Notice

- Homework assignments Week 14:
  - 1 multiple choice question (2%) and 1 programming work (6%).
  - Due date: 21 p.m. on January 1, 2022.

Outline

- **Lecture 9: Proximal operators**
  - The basic idea
  - The Moreau decomposition
  - Examples of proximal operators
- **Lecture 10: Proximal gradient algorithms**
  - A basic form of the proximal gradient algorithm
  - The fast proximal gradient algorithm
  - Application: graphical lasso estimation
- **Solutions to Week 13 quizzes and Week 12 homework assignments**
- **Lecture 11: Stochastic methods**
  - The stochastic gradient descent algorithm
  - Adaptive methods

Computation in Data Science: Week 14
Lecture 9

Tso-Jung Yen

Institute of Statistical Science
Academia Sinica

tjyen@stat.sinica.edu.tw

Data Science Degree Program

National Taiwan University

December 22, 2021

Proximal Operators

- **Motivation:**
  - Consider the lasso estimation problem:

$$\min_{\boldsymbol{\theta}} \qquad l(\boldsymbol{\theta}) + \alpha||\boldsymbol{\theta}||_1, \tag{1}$$

    where $\boldsymbol{\theta}$ is a p-dimensional vector, and loss function l is assumed to be a differentiable function.
  - If $\boldsymbol{\theta}^*$ is a solution to (1), then we must have

$$\nabla l(\boldsymbol{\theta}^*) + \alpha \mathbf{u}^* = \mathbf{0}, \tag{2}$$

    where $\mathbf{u}^* \in \partial||\boldsymbol{\theta}^*||_1$ is a subgradient of $||.||_1$ evaluated at $\boldsymbol{\theta}^*$.
  - Can we build an iterative scheme to solve (2)?

Proximal Operators

- **Motivation (contd):**

  - By applying the Taylor series expansion to the loss function l around point $\boldsymbol{\theta}'$ one has

$$l(\boldsymbol{\theta}) \quad \approx \quad \left\{ l(\boldsymbol{\theta}') + \nabla l(\boldsymbol{\theta}')^T(\boldsymbol{\theta} - \boldsymbol{\theta}') + \frac{1}{2c}||\boldsymbol{\theta} - \boldsymbol{\theta}'||_2^2 \right\}.$$

  - Following a similar way of building **gradient descent algorithms**, we define

$$\begin{aligned} \boldsymbol{\theta}^{r+1} \quad &= \quad \arg\min_{\boldsymbol{\theta}} \left\{ l(\boldsymbol{\theta}^r) + \nabla l(\boldsymbol{\theta}^r)^T(\boldsymbol{\theta} - \boldsymbol{\theta}^r) + \frac{1}{2c_r}||\boldsymbol{\theta} - \boldsymbol{\theta}^r||_2^2 + \alpha||\boldsymbol{\theta}||_1 \right\} \\ &= \quad \boldsymbol{\theta}^r - c_r \nabla l(\boldsymbol{\theta}^r) - c_r \alpha \mathbf{u}^{r+1} \\ &= \quad \arg\min_{\boldsymbol{\theta}} \left\{ \alpha c_r ||\boldsymbol{\theta}||_1 + \frac{1}{2} \left|\left| \boldsymbol{\theta} - [\boldsymbol{\theta}^r - c_r \nabla l(\boldsymbol{\theta}^r)] \right|\right|_2^2 \right\}, \end{aligned} \tag{3}$$

    to find a solution to (1).

    Here $\mathbf{u}^{r+1} \in \partial||\boldsymbol{\theta}^{r+1}||_1$ is a **subgradient** of $||.||_1$ evaluated at $\boldsymbol{\theta}^{r+1}$.

  - The iterative scheme (3) is an example of the **proximal gradient algorithm**, which exploits the idea of **proximal operators** to solve the final line of (3).

Proximal Operators

- **Definition of the proximal operator:**
  - The proximal operator of a function g of $\mathbf{x}$ is defined as

$$\mathrm{prox}_g(\mathbf{x}) = \arg\min_{\boldsymbol{\theta}} \left\{ g(\boldsymbol{\theta}) + \frac{1}{2}||\boldsymbol{\theta} - \mathbf{x}||_2^2 \right\}. \tag{4}$$

  - To express the final line of (3) in terms of (4), we have $g(\boldsymbol{\theta}) = \alpha c_r ||\boldsymbol{\theta}||_1$ and $\mathbf{x} = \boldsymbol{\theta}^r - c_r \nabla l(\boldsymbol{\theta}^r)$.
  - *Remark:* Here $\boldsymbol{\theta}$, $\mathrm{prox}_g(\mathbf{x})$ and $\mathbf{x}$ should have the *same* shape.

Proximal Operators

- **Projection via the proximal operator:**
  - Define

$$\iota\{\boldsymbol{\theta} \in \mathcal{C}\} = \begin{cases} 0 & \text{if} \quad \boldsymbol{\theta} \in \mathcal{C} \\ \infty & \text{otherwise} \end{cases} .$$

  - For $g(\boldsymbol{\theta}) = \iota\{\boldsymbol{\theta} \in \mathcal{C}\}$ define

$$\begin{aligned} \text{prox}_{\mathcal{C}}(\mathbf{x}) &= \arg\min_{\boldsymbol{\theta}} \left\{ \iota\{\boldsymbol{\theta} \in \mathcal{C}\} + \frac{1}{2}||\boldsymbol{\theta} - \mathbf{x}||_2^2 \right\} \\ &= \arg\min_{\boldsymbol{\theta} \in \mathcal{C}} \left\{ \frac{1}{2}||\boldsymbol{\theta} - \mathbf{x}||_2^2 \right\}. \end{aligned} \tag{5}$$

    (5) is the **projection** of $\mathbf{x}$ onto $\mathcal{C}$.

    The proximal operator $\text{prox}_{\mathcal{C}}(\mathbf{x})$ is the point in $\mathcal{C}$ that is the closest to the object $\mathbf{x}$ (measured in Euclidean distance).

