Second order methods

DSA5103 Lecture 11

Yangjing Zhang

30-Mar-2023

NUS

Today's content

1. Rate of convergence
2. Pure Newton's method
3. Practical Newton's method
4. Proximal Newton method

Rate of convergence

Rate of convergence: Q-linear

- One of the key measures of performance of an algorithm is its rate of convergence
- Let $\{x^{(k)}\}$ be a sequence in $\mathbb{R}^n$ that converges to x^*
- We say the convergence is Q-linear if there exists $r \in (0, 1)$ such that

$$\frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} \leq r \text{ for all } k \text{ sufficiently large}$$

- The distance to the solution x^* decreases at each iteration by at least a constant factor
- Q: quotient (of successive errors)

Example. The sequence $1 + 0.8^k$ converges Q-linearly to 1.

Rate of convergence: Q-superlinear

- We say the convergence is Q-superlinear if

$$\lim_{k\to\infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = 0$$

Example. The sequence $1 + k^{-k}$ converges Q-superlinearly to 1.

Rate of convergence: Q-superlinear

- We say the convergence is Q-superlinear if

$$\lim_{k \to \infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = 0$$

Example. The sequence $1 + k^{-k}$ converges Q-superlinearly to 1.

- Any sequence that converges Q-superlinearly also converges Q-linearly

Rate of convergence: Q-quadratic

- We say the convergence is Q-quadratic if there exists $M > 0$ such that

$$\frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|^2} \leq M \text{ for all } k \text{ sufficiently large}$$

  - ▷ M is not necessarily less than 1
  - ▷ a quadratically convergent sequence will always eventually converge faster than a linearly convergent sequence

Example. The sequence $1 + (0.8)^{2^k}$ converges Q-quadratically to 1.

Rate of convergence: Q-quadratic

- We say the convergence is Q-quadratic if there exists $M > 0$ such that
$$\frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|^2} \leq M \text{ for all } k \text{ sufficiently large}$$
  - M is not necessarily less than 1
  - a quadratically convergent sequence will always eventually converge faster than a linearly convergent sequence

Example. The sequence $1 + (0.8)^{2^k}$ converges Q-quadratically to 1.

- Any sequence that converges Q-quadratically also converges Q-superlinearly

Rate of convergence

- The convergence speed depends on constants, e.g., r, M
- Eventually, Q-quadratic $\overset{\text{faster than}}{>}$ Q-superlinear $\overset{\text{faster than}}{>}$ Q-linear
- We normally omit the letter Q and simply talk about superlinear convergence, quadratic convergence, etc.

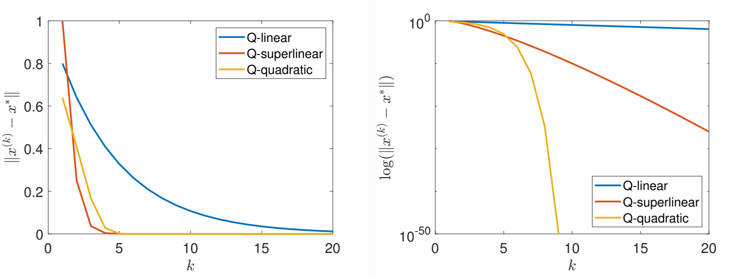


Figure 1: Q-linear $1 + 0.8^k$, Q-superlinear $1 + k^{-k}$, Q-quadratic $1 + (0.8)^{2^k}$.

Rate of convergence

Example. Show that the sequence $\frac{1}{k}$ is not Q-linearly convergent, though it does converge to zero.

Example. Show that the sequence $\frac{1}{k!}$ converges Q-superlinearly. Does it converge Q-quadratically? $(k! = 1 \cdot 2 \cdot 3 \cdots k)$

Pure Newton's method

First/second order methods

- **First order methods**: any method that uses first order derivatives
  - ▷ Gradient descent methods, PG, APG, BCD, ADMM are usually regarded as first order methods
- **Second order methods**: any method that uses any second order derivative

A general algorithmic framework for $\min_{x} f(x)$

$$x^{(k+1)} = x^{(k)} - \alpha_k P_k \nabla f(x^{(k)})$$

- Lot of freedom in choosing the preconditioning matrix $P_k \succeq 0$
- If $P_k = I$, it is gradient (steepest) descent method
- If $P_k =$ inverse of Hessian, it is Newton's method
- A wise choice of P_k will improve the convergence speed a lot

