Numerical Optimization

Unit 4: Quasi-Newton and Conjugate Gradient Methods

Che-Rung Lee

Department of Computer Science
National Tsing Hua University

October 12, 2022

Three problems of Newton's method

Three problems of Newton's method:

1. Hessian matrix H may not be positive definite.
2. Hessian matrix H is expensive to compute.
3. The system $\vec{p} = -H^{-1}\vec{g}$ is expensive to solve.

We want to discuss methods to solve the second and the third problems.

Secant equation

- Recall that in the one dimensional optimization problem , the secant method approximate $f''(x_k)$ by $\tilde{h}_k = \dfrac{f'(x_k) - f'(x_{k-1})}{x_k - x_{k-1}}$ and we use $\tilde{h}_k$ in the secant's method.

$$f'(x_k) - f'(x_{k-1}) = \tilde{h}_k(x_k - x_{k-1}) = \tilde{h}_k s_{k-1}.$$

- In multivariable optimization, we want to find an "approximate" Hessian matrix B_k such that

$$\vec{y}_{k-1} = \nabla f_k - \nabla f_{k-1} = B_k \vec{s}_{k-1}. \tag{1}$$

- The above equation is called the "secant equation" in multivariable function.

BFGS

- The BFGS method (1970) (Broyden, Fletcher, Goldtarb, Shanno) solves the following optimization problem,

$$\min_{B} \|B - B_k\| \tag{2}$$

$$\text{subject to } B = B^T \text{ and } Bs_k = y_k \tag{3}$$

- The norm used in the above problem is weighted Frobenius norm,

$$\|A\| = \|W^{1/2} A W^{1/2}\|_F$$

where $\|C\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}^2$.

- The wight used in the above definition is $W = \bar{G}^{-1}$, where

$$\bar{G} = \int_0^1 \nabla^2 f(\vec{x}_k + \tau \alpha_k \vec{p}_k) d\tau$$

is the average Hessian.

BFGS Derivation

- You can use Taylor's theorem or integration law to show that

$$\nabla f(\vec{x}_{k+1}) - \nabla f(\vec{x}_k) = \int_0^1 \nabla^2 f(\vec{x}_k + \tau \alpha_k \vec{p}_k) \alpha_k \vec{p}_k d\tau$$

$$\Rightarrow \vec{y}_k = \bar{G} \vec{s}_k$$

- The solution of above equation leads to the DFP (Davidon–Fletcher–Powell) algorithm .
- BFGS uses a similar idea to solve the inverse Hessian directly. Let H be B_{k+1}^{-1}, and $H_k = B_k^{-1}$.

$$\min_H \| H - H_k \| \tag{4}$$

$$\text{subject to } H = H^T \text{ and } Hy_k = s_k \tag{5}$$

- The inverse of B_{k+1} can be derived explicitly.

$$B_{k+1}^{-1} = (I - \rho_k \vec{s}_k \vec{y}_k^T) B_k^{-1} (I - \rho_k \vec{y}_k \vec{s}_k^T) + \rho_k \vec{s}_k \vec{s}_k^T \text{ where } \rho_k = \frac{1}{\vec{y}_k^T \vec{s}_k}$$

Convergence of BFGS

Theorem (Convergence of BFGS)

Suppose $f = \mathbb{R}^n \to \mathbb{R}$ *is twice continuously differentiable. Consider the iteration* $\vec{x}_{k+1} = \vec{x}_k + \vec{p}_k$ *where* $\vec{p}_k = -B_k^{-1}\nabla f_k$. *If* $\{\vec{x}_k\}$ *converges to* $\vec{x}^*$ *s.t.* $\nabla f(\vec{x}^*) = 0$ *and* $\nabla^2 f(\vec{x}^*)$ *is positive definite, then* $\{\vec{x}_k\}$ *converges superlinearly if and only if* $\lim_{k\to\infty} \dfrac{\|\nabla f_k + \nabla^2 f_k \vec{p}_k\|}{\|\vec{p}_k\|} = 0$

BFGS example

Consider $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{c}^T\vec{x} = \frac{1}{2}\vec{x}^T \begin{bmatrix} 1/4 & 0 \\ 0 & 1 \end{bmatrix} \vec{x} - \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \vec{x}$.

The optimal solution is at $\vec{x}^* = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$.