Proximal Operators

- **The Moreau decomposition:**
  - Assume g is a convex function.
  - An important fact about the proximal operator of g is that, the object $\mathbf{x}$ can be decomposed as sum of the proximal operators of the convex function and its conjugate:

$$\mathbf{x} = \text{prox}_g(\mathbf{x}) + \text{prox}_{g^*}(\mathbf{x}), \tag{6}$$

where g^* is defined by

$$g^*(\boldsymbol{\phi}) = \max_{\boldsymbol{\theta}} \{\boldsymbol{\theta}^T \boldsymbol{\phi} - g(\boldsymbol{\theta})\}.$$

(6) is called the **Moreau decomposition** of $\mathbf{x}$.

Proximal Operators

- **The first example (the proximal operator of the ridge penalty function):**
  - Consider the ridge penalty function

$$g(\boldsymbol{\theta}) = \frac{\alpha}{2}||\mathbf{W}\boldsymbol{\theta} + \mathbf{c}||_2^2,$$

    where $\boldsymbol{\theta}$ is a p-dimensional vector, $\mathbf{W}$ is a $m \times p$ matrix and $\alpha \geq 0$ is a constant.

  - The proximal operator of $g(\boldsymbol{\theta})$ is the solution to the following optimization problem:

$$\min_{\boldsymbol{\theta}} \qquad \frac{1}{2}||\boldsymbol{\theta} - \mathbf{x}||_2^2 + \frac{\alpha}{2}||\mathbf{W}\boldsymbol{\theta} + \mathbf{c}||_2^2. \tag{7}$$

  - Let $\boldsymbol{\theta}^*$ denote the solution to (7). Then $\boldsymbol{\theta}^*$ must satisfy the following equations:

$$\boldsymbol{\theta}^* - \mathbf{x} + \alpha\mathbf{W}^T(\mathbf{W}\boldsymbol{\theta}^* + \mathbf{c}) = \mathbf{0},$$

    which implies

$$\text{prox}_g(\mathbf{x}) = \boldsymbol{\theta}^* = (\alpha\mathbf{W}^T\mathbf{W} + \mathbf{I}_{p\times p})^{-1}(\mathbf{x} - \alpha\mathbf{W}^T\mathbf{c}).$$

Examples

- **Proximal operators of vector norms:**
  - Consider the following proximal operator of the penalty function $\alpha||\boldsymbol{\theta}||$ on vector $\mathbf{x} \in \mathbb{R}^p$:

$$\text{prox}_{\alpha||.||}(\mathbf{x}) = \arg\min_{\boldsymbol{\theta}} \left\{ \alpha||\boldsymbol{\theta}|| + \frac{1}{2}||\boldsymbol{\theta} - \mathbf{x}||_2^2 \right\},$$

    where $\alpha \geq 0$ is a constant. We assume $\boldsymbol{\theta} \in \mathbb{R}^p$.

  - Since the norm $||\boldsymbol{\theta}||$ is convex, if $\boldsymbol{\theta}^*$ is a solution to the above optimization problem, we must have

$$\mathbf{0} \in \alpha\partial||\boldsymbol{\theta}^*|| + (\boldsymbol{\theta}^* - \mathbf{x}),$$

    which implies

$$\text{prox}_{\alpha||.||}(\mathbf{x}) = \boldsymbol{\theta}^* = \mathbf{x} - \alpha\mathbf{u}^* \qquad \text{where } \mathbf{u}^* \in \partial||\boldsymbol{\theta}^*||. \tag{8}$$

  - A key point is to find a pair $(\boldsymbol{\theta}^*, \mathbf{u}^*)$ that simultaneously satisfy the following two conditions:

$$\begin{aligned} \boldsymbol{\theta}^* - \mathbf{x} + \alpha\mathbf{u}^* &= 0, \\ \mathbf{u}^* &\in \partial||\boldsymbol{\theta}^*||. \end{aligned} \tag{9}$$

Examples

- **Proximal operators of vector norms (contd):**
  - **Example 1 (l_1-norm):** For l_1-norm penalty function $\alpha||\boldsymbol{\theta}||_1$, remember that

    $$\partial||\boldsymbol{\theta}||_1 = \{\mathbf{u} : ||\mathbf{u}||_\infty \leq 1 \text{ and } \mathbf{u}^T\boldsymbol{\theta} = ||\boldsymbol{\theta}||_1\},$$

    which implies that

    $$(\mathbf{u})_j = \begin{cases} 1 & \text{if } (\boldsymbol{\theta})_j > 0 \\ -1 & \text{if } (\boldsymbol{\theta})_j < 0 \\ \text{any point} \in [-1, 1] & \text{otherwise} \end{cases} . \tag{10}$$

  - We consider to find pair $(\boldsymbol{\theta}^*, \mathbf{u}^*)$ that simultaneously satisfy the conditions (9).

Examples

- **Proximal operators of vector norms (contd):**
  - **Example 1 (contd):**
    - **Case 1 ($(\mathbf{x})_j > \alpha$):** Because $(\mathbf{u}^*)_j \in [-1, 1]$, we always have $(\boldsymbol{\theta}^*)_j = (\mathbf{x})_j - \alpha(\mathbf{u}^*)_j > 0$. In this case, the pair

      $$(\mathbf{u}^*)_j = 1 \text{ and } (\boldsymbol{\theta}^*)_j = (\mathbf{x})_j - \alpha > 0$$

      satisfy the conditions (9).
    - **Case 2 ($(\mathbf{x})_j < -\alpha$):** Because $(\mathbf{u}^*)_j \in [-1, 1]$, we always have $(\boldsymbol{\theta}^*)_j = (\mathbf{x})_j - \alpha(\mathbf{u}^*)_j < 0$. In this case, the pair

      $$(\mathbf{u}^*)_j = -1 \text{ and } (\boldsymbol{\theta}^*)_j = (\mathbf{x})_j + \alpha < 0$$

      satisfy the conditions (9).
    - **Case 3 ($-\alpha \leq (\mathbf{x})_j \leq \alpha$):** In this case we can express $(\mathbf{x})_j = \alpha v$ with some $v \in [-1, 1]$. To satisfy (9) we must choose

      $$(\mathbf{u}^*)_j = v \text{ and } \boldsymbol{\theta}^* = (\mathbf{x})_j - \alpha(\mathbf{u}^*)_j = \alpha v - \alpha v = 0.$$

Examples

- **Proximal operators of vector norms (contd):**
  - **Example 1 (contd):** The results shown above imply that

$$[\mathrm{prox}_{\alpha||.||_1}(\mathbf{x})]_j = \begin{cases} (\mathbf{x})_j - \alpha & \text{if } (\mathbf{x})_j > \alpha \\ (\mathbf{x})_j + \alpha & \text{if } (\mathbf{x})_j < -\alpha \\ 0 & \text{if } -\alpha \le (\mathbf{x})_j \le \alpha \end{cases}, \tag{11}$$

which further implies

$$[\mathrm{prox}_{\alpha||.||_1}(\mathbf{x})]_j = \mathrm{sign}[(\mathbf{x})_j](|(\mathbf{x})_j| - \alpha)_+,$$

i.e. each element of the proximal operator of the l_1-norm is a soft thresholding function.