Recall gradient descent

$$\min_x \; f(x)$$

- $x^{(k)}$, $x^{(k+1)}$, $d = x^{(k+1)} - x^{(k)}$
- Consider the approximation

$$f(x^{(k)} + d) \approx f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T d + \frac{1}{2\alpha_k}\|d\|^2$$

- $\arg\min_d \left\{ f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T d + \frac{1}{2\alpha_k}\|d\|^2 \right\} = -\alpha_k \nabla f(x^{(k)})$

- $x^{(k+1)} = x^{(k)} + d = x^{(k)} - \alpha_k \nabla f(x^{(k)})$ gradient descent

Newton's method derivation

- Consider the second order approximation

$$f(x^{(k)} + d) \approx f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T d + \frac{1}{2} d^T H_f(x^{(k)}) d$$

where $H_f\left(x^{(k)}\right) = \nabla^2 f(x^{(k)}) = \left[\frac{\partial f}{\partial x_i \partial x_j}(x^{(k)})\right]_{ij}$ is the Hessian matrix of f at $x^{(k)}$

- $\min_d \left\{ f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T d + \frac{1}{2} d^T H_f(x^{(k)}) d \right\}$

$\iff \nabla f(x^{(k)}) + H_f(x^{(k)}) d = 0$

If $H_f(x^{(k)}) \succ 0$, then $d = -\underbrace{\left(H_f(x^{(k)})\right)}_{\text{precond. matrix}}^{-1} \nabla f(x^{(k)})$, called a Newton direction

- $x^{(k+1)} = x^{(k)} - \left(H_f(x^{(k)})\right)^{-1} \nabla f(x^{(k)})$ Newton iteration

Newton's method intuition

Intuition: second order approximation and then minimize

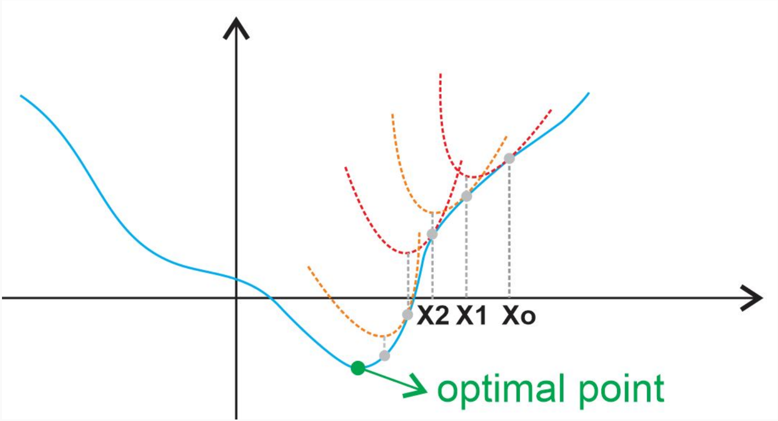


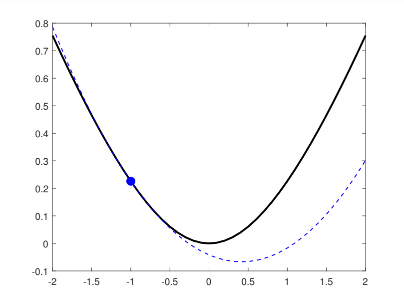
Figure 2: Image from internet

Newton's method intuition

Intuition: second order approximation and then minimize

For example, convex $f : \mathbb{R} \to \mathbb{R}$

$$f(x) = \sqrt{1+x^2} - 1 - \log(\sqrt{1+x^2}+1)/2)$$
$$\approx f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \frac{1}{2} f''(x^{(k)})(x - x^{(k)})^2$$
$$\approx 0.23 - 0.41(x+1) + \frac{0.29}{2}(x+1)^2$$



