10-725/36-725: Convex Optimization **Fall 2018**

Lecture 6: September 17

Lecturer: Lecturer: Ryan Tibshirani *Scribes: Heejong Bong, Wanshan Li, Shamindra Shrotriya*

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

6.1 Convergence Analysis of Optimization Algorithms

In our previous class we learned the following theorem for f is differentiable with Lipschitz and nonconvex.

Theorem (Gradient Descent with fixed step size)**.** *Gradient Descent with fixed step size $t \leq \frac{1}{L}$ results satisfies*

$$\min_{i=0,1,2,\ldots k} \|\nabla f(x^{(i)})\|_2 \leq \sqrt{\frac{2(f(x^{(k)}) - f(x^{(0)})}{t(k+1)}}$$

Thus gradient descent has rate $O(\frac{1}{\sqrt{k}})$, or $O(\frac{1}{\epsilon^2})$, even in the nonconvex case for finding stationary points

Proof (Sketch). From the lectures the key steps for the proof are as follows:

1. ∇f Lipschitz with constant L means:

$$f(y) \leq f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|_2^2 \text{ all } x, y$$

2. Plugging in $y = x^+ = x - t\nabla f(x)$,

$$f\left(x^+\right) \leq f(x) - \left(1 - \frac{Lt}{2}\right) t\|\nabla f(x)\|_2^2$$

3. Taking $0 < t \leq 1/L$, and rearranging

$$\|\nabla f(x)\|_2^2 \leq \frac{2}{t}\left(f(x) - f\left(x^+\right)\right)$$

4. Summing over iterations

$$\sum_{i=0}^{k} \left\|\nabla f\left(x^{(i)}\right)\right\|_2^2 \leq \frac{2}{t}\left(f\left(x^{(0)}\right) - f\left(x^{(k+1)}\right)\right) \leq \frac{2}{t}\left(f\left(x^{(0)}\right) - f^\star\right)$$

5. Lower bound sum by $(k+1)\min_{i=0,\ldots,k}\left\|\nabla f\left(x^{(i)}\right)\right\|_2^2$

□

6.1.1 Anatomy of Convergence Rate Proof

The proof for the previous theorem follows the following general construction (framework):

Step 1. Start with some quadratic upper and lower bound on $f(y)$ and $f(x)$ (current iteration)

- **Lipschitz gradient:** $\implies$ we can upper bound $f(y)$
- **Strong convexity:** $\implies$ we can lower bound $f(y)$

Step 2. Establish some 'sufficient descent' property of $f(x^+)$ (next iteration)

- e.g. in **gradient descent** with $t \leq \frac{1}{L}$ we have $f(x^+) \leq f(x) - \frac{t}{2}\|\Delta f(x)\|_2^2$

Step 3. Iterate/ recurse the above process to get a global statement about $f(x^*)$ **or** x^*

In the non-convex case above we leveraged Step 3 to get a property with a stationary point. Note that not every stationary point is useful or interesting (e.g. local minima) and stationarity in itself is a pretty weak property. Most of the time it is sufficient and most cases settle in a local minima.

6.2 Gradient Boosting

6.2.1 Gradient Boosting - Problem Setup

Suppose that we are given responses $y_i \in \mathbb{R}$ and features $x_i \in \mathbb{R}^p \quad \forall i \in \{1, 2, \ldots, n\}$ We want to construct a flexible (*nonlinear*) model for response based on features. Weighted sum of trees:

$$u_i = \sum_{j=1}^{m} \beta_j \cdot T_j(x_i) \quad \forall i \in \{1, 2, \ldots, n\}$$

Each tree T_j inputs x_i and outputs a predicted response. Typically trees are pretty short.

We have flexibility to pick a suitable loss function L to reflect setting e.g for continuous responses we can take the squared error loss function $L(y_i, u_i) = (y_i u_i)^2$. We then want to ideally solve

$$\min_{\beta} \sum L\left(y_i, \sum_{j=1}^{M} \beta_j \cdot T_j(x_i)\right)$$

Indexes all trees of a fixed size (e.g., depth = 5), so M is huge. Space is simply too big to optimize!

6.2.2 Gradient Boosting - Procedure

Key idea: Here we essentially do gradient descent on some loss function where we map gradients to vector of projections by a tree. This essentially forms a big cluster of weighted summed trees.

More specifically we have the following procedure:

First think of optimization as $\min_u f(u)$, over predicted values u, subject to u coming from trees. Start with initial model, a single tree $u^{(0)} = T_0$ and then repeat the following:

- Compute negative gradient d at latest prediction $u^{(k-1)}$

$$d_i = -\left[\frac{\partial L(y_i, u_i)}{\partial u_i}\right]\Bigg|_{u_i = u_i^{(k-1)}} \quad \forall i \in \{1, 2, \ldots, n\}$$

- Find a tree T_k that is close to a, i.e., according to

$$\min_{\text{trees } T} (d_i - T(x_i))^2$$

  This is not hard to (approximately) solve for a single tree

- Compute step size α_k, and update our prediction:

$$u^{(k)} = u^{(k-1)} + \alpha_k T_k$$

  Note: predictions are weighted sums of trees, as desired

6.3 Subgradients

Key idea: Subgradients are a generalization of gradients. Our focus is mainly non-differentiable convex functions and subgradients provide a fairly general concept of optimality.