  - For simplicity we define

$$[\mathrm{ST}_\alpha(\mathbf{x})]_j = \mathrm{sign}[(\mathbf{x})_j](|(\mathbf{x})_j| - \alpha)_+.$$

Examples

- **Proximal operators of vector norms (contd):**
  - **Example 2 (l_2-norm):** For l_2-norm penalty function $\alpha||\boldsymbol{\theta}||_2$, remember that

$$\partial||\boldsymbol{\theta}||_2 = \{\mathbf{u} : ||\mathbf{u}||_2 \leq 1 \text{ and } \mathbf{u}^T\boldsymbol{\theta} = ||\boldsymbol{\theta}||_2\},$$

    which implies that the subgradient $\mathbf{u}$ can be expressed as

$$\mathbf{u} = \begin{cases} \boldsymbol{\theta}/||\boldsymbol{\theta}||_2 & \text{if } \boldsymbol{\theta} \neq \mathbf{0} \\ \text{any vector } \in \mathcal{B}_2(1) = \{\mathbf{v} : ||\mathbf{v}||_2 \leq 1\} & \text{if } \boldsymbol{\theta} = \mathbf{0} \end{cases}.$$

  - We consider two cases $||\mathbf{x}||_2 > \alpha$ and $||\mathbf{x}||_2 \leq \alpha$.

Examples

- **Proximal operators of vector norms (contd):**
  - **Example 2 (contd):**
    - **Case 1 ($||\mathbf{x}||_2 > \alpha$):** In this case we have $||\mathbf{x}||_2 - \alpha||\mathbf{u}^*||_2 \neq 0$, which implies $\boldsymbol{\theta}^* = \mathbf{x} - \alpha\mathbf{u}^* \neq \mathbf{0}$. The pair

      $$\mathbf{u}^* = \frac{\boldsymbol{\theta}^*}{||\boldsymbol{\theta}^*||_2} \text{ and } \boldsymbol{\theta}^* = \mathbf{x} - \alpha\mathbf{u}^* \Rightarrow \boldsymbol{\theta}^* = (||\mathbf{x}||_2 - \alpha)\frac{\mathbf{x}}{||\mathbf{x}||_2} \neq \mathbf{0} \quad (12)$$

      satisfy the conditions (9).
    - **Case 2 ($||\mathbf{x}||_2 \leq \alpha$):** In this case we may express $\mathbf{x} = \alpha\mathbf{v}$ with some $\mathbf{v} \in \mathcal{B}_2(1)$. To satisfy conditions (9) we may choose the pair

      $$\mathbf{u}^* = \mathbf{v} \text{ and } \boldsymbol{\theta}^* = \mathbf{x} - \alpha\mathbf{u}^* = \alpha\mathbf{v} - \alpha\mathbf{v} = \mathbf{0}.$$

  - *Remark:* Note that if we still choose $\boldsymbol{\theta}^* \neq \mathbf{0}$ in **Case 2**, then $\boldsymbol{\theta}^*$ will have different signs from $\mathbf{x}$ (see eq. (12)), which is no good for minimizing the objective function.
  - In summary, we have

    $$\text{prox}_{\alpha||.||_2}(\mathbf{x}) \quad = \quad (||\mathbf{x}||_2 - \alpha)_+ \frac{\mathbf{x}}{||\mathbf{x}||_2}.$$

Examples

- **Quiz:**

1. Assume $\theta > 0$ and $\alpha > 0$. Consider the function:

$$g(\theta) = -\alpha \log \theta.$$

The proximal operator of $g(\theta)$ on scalar $x \in \mathbb{R}$ is defined as

$$\mathsf{prox}_g(x) = \arg\min_{\theta} \left\{ g(\theta) + \frac{1}{2}(\theta - x)^2 \right\}.$$

Which of the following statements are *true*?

a. We have

$$\mathsf{prox}_g(x) = \frac{\alpha}{|x|}.$$

b. We have

$$\mathsf{prox}_g(x) = \frac{x + \sqrt{x^2 + 4\alpha}}{2}.$$

c. We have

$$\mathsf{prox}_g(x) = \exp\left(- \frac{x}{\alpha} \right).$$

Examples

- **Proximal operators on functions of matrix-valued variables:**
  - Let $\mathbf{\Theta}$ be an $p \times p$ matrix. Let

    $$\boldsymbol{\lambda}(\mathbf{\Theta}) = (\lambda_1(\mathbf{\Theta}), \lambda_2(\mathbf{\Theta}), \cdots, \lambda_p(\mathbf{\Theta}))$$

    denote the singular values of $\mathbf{\Theta}$.
  - The vector $\boldsymbol{\lambda}(\mathbf{\Theta})$ is assumed to be arranged in a *descending* order.
  - We only focus on the function g that is a function of singular values of its argument:

    $$g(\mathbf{\Theta}) = f(\boldsymbol{\lambda}(\mathbf{\Theta}))$$

    to indicate that g only acts on $\boldsymbol{\lambda}(\mathbf{\Theta})$.
  - The proximal operator of g of a matrix $\mathbf{X}$:

    $$\begin{aligned} \mathrm{prox}_g(\mathbf{X}) &= \arg\min_{\mathbf{\Theta}} \left\{ g(\mathbf{\Theta}) + \frac{1}{2}||\mathbf{X} - \mathbf{\Theta}||_F^2 \right\} \\ &= \mathbf{U}\mathrm{diag}\{\mathrm{prox}_f(\boldsymbol{\lambda}(\mathbf{X}))\}\mathbf{U}^T. \end{aligned}$$

