Numerical Optimization

Instructor : Sung Chan Jun

Week #8 : October 21 – 25, 2019

Announcements

- No Class
  - Date : October 23 (Wednesday), 2019
- Makeup Class
  - Date : October 21 (Monday), 2019
  - Time : 7:00 PM – 8:15 PM
  - No attendance check

Numerical Optimization (2019 Fall)

BioComputing

Course Syllabus (tentative)

1st week	Sept. 2, 4	Introduction of optimization	
2nd week	Sept. 9, 11	Univariate Optimization	
3rd week	Sept. 16, 18	Univariate Optimization	
4th week	Sept. 23, 25	Unconstrained Multivariate Optimization	
5th week	Sept. 30, Oct. 2	Unconstrained Multivariate Optimization	
6th week	Oct. 7, 9	Unconstrained Multivariate Optimization	National Holiday (Oct. 9)
7th week	Oct. 14, 16	Unconstrained Multivariate Optimization	Midterm (Oct. 16)
8th week	Oct. 21, 23	Unconstrained Multivariate Optimization	

Course Syllabus (tentative)

9th week	Oct. 28, 30	Constrained Multivariate Optimization	
10th week	Nov. 4, 6	Constrained Multivariate Optimization	
11th week	Nov. 11, 13	Constrained Multivariate Optimization	
12th week	Nov. 18, 20	Global Optimization	
13th week	Nov. 25, 27	Global Optimization	
14th week	Dec. 2, 4	Global Optimization, Wrap-up	
15th week	Dec. 9	Final Exam	Final Exam (Dec. 9)

Recall Last Week

- Multivariate Optimization: Second Derivative methods
  - $H(\mathbf{x}_k)$ (2^{nd} order derivative) is approximated as $\mathbf{B}_k$
    - $\mathbf{B}_k$ should consist of gradients (1^{st} order derivatives) and function evaluations. That is, $H(\mathbf{x}_k) \approx \mathbf{B}_k$. Then computing Hessian should be much cheaper.
    - Computing inverse of approximation $\mathbf{B}_k$ should be done easily.
  - Then use approximate Hessian $\mathbf{B}_k$ as follows:
    - $\underbrace{H(\mathbf{x}_k)\,\mathbf{p}_k = -\nabla f(\mathbf{x}_k)}_{\text{Newton's}} \rightarrow \underbrace{\mathbf{B}_k\,\mathbf{p}_k = -\nabla f(\mathbf{x}_k)}_{\text{Modified version of Newton's}}$

Recall Last Week

- Multivariate Optimization: Quasi-Newton's
  - Investigation of Hessian
    - Taylor expansion : $\nabla f(\mathbf{x}_k+\alpha_k\mathbf{p}_k) = \nabla f(\mathbf{x}_k) + H(\mathbf{x}_k)\,\alpha_k\mathbf{p}_k + O(|\mathbf{p}_k|^2)$.
    - Then $\nabla f(\mathbf{x}_k+\alpha_k\mathbf{p}_k) \approx \nabla f(\mathbf{x}_k) + H(\mathbf{x}_k)\,\alpha_k\mathbf{p}_k \rightarrow H(\mathbf{x}_k)\,\alpha_k\mathbf{p}_k \approx \nabla f(\mathbf{x}_k+\alpha_k\mathbf{p}_k) - \nabla f(\mathbf{x}_k)$
    - In other expression, since $\mathbf{x}_{k+1} = \mathbf{x}_k+\alpha_k\mathbf{p}_k$, $\boxed{H(\mathbf{x}_k)\,(\mathbf{x}_{k+1} - \mathbf{x}_k) \approx \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)}$
      - Thus, Hessian may satisfies secant equation approximately. **`Secant Equation'**
  - Seek $\mathbf{B}_k$ which is an approximation to $H(\mathbf{x}_k)$, that is, $\mathbf{B}_k \approx H(\mathbf{x}_k)$.

• Symmetry & positive definite • $\mathbf{B}_{k+1} - \mathbf{B}_k$ has low rank • Satisfy secant equation	⇒	• To keep Hessian property. • To be easy to update $\mathbf{B}_{k+1}$ from $\mathbf{B}_k$. • Hessian satisfies secant equation approximately

Recall Last Week

- Multivariate Optimization – Quasi Newton's
  - $H(\mathbf{x}_k) \approx \mathbf{B}_k$ (approximation of Hessian)
    - $\mathbf{B}_k$ is updated with $\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{U}_k$. (since $\mathbf{B}_{k+1} - \mathbf{B}_k$ has low rank)
      - $\mathbf{U}_k$ is low rank and is depending on $\mathbf{B}_k$, $\nabla f(\mathbf{x}_{k+1})$, $\nabla f(\mathbf{x}_k)$, $\mathbf{x}_{k+1}$, and $\mathbf{x}_k$.
      - Such $\mathbf{B}_k$ is applied to $\mathbf{B}_k\,\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$ in place of $H(\mathbf{x}_k)$. "**Quasi Newton's**"
    - How to give $\mathbf{U}_k$?
      - SR1-update (Rank 1 update)
      - BFGS-update (Rank 2 update)
      - DFP-update (Rank 2 update)

