\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{U})[\boldsymbol{I}_n - \boldsymbol{V}(\boldsymbol{I}_m + \boldsymbol{U}\boldsymbol{V})^{-1}\boldsymbol{U}]$

Corollary ($\boldsymbol{A} \in \mathbb{R}^{m \times m}$ be invertible, $\boldsymbol{U} \in \mathbb{R}^{m \times n}$, $\boldsymbol{V} \in \mathbb{R}^{n \times m}$, $\boldsymbol{u}$ and $\boldsymbol{v} \in \mathbb{R}^m$)

$(\boldsymbol{A} + \boldsymbol{U}\boldsymbol{V})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}(\boldsymbol{I}_n + \boldsymbol{V}\boldsymbol{A}^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}\boldsymbol{A}^{-1}$.

$(\boldsymbol{A} + \boldsymbol{u}\boldsymbol{v}^\top)^{-1} = \boldsymbol{A}^{-1} - \frac{(\boldsymbol{A}^{-1}\boldsymbol{u})(\boldsymbol{v}^\top \boldsymbol{A}^{-1})}{1 + \boldsymbol{v}^\top \boldsymbol{A}^{-1}\boldsymbol{u}}$. (*proof. handwriting the details*)

BFGS Algorithm

inverse of $\boldsymbol{B}_{k+1}$

Assume $\boldsymbol{B}_k^{-1}$ is given (dented by $\boldsymbol{H}_k$). By applying S-M formula twice, we have

$$\begin{aligned}\boldsymbol{B}_{k+1}^{-1} &= \boldsymbol{H}_{k+1} \\ &= \boldsymbol{H}_k + \left(1 + \frac{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}{\boldsymbol{y}^{(k)\top}\boldsymbol{s}^{(k)}}\right)\frac{\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}} - \frac{\boldsymbol{H}_k\boldsymbol{y}^{(k)}\boldsymbol{s}^{(k)\top} + (\boldsymbol{H}_k\boldsymbol{y}^{(k)}\boldsymbol{s}^{(k)\top})^\top}{\boldsymbol{y}^{(k)\top}\boldsymbol{s}^{(k)}},\end{aligned}$$

discussion on DFP and BFGS

- Both are rank-two formulae.
- DFP: $\boldsymbol{H}_{k+1} \approx [\boldsymbol{F}(\boldsymbol{x}^{(k+1)})]^{-1}$, $\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}} - \frac{\boldsymbol{H}_k\boldsymbol{y}^{(k)}\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k}{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}$.
- BFGS: $\boldsymbol{B}_{k+1} \approx \boldsymbol{F}(\boldsymbol{x}^{(k+1)})$, $\quad \boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \frac{\boldsymbol{y}^{(k)}\boldsymbol{y}^{(k)\top}}{\boldsymbol{y}^{(k)\top}\boldsymbol{s}^{(k)}} - \frac{\boldsymbol{B}_k\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k}{\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k\boldsymbol{s}^{(k)}}$.

  Question: $\boldsymbol{H}_{k+1} = \boldsymbol{B}_{k+1}^{-1}$?
- Combinations of DFP and BFGS:
  $\bar{\boldsymbol{H}}_{k+1} = \alpha\boldsymbol{H}_{k+1} + (1-\alpha)\boldsymbol{B}_{k+1}^{-1}$ or $\bar{\boldsymbol{B}}_{k+1} = \alpha\boldsymbol{B}_{k+1} + (1-\alpha)\boldsymbol{H}_{k+1}^{-1}$

★ BFGS fulfills all premises of quasi-Newton method, e.g., positive definiteness, symmetry when minimizing quadratic problem.

BFGS Algorithm

Example (apply BFGS algorithm to minimize f)

$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \begin{bmatrix} 5 & -3 \\ -3 & 2 \end{bmatrix} \boldsymbol{x} - \boldsymbol{x}^\top \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \log(\pi)$ with $\boldsymbol{H}_0 = \boldsymbol{I}_2$ and $\boldsymbol{x}^{(0)} = [0, 0]^\top$.

S1: $\boldsymbol{d}^{(0)} = -\boldsymbol{g}^{(0)} = -(\boldsymbol{Q}\boldsymbol{x}^{(0)} - \boldsymbol{b})$,

$\alpha_0 = -\frac{\boldsymbol{g}^{(0)\top}\boldsymbol{d}^{(0)}}{\boldsymbol{d}^{(0)\top}\boldsymbol{Q}\boldsymbol{d}^{(0)}} = \frac{1}{2}$, $\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0\boldsymbol{d}^{(0)} = \left[0, \frac{1}{2}\right]^\top$.