Definition (Subgradient)**.** *The subgradient of a convex function f at $x \in \text{dom}(f)$ is:*

$$g \in \mathbb{R}^n \, s.t. \; f(y) \geq f(x) + g^T(y - x) \quad \forall y$$

6.3.1 Useful facts

1. The subgradient of f always exists, for points that are in the relative interior of $\text{dom}(f)$.
2. If f is differentiable at x, then subgradient $g = \nabla f(x)$ uniquely.
3. For nonconvex function, the same defnition works, but the subgradient need not exist.

6.3.2 Examples of Subgradients

The following are some useful examples of subgradients.

1. $f : \mathbb{R} \to \mathbb{R} : x \mapsto |x|$.

$$\begin{cases} \text{Unique subgradient } g = \text{sign}(x), \text{ for } x \neq 0, \\ g \text{ is any element of } [-1, 1], \text{ for } x = 0. \end{cases}$$

2. $f : \mathbb{R}^n \to \mathbb{R} : x \mapsto \|x\|_2$.

$$\begin{cases} \text{Unique subgradient } g = x/\|x\|_2, \text{ for } x \neq 0, \\ g \text{ is any element of } \{z : \|z\|_2 \leq 1\}, \text{ for } x = 0. \end{cases}$$

3. $f : \mathbb{R}^n \to \mathbb{R} : x \mapsto \|x\|_1$. For $1 \le i \le n$

$$\begin{cases} \text{Unique subgradient } g_i = \text{sign}(x), \text{ for } x_i \neq 0, \\ g_i \text{ is any element of } [-1, 1], \text{ for } x_i = 0. \end{cases}$$

4. $f = \max\{f_1, f_2\}$, where $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ are convex and differentiable.

$$\begin{cases} g = \nabla f_1(x), \ f_1(x) > f_2(x), \\ g = \nabla f_2(x), \ f_2(x) > f_1(x), \\ g \text{ is any point on line segment between } \nabla f_1(x) \text{ and } \nabla f_2(x), \ f_1(x) = f_2(x). \end{cases}$$

To see this, notice that when $f_1(x) > f_2(x)$, f acts like f_1 in a neighborhood of x, because convex function is continuous in the interior of its domain.

6.4 Subdifferentials

Definition (Subdifferential)**.** *The subdifferential of a convex function f at $x \in \text{dom}(f)$ is*

$$\partial f(x) = \{g \in \mathbb{R}^n : \ g \text{ is a subgradient of } f \text{ at } x\}.$$

6.4.1 Useful facts

1. The subdifferential is nonempty for convex function f (again, in the relative interior of $\text{dom}(f)$).
2. $\partial f(x)$ is closed and convex, even for nonconvex function f.
3. If f is differentiable at x and $\nabla f(x) = g$, then $\partial f(x) = \{g\}$.
4. If $\partial f(x) = \{g\}$, then f is differentiable at x and $\nabla f(x) = g$.

Proposition (Connection to convex geometry)**.** *Suppose C is a convex set in $\mathbb{R}^n$. Then, for $x \in C$, $\partial I_C(x) = N_C(x)$, where*

$$I_C(x) = \begin{cases} 0, \ x \in C, \\ \infty, \ x \notin C, \end{cases}$$

and

$$N_C(x) = \{g \in \mathbb{R}^n : g^\top x \ge g^\top y \text{ for any } y \in C\}.$$

Recall that $N_C(x)$ is called normal cone of set C at point x.

Proof. By definition, g is the subgradient at x if and onlly if,

$$I_C(y) \ge I_C(x) + g^\top (y - x), \ \forall y.$$

For $y \notin C$, $I_C(y) = \infty$ and the inequality holds, while for $y \in C$, $I_C(y) = 0$ and the inequality is equivalent to

$$0 \ge g^\top (y - x).$$

□

Proposition (Subgradient calculus)**.** *Some useful facts of subgradient calculus:*

- *Scaling:* $\partial(af) = a \cdot \partial f$ *provided* $a > 0$ *(to make* af *convex).*
- *Addition:* $\partial(f_1 + f_2) = \partial f_1 + \partial f_2$.
- *Affine composition: if* $g(x) = f(Ax + b)$*, then* $\partial g(x) = A^\top \partial f(Ax + b)$.
- *Finite pointwise maximum:* $f(x) = \max_{1 \le i \le n} f_i(x)$*, then*

$$\partial f(x) = \operatorname{conv}\left(\bigcup_{i: f_i(x) = f(x)} \partial f_i(x)\right),$$

  i.e., convex hull of the union of subdifferentials of active functions at x.

