Optimization Methods (CS1.404), Spring 2024
Lecture 16

Naresh Manwani

Machine Learning Lab, IIIT-H

March 11th, 2024

Conjugate Gradient Algorithm for Non-Quadratic Problems

- To minimize a non-quadratic function, we first find a quadratic approximation at $\mathbf{x}_k$ using Taylor series and minimize it using conjugate descent to find $\mathbf{x}_{k+1}$.
- We replace H by Hessian at that iteration.
- The conjugate descent algorithm requires computation of Hessian at each iteration which makes it computationally expensive.
- An efficient implementation of conjugate descent eliminates the evaluation of Hessian at each step.
- Note that in conjugate descent algorithm, Hessian appears in the expression of α_k and β_k.
- Because $\alpha_k = \arg\min_{\alpha \geq 0} f(\mathbf{x}_k + \alpha \mathbf{d}_k)$, closed form formula for α_k can be replaced by numerical line search method.
- To eliminate Hessian from the formula of β_k, there are three possible ways.

Hestenes-Stiefel Formula

- Recall that $\beta_k = \frac{\mathbf{g}_{k+1}^T H \mathbf{d}_k}{\mathbf{d}_k^T H \mathbf{d}_k}$.
- Here, we replace $H\mathbf{d}_k$ by the term $\frac{\mathbf{g}_{k+1} - \mathbf{g}_k}{\alpha_k}$.
  $\left(\frac{\mathbf{g}_{k+1} - \mathbf{g}_k}{\alpha_k} = \frac{H\mathbf{x}_{k+1} + \mathbf{c} - H\mathbf{x}_k - \mathbf{c}}{\alpha_k} = \frac{H(\mathbf{x}_{k+1} - \mathbf{x}_k)}{\alpha_k} = H\mathbf{d}_k\right)$
- Using this in the β_k formula, we get $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{d}_k^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}$.
- **For quadratic functions, $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{d}_k^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}$ is same as $\beta_k = \frac{\mathbf{g}_{k+1}^T H \mathbf{d}_k}{\mathbf{d}_k^T H \mathbf{d}_k}$.**

Hestenes-Stiefel Approach

Initialize: The starting point $\mathbf{x}_0$ and the tolerance parameter $\epsilon > 0$,
Set $k = 0$
Assign $\mathbf{d}_0 = -\mathbf{g}_0$
while $\|\mathbf{g}_k\| > \epsilon$ **do**
    $\alpha_k = \arg\min_{\alpha > 0} f(\mathbf{x}_k + \alpha \mathbf{d}_k)$
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
    Compute $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$
    if $(k < n - 1)$ **then**
        $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{d}_k^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}$
        $\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k$
        $k = k + 1$
    else
        $\mathbf{x}_0 = \mathbf{x}_{k+1}$
        $\mathbf{d}_0 = -\mathbf{g}_{k+1}$
        $k = 0$
    end if
end while
Output: $\mathbf{x}^* = \mathbf{x}_k$, a stationary point of f.

- Starting from Hestenes-Stiefel formula, we multiply out the denominator to get $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{d}_k^T\mathbf{g}_{k+1}-\mathbf{d}_k^T\mathbf{g}_k}$.
- But, we know that $\mathbf{d}_k^T\mathbf{g}_{k+1} = 0$.
- Also, since $\mathbf{d}_k = -\mathbf{g}_k + \beta_{k-1}\mathbf{d}_{k=1}$, we get

$$\mathbf{g}_k^T\mathbf{d}_k = -\mathbf{g}_k^T\mathbf{g}_k + \beta_{k-1}\mathbf{g}_k^T\mathbf{d}_{k-1} = -\mathbf{g}_k^T\mathbf{g}_k$$

- Thus, we get $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{g}_k^T\mathbf{g}_k}$.
- This expression for β_k is called Polak-Ribiere Formula.
- **For quadratic functions,** $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{g}_k^T\mathbf{g}_k}$ **is same as** $\beta_k = \frac{\mathbf{g}_{k+1}^T H\mathbf{d}_k}{\mathbf{d}_k^T H\mathbf{d}_k}$**.**

Polak-Ribiere Approach

Initialize: The starting point $\mathbf{x}_0$ and the tolerance parameter $\epsilon > 0$,
Set $k = 0$
Assign $\mathbf{d}_0 = -\mathbf{g}_0$
while $\|\mathbf{g}_k\| > \epsilon$ **do**
    $\alpha_k = \arg\min_{\alpha > 0} \; f(\mathbf{x}_k + \alpha \mathbf{d}_k)$
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
    Compute $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$
    if $(k < n - 1)$ **then**
        $\beta_k = \frac{\mathbf{g}_{k+1}^T(\mathbf{g}_{k+1} - \mathbf{g}_k)}{\mathbf{g}_k^T \mathbf{g}_k}$
        $\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k$
        $k = k + 1$
    else
        $\mathbf{x}_0 = \mathbf{x}_{k+1}$
        $\mathbf{d}_0 = -\mathbf{g}_{k+1}$
        $k = 0$
    end if
end while
Output: $\mathbf{x}^* = \mathbf{x}_k$, a stationary point of f.

