10-725: Convex Optimization **Spring 2023**

Lecture 7: February 7th

Lecturer: Siva Balakrishnan

Our first goal for today will be to try to better understand the prox. algorithm we defined in the previous lecture. Then we will turn our attention to another popular example of proximal GD.

7.1 Some Properties of the Proximal Operator

Recall, for a function h,

$$\text{prox}_h(x) = \arg\min_u \frac{1}{2}\|x - u\|_2^2 + h(u).$$

The first fact that we will show is that (like a projection) the prox operation is a contraction, i.e.

Lemma 7.1 *For a convex function h,*

$$\|prox_h(x) - prox_h(y)\|_2 \leq \|x - y\|_2.$$

Proof: We'll prove the stronger claim that,

$$\|\text{prox}_h(x) - \text{prox}_h(y)\|_2^2 \leq \langle x - y, \text{prox}_h(x) - \text{prox}_h(y)\rangle,$$

from which the original claim follows (by applying the Cauchy-Schwarz inequality to the RHS).

To see this we simply use subgradient optimality conditions. Let $u = \text{prox}_h(x)$, $v = \text{prox}_h(y)$, then we know that by subgradient optimality conditions,

$$x - u \in \partial h(u),$$
$$y - v \in \partial h(v).$$

By monotonicity of the gradient of h we know that for any elements (a, b) of the subdifferential of h at u, v,

$$\langle a - b, u - v\rangle \geq 0.$$

Applying this to the two vectors above we see that,

$$\langle x - u - y + v, u - v \rangle \geq 0,$$

from which the stronger claim (and hence the lemma) follows. ■

Another important property of the prox. update is that it only has one fixed point, i.e. only if we're already at an optimal point will the prox. algorithm stop. Before we can show this, we'll need to define an important quantity which is the *gradient mapping.* We'd like to analyze prox. GD but it differs from GD in that it's a bit difficult to make sense of the prox. GD updates. Define,

$$G_\eta(x) = \frac{1}{\eta}\left[x - \text{prox}_{\eta h}(x - \eta \nabla g(x))\right].$$

This might seem quite unintuitive a quantity (it is), but it allows us to write,

$$x^{t+1} = x^t - \eta_t G_{\eta_t}(x^t).$$

This in turn makes prox. GD updates look a bit more familiar. Provided that we can get some handle on the gradient mapping we might then be able to analyze prox. GD in a similar fashion to how we analyzed GD.

Now, lets use the gradient mapping to argue that the only prox. GD fixed points are optimal solutions to our original program.

Lemma 7.2

$$G_\eta(x^*) = 0 \iff 0 \in \nabla g(x^*) + \partial h(x^*)$$

Proof: For any point $\widetilde{x}$, we know that,

$$G_\eta(\widetilde{x}) = \frac{1}{\eta}\left[\widetilde{x} - \text{prox}_{\eta h}(\widetilde{x} - \eta \nabla g(\widetilde{x}))\right],$$

and this in turn (by optimality conditions for prox) means that,

$$\widetilde{x} - \eta \nabla g(\widetilde{x}) - \widetilde{x} + \eta G_\eta(\widetilde{x}) \in \eta \partial h(\widetilde{x} - \eta G_\eta(\widetilde{x})),$$

i.e. for any point $\widetilde{x}$ we must have that,

$$G_\eta(\widetilde{x}) \in \nabla g(\widetilde{x}) + \partial h(\widetilde{x} - \eta G_\eta(\widetilde{x})). \tag{7.1}$$

It is worth thinking about this equation for a bit and noticing how it differs from a usual sub-gradient, i.e. $v_{\widetilde{x}}$ would be a valid subgradient if it was in the collection $\nabla g(\widetilde{x}) + \partial h(\widetilde{x})$, but the proximal gradient mapping satisfies a slightly different condition.

Now we see that if $G_\eta(x^*) = 0$ for some x^* then x^* must satisfy,

$$0 \in \nabla g(x^*) + \partial h(x^*).$$

Conversely, if $0 \in \nabla g(x^*) + \partial h(x^*)$, then we have that,

$$x^* - \eta \nabla g(x^*) - x^* \in \eta \partial h(x^*),$$

which in turn means that,

$$x^* = \mathrm{prox}_{\eta h}(x^* - \eta \nabla g(x^*)).$$

This final expression yields that, $G_\eta(x^*) = 0$ as desired. ■

7.1.1 Main Descent Lemma

Our eventual goal will be to analyze the prox algorithm when g is β-smooth, and h is convex (you will do another case in HW3?). The main technical hurdle will be to prove a descent lemma which works when we replace gradients by generalized gradients.