Examples

- **Matrix-valued proximal operators (contd):**
  - **Example 1 (The log-barrier function):** Assume $\mathbf{\Theta} \in \mathcal{S}^p_{++}$, i.e. $\mathbf{\Theta}$ is a symmetric positive definite $p \times p$ matrix. For $\alpha > 0$, consider the following function:

$$g(\mathbf{\Theta}) = -\alpha \log \det(\mathbf{\Theta}).$$

Then for a symmetric $p \times p$ matrix $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\lambda})\mathbf{U}^T$ we have

$$\begin{aligned} \mathrm{prox}_g(\mathbf{X}) &= \arg\min_{\mathbf{\Theta}} \left\{ -\alpha \log \det(\mathbf{\Theta}) + \frac{1}{2} ||\mathbf{\Theta} - \mathbf{X}||_F^2 \right\} \\ &= \mathbf{U}\mathrm{diag}\{\mathrm{prox}_{-\alpha \log}(\boldsymbol{\lambda}(\mathbf{X}))\}\mathbf{U}^T \\ &= \sum_{j=1}^{p} \left(\frac{\lambda_j(\mathbf{X}) + \sqrt{[\lambda_j(\mathbf{X})]^2 + 4\alpha}}{2} \right) \mathbf{u}_{[j]} \mathbf{u}_{[j]}^T. \end{aligned}$$

Examples

- **Matrix-valued proximal operators (contd):**
  - **Example 2 (The Schatten 1-norm):** Assume $\mathbf{\Theta} \in \mathcal{S}_+^p$. Consider the following function:

$$g(\mathbf{\Theta}) = \alpha \sum_{j=1}^{p} |\lambda_j(\mathbf{\Theta})|.$$

Then for a symmetric $p \times p$ matrix $\mathbf{X} = \mathbf{U}\text{diag}(\boldsymbol{\lambda})\mathbf{U}^T$ we have

$$\begin{aligned} \text{prox}_g(\mathbf{X}) &= \arg\min_{\mathbf{\Theta}} \left\{ \alpha ||\mathbf{\Theta}||_{S_1} + \frac{1}{2} ||\mathbf{\Theta} - \mathbf{X}||_F^2 \right\} \\ &= \mathbf{U}\text{diag}\{\text{prox}_{\alpha||.||_1}(\boldsymbol{\lambda}(\mathbf{X}))\}\mathbf{U}^T \\ &= \sum_{j=1}^{p} \text{sign}[\lambda_j(\mathbf{X})](|\lambda_j(\mathbf{X})| - \alpha)_+ \mathbf{u}_{[j]}\mathbf{u}_{[j]}^T. \end{aligned}$$

Examples

- **Matrix-valued proximal operators (contd):**
  - **Example 3 (Projection onto $\mathcal{S}_+^p$):** Assume $\mathbf{\Theta}$ is a $p \times p$ matrix. Consider the following indicator function of $\mathbf{\Theta}$:

$$g(\mathbf{\Theta}) = \iota\{\mathbf{\Theta} \in \mathcal{S}_+^p\},$$

which is equivalent to the following function

$$f(\boldsymbol{\lambda}(\mathbf{\Theta})) = \iota\{\lambda_p(\mathbf{\Theta}) \geq 0\}.$$

Then for a $p \times p$ matrix $\mathbf{X} = \mathbf{U}\mathrm{diag}(\boldsymbol{\lambda})\mathbf{U}^T$ we have

$$\begin{aligned}
\mathrm{prox}_g(\mathbf{X}) &= \arg\min_{\mathbf{\Theta}} \left\{ \iota\{\mathbf{\Theta} \in \mathcal{S}_+^p\} + \frac{1}{2}||\mathbf{\Theta} - \mathbf{X}||_F^2 \right\} \\
&= \mathbf{U}\mathrm{diag}\{\mathrm{prox}_{\iota\{\lambda_p(.)\geq 0\}}(\boldsymbol{\lambda}(\mathbf{X}))\}\mathbf{U}^T \\
&= \sum_{j=1}^{p} [\lambda_j(\mathbf{X})]_+ \mathbf{u}_{[j]}\mathbf{u}_{[j]}^T.
\end{aligned}$$

Computation in Data Science: Week 14
Lecture 10

Tso-Jung Yen

Institute of Statistical Science
Academia Sinica

tjyen@stat.sinica.edu.tw

Data Science Degree Program

National Taiwan University

December 22, 2021

Proximal Gradient Algorithms

- **Basic idea:**
  - Consider the following optimization problem:

$$\min_{\boldsymbol{\theta}}\{l(\boldsymbol{\theta})+g(\boldsymbol{\theta})\}. \tag{13}$$

  - As mentioned previously in (3), we may run the following iterative scheme to find optimizer of (13):

$$\begin{aligned}
\boldsymbol{\theta}^{r+1} &= \arg\min_{\boldsymbol{\theta}}\left\{l(\boldsymbol{\theta}^r)+[\nabla l(\boldsymbol{\theta}^r)]^T(\boldsymbol{\theta}-\boldsymbol{\theta}^r)+\frac{1}{2c_r}||\boldsymbol{\theta}-\boldsymbol{\theta}^r||^2+g(\boldsymbol{\theta})\right\}\\
&= \arg\min_{\boldsymbol{\theta}}\left\{c_r g(\boldsymbol{\theta})+\frac{1}{2}\left|\left|\boldsymbol{\theta}-[\boldsymbol{\theta}^r-c_r\nabla l(\boldsymbol{\theta}^r)]\right|\right|_2^2\right\}. \qquad (14)
\end{aligned}$$

  - By using definition of the proximal operator, (14) can be expressed as

$$\begin{aligned}
\boldsymbol{\theta}^{r+1} &= \arg\min_{\boldsymbol{\theta}}\left\{c_r g(\boldsymbol{\theta})+\frac{1}{2}\left|\left|\boldsymbol{\theta}-[\boldsymbol{\theta}^r-c_r\nabla l(\boldsymbol{\theta}^r)]\right|\right|_2^2\right\}\\
&= \mathsf{prox}_{c_r g}\Big(\boldsymbol{\theta}^r-c_r\nabla l(\boldsymbol{\theta}^r)\Big). \qquad (15)
\end{aligned}$$

  - The iterative scheme (15) is called the **proximal gradient algorithm**.