At $x^{(0)} = -1$

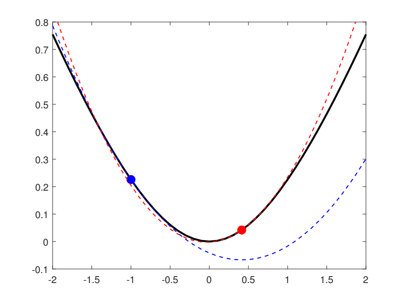
- $f(-1) = 0.23$
- $f'(-1) = -0.41$
- $f''(-1) = 0.29$
- Set $-0.41 + 0.29(x+1) = 0$, $x^{(1)} = 0.41$

Newton's method intuition

Intuition: second order approximation and then minimize

$$f(x) \approx f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \frac{1}{2} f''(x^{(k)})(x - x^{(k)})^2$$

$$\approx 0.04 + 0.20(x - 0.41) + \frac{0.44}{2}(x - 0.41)^2$$



At $x^{(1)} = 0.41$

- $f(0.41) = 0.04$
- $f'(0.41) = 0.20$
- $f''(0.41) = 0.44$
- Set $0.20 + 0.44(x - 0.41) = 0,$ $x^{(2)} = -0.04$

When $x^{(k)}$ is close to the optimal point, the second order approximation seems to be very well

Newton direction

Newton direction $p^{(k)}$ is computed from

$$\nabla f(x^{(k)}) + H_f(x^{(k)})p = 0$$

- If the Hessian matrix $H_f\left(x^{(k)}\right) = \nabla^2 f(x^{(k)})$ is positive definite, then $p^{(k)}$ is a descent direction[1] of f at $x^{(k)}$

[1] lecture 4, page 4

Newton direction

Newton direction $p^{(k)}$ is computed from

$$\nabla f(x^{(k)}) + H_f(x^{(k)})p = 0$$

- If the Hessian matrix $H_f\big(x^{(k)}\big) = \nabla^2 f(x^{(k)})$ is positive definite, then $p^{(k)}$ is a descent direction[1] of f at $x^{(k)}$

$$\Big(\nabla f(x^{(k)})\Big)^T p^{(k)} = -\Big(p^{(k)}\Big)^T H_f\big(x^{(k)}\big)p^{(k)} < 0$$

- f convex $\Rightarrow H_f(x) \succeq 0$; f strictly convex $\Rightarrow H_f(x) \succ 0$
- In general, the Hessian may not always be positive definite, and $p^{(k)}$ may not always be a descent direction (need additional strategies)

[1] lecture 4, page 4

Newton direction

Newton direction $p^{(k)}$ is computed from

$$\nabla f(x^{(k)}) + H_f(x^{(k)})p = 0$$

- When $H_f(x^*) \succ 0$, for all x close enough to x^*, $H_f(x) \succ 0$, under the assumption that $H_f(\cdot)$ is locally Lipschitz continuous[2] at x^*

$$\|H_f(x^*) - H_f(x)\| \leq L\|x^* - x\|$$

[2] g is locally Lipschitz continuous at x_0 if there exit $L > 0$ and some neighbourhood $\mathcal{N}$ of x_0 such that $\|g(x_1) - g(x_2)\| \leq L\|x_1 - x_2\|$ for any $x_1, x_2 \in \mathcal{N}$

Newton direction

Newton direction $p^{(k)}$ is computed from

$$\nabla f(x^{(k)}) + H_f(x^{(k)})p = 0$$

- When $H_f(x^*) \succ 0$, for all x close enough to x^*, $H_f(x) \succ 0$, under the assumption that $H_f(\cdot)$ is locally Lipschitz continuous[2] at x^*

$$\|H_f(x^*) - H_f(x)\| \leq L\|x^* - x\|$$

- For the time being, the step length $= 1$, and we first discuss the local rate-of-convergence of pure Newton's method

[2] g is locally Lipschitz continuous at x_0 if there exit $L > 0$ and some neighbourhood $\mathcal{N}$ of x_0 such that $\|g(x_1) - g(x_2)\| \leq L\|x_1 - x_2\|$ for any $x_1, x_2 \in \mathcal{N}$