Lemma 7.3 *For any $\eta \leq 1/\beta$, and any z*

$$f(x - \eta G_\eta(x)) \leq f(z) + G_\eta(x)^T(x - z) - \frac{\eta}{2}\|G_\eta(x)\|_2^2.$$

Once we prove this lemma the analysis will mirror the smooth case, but notice that this is a very non-obvious statement since f is not smooth, and since G_η is not a gradient map (or even a valid subgradient).

It is a generalization of the descent lemma. If we plug in $z = x$, then we recover the more familiar expression:

$$f(x - \eta G_\eta(x)) \leq f(x) - \frac{\eta}{2}\|G_\eta(x)\|_2^2.$$

This justifies referring to this algorithm as a descent algorithm (at least for appropriate step-size choices the function value reduces in each iteration).

Proof: We begin by observing,

$$\begin{aligned}
f(x - \eta G_\eta(x)) &= g(x - \eta G_\eta(x)) + h(x - \eta G_\eta(x)) \\
&\leq g(x) - \eta \nabla g(x)^T G_\eta(x) + \frac{\eta^2 \beta}{2}\|G_\eta(x)\|_2^2 + h(x - \eta G_\eta(x)) \\
&\leq g(z) + \nabla g(x)^T(x - z) - \eta \nabla g(x)^T G_\eta(x) + \frac{\eta^2 \beta}{2}\|G_\eta(x)\|_2^2 + h(x - \eta G_\eta(x))
\end{aligned}$$

just by using smoothness on the β-smooth function h. Now, just by convexity of h we know that,

$$h(x - \eta G_\eta(x)) \leq h(z) - \theta^T(z - x + \eta G_\eta(x)),$$

where θ is any element of $\partial h(x - \eta G_\eta(x))$. In (7.1) we showed that, $G_\eta(x) - \nabla g(x) \in h(x - \eta G_\eta(x))$, so we obtain the bound,

$$h(x - \eta G_\eta(x)) \leq h(z) - \eta \|G_\eta(x)\|_2^2 + \eta \nabla g(x)^T G_\eta(x) - (G_\eta(x) - \nabla g(x))^T (z - x).$$

Putting the pieces together we obtain the desired claim. ■

7.1.2 Analyzing the Prox. Algorithm

With the descent lemma in place its straightforward to analyze prox. GD when applied to β-smooth g (and convex h).

Theorem 7.4 *For β-smooth g, convex h the prox. GD algorithm with step-size $\eta = 1/\beta$ achieves the following guarantee:*

$$f(x^k) - f(x^*) \leq \frac{\beta \|x^0 - x^*\|_2^2}{2k}.$$

Proof: As usual we begin with the familiar manipulations,

$$\|x^{t+1} - x^*\|_2^2 = \|x^t - x^*\|_2^2 + 2\eta \left(\frac{\eta}{2} \|G_\eta(x^t)\|_2^2 - (x^t - x^*)^T G_\eta(x^t) \right).$$

Our lemma (applied with $z = x^*$) gives an upper bound on the last term, which in turn yields:

$$\|x^{t+1} - x^*\|_2^2 \leq \|x^t - x^*\|_2^2 + 2\eta (f(x^*) - f(x^{t+1})).$$

Re-arranging and telescoping we obtain the final result. ■

What should be surprising is that we've attained a $1/k$ rate of convergence for a class of non-smooth functions (i.e. f is certainly not smooth). As always though, we're exploiting a very particular type of structure (that arises often in regularized loss minimization problems). We also no longer operate in the first-order model (and are not bound by the oracle lower bounds discussed earlier).

7.1.3 Descent Without Smoothness

For some more intuition about why the proximal algorithm is so powerful – consider the case when $g = 0$, i.e. we're just doing proximal minimization of a potentially non-smooth function h. Since the function h is not smooth, we could apply the subgradient method, but the crucial observation we belaboured was that an arbitrary subgradient would in general not be a descent direction, and we could never hope to prove an analogue of our "main descent lemma" (which we proved for GD under smoothness).

Now, lets try to understand the proximal minimization steps (this is really a special case of the facts we proved above setting $g = 0$ but the magic is much clearer). We simply iterate, for any fixed $\eta > 0$,

$$x^{t+1} = \mathrm{prox}_{\eta h}(x^t) := \arg\min_x \frac{1}{2}\|x - x^t\|_2^2 + \eta h(x).$$

Of course, we could take η very large, and this would converge to a minimizer in one step (but that problem is as hard to solve as the minimization of h directly). In practice, one might hope taking a smaller value of η might yield easier to solve sub-problems which still converge to a minimizer of h.