Proximal Gradient Algorithms

- To establish the descent property for the sequence generated by the iterative scheme (14), we need the following two conditions:
  - **(a)** The loss function l is differentiable and its gradient satisfies the **Lipschitz continuous gradient condition**:

    $$||\nabla l(\boldsymbol{\theta}) - \nabla l(\mathbf{y})||_2 \leq M||\boldsymbol{\theta} - \mathbf{y}||_2$$

    for any $\boldsymbol{\theta}, \mathbf{y} \in \mathbb{R}^p$.
  - **(b)** g is convex.

- **The descent property:**
  - Under (a) and (b), for $\{\boldsymbol{\theta}^r\}_r$ generated from (14), we have

    $$l(\boldsymbol{\theta}^{r+1}) + g(\boldsymbol{\theta}^{r+1}) \leq [l(\boldsymbol{\theta}^r) + g(\boldsymbol{\theta}^r)] - \left(\frac{1}{c_r} - \frac{M}{2}\right)||\boldsymbol{\theta}^{r+1} - \boldsymbol{\theta}^r||_2^2. \quad (16)$$

Proximal Gradient Algorithms

- **Stepsize:**
  - To make the decent property (16) hold we have to have

$$c_r \leq \frac{2}{M},$$

which implies if Lipschitz constant M is available, we may let the stepsize $c_r = 1/M$. In this case, the descent property (16) becomes

$$l(\boldsymbol{\theta}^{r+1}) + g(\boldsymbol{\theta}^{r+1}) \leq [l(\boldsymbol{\theta}^{r}) + g(\boldsymbol{\theta}^{r})] - \frac{M}{2}||\boldsymbol{\theta}^{r+1} - \boldsymbol{\theta}^{r}||_2^2.$$

  - *Remark:* The line search method also works for choosing the value of the stepsize for the **proximal gradient algorithm**.

Proximal Gradient Algorithms

- **Gradient mapping:**
  - For practical purposes we define

$$\zeta_c(\boldsymbol{\theta}) = \frac{1}{c}\Big[\boldsymbol{\theta} - \mathsf{prox}_{cg}(\boldsymbol{\theta} - c\nabla l(\boldsymbol{\theta}))\Big]. \tag{17}$$

    Here (17) is called the **gradient mapping** of $\boldsymbol{\theta}$.

  - If $g(\boldsymbol{\theta}) = 0$ then the gradient mapping (17) becomes

$$\zeta_c(\boldsymbol{\theta}) = \frac{1}{c}\Big[\boldsymbol{\theta} - \boldsymbol{\theta} + c\nabla l(\boldsymbol{\theta})\Big] = \nabla l(\boldsymbol{\theta}),$$

    i.e. the gradient of l evaluated at $\boldsymbol{\theta}$.

  - The gradient mapping (17) allows us to express update $\boldsymbol{\theta}^{r+1}$ in the iterative scheme (15) as

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^r - c_r\zeta_{c_r}(\boldsymbol{\theta}^r). \tag{18}$$

Proximal Gradient Algorithms

- **Gradient mapping (contd):**
  - From (18) we have

$$||\boldsymbol{\theta}^{r+1} - \boldsymbol{\theta}^r||_2 = c_r||\zeta_{c_r}(\boldsymbol{\theta}^r)||_2.$$

  - Given $c_r = 1/M$, we may express the descent property (16) in terms of the gradient mapping (17) as

$$\frac{1}{2M}||\zeta_{c_r}(\boldsymbol{\theta}^r)||_2^2 \leq l(\boldsymbol{\theta}^r) + g(\boldsymbol{\theta}^r) - [l(\boldsymbol{\theta}^{r+1}) + g(\boldsymbol{\theta}^{r+1})].$$

- **Stopping criteria:** We may use

$$||\boldsymbol{\theta}^{r+1} - \boldsymbol{\theta}^r||_2 \leq \epsilon$$

(or equivalently $||\zeta_{c_r}(\boldsymbol{\theta}^r)||_2 \leq \epsilon/c_r$) as a criterion for stopping the iterative scheme (14).

Proximal Gradient Algorithms

- **Convergence analysis:**
  - If the conditions (a) and (b) are satisfied, and l is a convex function of $\boldsymbol{\theta}$, then given $c_r = 1/M$ we have

$$l(\boldsymbol{\theta}^r) + g(\boldsymbol{\theta}^r) - [l(\boldsymbol{\theta}^*) + g(\boldsymbol{\theta}^*)] \leq \frac{M||\boldsymbol{\theta}^0 - \boldsymbol{\theta}^*||_2^2}{2r},$$

    where $\boldsymbol{\theta}^*$ is the solution to the optimization problem $\min_{\boldsymbol{\theta}}\{l(\boldsymbol{\theta}) + g(\boldsymbol{\theta})\}$ and $\boldsymbol{\theta}^0$ is the initial value for running the iterative scheme (14).
  - In addition, we have

$$\min_{k=0,1,\cdots,r} ||\zeta_{c_k}(\boldsymbol{\theta}^k)||_2 = O(r^{-1}).$$

Proximal Gradient Algorithms

- **The fast proximal gradient algorithm:**
  - Now consider the iterative scheme:

$$\boldsymbol{\theta}^{r+1} \quad = \quad \text{prox}_{c_r g}\Big(\boldsymbol{\gamma}^r - c_r \nabla l(\boldsymbol{\gamma}^r)\Big). \tag{19}$$

  - The iterative scheme (19) becomes the proximal gradient algorithm when $\boldsymbol{\gamma}^r = \boldsymbol{\theta}^r$.
  - Define sequence $\{b_r\}_r$ with $b_0 = 1$ and

$$b_{r+1} = \frac{1 + \sqrt{1 + 4b_r^2}}{2}. \tag{20}$$

  - The **fast proximal gradient algorithm** uses

$$\boldsymbol{\gamma}^{r+1} = \boldsymbol{\theta}^{r+1} + \left(\frac{b_r - 1}{b_{r+1}}\right)(\boldsymbol{\theta}^{r+1} - \boldsymbol{\theta}^r) \tag{21}$$

  with $\boldsymbol{\gamma}^0 = \boldsymbol{\theta}^0$ to run the iterative scheme (19).

