10-425/625: Introduction to Convex Optimization (Fall 2023)

Lecture 11: Projected Gradient Descent

Instructor:[1] *Matt Gormley* *October 2, 2023*

Today our focus will be first on constrained optimization. We will start with a general projection problem. Then we will introduce the projected gradient descent algorithm.

11.1 Optimality Conditions

Recall from a previous lecture

We are interested in solving a problem:

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

where f is a convex function, and C is a convex set. What can I say about a solution x^* to this problem?

1. **Unconstrained Case:** Suppose first that $C = \mathbb{R}^d$, and that $\text{dom}(f) = \mathbb{R}^d$ then our characterization should be familiar to us from usual calculus classes.

   Theorem 11.1. *x^* is optimal, if (and only if) $0 \in \partial f(x^*)$.*

2. **Constrained, Differentiable Case:** Now suppose that $C \subset \mathbb{R}^d$ and we wish to solve the constrained optimization problem: $\min_{x \in C} f(x)$.

   Theorem 11.2. *A feasible point x^* is optimal, if and only if $\nabla f(x^*)^T (y - x^*) \geq 0$ for all $y \in C$.*

   If you recall the definition of the normal cone, then you will see that this condition says that,

   $$-\nabla f(x^*) \in N_C(x^*).$$

[1] These notes were originally written by Siva Balakrishnan for 10-725 Spring 2023 (original version: here) and were edited and adapted for 10-425/625.

3. **General, Constrained Case:** Now we consider the case where the function is nondifferentiable.

   Theorem 11.3. *A feasible point x^* is optimal, if and only if $0 \in \partial f(x^*) + N_C(x^*)$.*

   Here we are adding two sets, i.e. $C+D = \{y : y = u+v, u \in C, v \in D\}$.

11.1.1 Optimality Conditions for Projection

Here is a very basic/important problem. It arises in signal processing and statistics as a basic denoising scheme. For some convex set K, and observation y we would like to solve the constrained minimization problem,

$$\min_{x \in K} \frac{1}{2}\|y - x\|^2.$$

This finds the closest point in K to y, and is called the *projection* of y onto K. We will denote the solution x^* to the above program by $P_K(y)$.

Let us first write out the optimality conditions, and then use them to show a nice property of this projection operation. Since f is differentiable, $\nabla f(x^*) = x^* - y$ and we have that,

$$\begin{aligned} -(x^* - y) &\in N_C(x^*) \\ \Rightarrow 0 &\in x^* - y + N_C(x^*). \end{aligned}$$

Equivalently, this means that, $(y - x^*)^T(a - x^*) \leq 0$ for all $a \in K$. This can be easily understood with a picture.

Theorem 11.4. *Projection onto a convex set is a contraction, i.e. for any pair of points a, b,*

$$\|P_K(a) - P_K(b)\| \leq \|a - b\|.$$

That is the distance between the projections must be less than or equal to the distance between the original points (i.e. a contraction of the distance). **Proof:** From the optimality conditions we have that for any $x \in K$,

$$\begin{aligned} (a - P_K(a))^T(x - P_K(a)) &\leq 0 \\ (b - P_K(b))^T(x - P_K(b)) &\leq 0. \end{aligned}$$

As a consequence we can see that,

$$(a - P_K(a))^T(P_K(b) - P_K(a)) \leq 0$$
$$(b - P_K(b))^T(P_K(a) - P_K(b)) \leq 0.$$

Adding these inequalities we obtain that,

$$(b - a + (P_K(a) - P_K(b)))^T(P_K(a) - P_K(b)) \leq 0.$$

Now, re-arranging and applying the Cauchy-Schwarz inequality, we see that,

$$\|P_K(a) - P_K(b)\|^2 \leq (a - b)^T(P_K(a) - P_K(b)) \leq \|a - b\|\|P_K(a) - P_K(b)\|,$$

which is our desired conclusion. ■

Background: (Cauchy-Schwarz Inequality) For any pair of vectors $u, v \in \mathbb{R}^n$, we have:

$$|u^T v| \leq \|u\|_2 \|v\|_2$$

11.2 Projected Gradient Descent

Suppose our goal is to now minimize a convex function f, over a convex set C, i.e. we want to solve:

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

A natural first attempt might be to ignore the constraint and run gradient descent (or the subgradient method), but of course this won't solve our constrained problem in general. Since our iterates could leave the set C, we could end up at an infeasible solution.

A second natural attempt might be to run gradient descent, but choose the step size so that you never step outside the constrained region, but this won't converge to the minimizer of the constrained problem.