We define the gradient mapping $G_\eta(x)$ as before (though its expression is a bit simpler):

$$G_\eta(x) = \frac{1}{\eta}\left(x - \mathrm{prox}_{\eta h}(x)\right).$$

Lemma 7.5 (Descent Without Smoothness) *For h convex, we have the following guarantees,*

$$\begin{aligned} h(x^{t+1}) &\leq h(x^t) - \eta\|G_\eta(x^t)\|_2^2, \\ h(x^{t+1}) &\leq h(x^*) - \eta\|G_\eta(x^t)\|_2^2 + G_\eta(x^t)^T(x^t - x^*). \end{aligned}$$

It is worth appreciating this result, since it highlights the key idea of the prox. method. Without any smoothness assumptions whatsoever we obtain a descent lemma. The key caveat being that it's no longer an "explicit" step, rather we solve a minimization for each step (taking a so-called "implicit" step).

Proof: We know that for any z, any $u \in \partial h(x^{t+1})$,

$$h(x^{t+1}) \leq h(z) - \eta u^T(z - x^{t+1}).$$

In our setting we know that, $G_\eta(x^t) \in \partial h(x^{t+1})$. Taking $z = x^t$ we recover the first claim, and taking $z = x^*$ we recover the second claim. ■

Theorem 7.6 *After k iterations the proximal method, for convex h, achieves the guarantee:*

$$h(x^k) - h(x^*) \leq \frac{\|x^0 - x^*\|_2^2}{2\eta k}.$$

Proof: As usual,

$$\begin{aligned}\|x^{t+1} - x^*\|_2^2 &= \|x^t - x^*\|_2^2 + \eta^2 \|G_\eta(x^t)\|_2^2 - 2\eta G_\eta(x^t)^T(x^t - x^*) \\ &\leq \|x^t - x^*\|_2^2 + 2\eta(h(x^{t+1}) - h(x^*)),\end{aligned}$$

just by applying the second claim of the descent lemma. Re-arranging and summing we obtain that,

$$h(x^k) - h(x^*) \leq \frac{\|x^0 - x^*\|_2^2}{2\eta k}.$$

■

7.2 Another example – Matrix Completion

Another nice example of the proximal GD algorithm comes from the problem of matrix completion. In matrix completion, we observe some subset of indices of a matrix M^* (possibly with some noise), and we would like to "fill in" the matrix. For some subset of indices $\mathcal{I}$, we observe $\{Y_{ij} : (i,j) \in \mathcal{I}\}$.

Of course there are many possible ways to do this, and a typical assumption is that M^* has low rank, and so we'd like to find a matrix that is close to Y on the observed indices but has small rank. The rank of a matrix is a non-convex function, and so we'll instead choose to use a convex relaxation of the rank, which is called the trace norm or nuclear norm.

For a matrix $M \in \mathbb{R}^{n \times d}$ if we write its SVD as $M = U\Sigma V^T$ then the nuclear norm is simply $\|M\|_{\text{tr}} = \sum_{i=1}^d \sigma_i(M)$, i.e. the sum of its singular values.

With all of this background, we'd like to minimize the following objective:

$$\min_M \sum_{(i,j)\in\mathcal{I}} (Y_{ij} - M_{ij})^2 + \lambda \|M\|_{\text{tr}}.$$

This is a convex objective, but just like the LASSO the regularizer is not smooth.

We can still hope for a fast (at least compared to the subgradient method) algorithm if we can compute the proximal operator corresponding to the regularizer. You will show in your HW that the program,

$$\min_M \frac{1}{2}\|Y - M\|_F^2 + \lambda \|M\|_{\text{tr}},$$

has a closed form solution. The optimal M comes from soft-thresholding the singular values of Y, i.e. $\text{prox}_{\lambda h}(Y) = U\Sigma_\lambda V^T$, where $Y = U\Sigma V^T$ is the SVD of Y and $\Sigma_\lambda = \max\{0, \Sigma - \lambda\}$ which just subtracts λ from every singular value of Y (and stops if it hits 0).

Now, we have a simple proximal GD algorithm to optimize our original objective:

1. We compute $Z^{t+1} = X^t - \eta(X^t - Y) \circ \Omega$, where $\Omega_{ij} = 1$ if $(i, j) \in \mathbb{I}$ and 0 otherwise.
2. We then compute $X^{t+1} = \text{prox}_{\lambda h}(Z^{t+1})$.

You can take the step-size $\eta = 1$ since the objective is 1-smooth, and then this algorithm is known as soft-impute and is one of the faster ways to solve the matrix completion problem.