Proximal Gradient Algorithms

- **The fast proximal gradient algorithm (contd):**
  - Under similar conditions given above, for sequence $\{\boldsymbol{\theta}^r\}_r$ generated by the **fast proximal gradient algorithm** (19), (20) and (21), we have

$$l(\boldsymbol{\theta}^r) + g(\boldsymbol{\theta}^r) - [l(\boldsymbol{\theta}^*) + g(\boldsymbol{\theta}^*)] \leq \frac{2M||\boldsymbol{\theta}^0 - \boldsymbol{\theta}^*||_2^2}{(r+1)^2}.$$

  - *Remark 1:* The computation costs for the **fast proximal gradient algorithm** and the **proximal gradient algorithm** at each iteration have the same order magnitude.
  - *Remark 2:* **FISTA** (fast iterative shrinkage-thresholding algorithm, Beck and Teboulle (2009)) is a special case of the **fast proximal gradient algorithm** designed for carrying out lasso-type estimation.

Graphical Lasso

- **Example:**
  - The graphical lasso estimate for the precision matrix $\mathbf{\Theta}$ is the minimizer for the following optimization problem:

$$\min_{\mathbf{\Theta}} \quad -\log\det(\mathbf{\Theta}) + \mathrm{tr}(\mathbf{W}\mathbf{\Theta}) + \alpha||\mathrm{vec}(\mathbf{\Theta} - \mathbf{I} \circ \mathbf{\Theta})||_1, \qquad (22)$$

    where $\mathbf{W}$ is the sample covariance matrix and is assumed to be a symmetric matrix.
  - Here $\circ$ is the Hadamard product, and $\mathbf{\Theta} - \mathbf{I} \circ \mathbf{\Theta}$ means the diagonal terms of $\mathbf{\Theta}$ are eliminated.
  - Define

$$\begin{aligned} l(\mathbf{\Theta}) &= \mathrm{tr}(\mathbf{W}\mathbf{\Theta}), \\ g_1(\mathbf{\Theta}) &= -\log\det(\mathbf{\Theta}), \\ g_2(\mathbf{\Theta}) &= \alpha||\mathrm{vec}(\mathbf{\Theta} - \mathbf{I} \circ \mathbf{\Theta})||_1. \end{aligned}$$

    With definitions given above and a constraint $\mathbf{\Gamma} = \mathbf{\Theta}$, the graphical lasso estimation problem (22) becomes

$$\min_{\mathbf{\Theta},\mathbf{\Gamma}} \quad l(\mathbf{\Gamma}) + g_1(\mathbf{\Gamma}) + g_2(\mathbf{\Theta}) + \iota\{\mathbf{\Gamma} = \mathbf{\Theta}\}. \qquad (23)$$

Graphical Lasso

- **Example (contd):**
  - We may find a minimizer of the problem (23) by running the following iterative scheme:

$$\begin{aligned}\boldsymbol{\Gamma}^{r+1} &= \arg\min_{\boldsymbol{\Gamma}} \Big\{ l(\boldsymbol{\Theta}^r) + \mathsf{tr}(\nabla l(\boldsymbol{\Theta}^r)^T(\boldsymbol{\Gamma} - \boldsymbol{\Theta}^r)) \\ &\quad + \frac{1}{2c_r}||\boldsymbol{\Gamma} - \boldsymbol{\Theta}^r||_F^2 + g_1(\boldsymbol{\Gamma}) \Big\} \\ \boldsymbol{\Theta}^{r+1} &= \arg\min_{\boldsymbol{\Theta}} \Big\{ g_2(\boldsymbol{\Theta}) + \frac{1}{2c_r}||\boldsymbol{\Theta} - \boldsymbol{\Gamma}^{r+1}||_F^2 \Big\}. \qquad (24)\end{aligned}$$

  - Note that

$$\frac{\partial l(\boldsymbol{\Theta})}{\partial \theta_{ij}} = \mathsf{tr}(\mathbf{e}_j^T \mathbf{W} \mathbf{e}_i) = w_{ji} \Rightarrow \nabla l(\boldsymbol{\Theta}) = \mathbf{W},$$

  since we have assumed **W** is symmetric.

  - In addition, we may let $c_r = 1$ since

$$||\nabla l(\boldsymbol{\Theta}_1) - \nabla l(\boldsymbol{\Theta}_2)||_F = ||\mathbf{W} - \mathbf{W}||_F = 0 \leq ||\boldsymbol{\Theta}_1 - \boldsymbol{\Theta}_2||_F.$$

Graphical Lasso

- **Example (contd):**
  - Assume $\mathbf{\Theta}^r - \mathbf{W}$ has eigenvalue decomposition $\sum_{j=1}^{p} \lambda_j^r \mathbf{u}_{[j]}^r (\mathbf{u}_{[j]}^r)^T$. Now the first line of the iterative scheme (24) becomes

$$\begin{aligned}
\mathbf{\Gamma}^{r+1} &= \arg\min_{\mathbf{\Gamma}} \left\{ g_1(\mathbf{\Gamma}) + \frac{1}{2} ||\mathbf{\Gamma} - [\mathbf{\Theta}^r - \mathbf{W}]||_F^2 \right\} \\
&= \mathsf{prox}_{g_1}(\mathbf{\Theta}^r - \mathbf{W}) \\
&= \sum_{j=1}^{p} \left(\frac{\lambda_j^r + \sqrt{(\lambda_j^r)^2 + 4}}{2} \right) \mathbf{u}_{[j]}^r (\mathbf{u}_{[j]}^r)^T.
\end{aligned}$$

  - The second line of the iterative scheme (24) is

$$\begin{aligned}
\mathbf{\Theta}^{r+1} &= \arg\min_{\mathbf{\Theta}} \left\{ g_2(\mathbf{\Theta}) + \frac{1}{2} ||\mathbf{\Theta} - \mathbf{\Gamma}^{r+1}||_F^2 \right\} \\
&= \mathsf{prox}_{g_2}(\mathbf{\Gamma}^{r+1}) \\
&= \mathsf{ST}_{\alpha}(\mathbf{\Gamma}^{r+1} - \mathbf{I} \circ \mathbf{\Gamma}^{r+1}) + \mathbf{I} \circ \mathbf{\Gamma}^{r+1},
\end{aligned}$$

    where $\mathsf{ST}_{\alpha}(.)$ is the soft-thresholding operator.
  - In summary we have

$$\mathbf{\Theta}^{r+1} = \mathsf{prox}_{g_2}(\mathsf{prox}_{g_1}(\mathbf{\Theta}^r - \mathbf{W})).$$