Newton's method

Consider $\min_{x\in\mathbb{R}^n} f(x)$. Assume that

- $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable
- At the solution x^*, the sufficient conditions[3] are satisfied

$$\nabla f(x^*) = 0, \quad H_f(x^*) = \nabla^2 f(x^*) \succ 0$$

- $H_f(\cdot)$ is locally Lipschitz continuous at x^*

[3] lecture 1, page 27

Newton's method

Consider $\min\limits_{x \in \mathbb{R}^n} f(x)$. Assume that

- $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable
- At the solution x^*, the sufficient conditions[3] are satisfied

$$\nabla f(x^*) = 0, \quad H_f(x^*) = \nabla^2 f(x^*) \succ 0$$

- $H_f(\cdot)$ is locally Lipschitz continuous at x^*

$$\begin{cases} \nabla f(x^{(k)}) + H_f(x^{(k)}) p^{(k)} = 0 \\ x^{(k+1)} = x^{(k)} + p^{(k)} \end{cases}$$

If $H_f(x^{(k)}) = I$, $p = -\nabla f(x^{(k)})$ reduces to steepest descent direction.

[3] lecture 1, page 27

Local convergence

Theorem. Suppose that

1. f is twice differentiable,
2. the Hessian $H_f(\cdot)$ is locally Lipschitz continuous at a solution x^*,
3. and the sufficient conditions are satisfied at x^*.

Consider the Newton iteration

$$x^{(k+1)} = x^{(k)} + p^{(k)}$$

where $p^{(k)}$ is a solution to the linear system $H_f\big(x^{(k)}\big)p = -\nabla f(x^{(k)})$. Then

1. if the starting point $x^{(0)}$ is sufficiently close to x^*, the sequence of iterates $\{x^{(k)}\}$ converges Q-quadratically to x^*;
2. the sequence of gradient norms $\{\|\nabla f(x^{(k)})\|\}$ converges Q-quadratically to zero.

Algorithm framework

Algorithm (Pure Newton's method)

Choose $x^{(0)} \in \mathbb{R}^n$, $\epsilon > 0$. Set $k \leftarrow 0$

while $\|\nabla f(x^{(k)})\| > \epsilon$ **do**

Compute Newton direction $p^{(k)}$ by solving $H_f\big(x^{(k)}\big)p = -\nabla f(x^{(k)})$

$x^{(k+1)} \leftarrow x^{(k)} + p^{(k)}$

$k \leftarrow k+1$

end(while)

return $x^{(k)}$

- The Newton method with unit steps converges Q-quadratically once it approaches x^*
- It only looks for stationary point $\nabla f(x) = 0$ (global min if f convex)
- It may fail to converge to a solution from remote starting points, need globalization strategies (line search, Hessian modification, trust-region)

Example

Example. $\min_{x=(x_1;x_2)\in\mathbb{R}^2} f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 2x_2$

At $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, compute the steepest descent direction and the Newton direction.

Example

Example. $\min_{x=(x_1;x_2)\in\mathbb{R}^2} \; f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 2x_2$

At $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, compute the steepest descent direction and the Newton direction.

Solution. Compute the gradient and Hessian

$$\nabla f(x) = \begin{bmatrix} 2x_1 - 2x_2 \\ 4x_2 - 2x_1 - 2 \end{bmatrix}, \quad H_f(x) = \begin{bmatrix} 2 & -2 \\ -2 & 4 \end{bmatrix} \succ 0$$

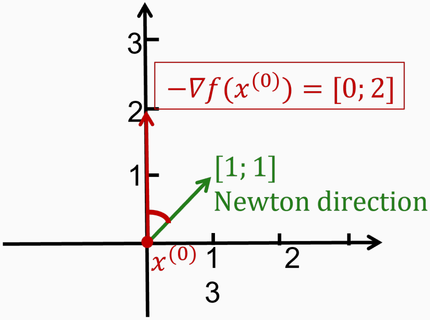
Then the steepest descent direction is $-\nabla f(x^{(0)}) = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$

and the Newton direction is

$$-\Big(H_f(x^{(0)})\Big)^{-1} \nabla f(x^{(0)}) = \begin{bmatrix} 2 & -2 \\ -2 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Example

Example. $\min_{x=(x_1;x_2)\in\mathbb{R}^2} f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 2x_2$