Recall Last Week

- Quasi-Newton's : SR1 (symmetric-rank-1) update
  - $\mathbf{B}_{k+1} = \mathbf{B}_k + \sigma\mathbf{v}\mathbf{v}^T$ ($\mathbf{v}$: vector, σ: either 1 or -1)
    - $\mathbf{v} = \delta(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)$
    - $\delta^2 = |1/(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k|$ and $\sigma = \text{sign}[(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k]$

  Finally, we have $\boxed{\mathbf{B}_{k+1} = \mathbf{B}_k + \dfrac{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T}{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k}}$

  - How to compute inverse of $\mathbf{B}_k$, that is, $(\mathbf{B}_k)^{-1} = \mathbf{D}_k$

$$\boxed{\mathbf{B}_k\mathbf{p}_k = -\nabla f(\mathbf{x}_k)} \Rightarrow \boxed{\mathbf{p}_k = -\mathbf{D}_k\nabla f(\mathbf{x}_k)}$$

$$\mathbf{D}_{k+1} = \mathbf{B}_{k+1}^{-1} = \left[\mathbf{B}_k + \frac{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T}{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k}\right]^{-1}$$

$$= \mathbf{B}_k^{-1} + \frac{(\mathbf{s}_k - \mathbf{B}_k^{-1}\mathbf{y}_k)(\mathbf{s}_k - \mathbf{B}_k^{-1}\mathbf{y}_k)^T}{(\mathbf{s}_k - \mathbf{B}_k^{-1}\mathbf{y}_k)^T\mathbf{y}_k} = \mathbf{D}_k + \frac{(\mathbf{s}_k - \mathbf{D}_k\mathbf{y}_k)(\mathbf{s}_k - \mathbf{D}_k\mathbf{y}_k)^T}{(\mathbf{s}_k - \mathbf{D}_k\mathbf{y}_k)^T\mathbf{y}_k}$$

Recall Last Week

- Quasi-Newton's : SR1 (symmetric-rank-1) update
  - $\mathbf{B}_{k+1}$ may be not positive definite even if $\mathbf{B}_k$ is positive definite.
  - Possible for denominator to be zero, i.e, $(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k = 0$. When it occurs, the method will be break-down.
  - Strategy of SR1 to avoid breaking down
    - If $(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k \geq r|\mathbf{s}_k|\cdot|\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k| > 0$ $(r\in(0,1))$, then accept update.
    - Otherwise, reject update and then $\mathbf{B}_{k+1} := \mathbf{B}_k$.

Multivariate Optimization: Quasi-Newton's Method

- Rank-2 update
  - BFGS update

    (Broyden, Fletcher, Goldfarb, and Shanno)

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \text{ assuming } \mathbf{s}_k^T\mathbf{y}_k > 0$$

$$\mathbf{s}_k := \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k := \nabla f_{k+1} - \nabla f_k$$

Numerical Optimization (2019 Fall)

Multivariate Optimization:
Quasi-Newton's Method

(BFGS)

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \text{ assuming } s_k^T y_k > 0$$

$$s_k := x_{k+1} - x_k, \quad y_k := \nabla f_{k+1} - \nabla f_k$$

$$\boxed{B_k p_k = -\nabla f(x_k)} \Rightarrow \boxed{p_k = -B_k^{-1} \nabla f(x_k)}$$

Assuming $D_k := B_k^{-1}$

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

$$\rho_k := 1/(y_k^T s_k)$$

(Inverse Version of BFGS)

Sherman-Morrison Identities

(Sherman-Morrison Identity)

- If **A** is nonsingular and **c**, **d** are n x 1 matrices, then

$$(\mathbf{A} + \mathbf{c}\mathbf{d}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{c}\mathbf{d}^T\mathbf{A}^{-1}}{1 + \mathbf{d}^T\mathbf{A}^{-1}\mathbf{c}} \quad \text{when } 1 + \mathbf{d}^T\mathbf{A}^{-1}\mathbf{c} \neq 0$$

(Sherman-Morrison-Woodbury Identity)

- If **A** is a n x n nonsingular matrix, **C** & **D** are n x k matrices, and $(\mathbf{I} + \mathbf{D}^T\mathbf{A}^{-1}\mathbf{C})$ is nonsingular, then

$$(\mathbf{A} + \mathbf{C}\mathbf{D}^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{C}(\mathbf{I} + \mathbf{D}^T\mathbf{A}^{-1}\mathbf{C})^{-1}\mathbf{D}^T\mathbf{A}^{-1}.$$

- Sherman-Morrison-Woodbury is a generalization of Sherman-Morrison.
- When $\mathbf{A}^{-1}$ is known and minor update in **A** is needed, Sherman–Morrison shows how the previously computed information in $\mathbf{A}^{-1}$ can be updated to produce the new inverse.