Fletcher Reeves Formula

- Starting with the Polak-Ribiere Formula, we get $\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1} - \mathbf{g}_{k+1}^T \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{g}_k}$.
- We know that $\mathbf{d}_k = -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1}$. Thus, $\mathbf{g}_{k+1}^T \mathbf{d}_k = -\mathbf{g}_{k+1}^T \mathbf{g}_k + \beta_k \mathbf{g}_{k+1}^T \mathbf{d}_{k-1}$.
- But, we know that $\mathbf{g}_{k+1}^T \mathbf{d}_k = \mathbf{g}_{k+1}^T \mathbf{d}_{k-1} = 0$.
- Thus, $\mathbf{g}_{k+1}^T \mathbf{g}_k = 0$.
- This leads to $\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k}$.
- This is called Fletcher Reeves formula.
- **For quadratic functions,** $\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k}$ **is same as** $\beta_k = \frac{\mathbf{g}_{k+1}^T H \mathbf{d}_k}{\mathbf{d}_k^T H \mathbf{d}_k}$**.**

Initialize: The starting point $\mathbf{x}_0$ and the tolerance parameter $\epsilon > 0$, Set $k = 0$
Assign $\mathbf{d}_0 = -\mathbf{g}_0$
while $\|\mathbf{g}_k\| > \epsilon$ **do**
    $\alpha_k = \arg\min_{\alpha > 0} \; f(\mathbf{x}_k + \alpha \mathbf{d}_k)$
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
    Compute $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$
    if $(k < n - 1)$ **then**
        $\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k}$
        $\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k$
        $k = k + 1$
    else
        $\mathbf{x}_0 = \mathbf{x}_{k+1}$
        $\mathbf{d}_0 = -\mathbf{g}_{k+1}$
        $k = 0$
    end if
end while
Output: $\mathbf{x}^* = \mathbf{x}_k$, a stationary point of f.

- Conjugate direction methods can be regarded as being between the method of steepest descent (first-order method that uses gradient) and Newton's method (second-order method that uses Hessian as well).
  - Steepest descent is slow.
  - Newton method is fast, but we need to calculate the inverse of the Hessian matrix.
  - **Conjugate gradient uses gradient only and faster than steepest descent.**
- Conjugate gradient method attempts to accelerate gradient descent by building in momentum.
  - Recall $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
  - Using $\mathbf{d}_k = -\mathbf{g}_k + \beta_{k-1}\mathbf{d}_{k-1}$, we get

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{d}_k \\ &= \mathbf{x}_k - \alpha_k \mathbf{g}_k + \alpha_k \beta_{k-1}\mathbf{d}_{k-1}\end{aligned}$$

  - Using $\mathbf{d}_{k-1} = \frac{\mathbf{x}_k - \mathbf{x}_{k-1}}{\alpha_{k-1}}$, we get

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k + \underbrace{\frac{\alpha_k \beta_{k-1}}{\alpha_{k-1}}(\mathbf{x}_k - \mathbf{x}_{k-1})}_{\text{momentum term}}$$

- **Newton Method:** Given a function $f \in \mathbb{C}^2(\mathbb{R}^n)$, Newton method finds the descent direction by solving $H_k \mathbf{d}_k = -\mathbf{g}_k$, where $H_k = \nabla^2 f(\mathbf{x}_k)$ and $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$.
- **Quasi Newton Method:** Given a function $f \in \mathbb{C}^1(\mathbb{R}^n)$, quasi-Newton method finds descent direction as $\mathbf{d}_k = -B_k \mathbf{g}_k$, where B_k is a positive definite matrix.
  - B_k^{-1} is either H_k or its approximation.
  - $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k^{QN} = \mathbf{x}_k - \alpha_k B_k \mathbf{g}_k$
  - Given $\mathbf{x}_k$, $\mathbf{x}_{k+1}$, $\mathbf{g}_k$, $\mathbf{g}_{k+1}$ and B_k, how to get symmetric positive definite B_{k+1}?
  - Are there any conditions that B_{k+1} needs to satisfy?

