10-725/36-725: Convex Optimization **Fall 2015**

Lecture 8: September, 24

Lecturer: Ryan Tibshirani *Scribes: Aman Gupta, Hemank Lamba*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

8.1 Previous Lecture: Subgradient Method

Given the problem,

$$\min_x f(x)$$

with $f(x)$ convex, and $dom(f) = \mathbb{R}^n$.

- Choose an initial $x^{(0)} \in \mathbb{R}^n$, and repeat:
- $x^k = x^{k-1} - t_k.g^{k-1}$, k=1,2,3, ..., where $g^{k-1} \in \partial f(x^{k-1})$

If f is Lipschitz, then subgradient method has a convergence rate $\mathcal{O}(\frac{1}{\epsilon^2})$. Advantage of using subgradient method is that it is very generic, whereas it can be very slow.

In this lecture, we consider the case of decomposable functions, and show how for such a function we can achieve a convergence rate of $\mathcal{O}(\frac{1}{\epsilon})$. We also introduce the concept of acceleration, which helps in achieving the optimal convergence rate as specified by Nesterov [1].

8.2 Proximal Gradient Descent

We can improve on the rather slow subgradient method by turning to proximal gradient descent, an algorithm with an improved running time and the ability to a decomposable objective fuction that may not necessarily be differentiable.

8.2.1 Decomposable Functions

Consider an objective function that is decomposable into two functions in the following manner:

$$f(x) = g(x) + h(x) \tag{8.1}$$

where g is a convex and differentiable function, and h is convex and possibly non-differentiable, but simple. An example for a simple h is the l1-norm of a vector. With the proximal gradient descent method, we can achieve a convergence rate of $\mathcal{O}(\frac{1}{\epsilon})$. By adding acceleration, this can be improved to $\mathcal{O}(\frac{1}{\sqrt{\epsilon}})$.

[1] This part is covered in the next set of scribe notes

Simple gradient descent works with a convex and differentiable f, using gradient information to take steps towards the optima. This step is derived using a quadratic approximation of the objective function $f(x)$, after replacing $\nabla^2 f$ with a spherical term $\frac{1}{t}I$:

$$x^+ = \underset{z}{\text{argmin}} \quad \underbrace{f(x) + \nabla f(x)^T(z-x) + \frac{1}{2t}\|z-x\|_2^2}_{\widetilde{f}_t(z)} \tag{8.2}$$

If f is not differentiable, but is decomposable into two functions g and h as described above, we can still use a quadratic approximation of the smooth part g to define a step towards the minimum value

$$\begin{aligned} x^+ &= \underset{z}{\text{argmin}} \quad \widetilde{g}_t(z) + h(z) \\ &= \underset{z}{\text{argmin}} \quad g(x) + \nabla g(x)^T(z-x) + \frac{1}{2t}\|z-x\|_2^2 + h(z) \end{aligned} \tag{8.3}$$

8.2.2 Proximal Mapping

Equation 8.3 can be rewritten as:

$$\begin{aligned} x^+ &= \underset{z}{\text{argmin}} \quad \frac{1}{2t}\|z-(x-t\nabla g(x))\|_2^2 + h(z) \\ &= prox_{h,t}(x - t\nabla g(x)) \end{aligned} \tag{8.4}$$

Where $prox$ is a function of h and t, and is referred to as the proximal map of h:

$$prox_{h,t}(x) = \underset{z}{\text{argmin}} \quad \frac{1}{2t}\|z-x\|_2^2 + h(z) \tag{8.5}$$

In equation 8.4, the first term is minimized when z is close to the gradient update of the smooth part g, and the second term is minimized when the value of h is as small as possible.

8.2.3 Proximal Gradient Descent

Proximal gradient descent can be defined as follows: Choose initial $x^{(0)}$ and then repeat:

$$x^{(k)} = \ prox_{t_k}(x^{(k-1)} - t_k \nabla g(x^{(k-1)})), \quad k = 1, 2, 3, \ldots \tag{8.6}$$

To make the update look familiar, we define the update in terms of the generalized gradient:

$$x^{(k)} = x^{(k-1)} - t_k \cdot G_{t_k}(x^{(k-1)}) \tag{8.7}$$

where G_t is a generalized gradient of f,

$$G_t(x) = \frac{x - prox_t(x - t\nabla g(x))}{t} \tag{8.8}$$

Even though we swapped one minimization problem for another, this approach can be advantageous in the following manner:

- The proximal map $prox_t(\cdot)$ can be computed analytically for a lot of h functions
- $prox_t(\cdot)$ doesn't depend on g, only on h
- g can be a complicated function; all we need to do is to compute its gradient

8.2.4 Iterative soft-thresholding algorithm (ISTA)

Let's consider an example where we are trying to solve the lasso problem. As we might recall the lasso problem is given as follows:
For a given $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$, we want to optimize:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2} \underbrace{\|y - X\beta\|_2^2}_{g(\beta)} + \underbrace{\lambda \|\beta\|_1}_{h(\beta)} \tag{8.9}$$

For proximal gradient descent algorithm to work, we first need to find the proximal mapping for the given objective function. We know that the proximal mapping can be computed as follows:

$$\begin{aligned} prox_t(\beta) &= \underset{z \in \mathbb{R}^n 0}{argmin} \frac{1}{2t} \|\beta - z\|_2^2 + \lambda \|z\|_1 \\ &\equiv \underset{z \in \mathbb{R}^n 0}{argmin} \frac{1}{2} \|\beta - z\|_2^2 + \lambda t \|z\|_1 \\ &= S_{\lambda t}(\beta) \end{aligned} \tag{8.10}$$

where, we know that $S_\lambda(\beta)$ is the soft-thresholding operator given by:

$$[S_\lambda(\beta)]_i = \begin{cases} \beta_i - \lambda & \text{if } \beta_i > \lambda \\ 0 & \text{if } -\lambda \leq \beta_i \leq \lambda \\ \beta_i + \lambda & \text{if } \beta_i < -\lambda \end{cases} \tag{8.11}$$

Therefore, the proximal map for lasso objective is the soft-thresholding operator of β parameterized by λt. The gradient of g, is

$$\nabla g(\beta) = -X^T(y - X\beta) \tag{8.12}$$

Therefore, our proximal gradient descent update becomes:

$$\beta^+ = S_{\lambda t}(\beta + t \cdot X^T(y - X\beta)) \tag{8.13}$$

The proximal gradient descent algorithm to solve the lasso problem is very simple, and is known as the *Iterative Soft Thresholding Algorithm*(ISTA). We compare the subgradient method and proximal gradient descent method (ISTA) in Fig 8.1. It can clearly be seen that the proximal gradient descent algorithm performs much better. Thus, in the case of decomposable functions where the prox operator of $h(z)$ is easy to compute, it is advantageous to use proximal gradient descent than subgradient method.

8.2.5 Convergence Analysis

As mentioned above for the cases of decomposable function $f(x) = g(x) + h(x)$, we assume:

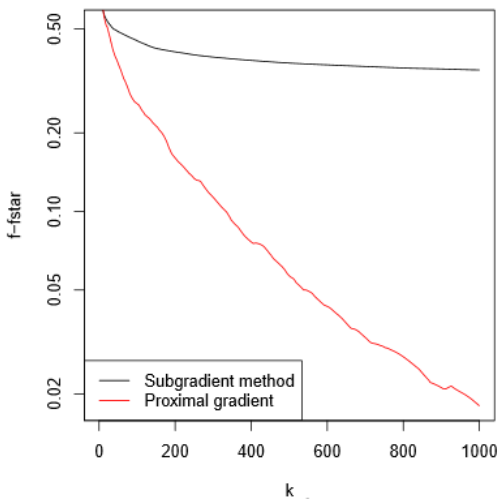

Figure 8.1: Example of proximal gradient descent (ISTA) vs. subgradient method convergence rates

- The function g is convex, differentiable, $dom(g) = \mathbb{R}^n$, and ∇g is Lipschitz continuous with L.
- The function h is convex and its proximal map can be easily computed.

With the above assumptions, the proximal gradient descent with fixed step size $t \leq 1/L$ satisfies:

$$f(x^{(k)}) - f* \leq \frac{\|x^{(0)} - x*\|_2^2}{2tk} \tag{8.14}$$

This implies that the proximal gradient descent has a convergence rate of $\mathcal{O}(1/k)$ or $\mathcal{O}(1/\epsilon)$. The convergence rate is very similar to the convergence rate of gradient descent. However, we should be careful as this specifies the number of iterations and not the number of operations.

8.2.6 Backtracking Line Search

Similar to gradient descent, backtracking line search to determine the step size for each step towards the minima. However, the search is applied only on the smooth part g of the function f.