Multivariate Optimization:
Quasi-Newton's Method

- BFGS update derivation
  - Seek $\mathbf{D}$ minimizing $|\mathbf{D} - \mathbf{D}_k|_{WF}$ under two conditions : $\mathbf{D} = \mathbf{D}^T$, $\mathbf{D}\mathbf{y}_k = \mathbf{s}_k$ and $\mathbf{D}$ is positive definite. ($\mathbf{y}_k := \nabla f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) - \nabla f(\mathbf{x}_k)$, $\mathbf{s}_k := \alpha_k \mathbf{p}_k$)
    - Take into account $\mathbf{W}$-weighted Frobenius matrix norm $|\ |_{\mathbf{WF}}$ such that $\mathbf{W}\mathbf{s}_k = \mathbf{y}_k$.
    - $\mathbf{W} := \mathbf{G}_k$ (averaged Hessian) defined by $\mathbf{G}_k = \left[\int_0^1 \nabla^2 f(\mathbf{x}_k + \tau\alpha_k \mathbf{p}_k) d\tau\right]$

    satisfy $\mathbf{W}\mathbf{s}_k = \mathbf{y}_k$.
  - (BFGS) $$\mathbf{D}_{k+1} = (\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^T)\mathbf{D}_k(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T) + \rho_k \mathbf{s}_k \mathbf{s}_k^T$$ $$\rho_k := 1/(\mathbf{y}_k^T \mathbf{s}_k)$$
  - It is noted that Different variants are obtained by different choices of weighting matrix $\mathbf{W}$.

Multivariate Optimization:
Quasi-Newton's : BFGS

Recall : 2nd Wolfe Condition

$$\boxed{\nabla f(\mathbf{x}_k + \alpha_k \mathbf{p}_k)\cdot \mathbf{p}_k \geq c_2 \nabla f(\mathbf{x}_k)\cdot \mathbf{p}_k, \quad 0 < c_1 < c_2 < 1}$$

- Remarks
  - Wolfe condition yields $\mathbf{s}_k^T \mathbf{y}_k > 0$.

    (*Proof*)

    If α_k satisfies the Wolfe conditions, by 2nd condition $\nabla f^T_{k+1} \mathbf{s}_k \geq c_2 \nabla f^T_k \mathbf{s}_k$ (since $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k = \alpha_k \mathbf{p}_k$)

    it gives $\mathbf{y}_k^T \mathbf{s}_k = (\nabla f_{k+1} - \nabla f_k)^T \mathbf{s}_k \geq c_2 \nabla f^T_k \mathbf{s}_k - \nabla f_k^T \mathbf{s}_k = (c_2 - 1)\nabla f^T_k \mathbf{s}_k$.

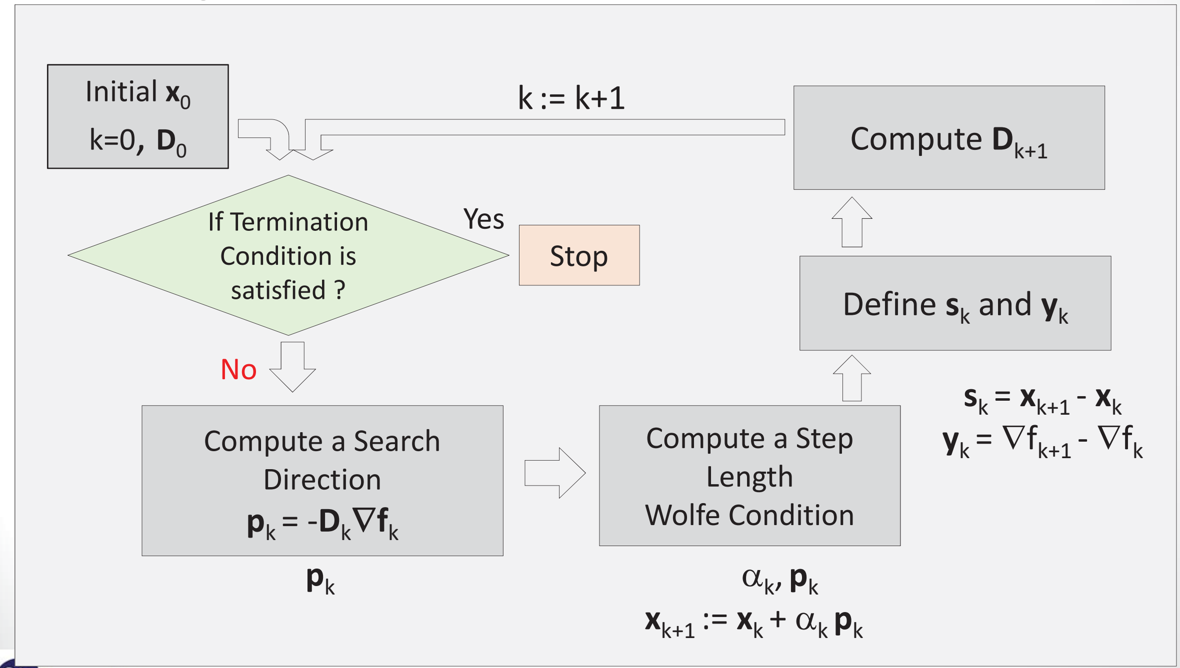
    Now we get $y_k^T \mathbf{s}_k \geq (c_2 - 1)\alpha_k \nabla f^T_k \mathbf{p}_k$. (since $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k = \alpha_k \mathbf{p}_k$ and $\mathbf{y}_k = \nabla f_{k+1} - \nabla f_k$)

