Proximal Gradient Descent and Frank-Wolfe Method

Arpit Agarwal

July 2, 2016

In this note I will discuss projected gradient descent and also proximal gradient descent. I would also discuss conditional gradient descent (a.k.a Frank Wolfe method). These notes are taken from multiple sources including [Bub14], the convex optimization course at CMU [1], and lecture notes by Laurent El Ghaoui[2].

Notation: In this note we will call functions that have L-Lipschitz continuous gradients as L-smooth functions.

1 Projected Gradient Descent

Projected gradient descent is one of the simplest method for constraint optimization. It follows gradient descent closely except that we project an iterate back to the constraint set in each iteration. Consider the following optimization problem:

$$\min_{x \in C} f(x)$$

In projected gradient descent we generate a sequence of points $\{x_k\}$, for $k = 1, \cdots$, using the following update:

$$x_+ = x_k - t_k \nabla f(x_k) \tag{1}$$
$$x_{k+1} = P_C(x_+)\,, \tag{2}$$

where t_k is the step size and P_C is the projection operation

$$P_C(x) = \min_{y \in C} \frac{1}{2}\|x - y\|_2\,.$$

Example 1.1. *Let $C = \{x \in \mathbb{R}^n : Ax = b\}$ for some non-singular A, and consider the minimization of a function f over this set.*

[1] http://www.cs.cmu.edu/ ggordon/10725-F12/
[2] http://people.eecs.berkeley.edu/ elghaoui/Teaching/EE227A/lecture18.pdf

$$P_C(x_+) = x_+ - A^\top (AA^\top)^{-1}(Ax_+ - b)\,.$$

Therefore, the projected gradient descent iterates are of the form

$$\begin{aligned} x_{k+1} &= x_+ - A^\top (AA^\top)^{-1}(Ax_+ - b) && (3)\\ &= x_k - t_k(I - A^\top (AA^\top)^{-1}A)\nabla f(x_k)\,. && (4) \end{aligned}$$

The projection can be computed in the closed form for some sets such as hyperplanes, norm balls etc. However, for other sets the projection operation might not be easy to compute.

2 Proximal Gradient Descent

In the earlier section we saw the projected gradient descent. Proximal gradient descent is a generalization of it, where we use the proximal operator in place of the projection operator.

2.1 Proximal Operator

For a convex function h, we define the proximal operator as:

$$\text{prox}_h(x) = \text{argmin}_{u\in\mathbb{R}^n} h(u) + \frac{1}{2}\|u - x\|_2^2$$

The following is a necessary and sufficient condition for the above minimization.

$$y = \text{prox}_h(x) \iff x - y \in \partial f(y)\,.$$

Examples:

1. If $h(x) = 0$, then $\text{prox}_h(x) = x$.
2. If $h(x) = I_C(x)$, where $I_C(x)$ is the indicator function of constraint set C

$$I_C(x) = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{o/w} \end{cases},$$

   then $\text{prox}_h(x) = P_C(x)$, i.e. the proximal operator behaves as the projection operator.

3. If $h(x) = \lambda\|x\|_1$, then

$$\text{prox}_h(x)_i = \begin{cases} x_i - \lambda & \text{if } x_i > \lambda \\ 0 & \text{if } -\lambda \le x_i \le \lambda \\ x_i + \lambda & \text{o/w} \end{cases},$$

One property of the proximal operator is that it is non-expansive. Formally, let $u = \text{prox}_h(x)$ and $\hat{u} = \text{prox}_h(\hat{x})$, then

$$\|u - \hat{u}\|_2 \leq \|x - \hat{x}\|_2 .$$

2.2 Solving an Optimization Problem

Suppose that we have the following minimization problem

$$\min_{x \in \mathbb{R}^n} f(x) + h(x) ,$$

where $f(x)$ is a convex and differentiable function; and h is a convex function.

In proximal gradient descent we generate a sequence of points $\{x_k\}$, for $k = 1, \cdots$, using the following update:

$$x_+ = x_k - t_k \nabla f(x_k) , \tag{5}$$
$$x_{k+1} = \text{prox}_{t_k h}(x_+) , \tag{6}$$

where t_k is the step size and $\text{prox}_{t_k h}$ is the proximal operator for the function $t_k h$.

Examples:

1. If $h(x) = 0$, then $\text{prox}_h(x) = x$ and proximal gradient descent reduces to gradient descent.
2. If $h(x) = I_C(x)$, then $\text{prox}_h(x) = P_C(x)$ and proximal gradient descent reduces to projected gradient descent.
3. If $h(x) = \lambda \|x\|_1$, i.e. we want to minimize a function with lasso regularization, then

$$x_+ = x_k - t_k \nabla f(x_k) , \tag{7}$$
$$x_{(k+1)i} = \begin{cases} x_{+i} - \lambda t_k & \text{if } x_{+k} > \lambda t_k \\ 0 & \text{if } -\lambda t_k \leq x_{+k} \leq \lambda t_k \\ x_{+i} + \lambda t_k & \text{o/w} \end{cases} , \tag{8}$$

where x_{ki} denotes the i-th component of the vector x_k.

This algorithm is called Iterative Soft Thresholding Algorithm (ISTA).

2.3 Convergence Analysis

The proximal gradient updates can also been seen in the following form:

$$x_{k+1} = x_k - t_k G_{t_k}(x_k)\,,$$

where $G_{t_k}(x) = \frac{1}{t_k}(x - \text{prox}_{t_k h}(x - t\nabla f(x)))$.