- Let $\vec{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. $\vec{g}_0 = \nabla f(\vec{x}_0) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$
- Initial guess of Hessian $B_0 = I$. $p_0 = -B_0^{-1} g_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.
- Use exact line search $\phi(\alpha) = \frac{1}{2}\alpha^2 \vec{p}_0^T Q\vec{p}_0 - \alpha\vec{c}^T\vec{p}_0$. Get $\alpha_0 = 1.6$.
- $\vec{x}_1 = \vec{x}_0 + \alpha_0\vec{p}_0 = \begin{bmatrix} 1.6 \\ 1.6 \end{bmatrix}$. $\vec{g}_1 = \nabla f(\vec{x}_1) = Q\vec{x}_1 - \vec{c} = \begin{bmatrix} -0.6 \\ 0.6 \end{bmatrix}$.
- $\vec{s}_0 = \vec{x}_1 - \vec{x}_0 = \begin{bmatrix} 1.6 \\ 1.6 \end{bmatrix}$ and $\vec{y}_0 = \vec{g}_1 - \vec{g}_0 = \begin{bmatrix} 0.4 \\ 1.6 \end{bmatrix}$

BFGS example — continue

- $B_1 = B_0 - \frac{B_0 \vec{s}_0 \vec{s}_0^T B_0}{\vec{s}_0^T B_0 \vec{s}_0} + \frac{\vec{y}_0 \vec{y}_0^T}{\vec{y}_0^T \vec{s}_0} = I - \frac{\vec{s}_0 \vec{s}_0^T}{\vec{s}_0^T \vec{s}_0} + \frac{\vec{y}_0 \vec{y}_0^T}{\vec{y}_0^T \vec{s}_0} = \begin{bmatrix} 0.55 & -0.30 \\ -0.30 & 1.30 \end{bmatrix}$
- $p_1 = -B_1^{-1} g_1 = \begin{bmatrix} 0.96 \\ -0.24 \end{bmatrix}$.
- Use exact line search

$$\phi(\alpha) = \frac{1}{2}(\vec{x}_1 + \alpha \vec{p}_1)^T Q (\vec{x}_1 + \alpha \vec{p}_1) - \vec{c}^T (\vec{x}_1 + \alpha \vec{p}_1).$$

  Get $\alpha_1 = 2.5$.
- $\vec{x}_2 = \vec{x}_1 + \alpha_1 \vec{p}_1 = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$. $\vec{g}_2 = \nabla f(\vec{x}_2) = Q\vec{x}_2 - \vec{c} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.
- $\vec{s}_1 = \vec{x}_2 - \vec{x}_1 = \begin{bmatrix} 2.4 \\ -0.6 \end{bmatrix}$ and $\vec{y}_1 = \vec{g}_2 - \vec{g}_1 = \begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}$
- $B_2 = B_1 - \frac{B_1 \vec{s}_1 \vec{s}_1^T B_1}{\vec{s}_1^T B_1 \vec{s}_1} + \frac{\vec{y}_1 \vec{y}_1^T}{\vec{y}_1^T \vec{s}_1} = \begin{bmatrix} 0.25 & 0 \\ 0 & 1 \end{bmatrix} = Q$

The SR1 update

- B_{k+1} and B_k should be "similar": the symmetric rank 1 (SR1) update: where $\sigma_k = +1$ or -1 and $\vec{u}$ is a vector.

$$B_{k+1} = B_k + \sigma_k \vec{u}\vec{u}^T, \tag{6}$$

- What is $\vec{u}$?
  - Let $\vec{y}_k = \nabla f_{k+1} - \nabla f_k$ and $\vec{s}_k = \vec{x}_{k+1} - \vec{x}_k$.
  - The secant equation can be written as $\vec{y}_k = B_{k+1}\vec{s}_k = (B_k + \sigma_k \vec{u}\vec{u}^T)\vec{s}_k = B_k\vec{s}_k + \sigma_k \vec{u}\vec{u}^T\vec{s}_k$.
  - $\vec{y}_k - B_k\vec{s}_k = (\sigma\vec{u}^T\vec{s}_k)\vec{u} \Rightarrow \vec{u}$ is parallel to $\vec{y}_k - B_k\vec{s}_k$.
  - Let $\vec{u} = \delta(\vec{y}_k - B_k\vec{s}_k)$. Using (1) and (6), one can derive

$$\sigma = \mathrm{sign}(\vec{y}_k^T\vec{s}_k - \vec{s}_k^T B_k\vec{s}_k) \tag{7}$$

$$\delta = \pm(\vec{y}_k^T\vec{s}_k - \vec{s}_k^T B_k\vec{s}_k)^{-1/2} \tag{8}$$

- By substituting (7) and (8) back to (6), one can show that

$$B_{k+1} = B_k + \frac{(\vec{y}_k - B_k\vec{s}_k)(\vec{y}_k - B_k\vec{s}_k)^T}{(\vec{y}_k - B_k\vec{s}_k)^T\vec{s}_k} \tag{9}$$