- We find quadratic approximation of f at $\mathbf{x}_{k+1}$ using B_{K+1} as follows. $f_{k+1}(\mathbf{x}) = f(\mathbf{x}_{k+1}) + \mathbf{g}_{k+1}^T(\mathbf{x} - \mathbf{x}_{k+1}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_{k+1})^T B_{k+1}^{-1}(\mathbf{x} - \mathbf{x}_{k+1})$
- We require that $\nabla f_{k+1}(\mathbf{x}_k) = \mathbf{g}_k$ and $\nabla f_{k+1}(\mathbf{x}_{k+1}) = \mathbf{g}_{k+1}$.
- Therefore, using the first condition, we require $\nabla f_{k+1}(\mathbf{x}_k) = \mathbf{g}_k = \mathbf{g}_{k+1} + B_{k+1}^{-1}(\mathbf{x}_k - \mathbf{x}_{k-1})$.
- Letting $\gamma_k = \mathbf{g}_{k+1} - \mathbf{g}_k$ and $\boldsymbol{\delta}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, we get $B_{k+1}\gamma_k = \boldsymbol{\delta}_k$.
- This condition is also called **Quasi-Newton condition**.
- B_{k+1} should be positive definite. Thus, $\gamma_k^T B_{k+1}\gamma_k = \gamma_k^T\boldsymbol{\delta}_k > 0$.
  - From Wolfe line search condition

$$\mathbf{g}_{k+1}^T\mathbf{d}_k \geq c_2\mathbf{g}_k^T\mathbf{d}_k, \quad \text{where } c_2 \in (0,1)$$
$$\Rightarrow (\mathbf{g}_{k+1} - \mathbf{g}_k)^T\mathbf{d}_k \geq (c_2 - 1)\mathbf{g}_k^T\mathbf{d}_k$$

  We know that $c_2 - 1 < 0$ and $\mathbf{g}_k^T\mathbf{d}_k = -\mathbf{g}_k^T B_k\mathbf{g}_k < 0$ as B_k is positive definite matrix. Thus, we get

$$(\mathbf{g}_{k+1} - \mathbf{g}_k)^T\mathbf{d}_k \geq (c_2 - 1)\mathbf{g}_k^T\mathbf{d}_k > 0 \Rightarrow \gamma_k^T\mathbf{d}_k > 0$$
$$\Rightarrow \gamma_k^T\boldsymbol{\delta}_k > 0, \quad \text{using } \mathbf{d}_k = \frac{1}{\alpha_k}(\mathbf{x}_{k+1} - \mathbf{x}_k) = \frac{1}{\alpha_k}\boldsymbol{\delta}_k$$

  - When Wolfe condition is satisfied in a line search, $\exists B_{k+1}$ which satisfies Quasi-Newton condition.

Symmetric Rank One Correction

- Here, we want to update B_k to B_{k+1} by adding a rank one matrix $a_k \mathbf{z}_k \mathbf{z}_k^T$, where $a_k \in \mathbb{R}(a \neq 0)$ and $\mathbf{z}_k \in \mathbb{R}^n(\mathbf{z} \neq \mathbf{0})$. Thus,

$$B_{k+1} = B_k + a_k \mathbf{z}_k \mathbf{z}_k^T$$

- Now, we choose a_k and $\mathbf{z}_k$ such that B_{k+1} satisfies Quasi-Newton condition. Thus, we want

$$\begin{aligned} B_{k+1}\boldsymbol{\gamma}_k &= \boldsymbol{\delta}_k \\ \Rightarrow (B_k + a_k \mathbf{z}_k \mathbf{z}_k^T)\boldsymbol{\gamma}_k &= \boldsymbol{\delta}_k \\ \Rightarrow a_k \mathbf{z}_k \mathbf{z}_k^T \boldsymbol{\gamma}_k &= \boldsymbol{\delta}_k - B_k \boldsymbol{\gamma}_k \end{aligned}$$

- Let $\mathbf{z}_k = \boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k$. Therefore, $a_k \mathbf{z}_k^T \boldsymbol{\gamma}_k = 1$.
- That gives $\alpha_k = \frac{1}{(\boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k)^T\boldsymbol{\gamma}_k}$.