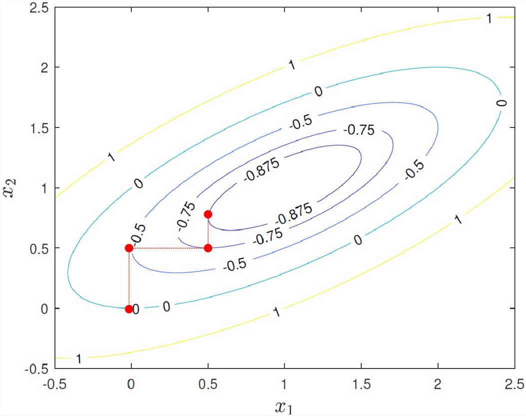


At $x^{(0)} = [0;0]$

- Steepest descent direction: $[0;2]$
- Newton direction: $[1;1]$
- Newton iterate $x^{(1)} = [1;1]$
- $\nabla f(x^{(1)}) = 0$ minimizer

Contour plot

Contour plot of $f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 2x_2$



Steepest descent method with exact line search (zig-zag path)

$x^{(0)} = [0; 0]$

$x^{(1)} = [0; \frac{1}{2}]$

$x^{(2)} = [\frac{1}{2}; \frac{1}{2}]$

$x^{(3)} = [\frac{1}{2}; \frac{3}{4}]$

$\ldots$

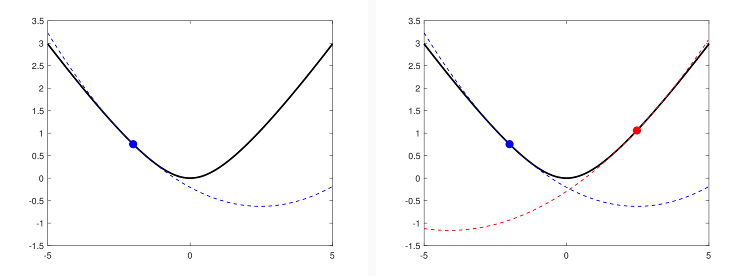
Newton's method: $x^{(0)} = [0; 0]$, $x^{(1)} = [1; 1]$ global min

Example: divergent

Example. $f(x) = \sqrt{1+x^2} - 1 - \log(\sqrt{1+x^2}+1)/2)$

When starting from a remote point, the iterates overshoot.

$x^{(0)} = -2$, $x^{(1)} = 2.5$, $\ldots$



Remarks

Globalization

- Pure Newton's method has unit step length and may diverge from a remote starting point
- To globalize the method, consider line search rules

Hessian modification

- The Hessian may not always be positive definite, and the Newton direction may not always be a descent direction
- Consider modified Hessian

Numerical linear algebra

- Do not invert a matrix $\left[H_f\left(x^{(k)}\right)\right]^{-1}$
- Solve the linear system by iterative methods

$$H_f\left(x^{(k)}\right)p = -\nabla f(x^{(k)})$$

- If $H_f\left(x^{(k)}\right)$ is close to being singular, the linear system is difficult to solve

Practical Newton's method

Practical Newton's method

- Line search Newton's method with modification
- Trust-region Newton's method
- Quasi-Newton method

Line search Newton's method with modification

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}$$

$p^{(k)}$ is from $\left(H_f\left(x^{(k)}\right) + \tau_k I\right) p = -\nabla f(x^{(k)})$

α_k is the step length by certain line search rule

1. **Hessian modification**. If the Hessian $H_f\left(x^{(k)}\right)$ is not positive definite, or is close to being singular, then we can add a positive diagonal matrix $H_f\left(x^{(k)}\right) + \tau_k I$
2. **Line search**. Choose the step length α_k by certain line search rule, for example, by an Armijo backtracking line search[4]

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

   the line search should always try the unit step length $\alpha = 1$ first, so that this step length is used when acceptable

[4] lecture 2, page 25

Algorithm framework

Algorithm (Line search Newton's method with modification)

Choose $x^{(0)}$, $\epsilon > 0$, $\rho \in (0,1)$, $c_1 \in (0,1)$; Set $k \leftarrow 0$

while $\|\nabla f(x^{(k)})\| > \epsilon$ **do**

Choose $\tau_k \geq 0$ such that $B_k = H_f\big(x^{(k)}\big) + \tau_k I \succ 0$

Compute $p^{(k)}$ by solving $B_k p = -\nabla f(x^{(k)})$

$\alpha \leftarrow 1$ (line search always try unit step length first)

repeat until $f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$

$\alpha \leftarrow \rho\alpha$

end(repeat)