To perform backtracking line search, first decide a shrinking parameter $0 < \beta < 1$, and at each iteration, start with $t = 1$, and while

$$g(x - tG_t(x)) > g(x) - t\nabla g(x)^T G_t(x) + \frac{t}{2}\|G_t(x)\|_2^2 \tag{8.15}$$

shrink $t = \beta t$, where $G_t(x)$ is the generalised gradient as described in previous sections. Else, perform proximal gradient update. With the same assumptions as those for gradient descent, we get the same convergence bounds for proximal gradient descent:

$$f(x^{(k)}) - f* \leq \frac{\|x^{(0)} - x*\|_2^2}{2t_{\min}k} \tag{8.16}$$

where $t_{\min} = \min\{1, \beta/L\}$

8.3 Examples

8.3.1 Matrix Completion

Consider recommender system problem, where we are seeking to predict what ratings would a user give to a particular movie. We first create a matrix, where the rows indicate users, and columns indicate movies. This matrix is partially observed, the goal is to fill in the entire matrix. Formally, this problem can be written as: Given a matrix $Y \in \mathbb{R}^{m\times n}$ where you only observe the entries $Y_{ij}, (i,j) \in \Omega$. Suppose we want to fill in missing entries (e.g. for a recommender system), so we solve a *matrix completion* problem:

$$\min_{B\in\mathbb{R}^{m\times n}} \frac{1}{2}\sum_{(i,j)\in\Omega}(Y_{ij}-B_{ij})^2 + \lambda\|B\|_{\text{tr}} \tag{8.17}$$

where $\|B\|_{\text{tr}}$ is the trace (nuclear) norm of B:

$$\|B\|_{\text{tr}} = \sum_{i=1}^{r}\sigma_i(B) \tag{8.18}$$

and $r = \text{rank}(B)$ and $\sigma_1(B) \geq \cdots \geq \sigma_r(B) \geq 0$ are singular values of matrix X. In this problem, the first term ensures that we stay close to the actual observed values, whereas the trace norm ensures that the end matrix is low ranked.

This objective function, can now be re-written in terms of only observed values, which will be more efficient. Now, define $P_\Omega(B) = \begin{cases} B_{ij} & (i,j)\in\Omega \\ 0 & (i,j)\in\Omega \end{cases}$, which is the projection operator onto an observed index set Ω. Then, the criterion becomes:

$$f(B) = \frac{1}{2}\|P_\Omega(Y) - P_\Omega(B)\|_F^2 + \lambda\|B\|_{\text{tr}} \tag{8.19}$$

This form of the problem is nothing but a matrix representation of the original matrix completion equation. For proximal gradient algorithm, we need two things: ∇g and the prox function:

- Gradient: $\nabla g(B) = -(P_\Omega(Y) - P_\Omega(B))$

- Prox Function:

$$\text{prox}_t(B) = \underset{Z\in\mathbb{R}^{m\times n}}{\text{argmin}} \frac{1}{2t}\|B-Z\|_F^2 + \lambda\|Z\|_{\text{tr}} \tag{8.20}$$

Since trace norm for matrices is equivalent to l-1 norm, we can claim that $\text{prox}_t(B) = S_{\lambda t}(B)$, the matrix soft-thresholding at the level λt, where

$$S_\lambda(B) = U\Sigma_\lambda V^T \tag{8.21}$$

for the SVD $B = U\Sigma V^T$ and diagonal Σ_λ such that

$$(\Sigma_\lambda)_{ii} = \max\{\Sigma_{ii} - \lambda, 0\} \tag{8.22}$$

Subgradient optimality tells us that the optimal solution to 8.19 should satisfy, for some

$$0 \in Z - B + \lambda t\partial\|Z\|_{tr} \tag{8.23}$$

So, let's set $Z = S_{\lambda t}(B)$ and see that this indeed 8.23. Denoting the SVD of B and $B = U\Sigma V^T$, we can see that the soft-thresholded version Z has a singular value decomposition of $U\Sigma_{\lambda t}V^T$ where $\Sigma_{\lambda t}$ has diagonal

elements thresholded above at λt (this is directly from the definition of the soft-thresholding operator $S_{\lambda t}$). Plugging the SVD into 8.23, we need only verify the following:

$$0 \in U\Sigma_{\lambda t}V^T - U\Sigma V^T + \lambda t \partial \|U\Sigma_{\lambda t}V^T\| \tag{8.24}$$

$$\iff \frac{1}{\lambda t}\left[U\Sigma V^T - U\Sigma_{\lambda t}V^T\right] \in \partial \|U\Sigma_{\lambda t}V^T\| \tag{8.25}$$