    $\nabla f^T_k \mathbf{p}_k < 0$ (since $\mathbf{p}_k$ is a descending direction) and $c_2 - 1 < 0$. Thus, $(c_2 - 1)\alpha_k \nabla f^T_k \mathbf{p}_k > 0$.

    Finally, $y_k^T \mathbf{s}_k \geq (c_2 - 1)\alpha_k \nabla f^T_k \mathbf{p}_k > 0$.
  - If $\mathbf{B}_k$ and $\mathbf{D}_k$ are positive definite, then so are $\mathbf{B}_{k+1}$ and $\mathbf{D}_{k+1}$.

Multivariate Optimization:
Quasi-Newton's Method

- BFGS algorithm



Multivariate Optimization:
Quasi-Newton's Method

- Notes
  - Different variants are obtained by different choices of weighting matrix **W**.
  - Bad situations : when $\mathbf{s}_k^T \mathbf{y}_k$ is so tiny

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{\mathbf{B}_k \mathbf{s}_k \mathbf{s}_k^T \mathbf{B}_k}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k} + \frac{\mathbf{y}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{s}_k}, \text{ assuming } \mathbf{s}_k^T \mathbf{y}_k > 0$$
$$\mathbf{s}_k := \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k := \nabla f_{k+1} - \nabla f_k$$

$$\mathbf{D}_{k+1} = (\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^T)\mathbf{D}_k(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T) + \rho_k \mathbf{s}_k \mathbf{s}_k^T$$
$$\rho_k := 1/(\mathbf{y}_k^T \mathbf{s}_k)$$

    - Good news : BFGS has effective self-correcting property even if $\mathbf{D}_k$ is a poor approximation.
  - It is known that BFGS is the most effective among them.

Multivariate Optimization: Quasi-Newton's Method

- Convergence of BFGS
  - Assume that $f(\mathbf{x})$ is twice continuously differentiable and
    - $L = \{ \mathbf{x} \in \Re^n | f(\mathbf{x}) \leq f(\mathbf{x}_0)\}$ is convex, and $\exists$ m, M > 0 such that $m|\mathbf{z}|^2 \leq \mathbf{z}^T(\nabla^2 f(\mathbf{x}))\mathbf{z} \leq M|\mathbf{z}|^2$ for all $\mathbf{z} \in \Re^n$, $\mathbf{x} \in L$.
    - $\mathbf{B}_0$ is a symmetric positive definite matrix.
  - Then the sequence $\{\mathbf{x}_n\}$ of BFGS converges to the minimizer $\mathbf{x}^*$ of $f(\mathbf{x})$.
- Convergence rate of BFGS
  - Assume that $f(\mathbf{x})$ is twice continuously differentiable and
    - The sequence $\{\mathbf{x}_n\}$ converges to the minimizer $\mathbf{x}^*$ of $f(x)$.
    - $|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{x}^*)| \leq L|\mathbf{x} - \mathbf{x}^*|$ at $\mathbf{x}^*$.
  - Then the sequence $\{\mathbf{x}_n\}$ of BFGS converges super-linearly (rate > 1) to $\mathbf{x}^*$.

Multivariate Optimization: Quasi-Newton's Method

- Broyden Class

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + \phi_k (s_k^T B_k s_k) v_k v_k^T$$

ϕ_k is a scalar and $v_k = \left[\frac{y_k}{y_k^T s_k} - \frac{B_k s_k}{s_k^T B_k s_k} \right]$

Recall : BFGS update

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, \text{ assuming } s_k^T y_k > 0$$

$s_k := x_{k+1} - x_k,\ y_k := \nabla f_{k+1} - \nabla f_k$

  - $\phi_k = 0$ (BFGS) and $\phi_k = 1$ (DFP)
  - $\mathbf{B}_{k+1} = (1 - \phi_k)\, \mathbf{B}^{BFGS}{}_{k+1} + \phi_k\, \mathbf{B}^{DFP}{}_{k+1}$, $\phi_k \in (0,1)$

    "Restricted Broyden Class"

- (Question) What is DFP Quasi-Newton's method?