$\alpha_k \leftarrow \alpha$

$x^{(k+1)} \leftarrow x^{(k)} + \alpha_k p^{(k)}$

$k \leftarrow k + 1$

end(while)

return $x^{(k)}$

Global convergence

- Bounded modified factorization property is that

$$\|B_k\|\|B_k^{-1}\| \leq C, \text{ for some } C > 0 \text{ and all } k = 0, 1, \ldots$$

  whenever $\{H_f\big(x^{(k)}\big), k = 0, 1, \ldots\}$ is bounded.
- Line search Newton's method with modification converges globally.

Theorem. Suppose that f is twice continuously differentiable, the bounded modified factorization property holds, and the level set $\{x \mid f(x) \leq f(x^{(0)})\}$ is closed and bounded. Then for iterates generated by the line search Newton's method with modification, we have that

$$\lim_{k\to\infty} \nabla f(x^{(k)}) = 0$$

Global convergence

- Bounded modified factorization property is that
$$\|B_k\|\|B_k^{-1}\| \leq C, \text{ for some } C > 0 \text{ and all } k = 0, 1, \ldots$$
whenever $\{H_f\big(x^{(k)}\big), k = 0, 1, \ldots\}$ is bounded.
- Line search Newton's method with modification converges globally.

Theorem. Suppose that f is twice continuously differentiable, the bounded modified factorization property holds, and the level set $\{x \mid f(x) \leq f(x^{(0)})\}$ is closed and bounded. Then for iterates generated by the line search Newton's method with modification, we have that

$$\lim_{k\to\infty} \nabla f(x^{(k)}) = 0$$

Rate of convergence: suppose $x^{(k)} \to x^*$

1. If $H_f(x^*) \succ 0$, then $\tau_k = 0$ for all sufficiently large k, the algorithm reduces to a pure Newton's method, converging Q-quadratically.
2. If $H_f(x^*)$ is not positive definite, theconvergence rate may be only linear.

How to choose τ_k

- If $H_f\big(x^{(k)}\big) \succ 0$, then set $\tau_k = 0$
- If $\lambda_{\min}\Big(H_f\big(x^{(k)}\big)\Big) < 0$, then set $\tau_k > \lambda_{\min}\Big(H_f\big(x^{(k)}\big)\Big)$ such that $B_k = H_f\big(x^{(k)}\big) + \tau_k I \succ 0$
  - ▷ For example, set $\tau_k = 1.5$ works when

$$H_f\big(x^{(k)}\big) = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 6 & 3 \\ 1 & 3 & -0.1 \end{bmatrix} = Q \begin{bmatrix} -1.33 & & \\ & 2.87 & \\ & & 8.37 \end{bmatrix} Q^T$$

- $p^{(k)}$ by solving $B_k p = -\nabla f(x^{(k)})$ is a descent direction of f at $x^{(k)}$

$$\Big(\nabla f(x^{(k)})\Big)^T p^{(k)} = -\Big(p^{(k)}\Big)^T B_k p^{(k)} < 0$$

- If τ_k is very large, $B_k \approx \tau_k I$, and

$$p^{(k)} \approx -\tau_k^{-1} \nabla f(x^{(k)}) \text{ gradient direction}$$

Trust-region Newton's method

- Unlike line search methods, trust-region methods do not require the Hessian to be positive definite
- Taylor series

$$f(x^{(k)}+p) \approx f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T p + \frac{1}{2} p^T H_f(x^{(k)}) p$$

- The search direction $p^{(k)}$ is obtained by solving

$$\min_{p \in \mathbb{R}^n} \left(\nabla f(x^{(k)})\right)^T p + \frac{1}{2} p^T H_f(x^{(k)}) p \quad \text{s.t.} \quad \|p\| \leq \Delta_k \tag{1}$$

  where Δ_k is the trust-region radius. And the iteration is

$$x^{(k+1)} = x^{(k)} + p^{(k)}$$

- The trust-region problem (1) can be solved by trust-region Newton Conjugate Gradient method (not discussed here)
- Global convergence under mild assumptions $\lim_{k \to \infty} \|\nabla f(x^{(k)})\| = 0$