The Sherman—Morrison—Woodbury formula

- What we really need is not an approximation to H_k, but an approximation to H_k^{-1}.
- If we know B_{k-1}^{-1}, and $B_k = B_{k-1} + \sigma \vec{u}\vec{u}^T$, can we compute B_k^{-1} efficiently?
- The Sherman - Morrison - Woodbury formula.

$$
\begin{aligned}
\hat{A} &= A + \vec{a}\vec{b}^T \\
\hat{A}^{-1} &= A^{-1} - \frac{A^{-1}\vec{a}\vec{b}^T A^{-1}}{1 + \vec{b}^T A^{-1}\vec{a}}
\end{aligned}
$$

- Thus, the formula of SR1 update is

$$
B_{k+1}^{-1} = B_k^{-1} - \frac{(\vec{s}_k - B_k^{-1}\vec{y}_k)(\vec{s}_k - B_k^{-1}\vec{y}_k)^T}{\vec{y}_k^T B_k^{-1}\vec{y}_k - \vec{y}_k^T \vec{s}_k}
$$

Numerical properties of the SR1 update

Convergence for a quadratic function

Suppose $f(\vec{x}) = \vec{b}^T\vec{x} + \frac{1}{2}\vec{x}^T A\vec{x}$ and A is symmetric positive definite. Then for any starting point $\vec{x}_0$ and any starting H_0, SR1 converges to the minimizer in at most n steps, where n is the problem size, provided that $(\vec{s}_k - B_k^{-1}\vec{y}_k)^T\vec{y}_k \neq 0$ for all k.

Problems of the SR1 method

1. $(\vec{y}_k - B_k\vec{s}_k)^T\vec{s}_k$ may be 0 $\Rightarrow$ Just use $B_{k+1} = B_k$.
2. B_{k+1} may be indefinite $\Rightarrow$ Use it with the trusted region framework.

SR1 example

Consider $f(\vec{x}) = \frac{1}{2}\vec{x}^T Q\vec{x} - \vec{c}^T\vec{x} = \frac{1}{2}\vec{x}^T \begin{bmatrix} 1/4 & 0 \\ 0 & 1 \end{bmatrix} \vec{x} - \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \vec{x}$.

The optimal solution is at $\vec{x}^* = \begin{bmatrix} 4 \\ 1 \end{bmatrix}$.

- Let $\vec{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. $\vec{g}_0 = \nabla f(\vec{x}_0) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$
- Initial guess of Hessian $B_0 = I$. $\vec{p}_0 = -B_0^{-1}\vec{g}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.
- Use exact line search $\phi(\alpha) = \frac{1}{2}\alpha^2\vec{p}_0^T Q\vec{p}_0 - \alpha\vec{c}^T\vec{p}_0$. Get $\alpha_0 = 1.6$.
- $\vec{x}_1 = \vec{x}_0 + \alpha_0\vec{p}_0 = \begin{bmatrix} 1.6 \\ 1.6 \end{bmatrix}$. $\vec{g}_1 = \nabla f(\vec{x}_1) = Q\vec{x}_1 - \vec{c} = \begin{bmatrix} -0.6 \\ 0.6 \end{bmatrix}$.
- $\vec{s}_0 = \vec{x}_1 - \vec{x}_0 = \begin{bmatrix} 1.6 \\ 1.6 \end{bmatrix}$ and $\vec{y}_0 = \vec{g}_1 - \vec{g}_0 = \begin{bmatrix} 0.4 \\ 1.6 \end{bmatrix}$