Multivariate Optimization:
Derivative-based methods

Method of Steepest Descent	Newton's Method	Quasi Newton's Method
Direction $\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$ • Global convergence • Slow convergence near minimum	Direction $\mathbf{p}_k = -(\nabla^2 f(\mathbf{x}_k))^{-1}\nabla f(\mathbf{x}_k)$ • Fast convergence (quadratic) • Require expensive Hessian computing every iteration	Direction $\mathbf{p}_k = -\mathbf{B}_k^{-1}\nabla f(\mathbf{x}_k)$ $\mathbf{B}_k \approx (\nabla^2 f(\mathbf{x}_k))$ • Relatively fast convergence close to Newton's • Do not require Hessian computing

Multivariate Optimization:
Derivative Based Methods

- Homework #4

  Due date : October 30 (Wednesday), 2019 10:30 AM

  - Justify the inverse version of BFGS by using Sherman-Morrison identity.

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k^T\mathbf{B}_k}{\mathbf{s}_k^T\mathbf{B}_k\mathbf{s}_k} + \frac{\mathbf{y}_k\mathbf{y}_k^T}{\mathbf{y}_k^T\mathbf{s}_k},\ \text{assuming } \mathbf{s}_k^T\mathbf{y}_k > 0$$

$$\mathbf{s}_k := \mathbf{x}_{k+1} - \mathbf{x}_k,\quad \mathbf{y}_k := \nabla f_{k+1} - \nabla f_k$$

Assuming $\mathbf{H}_k := \mathbf{B}_k^{-1}$

$$\mathbf{D}_{k+1} = (\mathbf{I} - \rho_k\mathbf{s}_k\mathbf{y}_k^T)\mathbf{D}_k(\mathbf{I} - \rho_k\mathbf{y}_k\mathbf{s}_k^T) + \rho_k\mathbf{s}_k\mathbf{s}_k^T$$

$$\rho_k := 1/(\mathbf{y}_k^T\mathbf{s}_k)$$

(Inverse Version of BFGS)

Multivariate Optimization: Derivative Based Methods

- Homework #4 (**Implementation**)

  Due date : October 30 (Wednesday), 2019 10:30 AM

  - Implement the following numerical methods:
    - The method of steepest descent
    - Newton's method
    - Quasi Newton's method (BFGS)
  - Compare their performance for the following three problems:
    - $f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$
    - $f(x, y) = 40(y - x^2)^2 + (1-x)^2$
    - $f(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$
  - First start at (2.0, 2.0) at each function. Then use different starting points to discuss how approximate points are moving.

Multivariate Optimization: Conjugate Gradient Method

- Conjugate Gradient Method (CG)
  - Iterative method to solve a linear system $\mathbf{Ax} = \mathbf{b}$ for a square symmetric positive definite matrix $\mathbf{A}$.
  - It is interesting that
    - Solving linear system

      $$\boxed{\mathbf{Ax} = \mathbf{b}}$$

    $\Leftrightarrow$ Solving minimization problem

      $$\boxed{\min\ [\tfrac{1}{2}\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x}]}$$

Multivariate Optimization:
Conjugate Gradient Method

- Solving linear system ⇔ Solving minimization problem

$\mathbf{Ax} = \mathbf{b}$ ⇔ $\min\ [½\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x}]$

Proof)

Let $f(\mathbf{x}) = ½\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x}$.

Assume $\mathbf{x}_0$ is a solution of $\mathbf{Ax} = \mathbf{b}$.

Then $\nabla f(\mathbf{x}) = ½\mathbf{Ax} + ½(\mathbf{x}^T\mathbf{A})^T - (\mathbf{b}^T)^T = \mathbf{Ax} - \mathbf{b}$. $\nabla f(\mathbf{x}_0) = \mathbf{0}$.

(since $\mathbf{A}$ is symmetric and $\mathbf{x}_0$ is a solution of $\mathbf{Ax} = \mathbf{b}$.)

Also, hessian of $f(\mathbf{x})$ is $\nabla(\nabla f(\mathbf{x})) = (\mathbf{A}^T)^T = \mathbf{A} > 0$ at $\mathbf{x}_0$.

(since $\mathbf{A}$ is positive definite).

Due to optimality condition, $\mathbf{x}_0$ is a local minimum point of $f(\mathbf{x})$.

Assume $\mathbf{x}_0$ is a local minimum point of $f(\mathbf{x})$.

Due to optimality condition, $\nabla f(\mathbf{x}_0) = \mathbf{0}$. So, $\mathbf{Ax}_0 - \mathbf{b} = \mathbf{0}$.

Finally, $\mathbf{x}_0$ is a solution of $\mathbf{Ax} = \mathbf{b}$.

Recall : Optimality Conditions

- (**NC**) Necessary condition for a local minimum

grad(f(x)) = 0, H(x) ≥ 0.