Computation in Data Science: Week 14 Lecture 11

Tso-Jung Yen

Institute of Statistical Science
Academia Sinica

tjyen@stat.sinica.edu.tw

Data Science Degree Program

National Taiwan University

December 22, 2021

Stochastic Gradient Descent Algorithms

- **Problem setting:**
  - Consider the following optimization problem:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \qquad \frac{1}{n} \sum_{i=1}^{n} l(\boldsymbol{\theta}; \mathbf{x}_i), \tag{25}$$

    where $\boldsymbol{\theta} \in \mathbb{R}^p$ is a p-dimensional vector of parameters, and $\mathbf{x}_i$ is a data point containing information about the ith observation.
  - The problem (25) is a commonly-seen problem format in **statistics** and **machine learning**, e.g. maximum likelihood estimation.
  - For practical purposes, we assume the n observations are independently observed. Further define

$$h(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} l(\boldsymbol{\theta}; \mathbf{x}_i).$$

Stochastic Gradient Descent Algorithms

- **Problem setting (contd):**
  - Usually one can see the the objective function in (25) as

$$h(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} l(\boldsymbol{\theta}; \mathbf{x}_i) = \mathbb{E}_{\mathbf{x}}[l(\boldsymbol{\theta}; \mathbf{x})], \tag{26}$$

    where $\mathbb{E}_{\mathbf{x}}$ is an expectation operator such that $\mathbb{E}_{\mathbf{x}}(\delta_{\{\mathbf{x}=\mathbf{x}_i\}}) = \mathbb{P}(\mathbf{x} = \mathbf{x}_i) = n^{-1}$, where $\delta_{\{\mathbf{x}=\mathbf{x}_i\}} = 1$ if $\mathbf{x} = \mathbf{x}_i$, and $\delta_{\{\mathbf{x}=\mathbf{x}_i\}} = 0$ otherwise.
  - The representation (26) provides us a way to see the deterministic objective function $h(\boldsymbol{\theta})$ in problem (25) as the expectation of a random objective function $l(\boldsymbol{\theta}; \mathbf{x})$ with $\mathbb{P}(\mathbf{x} = \mathbf{x}_i) = n^{-1}$.

Stochastic Gradient Descent Algorithms

- To find a solution to (25), we consider the following iterative scheme:

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^{r} - c_r \mathbf{v}^r, \tag{27}$$

  where $\mathbf{v}^r$ is a p-dimensional vector.

- When $\mathbf{v}^r = \nabla h(\boldsymbol{\theta}^r)$, the iterative scheme (27) is an example of the gradient descent algorithm.

- When $\mathbf{v}^r$ is a random vector such that $\mathbb{E}[\mathbf{v}^r] = \nabla h(\boldsymbol{\theta}^r)$, the iterative scheme is an example of the **stochastic gradient descent algorithm (SGD)**.

Stochastic Gradient Descent Algorithms

- **Application:**
  - In practice, we use the following iterative scheme to compute $\boldsymbol{\theta}^{r+1}$: At the $(r+1)$th iteration, choose i_r *uniformly* from $\{1, 2, \cdots, n\}$ and define

    $$\mathbf{v}_{i_r}^r = \nabla l(\boldsymbol{\theta}^r; \mathbf{x}_{i_r}).$$

    In this case we have

    $$\mathbb{E}[\mathbf{v}_{i_r}^r] = \mathbb{E}_{\mathbf{X}}[\nabla l(\boldsymbol{\theta}^r; \mathbf{x}_{i_r})] = \frac{1}{n} \sum_{i=1}^{n} \nabla l(\boldsymbol{\theta}^r; \mathbf{x}_{i_r}) = \nabla h(\boldsymbol{\theta}^r).$$

  - According to theory of stochastic gradient descent algorithms (Chapter 8 of Beck, 2017), if $\mathbf{v}_{i_r}^r$ further satisfies some regularity conditions, then we can use the iterative scheme

    $$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^r - c_r \cdot \mathbf{v}_{i_r}^r \tag{28}$$

    to find a solution to the problem (25).

Stochastic Gradient Descent Algorithms

- *Remark 1:* In training deep neural network models, we usually sample a *batch* of $\mathbf{v}_{i_r}^r$'s to compute stochastic approximation of $\nabla h(\boldsymbol{\theta}^r)$, e.g. with batch size B, we sample $\{\mathbf{v}_{i_r}\}_{i=1}^B$ and then compute

$$\mathbf{g}^r = \frac{1}{B}\sum_{i=1}^{B} \mathbf{v}_{i_r}^r = \frac{1}{B}\sum_{i=1}^{B} \nabla l(\boldsymbol{\theta}^r; \mathbf{x}_{i_r}).$$

  Here $\mathbf{g}^r$ is a **stochastic approximation** to the gradient of the loss function $h(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}}[l(\boldsymbol{\theta}; \mathbf{x})]$ evaluated at $\boldsymbol{\theta}^r$.

  In practice we write $\boldsymbol{\theta}^{r+1}$ as

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^r - c_r \mathbf{g}^r. \tag{29}$$

- *Remark 2:* There is no concrete descent property for the stochastic gradient descent algorithm. Some theoretical results (e.g. Chapter 3 of Mahoney et al. (2018)) suggest $c_r = a/(r+b)$ with $a, b > 0$. In practice (in particular in deep learning) setting c_r heavily relies on **trial-and-error** and **heuristics**.

Stochastic Gradient Descent Algorithms

- In **TensorFlow 2.0**, the stochastic gradient descent algorithm is carried out using the function

  tf.keras.optimizers.SGD(learning_rate),

  where "learning_rate" has the same definition as c_r in (28) and (29).

- In **PyTorch**, the the stochastic gradient descent algorithm is carried out using the function

  torch.optim.SGD(lr),

  where "lr" has the same definition as c_r in (28) and (29).