Thus, using $\mathbf{x}_k$, $\mathbf{x}_{k+1}$, $\mathbf{g}_{k+1}$ and $\mathbf{g}_k$, we get

$$B_{k+1}^{SR1} = B_k + \frac{(\boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k)(\boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k)^T)}{(\boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k)^T\boldsymbol{\gamma}_k}$$

Quasi-Newton Method (Rank One Correction)

Initialize: The starting point $\mathbf{x}_0$, Symmetric positive definite matrix B_0 and the tolerance parameter $\epsilon > 0$, Set $k = 0$

while $\|\mathbf{g}_k\| > \epsilon$ **do**

  $\mathbf{d}_k = -B_k \mathbf{g}_k$

  Find α_k along $\mathbf{d}_k$ such that

  - $f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{x}_k)$
  - α_k satisfies Armijo-Wolfe condition

  $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$

  Find B_{k+1} as

$$B_{k+1} = B_k + \frac{(\boldsymbol{\delta}_k - B_k \boldsymbol{\gamma}_k)(\boldsymbol{\delta}_k - B_k \boldsymbol{\gamma}_k)^T)}{(\boldsymbol{\delta}_k - B_k \boldsymbol{\gamma}_k)^T \boldsymbol{\gamma}_k}$$

  $k = k + 1$

end while

Output: $\mathbf{x}^* = \mathbf{x}_k$, a stationary point of f.

Example: Rank One Correction

- Consider the problem min $f(x, y) = 4x^2 + y^2 - 2xy$
- For this problem, $\mathbf{x}^* = [0 \;\; 0]^T$, $H = \begin{bmatrix} 8 & -2 \\ -2 & 2 \end{bmatrix}$,

  $H^{-1} = \begin{bmatrix} 0.1667 & 0.1667 \\ 0.1667 & 0.6667 \end{bmatrix}$
- We run **rank one correction** approach with $\mathbf{x}_0 = [-2 \;\; -2]^T$ and B_0 as identity matrix.
- We see that the algorithm converges in 3 steps. Below are the updates in each step.

k	x_k	y_k	B_k	$\|\mathbf{g}_k\|$
0	-2	-2	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	12.0
1	0	-2	$\begin{bmatrix} 0.1833 & 0.2333 \\ 0.2333 & 0.9333 \end{bmatrix}$	5.65
2	0.1538	0.1536	$\begin{bmatrix} 0.1667 & 0.1667 \\ 0.1667 & 0.6667 \end{bmatrix}$	0.92
3	0	0	H^{-1}	0

Quasi-Newton Algorithm Applied on Quadratic Functions

- Consider the problem $\min_{\mathbf{x}} \ \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \mathbf{c}^T\mathbf{x}$, where H is a symmetric positive definite matrix.
- To solve this problem using rank one correction method, at every iteration k
  - B_{k+1} is symmetric positive definite.
  - B_{k+1} is obtained from $\mathbf{x}_k$, $\mathbf{x}_{k+1}$, $\mathbf{g}_{k+1}$ and $\mathbf{g}_k$.
  - B_{k+1} satisfies Quasi-Newton condition, $B_{k+1}\boldsymbol{\gamma}_k = \boldsymbol{\delta}_k$
- Note that $\mathbf{g}_{k+1} - \mathbf{g}_k = H\mathbf{x}_{k+1} + \mathbf{c} - H\mathbf{x}_k - \mathbf{c} = H(\mathbf{x}_{k+1} - \mathbf{x}_k)$. Which means, $\boldsymbol{\gamma}_k = H\boldsymbol{\delta}_k$.

Lemma: Hereditary Property

SR1 correction approach applied to quadratic function with positive definite Hessian H, we have

$$B_{k+1}\boldsymbol{\gamma}_i = \boldsymbol{\delta}_i,\ 0 \leq i \leq k.$$

When f is quadratic, the hereditary property is satisfied by SR1 regardless of how the line search is performed.

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY HYDERABAD

Convergence of SR1 Applied on Quadratic Functions

Theorem1: For Quadratic Functions

Consider SR1 quasi-Newton algorithm applied to a quadratic function with positive definite Hessian H. Then, for any starting point $\mathbf{x}_0$ and any symmetric starting matrix B_0, the sequence of iterates $\mathbf{x}_k$ generated by SR1 converges to the minimizer in n-steps, provided $(\boldsymbol{\delta}_k - B_k\boldsymbol{\gamma}_k)^T\boldsymbol{\gamma}_k \neq 0, \forall k$. Moreover, if n-steps are performed and $\boldsymbol{\delta}_0, \boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_{n-1}$ are linearly independent, then $B_n = H^{-1}$.