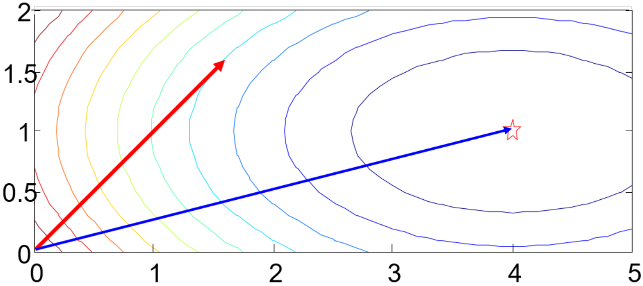
SR1 example — continue

SR1 update formula

$$B_1 = B_0 + \frac{(\vec{y}_0 - B_0\vec{s}_0)(\vec{y}_0 - B_0\vec{s}_0)^T}{(\vec{y}_0 - B_0\vec{s}_0)^T\vec{s}_0} = I + \frac{(\vec{y}_0 - \vec{s}_0)(\vec{y}_0 - \vec{s}_0)^T}{(\vec{y}_0 - \vec{s}_0)^T\vec{s}_0}.$$

$$B_1 = \begin{bmatrix} 0.25 & 0 \\ 0 & 1 \end{bmatrix} = Q$$

$$\vec{p}_1 = -B_1^{-1}\vec{g}_1 = \begin{bmatrix} 2.4 \\ -0.6 \end{bmatrix}$$



Enforce Positive Definite of B_k

- The curvature condition keeps B_k positive definite.

$$\vec{y}_k = B_{k+1}\vec{s}_k \Rightarrow \vec{s}_k^T B_{k+1}\vec{s}_k = \vec{s}_k^T \vec{y}_k > 0 \tag{10}$$

- If f is a convex function, (10) is satisfied automatically.
- For non-convex functions, we can enforce (10) by using line-search methods. For example, if Wolf conditions are satisfied,

$$\vec{y}_k^T \vec{s}_k \geq (c_2 - 1)\alpha_k \nabla f_k^T \vec{p}_k$$

  which will be positive. (why?)
  - Recall the second Wolf condition: For some $c_2 \in (c_1, 1)$.

$$\nabla f(\vec{x}_k + \alpha_k \vec{p}_k)^T \vec{p}_k \geq c_2 \nabla f_k^T \vec{p}_k$$

  - And the facts: $\vec{s}_k = \alpha_k \vec{p}_k$, and $\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k$.

Review of the quadratic model

- Consider a quadratic function

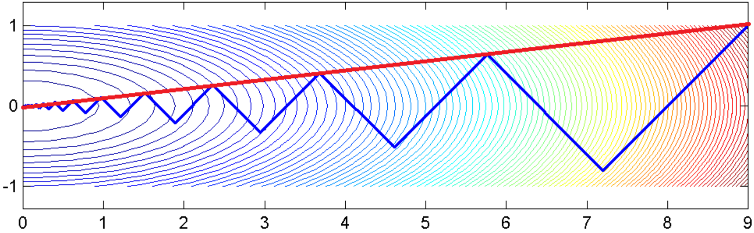
$$f(\vec{x}) = \frac{1}{2}\vec{x}^T A\vec{x} - \vec{x}^T\vec{b} + c$$

- To find the optimal solution of $f(\vec{x})$ is equivalent to find $\nabla f(\vec{x}) = A\vec{x} - \vec{b} = 0$, which is to solve the linear system $A\vec{x} = \vec{b}$.
- We call $\vec{r} = \vec{b} - A\vec{x}$ the *residual* for the linear system $A\vec{x} = \vec{b}$. The smaller $\|\vec{r}\|$ is, the better solution $\vec{x}$ is.

$$\begin{aligned}
\vec{r} &= \vec{b} - A\vec{x} = A\vec{x}^* - A\vec{x} \\
\|\vec{x}^* - \vec{x}\| &= \|A^{-1}\vec{r}\| \le \|A^{-1}\|\|\vec{r}\| \\
&= \|A^{-1}\|\|A\|\frac{\|\vec{r}\|}{\|A\|} = \kappa(A)\frac{\|\vec{r}\|}{\|A\|}
\end{aligned}$$

The steepest descent directions

- Recall the steepest descent method: $\vec{p}_k = -\nabla f(\vec{x}) = \vec{b} - A\vec{x}$ and $\alpha_k = -\dfrac{\vec{p}_k^T \vec{g}_k}{\vec{p}_k^T A_k \vec{p}_k}$.
- The trace of $\{x_k\}$ shows a zigzag pattern.