Adaptive Methods

- **AdaGrad (Duchi et al., 2011):**
  - The **AdaGrad** iterative scheme takes the following form for updating $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^{r} - c_r[\mathbf{H}^r]^{-1}\mathbf{g}^r, \tag{30}$$

where c_r is the learning rate,

$$\mathbf{H}^r = \mathrm{diag}(\mathbf{u}^r + \epsilon\mathbf{1}) \tag{31}$$

with $\epsilon \geq 0$ is a scale matrix, and

$$\mathbf{u}^r = \left[\sum_{s=1}^{r} \mathbf{g}^s \circ \mathbf{g}^s\right]^{1/2},$$

and $\mathbf{g}^r$ is a stochastic approximation to the gradient of the loss function.

Adaptive Methods

- **AdaGrad (Duchi et al., 2011):**
  - In **TensorFlow 2.0**, the **AdaGrad** algorithm is carried out using the function

    tf.keras.optimizers.Adagrad(learning_rate, epsilon),

    where "learning_rate" has the same definition as c_r in (30), and "epsilon" is the same as ϵ defined in (31).
  - In **PyTorch**, the **AdaGrad** algorithm is carried out using the function

    torch.optim.Adagrad(lr, eps),

    where "lr" has the same definition as c_r in (30), and "eps" is the same as ϵ defined in (31).

Adaptive Methods

- **RMSProp (Tieleman and Hinton, 2012):**
  - The **RMSProp** (Root Mean Square Propagation) iterative scheme takes the following form for updating $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^{r} - c_r[\mathbf{H}^r]^{-1}\mathbf{g}^r, \tag{32}$$

where c_r is the learning rate,

$$\mathbf{H}^r = \text{diag}\{[(\mathbf{u}^r)^{1/2} + \epsilon\mathbf{1}]\} \tag{33}$$

with $\epsilon \geq 0$, and

$$\mathbf{u}^r = \rho\mathbf{u}^{r-1} + (1-\rho)\mathbf{g}^r \circ \mathbf{g}^r \tag{34}$$

with $\rho \in [0, 1]$.

Adaptive Methods

- **RMSProp (Tieleman and Hinton, 2012):**
  - In **TensorFlow 2.0**, the **RMSProp** algorithm is carried out using the function

    tf.keras.optimizers.RMSprop(learning_rate, rho, epsilon),

    where "learning_rate" has the same definition as c_r in (32), "rho" is the same as ρ in (34), and "epsilon" is the same as ϵ in (33).
  - In **PyTorch**, the **RMSProp** algorithm is carried out using the function

    torch.optim.RMSprop(lr, alpha, eps),

    where "lr" has the same definition as c_r in (32), "alpha" is the same as ρ in (34), and "eps" is the same as ϵ in (33).

Adaptive Methods

- **Adam (Kingma and Ba, 2015):**
  - The **Adam** algorithm uses the following iterative scheme for weight updating:

$$\boldsymbol{\theta}^{r+1} = \boldsymbol{\theta}^r - c_r[\mathbf{H}^r]^{-1}\hat{\mathbf{g}}^r, \tag{35}$$

where

$$\begin{aligned} \hat{\mathbf{g}}^r &= (1-\beta_1^r)^{-1}\tilde{\mathbf{g}}^r \\ \tilde{\mathbf{g}}^r &= \beta_1\tilde{\mathbf{g}}^{r-1} + (1-\beta_1)\mathbf{g}^r, \end{aligned} \tag{36}$$

with $\beta_1 \in [0,1]$ and $\mathbf{g}^r$ is a stochastic approximation of the gradient vector of the loss function, and

$$\mathbf{H}^r = \mathrm{diag}\{[(1-\beta_2^r)^{-1}\mathbf{u}^r]^{1/2} + \epsilon\mathbf{1}\}, \tag{37}$$

where $\beta_2 \in [0,1]$, $\epsilon \geq 0$, and

$$\mathbf{u}^r = \beta_2\mathbf{u}^{r-1} + (1-\beta_2)\mathbf{g}^r \circ \mathbf{g}^r.$$

Adaptive Methods

- **Adam (Kingma and Ba, 2015):**
  - In **TensorFlow 2.0**, the **Adam** algorithm is carried out using the function

    tf.keras.optimizers.Adam(learning_rate, beta_1, beta_2, epsilon),

    where "learning_rate" has the same definition as c_r in (35), "beta_1" is the same as β_1 in (36), "beta_2" is the same as β_2 in (37), and "epsilon" is the same as ϵ in (37).
  - In **PyTorch**, the **Adam** algorithm is carried out using the function

    torch.optim.Adam(lr, betas=(beta1, beta2), eps),

    where "lr" has the same definition as c_r in (35), "beta1" is the same as β_1 in (36), "beta2" is the same as β_2 in (37), and "eps" is the same as ϵ in (37).

Adaptive Methods

- **When adaptive methods work?**
  - According to Wilson et al. (2017), adaptive methods may have better performances in **generative adversarial networks** or **reinforcement learning**.
  - These training schemes are not about to solve optimization (minimization) problems.
  - Dynamics of the adaptive methods may be accidentally well matched to the search procedures of these training schemes.

Week 14

- **References:**

  - H. H. Bauschke and P. L. Combettes (2010). *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer. The materials on proximal operators are from this book.
  - A. Beck (2017). *First-Order Methods in Optimization*. SIAM. The materials on proximal operators and proximal gradient algorithms are from this book.
  - A. Beck and M. Teboulle (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal of Imaging Science*, Vol. 2, page 183-202. The **FISTA** algorithm mentioned in the course slides is from this paper.
  - J. Duchi, E. Hazan, and Y. Singer (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, 2121-2159.
  - D. P. Kingma and J. L. Ba (2015). Adam: a method for stochastic optimization. *ICLR*.
  - M. W. Mahoney, J. C. Duchi and A. C. Gilbert (2018). *The Mathematics of Data*. American Mathematical Society. Parts of materials on stochastic gradient descent algorithms are from Chapter 3 (written by Professor J. C. Duchi) of this book.
  - N. Parikh and S. Boyd (2013). Proximal Algorithms. *Foundations and Trends in Optimization*, Vol. 1. No. 3, page 123-231. The materials on proximal operators are from this book.
  - T. Tieleman and G. Hinton (2012). Lecture 6.5-rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude. COURSERA: Neural Networks for Machine Learning, 4, 26-31.
  - A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht (2017). The marginal value of adaptive gradient methods in machine learning. *NIPS*.