- (**SC**) Sufficient condition for a local minimum

grad(f(x)) = 0, H(x) > 0.

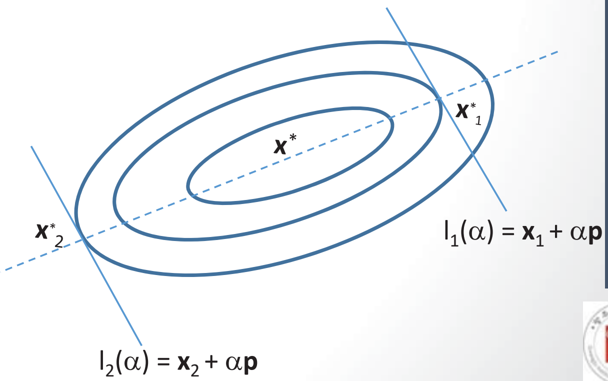
Multivariate Optimization:
Conjugate Gradient Method

- Conjugacy
  - A set of nonzero vectors $\{\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_L\}$ is conjugate with respect to symmetric positive definite matrix $\mathbf{A}$ if $\mathbf{p}_i^T \mathbf{A}\, \mathbf{p}_j = 0$, for all $i \neq j$.
  - Geometrical meaning of conjugacy : $f(\mathbf{x}) = ½\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x}$
    - When $\mathbf{x}^*_1$ and $\mathbf{x}^*_2$ are optimal points along two subspaces $S_1 = \{ \mathbf{x}_1 + \Sigma\, \alpha_i \mathbf{p}_i \mid \alpha_i \in R, i = 1, 2, \dots, L\}$ and $S_2 = \{ \mathbf{x}_2 + \Sigma\, \alpha_i \mathbf{p}_i \mid \alpha_i \in R, i = 1, 2, \dots, L\}$, respectively, then $(\mathbf{x}^*_1 - \mathbf{x}^*_2)$ are conjugate to $\{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_L\}$.

$$\left.\frac{\partial f(\mathbf{x}_1^* + \alpha_i \mathbf{p}_i)}{\partial \alpha_i}\right|_{\alpha_i = 0} = \nabla f(\mathbf{x}_1^*)^T \mathbf{p}_i = 0, \quad i = 1, 2, \cdots, L$$

$$\left.\frac{\partial f(\mathbf{x}_2^* + \alpha_i \mathbf{p}_i)}{\partial \alpha_i}\right|_{\alpha_i = 0} = \nabla f(\mathbf{x}_2^*)^T \mathbf{p}_i = 0, \quad i = 1, 2, \cdots, L$$

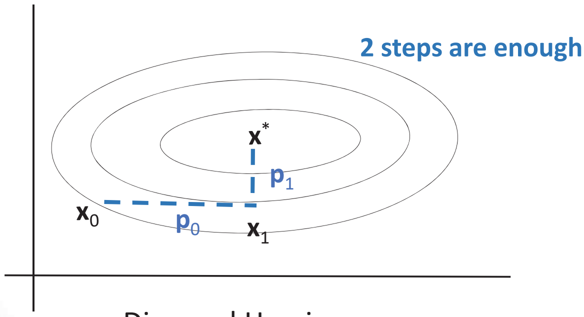
$$\begin{aligned} 0 &= \left(\nabla f(\mathbf{x}_1^*) - \nabla f(\mathbf{x}_2^*)\right)^T \mathbf{p}_i \\ &= (\mathbf{Ax}_1^* - \mathbf{b} - \mathbf{Ax}_2^* + \mathbf{b})^T \mathbf{p}_i \\ &= (\mathbf{x}_1^* - \mathbf{x}_2^*)^T \mathbf{Ap}_i, \quad i = 1, 2, \cdots, L \end{aligned}$$



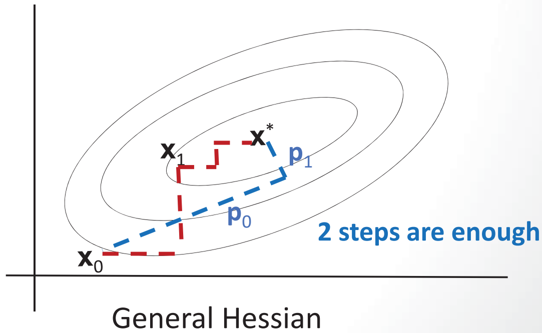
Multivariate Optimization: Conjugate Gradient Method

- Conjugate direction methods

For given a set of conjugate directions $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{n-1}\}$ with respect to a symmetric positive definite matrix **A** (n x n), the sequence $\{\mathbf{x}_k\}$ by setting $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$ converges to the minimum of the quadratic convex function ($f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}$) within at most n steps when α_k is given by exact search.



Diagonal Hessian



General Hessian

Transform Hessian into Diagonal.