Quasi-Newton method

$$H_f\big(x^{(k)}\big)p = -\nabla f(x^{(k)})$$

- Newton's methods require the Hessian of the objective function at each iteration, which is costly
- Solving the $n \times n$ linear system is also costly, especially when $H_f\big(x^{(k)}\big)$ is close to being singular
- Quasi-Newton methods, like steepest descent, require only the gradient of the objective function (but much faster than steepest descent)
- Key idea: use all the gradients computed throughout the algorithm to approximate the Hessian
- Here we introduce the most popular quasi-Newton algorithm BFGS, named for its discoverers Broyden, Fletcher, Goldfarb, and Shanno

Quasi-Newton method

- "**Quasi**" — **approximate** Hessian
- Quasi-Newton iteration is quite similar to the line search Newton's method
- The key difference is that the approximate Hessian B_k is used in place of the true Hessian, never compute true Hessian

$$B_k p^{(k)} = -\nabla f(x^{(k)})$$

$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)}$ step length α_k by line search

Update B_{k+1} from B_k by some strategies

- Need a wise strategy to approximate true Hessian

Secant equation

- At k-th iteration, we use an approximation B_k of the true Hessian and solve
$$B_k p^{(k)} = -\nabla f(x^{(k)})$$
- The quasi-Newton condition imposed on the update of B_k is
$$B_{k+1}\underbrace{(x^{(k+1)} - x^{(k)})}_{s_k} = \underbrace{\nabla f(x^{(k+1)}) - \nabla f(x^{(k)})}_{y_k}$$
It is called the secant equation
- Denote $H_k = B_k^{-1}$. It allows $p^{(k)}$ be calculated by matrix-vector multiplication $p^{(k)} = -H_k \nabla f(x^{(k)})$
- BFGS idea: approximate Hessian inverse H_{k+1}, imposing the condition
$$s_k = H_{k+1} y_k$$

Secant equation

- The updated H_{k+1} is given by an explicit solution of

$$\begin{aligned} \min_{H} \quad & \|H - H_k\|_W \\ \text{s.t.} \quad & H = H^T, \quad H y_k = s_k \end{aligned}$$

  where $\|H\|_W = \|W^{1/2} H W^{1/2}\|_F$ is a weighted Frobenius norm, and W is a specially chosen weighting matrix
- The unique solution H_{k+1} to the above problem is

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

BFGS

Algorithm (BFGS method)

Choose $x^{(0)}$, $\epsilon > 0$, H_0 (e.g., $= I$); Set $k \leftarrow 0$

while $\|\nabla f(x^{(k)})\| > \epsilon$ **do**

$p^{(k)} = -H_k \nabla f(x^{(k)})$

$x^{(k+1)} \leftarrow x^{(k)} + \alpha_k p^{(k)}$ where α_k is by line search

$s_k = x^{(k+1)} - x^{(k)}$, $y_k = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)})$

$\rho_k = \frac{1}{y_k^T s_k}$

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

$k \leftarrow k + 1$

end(while)

return $x^{(k)}$

- Each iteration $O(n^2)$ operations
- Rate of convergence: Q-superlinear

BFGS

- Each iteration $O(n^2)$ operations
- Rate of convergence: Q-superlinear
- Implementations:
  - ▷ "scipy.optimize.fmin_bfgs" in Python
  - ▷ "fminunc" in Matlab

Proximal Newton method

Proximal Newton method

gradient method	proximal gradient method
$min\, f(x)$	$min\, f(x) + g(x)$
Newton's method	**proximal Newton method**
$min\, f(x)$	$min\, f(x) + g(x)$

Proximal Newton method

$$\min_x \; f(x) + g(x)$$

$$x^{(k+1)} = \arg\min_x \Bigg\{ f(x^{(k)}) + \left(\nabla f(x^{(k)})\right)^T (x - x^{(k)}) \\ + \frac{1}{2}(x - x^{(k)})^T H_f(x^{(k)})(x - x^{(k)}) + g(x) \Bigg\}$$

- If $g = 0$, it reduces to pure Newton's method
- For each iteration, we essentially solve a subproblem

$$\min_x b^T x + x^T A x + g(x)$$

  for some b and A. It is not easy
- Solve with an optimization subroutine for each subproblem