- Relation of $\vec{p}_k$ and $\vec{p}_{k+1}$: Since $\vec{x}_{k+1} = \vec{x}_k + \alpha_k \vec{p}_k$,

$$\begin{aligned}
\vec{p}_{k+1} &= -\nabla f(\vec{x}_{k+1}) = \vec{b} - A(\vec{x}_k + \alpha_k \vec{p}_k) = \vec{b} - A\vec{x}_k - \alpha_k A\vec{p}_k \\
&= \vec{p}_k - \alpha_k A\vec{p}_k \\
\vec{p}_k^T \vec{p}_{k+1} &= \vec{p}_k^T(\vec{p}_k - \alpha_k A\vec{p}_k) = \vec{p}_k^T \vec{p}_k - \alpha_k \vec{p}_k^T A\vec{p}_k = 0
\end{aligned}$$

Conjugate gradient method (CG)

- A symmetric positive definite matrix can define an "inner product":

$$\langle \vec{a}, \vec{b} \rangle_A \equiv \vec{a}^T A \vec{b}.$$

- Vector $\vec{a}, \vec{b}$ are called A-conjugate or A-orthogonal if $\langle \vec{a}, \vec{b} \rangle_A = 0$
- Let $\vec{p}_{k+1} = -\vec{r}_{k+1} + \beta_k \vec{p}_k$. We want $\vec{p}_{k+1}$ and $\vec{p}_k$ to be A-conjugate.

$$\langle \vec{p}_{k+1}, \vec{p}_k \rangle_A = \vec{p}_k^T A(-\vec{r}_{k+1} + \beta_k \vec{p}_k) = -\vec{p}_k^T A \vec{r}_{k+1} + \beta_k \vec{p}_k^T A \vec{p}_k = 0.$$

$$\Rightarrow \beta_k = \frac{\vec{p}_k^T A \vec{r}_{k+1}}{\vec{p}_k^T A \vec{p}_k}$$

- Use the same $\alpha_k = \dfrac{-\vec{p}_k^T \vec{g}_k}{\vec{p}_k^T A_k \vec{p}_k}$ as the steepest descent method.
- To save one matrix-vector multiplication, residuals can be updated as

$$\vec{r}_{k+1} = \vec{b} - A\vec{x}_{k+1} = \vec{b} - A(\vec{x}_k + \alpha_k \vec{p}_k) = \vec{r}_k - \alpha_k A \vec{p}_k$$

The conjugate gradient algorithm

Put everything together...

The conjugate gradient algorithm

1. Given $\vec{x}_0$. Let $\vec{p}_0 = \vec{b} - A\vec{x}_0$ and $\vec{r}_0 = \vec{p}_0$.
2. For $k = 0, 1, 2, \ldots$ until $\|\vec{r}_k\| \leq \epsilon$

$$\begin{aligned}
\alpha_k &= \frac{\vec{p}_k^T \vec{r}_k}{\vec{p}_k^T A\vec{p}_k} \\
\vec{x}_{k+1} &= \vec{x}_k + \alpha_k \vec{p}_k \\
\vec{r}_{k+1} &= \vec{r}_k - \alpha_k A\vec{p}_k \\
\beta_k &= \frac{\vec{r}_{k+1}^T A\vec{p}_k}{\vec{p}_k^T A\vec{p}_k} \\
\vec{p}_{k+1} &= -\vec{r}_{k+1} + \beta_k \vec{p}_k
\end{aligned}$$

Example

$f(\vec{x}) = \frac{1}{2}\vec{x}^T \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \vec{x}$ and $\vec{x}_0 = \begin{pmatrix} 9 \\ 1 \end{pmatrix}$, in which

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix}, \vec{b} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Initially,

$$\vec{p}_0 = \vec{b} - A\vec{x}_0 = \begin{pmatrix} -9 \\ -9 \end{pmatrix} = \vec{r}_0.$$

The first iteration,

$$A\vec{p}_0 = \begin{pmatrix} -9 \\ -81 \end{pmatrix}$$

$$\alpha_0 = \frac{2 \times 81}{81 + 9 \times 81} = \frac{1}{5}$$

$$\vec{x}_1 = \begin{pmatrix} 9 \\ 1 \end{pmatrix} + \frac{1}{5}\begin{pmatrix} -9 \\ -9 \end{pmatrix} = \begin{pmatrix} 7.2 \\ -0.8 \end{pmatrix}$$