- *General poinwise maximum:* $f(x) = \max_{s \in S} f_s(x)$*, then*

$$\partial f(x) \supseteq \mathrm{Cl}\{\operatorname{conv}\left(\bigcup_{s: f_s(x) = f(x)} \partial f_s(x)\right)\},$$

  where Cl(C) means closure of C, and one should notice that $\operatorname{conv}\left(\bigcup_{s: f_s(x) = f(x)} \partial f_s(x)\right)$ *need not be closed if* S *is infinite. Under some regularity conditions on* S *and* f_s*, e.g.,* f_s *is continuous for all* s*, we can get equality.*

- *Norms:* $f(x) = \|x\|_p$*,* x *need not in a finite dimension, let* q *be such that* $1/p + 1/q = 1$*, then*

$$\|x\|_p = \max_{z: \|z\|_1 \le 1} z^\top x.$$

  Hence, $\partial f(x) = \arg\max_{z: \|z\|_1 \le 1} z^\top x$.

6.4.2 Importance of subgradient

- Convex analysis: KKT condition $\rightarrow$ optimality characterization; monotonicity; relationship to duality.
- Convex optimization: if you can compute subgradient, then you can minimize any convex functions.

6.5 Optimality conditions

Here we note some optimality criteria involving subgradients with a particular focus on convex functions.

Proposition (subgradient optimality condition)**.** *For any function* f*,*

$$f(x^*) = \min_x f(x) \iff 0 \in \partial f(x^*)$$

while the latter condition is called the subgradient optimality condition.

Note the implication for a convex and differentiable function f:

Proposition (first-order optimality condition)**.** *For any convex and differentiable function f and a convex set C, x^* is a solution for the optimization problem,*

$$\min_x f(x) \textit{subject tox} \in C,$$

if and only if

$$\nabla f(x)^T(y-x) \geq 0, \forall y \in C.$$

Proof. The given optimization problem is equivalent with

$$\min_x f(x) + I_C(x)$$

where I_C is an indicator function for the set, C.

The subgradient optimality applied,

$$\begin{aligned} 0 \in (\partial f(x) + I_C(x)) &\iff 0 \in \nabla f(x) + N_C(x) \\ &\iff -\nabla f(x) \in N_C(x) \\ &\iff -\nabla f(x)^T x \geq -\nabla f(x)^T y, \forall y \in C \\ &\iff \nabla f(x)^T(y-x) \geq 0, \forall y \in C \end{aligned}$$

where N_C is the normal cone of the set C □

Example (lasso optimality conditions)**.** Consider the following lasso problem:

$$\min_\beta \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

The subgradient optimality condition is given as follows:

$$\begin{aligned} 0 \in \partial(\frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1) &\iff 0 \in -X^T(y - X\beta) + \lambda\partial\|\beta\|_1 \\ &\iff X^T(y - X\beta) = \lambda v, \text{for some } v \in \partial\|\beta\|_1 \\ &\iff \begin{cases} X_i^T(y - X\beta) = \lambda \text{ sign}(\beta_i) \text{ if } \beta_i \neq 0, \\ |X_i^T(y - X\beta)| \leq \lambda \text{ if } \beta_i = 0, \end{cases} \end{aligned}$$

where

$$[\partial\|\beta\|_1]_i = \begin{cases} \{1\}, \beta_i > 0, \\ \{-1\}, \beta_i < 0, \\ [-1, 1], \beta_i = 0. \end{cases}$$

It does not only give a way to check lasso optimality, but also it is helpful in understanding the lasso estimator.

Example (soft-thresholding)**.** Consider a simplified lasso problem with $X = I$:

$$\min_\beta \frac{1}{2}\|y - \beta\|_2^2 + \lambda\|\beta\|_1.$$

A solution for this problem is given as

$$\beta = S_\lambda(y)$$

where S_λ is a soft-threshold operator such that

$$[S_\lambda(y)]_i = \begin{cases} y_i - \lambda \text{ if } y_i > \lambda, \\ y_i + \lambda \text{ if } y_i < -\lambda, \\ 0 \text{ otherwise.} \end{cases}$$

Check the subgradient optimality of this solution:

1. If $y_i > \lambda$, $\beta_i = y_i - \lambda > 0$, and hence $y_i - \beta_i = \lambda = \lambda \text{ sign}(\beta_i)$,
2. Similarly, it satisfies subgradient optimality condition if $y_i < -\lambda$,
3. If $|y_i| < \lambda$, $\beta_i = 0$, and hence $|y_i - \beta_i| = |y_i| < \lambda$.

Conversely, soft-thresholding can be derived by reverse engineer this argument.

Example (distance to a convex set)**.** For a convex set C,

$$\text{dist}(x, C) = \min_{y \in C} \|y - x\|_2.$$

Then,

$$\partial \text{dist}(x, C) = \{\frac{x - P_C(x)}{\|x - P_C(x)\|_2}\}$$

where $P_C(x)$ is a projection of x onto C. Hence, $\text{dist}(x, C)$ is differentiable and this is its gradient.