S2: $\boldsymbol{s}^{(0)} = \boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)} = [0, \frac{1}{2}]^\top$, $\boldsymbol{g}^{(1)} = \boldsymbol{Q}\boldsymbol{x}^{(1)} - \boldsymbol{b} = [\frac{-3}{2}, 0]^\top$,

$\boldsymbol{y}^{(0)} = \boldsymbol{g}^{(1)} - \boldsymbol{g}^{(0)} = [\frac{-3}{2}, 1]^\top$,

$\boldsymbol{H}_1 = \boldsymbol{H}_0 + \left(1 + \frac{\boldsymbol{y}^{(0)\top}\boldsymbol{H}_0\boldsymbol{y}^{(0)}}{\boldsymbol{y}^{(0)\top}\boldsymbol{s}^{(0)}}\right)\frac{\boldsymbol{s}^{(0)}\boldsymbol{s}^{(0)\top}}{\boldsymbol{s}^{(0)\top}\boldsymbol{y}^{(0)}} - \frac{\boldsymbol{s}^{(0)}\boldsymbol{y}^{(0)\top}\boldsymbol{H}_0 + \boldsymbol{H}_0\boldsymbol{y}^{(0)}\boldsymbol{s}^{(0)\top}}{\boldsymbol{y}^{(0)\top}\boldsymbol{s}^{(0)}} = \begin{bmatrix} 1 & \frac{3}{2} \\ \frac{3}{2} & \frac{11}{4} \end{bmatrix}$,

$\boldsymbol{d}^{(1)} = -\boldsymbol{H}_1\boldsymbol{g}^{(1)} = [\frac{3}{2}, \frac{9}{4}]^\top$,

$\alpha_1 = -\frac{\boldsymbol{g}^{(1)\top}\boldsymbol{d}^{(1)}}{\boldsymbol{d}^{(1)\top}\boldsymbol{Q}\boldsymbol{d}^{(1)}} = 2$,

$\boldsymbol{x}^{(2)} = \boldsymbol{x}^{(1)} + \alpha_1\boldsymbol{d}^{(1)} = [3, 5]^\top$,

$\because\ \boldsymbol{g}^{(2)} = \boldsymbol{0}$, $\therefore\ \boldsymbol{x}^{(2)}$ is the minimizer.

Comparisons of Quasi-Newton and Newton Methods

Quasi-Newton method	Newton method
only need function value and gradient	need function value, gradient and Hessian
$\boldsymbol{H}_k \succ 0 / \boldsymbol{B}_k \succ 0$ for several updates	$\boldsymbol{F}(\boldsymbol{x}^{(k)})$ may be not positive definite
$O(n^2)$ multiplications in each iteration	$O(n^3)$ multiplications in each iteration

Example (apply BFGS algorithm to minimize nonquadratic f)

$f(\boldsymbol{x}) = f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ with $\boldsymbol{x}^{(0)} = [-1.2, 1]^\top$ (note: exact minimizer $x^* = [1, 1]^\top$)

steepest descent	BFGS	Newton
1.827e-4	1.70e-3	3.48e-2
1.826e-4	1.17e-3	1.44e-2
1.824e-4	1.34e-4	1.82e-4
1.823e-4	1.01e-6	1.17e-8

After coding on Matlab, Table lists some values of $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_2$. To reach $\|\boldsymbol{g}^{(k)}\|_2 \leq 10^{-5}$, steepest descent needs 5264 iterations, BFGS needs 34 iterations, Newton needs 21 iterations.

★ BFGS has the superlinear rate of convergence on practical problems.

BFGS Algorithm

- ★ For nonquadratic problems, quasi-Newton algorithms may not converge in n steps.
- ★ Restart technique: As in the case of the conjugate gradient methods, we may reinitialize the direction vector to the negative gradient after every few iterations (e.g., n or $n+1$), and continue until the algorithm satisfies the stopping criterion.

Barzilai-Borwein (BB) Algorithm

quasi-Newton method with BFGS

$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{B}_k^{-1}\boldsymbol{g}^{(k)}$, where $\boldsymbol{B}_k \approx \boldsymbol{F}(\boldsymbol{x}^{(k)})$.

Question: instead of rank-one/two $\approx \boldsymbol{F}(\boldsymbol{x}^{(k)})$, how about $\alpha_k \boldsymbol{I} \approx \boldsymbol{F}(\boldsymbol{x}^{(k)})$?