Example–continue

$$\vec{r}_1 = \begin{pmatrix} -9 \\ -9 \end{pmatrix} - \frac{1}{5} \begin{pmatrix} -9 \\ -81 \end{pmatrix} = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix}$$

$$\beta_1 = \frac{7.2^2 \times 2}{9^2 \times 2} = \left(\frac{7.2}{9}\right)^2 = 0.64$$

$$\vec{p}_1 = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix} + 0.8 \times 0.8 \begin{pmatrix} -9 \\ -9 \end{pmatrix} = \begin{pmatrix} -1.8 \times 7.2 \\ 0.2 \times 7.2 \end{pmatrix}$$

The second iteration

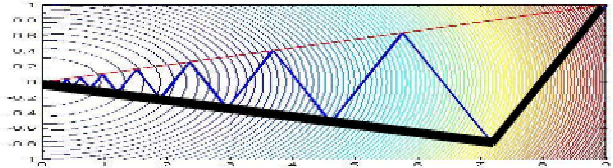
$$\alpha_1 = \frac{7.2^2 \times 2}{12.96^2 + 12.96 \times 1.44} = \frac{1}{1.8}$$

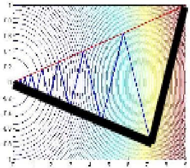
$$\vec{x}_2 = \begin{pmatrix} 7.2 \\ -0.8 \end{pmatrix} + \frac{1}{1.8} \begin{pmatrix} -12.96 \\ 1.44 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\vec{r}_2 = \begin{pmatrix} -7.2 \\ 7.2 \end{pmatrix} - \frac{1}{1.8} \begin{pmatrix} -1.8 \times 7.2 \\ 0.2 \times 7.2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Properties of the CG

Trace of the example (compared with Steepest-descent direction and Newton's direction.)





Theorem (Convergence)

For any $\vec{x} \in \mathbb{R}^n$, if A has m distinct eigenvalues, the CG will terminate at the solution at most m iterations. Moreover, if A has eigenvalues $\lambda_1 \le \lambda_2 \le \cdots \lambda_n$,

$$\|\vec{x}_{k+1} - \vec{x}^*\|_A^2 \le \left(\frac{\lambda_{n-k} - \lambda_1}{\lambda_{n-k} + \lambda_1}\right)^2 \|\vec{x}_0 - \vec{x}^*\|_A^2$$

Preconditioned CG (PCG)

- The convergence of the CG can be very small if If $\kappa(A)^{-1} = \frac{\lambda_{min}}{\lambda_{max}}$ is small.
- If we can find a matrix M such that the ratio of the smallest eigenvalue and the largest eigenvalue of $MA \approx I$, then the convergence can be faster.
- The original problem $A\vec{x} = \vec{b}$ becomes $MA\vec{x} = M\vec{b}$.

$$\vec{x} = (MA)^{-1}M\vec{b} = A^{-1}M^{-1}M\vec{b} = A^{-1}\vec{b}$$

Truncated Newton method

1. Hessian matrix A may fail to be positive definite.
2. The linear system $A\vec{x} = \vec{b}$ need not be solved "exactly". (Recall the modified Newton's method.)
3. Therefore, we can stop the iterations as soon as we found the indefiniteness of A or when $\|\vec{r}\| < \epsilon$.
4. More details will be discussed in the next unit.

Hessian free CG

- When the problem is large, generating and storing matrix A are expensive. (Matrix A may not be sparse in many cases.)
- We don't really need the Hessian matrix A explicitly. What we need is $A\vec{v}$.
- Matrix A is a special matrix $\nabla^2 f_k$. Recall the definition of the directional derivative (See note 2),

$$A\vec{v} = \nabla^2 f_k \vec{v} \approx \frac{\nabla f(\vec{x}_k + h\vec{v}) - \nabla f(\vec{x}_k)}{h}$$

  for some small enough h.
- Other methods that can solve large-scale problems include Limited memory BFGS, etc.

Nonlinear CG

- When the problem is not quadratic, similar methods can be used for the nonlinear optimization.
- Two differences:
  1. Step length α_k is computed by the line search algorithm.
  2. The formula of computing β_k.
     - Ex: The Fletcher-Reeves method, $\beta_k = \dfrac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}$.
     - Ex: The Polak-Ribière method, $\beta_k = \dfrac{\nabla f_{k+1}^T (\nabla f_{k+1} - \nabla f_k)}{\nabla f_k^T \nabla f_k}$.
     - Ex: The Hestenes-Stiefel method, $\beta_k = \dfrac{\nabla f_{k+1}^T (\nabla f_{k+1} - \nabla f_k)}{(\nabla f_{k+1} - \nabla f_k)^T \vec{p}_k}$