Take the left hand side and rearrange to make

$$\frac{1}{\lambda t}U\Sigma V^T - U\Sigma_{\lambda t}V^T = UV^T + \frac{1}{\lambda t}U(\frac{\Sigma - \Sigma_{\lambda t}}{\lambda t} - I)V^T = UV^T + W \tag{8.26}$$

Because the soft thresholding operation on $B = U\Sigma V^T$ gives:

$$\Sigma_{\lambda t} = \text{diag}\left((\Sigma_{11} - \lambda t)^+, (\Sigma_{22} - \lambda t)^+, \cdots\right) \tag{8.27}$$

we can verify the following

$$\frac{1}{\lambda t}(\Sigma - \Sigma_{\lambda t}) - I = \begin{pmatrix} \frac{1}{\lambda t}[\Sigma_{11} - (\Sigma_{11} - \lambda t)^+] & \cdots & \cdots & 0 \\ \vdots & \frac{1}{\lambda t}[\Sigma_{22} - (\Sigma_{22} - \lambda)^+] & \vdots & 0 \\ \vdots & \cdots & \ddots & 0 \\ 0 & 0 & 0 & \ddots \end{pmatrix} - I \tag{8.28}$$

$$= \begin{pmatrix} \frac{1}{\lambda t}\min[\lambda t, \Sigma_{11}] & \cdots & \cdots & 0 \\ \vdots & \frac{1}{\lambda t}\min[\lambda t, \Sigma_{22}] & \vdots & 0 \\ \vdots & \cdots & \ddots & 0 \\ 0 & 0 & 0 & \ddots \end{pmatrix} - I \tag{8.29}$$

$$= \begin{pmatrix} \min[1, \frac{\Sigma_{11}}{\lambda t}] - 1 & \cdots & \cdots & \\ \vdots & \min[1, \frac{\Sigma_{22}}{\lambda t}] - 1 & \vdots & 0 \\ \vdots & \cdots & \ddots & 0 \\ 0 & 0 & 0 & \ddots \end{pmatrix} \tag{8.30}$$

whose diagonal entries are all smaller than 0 so that $\|W\|_{\text{op}} \leq 0$. It is easy to see that $U^TW = WV = 0$. Therefore, the left hand side of 8.25 is indeed a subgradient of the trace norm of Z (right hand side of 8.25):

$$\frac{1}{\lambda t}\left[U\Sigma V^T - U\Sigma_{\lambda t}V^T\right] = U^TV + W \in \partial\|U\Sigma_{\lambda t}V^T\| \tag{8.31}$$

$$= \{UV^T + W_0 : \|W\|_{\text{op}} \leq 1, U^TW_0 = 0, W_0V = 0\} \tag{8.32}$$

The last equality should be used as a fact.

Using this, we can see that the proximal gradient update step is:

$$B^+ = S_{\lambda t}\left(B + t(P_\Omega(Y) - P_\Omega(B))\right) \tag{8.33}$$

We can also see that $\nabla g(B)$ is Lipschitz continuous with $L = 1$, so we can choose fixed step size $t = 1$ for a simpler fixed-step proximal update with proved rate $O(1/\epsilon)$. The update is now:

$$B^+ = S_{\lambda t}\left(t(P_\Omega(Y) - P_\Omega^\perp(B))\right) \tag{8.34}$$

where $P^\perp(\cdot)$ projects onto the unobserved set. This is the *soft-impute* algorithm, a simple and effective method for matrix completion. [2]

[2]The rest of the lecture is covered in the next set of notes.

References and Further Reading

Nesterov's four ideas (three acceleration methods):

- Y. Nesterov (1983), "A method for solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$"
- Y. Nesterov (1988), "On an approach to the construction of optimal methods of minimization of smooth convex functions"
- Y. Nesterov (2005), "Smooth minimization of non-smooth functions"
- Y. Nesterov (2007), "Gradient methods for minimizing composite objective function"

Extensions and/or analyses:

- A. Beck and M. Teboulle (2008), "A fast iterative shrinkage-thresholding algorithm for linear inverse problems"
- S. Becker and J. Bobin and E. Candes (2009), "NESTA: a fast and accurate first-order method for sparse recovery"
- P. Tseng (2008), "On accelerated proximal gradient methods for convex-concave optimization"

Helpful lecture notes/books:

- E. Candes, Lecture notes for Math 301, Stanford University, Winter 2010-2011
- Y. Nesterov (2004), "Introductory lectures on convex optimization: a basic course", Chapter 2
- L. Vandenberghe, Lecture notes for EE 236C, UCLA, Spring 2011-2012