The above implies that $G_t(x) \in \nabla f(x) + \partial h(x - t\nabla G_t(x))$. Therefore, the necessary and sufficient condition for optimality of x^* is that $G_t(x^*) = 0$.

To check for convergence one can see if $\|G_t(x)\|_2$ is very small.

One can show a $O(1/k)$ convergence of the proximal gradient method, when the function f is L-smooth.

Theorem 2.1. *Let the function f be L-smooth and convex, and h be convex, then the proximal gradient descent with $t_k = 1/L$ satisfies*

$$f(x_k) - f(x^*) \leq \frac{L||x_0 - x^*||^2}{2k}\,.$$

3 Frank-Wolfe Method

The Frank-Wolfe (also known as conditional gradient) method is used for a convex optimization problem when the constraint set is compact. Instead of solving the projection operation in each iteration, it solves a linear program over the constraint set.

We generate a sequence of points $\{x_k\}$, for $k = 1, \cdots$, using the following update:

$$y_k = \text{argmin}_{y \in C} \nabla f(x_k)^\top y\,, \tag{9}$$

$$x_{k+1} = (1 - \gamma_k)x_k + \gamma_k y_k\,, \tag{10}$$

where $0 \leq \gamma_k \leq 1$ is an update parameter. Note that if $x_k \in C$, then $x_{k+1} \in C$ due to the convexity of C.

There are other variants of this method in which one updates x_{k+1} using y_k and all the previous iterates. This is sometimes called the fully corrective variant of Frank-Wolfe.

3.1 Properties of the Frank-Wolfe Method

Here we list some of the most important properties of the Frank-Wolfe method:

1. **Sparse Updates:** Due to a property of the linear optimization in Equation 9, we have that y_k is a vertex of the convex set C. The update x_{k+1} is a convex combination of x_k and y_k. If we initialize x_0 with some vertex of C, then at iteration $k+1$, x_{k+1} will be a convex combination of atmost k vertices.

Therefore, each of the iterates will be sparse in the sense that they can be represented as a convex combination of a few vertices of C. This fact will be useful when one is working in a very high dimensional space, but there is a solution that can be represented as a convex combination of a few vertices.

2. **Computes suboptimality gap in each iteration:** The following quantity is an upper bound on the suboptimality gap of current iterate

$$\max_{y \in C} \nabla f(x_k)^\top (x_k - y)\,.$$

To see this consider the first-order condition for optimality

$$f(y) \geq f(x_k) + \nabla f(x_k)^\top (y - x_k)\,,$$

and minimizing it on both sides we get

$$f(x^*) \geq f(x_k) + \min_{y \in C} \nabla f(x_k)^\top (y - x_k)\,,$$

which gives the result. Note that this quantity can be computed easily in each iteration using y_k. This gives a condition for checking for convergence.

3. **Does not depend on the norm for smoothness condition:** The Frank-Wolfe method can be used even when the function is L-smooth in any arbitrary norm

$$\|\nabla f(x) - \nabla f(y)\|_* \leq L\|x - y\|\,,$$

where $\|\cdot\|$ is any arbitrary norm and $\|\cdot\|_*$ is the dual norm.

3.2 Example Application

Consider a LASSO problem in which we have N dictionary elements $d_1, \cdots, d_N \in \mathbb{R}^n$ and a signal $Z \in \mathbb{R}^n$ for some n. Consider the case when N is exponentially large. The problem is to represent Z as a sparse linear combination of the dictionary elements. More formally, we need to find a sparse $x \in \mathbb{R}^N$, such that

$$\min_{x \in \mathbb{R}^N} \|Z - \sum_{i=1}^{N} x_i d_i\|_2^2 \tag{11}$$

$$\text{s.t. } \|x\|_1 \leq s \tag{12}$$

Letting $D = [d_1, \cdots, d_N] \in \mathbb{R}^{n \times N}$ we can rewrite the above as

$$\min_{x \in \mathbb{R}^N} \|Z - Dx\|_2^2 \tag{13}$$

$$\text{s.t. } \|x\|_1 \leq s \tag{14}$$

At first, the problem seems intractable because it will take exponential time even to write the vector x. But this problem can be solved in polynomial time using the Frank-Wolfe method if the following problem can be solved in polynomial time for any vector y

$$\min_{i \in [N]} d_i^\top y \,. \tag{15}$$

Calculating $\nabla f(x_k) = D^\top (Dx_k - Z)$, and letting $z_k = Dx_k - Z$. The linear programming problem

$$\min_{x \in \mathbb{R}^N} y^\top D^\top z_k \tag{16}$$

$$\text{s.t. } \|y\|_1 \leq s \tag{17}$$

is equivalent to finding the largest component in the vector $|D^\top z_k|$ which can be solved in polynomial time using Equation 15. Due to a property of linear programs, the vector y_k will have just one non-zero component. And therefore, x_{k+1} will have atmost $k+1$ non-zero components. Due to this, the Frank-Wolfe updates can be made in polynomial time.

3.3 Convergence Analysis

The Frank-Wolfe method can be shown to have $O(1/k)$ convergence when the function f is L-smooth is any arbitrary norm.

Theorem 3.1. *Let the function f be convex and L-smooth w.r.t any arbitrary norm $\|\cdot\|$, $R = \sup_{x,y \in C} \|x - y\|$, and $\gamma_k = \frac{2}{k+1}$ for $k \geq 1$, then*

$$f(x_k) - f(x^*) \leq \frac{2LR^2}{k+1} \,.$$

References

[Bub14] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980*, 2014.