Multivariate Optimization: Conjugate Gradient Method

- Consider convex quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}.$$

- Motivation

Present a new conjugate direction ($\mathbf{p}_k$) in terms of residue ($\mathbf{r}_k := \mathbf{A}\mathbf{x}_k - \mathbf{b}$) and the previous conjugate direction ($\mathbf{p}_{k-1}$) as follows:

$$\mathbf{p}_k = -\mathbf{r}_k + \beta_k \mathbf{p}_{k-1}$$

- Conjugate gradient method is generating conjugate direction for each iteration, so it is a special case of conjugate direction method.
- We note that $\mathbf{p}_k = -\mathbf{r}_k$ (case $\beta_k=0$) is the steepest decreasing direction.

Multivariate Optimization:
Conjugate Gradient Method

- Consider convex quadratic function

$$f(\mathbf{x}) = ½\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}.$$

  - How to generate conjugate directions?
    - Key idea : determine β_k in order that a new vector $\mathbf{p}_k = -\mathbf{r}_k + \beta_k\mathbf{p}_{k-1}$ is a conjugate with respect to $\mathbf{A}$.
    - So β_k is estimated by $\beta_k = \dfrac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}$.

Proof : By multiplying $\mathbf{p}_{k-1}{}^T\mathbf{A}$ into both sides of $\mathbf{p}_k = -\mathbf{r}_k + \beta_k\mathbf{p}_{k-1}$,

we get $\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{p}_k = -\,\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{r}_k + \beta_k\,\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{p}_{k-1}$. Since $\mathbf{p}_k$ and $\mathbf{p}_{k-1}$ are conjugate with respect of $\mathbf{A}$, $\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{p}_k = 0$.

Then $0 = -\,\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{r}_k + \beta_k\,\mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{p}_{k-1}$. Thus, $\beta_k = \mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{r}_k / \mathbf{p}_{k-1}{}^T\mathbf{A}\,\mathbf{p}_{k-1}$.

Multivariate Optimization:
Conjugate Gradient Method

- Standard CG Algorithm ($f(\mathbf{x}) = ½\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}$)
  - Given $\mathbf{x}_0$
  - Set k:=0, $\mathbf{r}_0 := \mathbf{A}\mathbf{x}_0 - \mathbf{b}$, $\mathbf{p}_0 := -\mathbf{r}_0$ (initial search direction is $-\nabla f(\mathbf{x}_0)$)
  - While $\mathbf{r}_k \neq \mathbf{0}$

$$\alpha_k := \frac{-r_k^T p_k}{p_k^T A p_k} \Rightarrow x_{k+1} := x_k + \alpha_k p_k$$

Determine step length (exact line search)

$$r_{k+1} := Ax_{k+1} - b \Rightarrow \beta_{k+1} := \frac{r_{k+1}^T A p_k}{p_k^T A p_k}$$

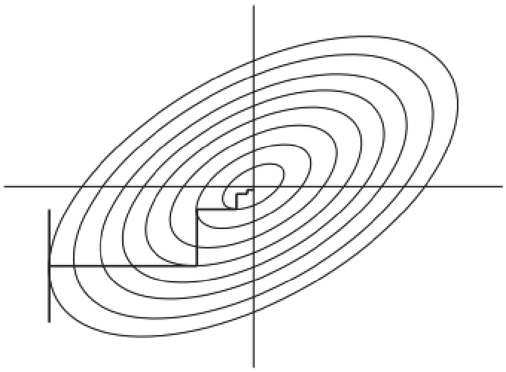
Compute residue

$$p_{k+1} := -r_{k+1} + \beta_{k+1} p_k \Rightarrow k := k+1$$

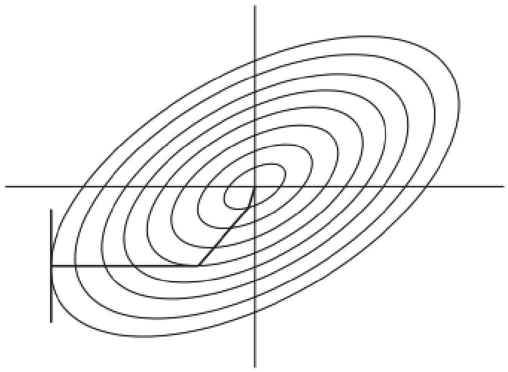
Search a new direction

(Exercise) Check $\boxed{\alpha_k = \dfrac{-r_k^T p_k}{p_k^T A p_k}}$

Multivariate Optimization:
Conjugate Gradient Method



Method of Steepest Descent



Conjugate Gradient Method

Multivariate Optimization:
Conjugate Gradient Method

- CG properties
  - Search directions are conjugate w.r.t matrix **A**.
  - Residue $\mathbf{r}_k$ and search direction $\mathbf{p}_i$ are orthogonal,

    that is, $\mathbf{r}_k^T\mathbf{p}_i = 0$ for i = 0, 1, …, k-1.
  - Residues $\mathbf{r}_i$ are mutually orthogonal,

    that is, $\mathbf{r}_k^T\mathbf{r}_i=0$ for i = 0, 1, …, k-1.
  - Identities (Please check!)