BB algorithm

Given $\boldsymbol{y}^{(k)}, \boldsymbol{s}^{(k)}$, solve systems

$$\text{BFGS} \begin{cases} \boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)} \\ \boldsymbol{B}_{k+1} = \boldsymbol{B}_k + a\boldsymbol{u}\boldsymbol{u}^\top + b\boldsymbol{v}\boldsymbol{v}^\top \end{cases} \xrightarrow{\text{new idea}} \begin{cases} \boldsymbol{B}_{k+1}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)} \\ \boldsymbol{B}_{k+1} = t_{k+1}\boldsymbol{I} \end{cases} \quad (*)$$

☹ (*) has no exact solution.

☺ (*) has a least squares solution as $t_{k+1} = \frac{(\boldsymbol{y}^{(k)})^\top \boldsymbol{s}^{(k)}}{(\boldsymbol{s}^{(k)})^\top \boldsymbol{s}^{(k)}}$ (why?)

iterative scheme of BB

$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k^{\mathrm{BB}}\boldsymbol{g}^{(k)}$, where $\alpha_k^{\mathrm{BB}} = \frac{1}{t_k} = \frac{\left(\boldsymbol{s}^{(k-1)}\right)^\top \boldsymbol{s}^{(k-1)}}{\left(\boldsymbol{y}^{(k-1)}\right)^\top \boldsymbol{s}^{(k-1)}}$.

Barzilai-Borwein (BB) Algorithm

Question: what is the relationship of BB and steepest descent (SD) methods when minimizing quadratic function $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x}$?

relationship of BB and SD

exact stepsize of SD method: $\alpha_k^{\mathrm{SD}} = \frac{(\boldsymbol{g}^{(k)})^\top \boldsymbol{g}^{(k)}}{(\boldsymbol{g}^{(k)})^\top \boldsymbol{Q}\boldsymbol{g}^{(k)}}$.

$\because \ \boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{g}^{(k)}$

$\therefore \ \boldsymbol{g}^{(k)} = \frac{\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}}{\alpha_k} = \frac{\boldsymbol{s}^{(k)}}{\alpha_k}$,

$\therefore \ \alpha_k^{\mathrm{SD}} = \frac{(\boldsymbol{g}^{(k)})^\top \boldsymbol{g}^{(k)}}{(\boldsymbol{g}^{(k)})^\top \boldsymbol{Q}\boldsymbol{g}^{(k)}} = \frac{(\boldsymbol{s}^{(k)})^\top \boldsymbol{s}^{(k)}}{(\boldsymbol{s}^{(k)})^\top \boldsymbol{Q}\boldsymbol{s}^{(k)}}$,

$\because \ \boldsymbol{Q}\boldsymbol{s}^{(k)} = \boldsymbol{Q}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}) = \boldsymbol{g}^{(k)} - \boldsymbol{g}^{(k+1)} = \boldsymbol{y}^{(k)}$

$\therefore \ \alpha_k^{\mathrm{SD}} = \frac{(\boldsymbol{s}^{(k)})^\top \boldsymbol{s}^{(k)}}{(\boldsymbol{s}^{(k)})^\top \boldsymbol{Q}\boldsymbol{s}^{(k)}} = \frac{(\boldsymbol{s}^{(k)})^\top \boldsymbol{s}^{(k)}}{(\boldsymbol{s}^{(k)})^\top \boldsymbol{y}^{(k)}} = \alpha_{k+1}^{\mathrm{BB}}$

★ BB stepsize at $k+1$ is the SD stepsize at k.

Homework

1. Derive the DFP formula:

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}}{\boldsymbol{s}^{(k)\top}\boldsymbol{y}^{(k)}} - \frac{\left[\boldsymbol{H}_k\boldsymbol{y}^{(k)}\right]\left[\boldsymbol{H}_k\boldsymbol{y}^{(k)}\right]^\top}{\boldsymbol{y}^{(k)\top}\boldsymbol{H}_k\boldsymbol{y}^{(k)}}$$

2. Derive the BFGS formula:

$$\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \frac{\boldsymbol{y}^{(k)}\boldsymbol{y}^{(k)\top}}{\boldsymbol{y}^{(k)\top}\boldsymbol{s}^{(k)}} - \frac{\boldsymbol{B}_k\boldsymbol{s}^{(k)}\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k}{\boldsymbol{s}^{(k)\top}\boldsymbol{B}_k\boldsymbol{s}^{(k)}}$$