$$\mathbf{r}_{k+1}^T\mathbf{A}\mathbf{p}_k = \mathbf{r}_{k+1}^T\mathbf{r}_{k+1} / \alpha_k$$
$$\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k = \mathbf{r}_k^T\mathbf{r}_k / \alpha_k$$

Multivariate Optimization:
Conjugate Gradient Method

- Standard CG Algorithm
  - Given $\mathbf{x}_0$
  - Set $\mathbf{r}_0 := \mathbf{A}\mathbf{x}_0 - \mathbf{b}$, $\mathbf{p}_0 := -\mathbf{r}_0$, $k := 0$
  - While $\mathbf{r}_k \neq 0$

$$\alpha_k := \frac{-\mathbf{r}_k^T\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k} \Rightarrow \mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k\mathbf{p}_k$$

$$\mathbf{r}_{k+1} := \mathbf{A}\mathbf{x}_{k+1} - \mathbf{b} \Rightarrow \beta_{k+1} := \frac{\mathbf{r}_{k+1}^T\mathbf{A}\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k}$$

$$\mathbf{p}_{k+1} := -\mathbf{r}_{k+1} + \beta_{k+1}\mathbf{p}_k \Rightarrow k := k+1$$

- Practical CG Algorithm
  - Given $\mathbf{x}_0$
  - Set $\mathbf{r}_0 := \mathbf{A}\mathbf{x}_0 - \mathbf{b}$, $\mathbf{p}_0 := -\mathbf{r}_0$, $k := 0$
  - While $\mathbf{r}_k \neq 0$

$$\alpha_k := \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k} \Rightarrow \mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k\mathbf{p}_k$$

$$\mathbf{r}_{k+1} := \mathbf{r}_k + \alpha_k\mathbf{A}\mathbf{p}_k \Rightarrow \beta_{k+1} := \frac{\mathbf{r}_{k+1}^T\mathbf{r}_{k+1}}{\mathbf{r}_k^T\mathbf{r}_k}$$

$$\mathbf{p}_{k+1} := -\mathbf{r}_{k+1} + \beta_{k+1}\mathbf{p}_k \Rightarrow k := k+1$$

$$\mathbf{r}_{k+1}^T\mathbf{A}\mathbf{p}_k = \mathbf{r}_{k+1}^T\mathbf{r}_{k+1} / \alpha_k$$
$$\mathbf{p}_k^T\mathbf{A}\mathbf{p}_k = \mathbf{r}_k^T\mathbf{r}_k / \alpha_k$$

These identities are applied here.

Multivariate Optimization:
Conjugate Gradient Method

- Convergence of CG
  - It converges within N-iterations when **A** is a symmetric p·d matrix of size N x N.
- Convergence rate of CG
  - When **A** has eigenvalues $\lambda_1 \le \lambda_2 \le \ldots \le \lambda_N$,

$$\left\|\mathbf{x}_k - \mathbf{x}^*\right\|_A \le 2\left(\frac{\sqrt{\kappa(\mathbf{A})}-1}{\sqrt{\kappa(\mathbf{A})}+1}\right)^k \left\|\mathbf{x}_0 - \mathbf{x}^*\right\|_A,\ \kappa(\mathbf{A}) = \frac{\lambda_N}{\lambda_1}$$

  - CG convergence depends on clustering of eigenvalues of **A**.
    - When κ(**A**) is big enough, i.e. eigenvalues are widely scattered,
      - It converges slowly.
    - When κ(**A**) is around 1, i.e. eigenvalues are well clustered,
      - It converges fast.

Multivariate Optimization: Conjugate Gradient Method

- How to speed-up CG when CG convergence is slow
  - One idea
    - To use preconditioner 'symmetric positive definte matrix **M**'
    - Transform original problem into new problem

      $\mathbf{A}\,\mathbf{x} = \mathbf{b} \;\Rightarrow\; (\mathbf{M}^{-1}\mathbf{A})\,\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$ or

      $\mathbf{A}\,\mathbf{x} = \mathbf{b} \;\Rightarrow\; (\mathbf{M}^{-1}\mathbf{A}\,\mathbf{M}^{-T})\,\mathbf{x}^{\wedge} = \mathbf{M}^{-1}\mathbf{b}$ and $\mathbf{x}^{\wedge} = \mathbf{M}^{T}\mathbf{x}$
    - In order for $\kappa(\mathbf{M}^{-1}\mathbf{A})$ or $\kappa(\mathbf{M}^{-1}\mathbf{A}\,\mathbf{M}^{-T})$ to be close to 1, **M** can be chosen properly, then CG can be faster than before.