The natural third attempt might be to simply take steps of GD or the subgradient method, but then project iterates back to the set C whenever they

leave the set C. If the objective is differentiable this is the projected GD algorithm, i.e. for a step-size η, we iterate:

$$y^{t+1} = x^t - \eta \nabla f(x^t),$$
$$x^{t+1} = P_C(y^{t+1}).$$

Here the projection step finds the point in C closest to y^{t+1}, i.e. it solves the optimization problem:

$$x^{t+1} = P_C(y^{t+1}) := \arg\min_{x \in C} \frac{1}{2}\|x - y^{t+1}\|_2^2.$$

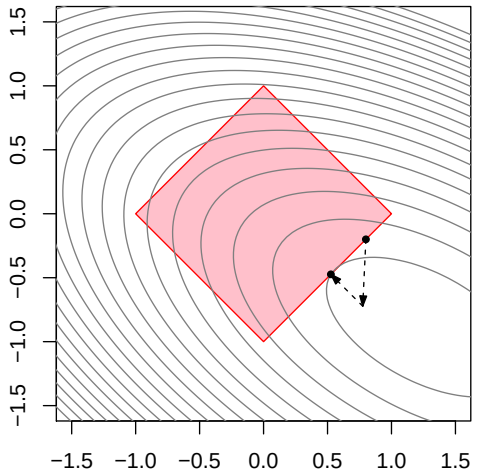

Figure 11.1: One step of projected gradient descent

Assuming this is a sensible method (it's not clear that it is) projected GD reduces solving one optimization problem to solving a sequence of optimization problems.

11.2.1 Efficient Projection onto a Set

Perhaps the immediate question is for what types of sets can we efficiently project onto. You'll explore some interesting examples in your HW.

Example 11.5 (Non-negative vectors)**.** Here is a very simple one: suppose we're solving an optimization problem but would like to constrain all the optimization variables to be positive. For instance, the problem of non-negative least squares is to solve:

$$\min_{x \geq 0} \frac{1}{2}\|Ax - b\|_2^2.$$

So the projected gradient descent algorithm alternates two steps:

$$y^{t+1} = x^t - \eta A^T(Ax^t - b),$$
$$x^{t+1} = P_C(y^{t+1}) \text{ where } C = \{\mathbf{x} \in \mathbb{R}^d : x_i \geq 0, \forall i\}.$$

Here in order to solve the projection problem we need to solve the following problem: given a vector y^{t+1}, find x^{t+1} which has all positive entries which is closest to y^{t+1}.

$$x^{t+1} = P_C(y^{t+1}) := \arg\min_{x \geq 0} \frac{1}{2}\|x - y^{t+1}\|_2^2.$$

This has a simple closed form:

$$x^{t+1} = \max\{0, y^{t+1}\},$$

where the max operation is applied elementwise. This is a simple example where projected GD is straightforward to implement, but it's still unclear if the algorithm inherits any of the properties of GD (in the unconstrained case).

Example 11.6 (The unit ball)**.** A d-dimensional unit ball $\mathcal{B}$ is a set in $\mathbb{R}^d$, defined as the set of points $x \in \mathbb{R}^d$ satisfying the following constraints:

$$\mathcal{B} = \{x \in \mathbb{R}^d : \|x\|_2^2 \leq 1\}$$

The projection can be efficiently computed as,

$$P_{\mathcal{B}}(x) = \begin{cases} x & \text{if } \|x\|_2 \leq 1; \\ \frac{x}{\|x\|_2} & \text{if } \|x\|_2 > 1. \end{cases}$$

A picture may help to convince you this is correct.

(How would you prove this is correct? For the sake of contradiction, you would assume when $\|x\|_2 > 1$, $\Pi_{\mathcal{B}}(x) = z \neq \frac{x}{\|x\|_2}$, then $z' = \langle z, \frac{x}{\|x\|_2}\rangle \widehat{x}$ where $\widehat{x} = \frac{x}{\|x\|_2} \in \mathcal{B}$. What would happen to this point z'? You can also prove this using the KKT conditions we'll consider later in the course.)

Example 11.7 (The probability simplex)**.** A probability distribution must sum-to-one and have all non-negative entries. The probability simplex in d dimensions is given by:

$$\mathcal{S} = \{x \in \mathbb{R}^d : x^T \mathbf{1} = \mathbf{1}, x \geq \mathbf{0}\}$$

We have that $x = P_{\mathcal{S}}(y)$ is a vector where $x_i = \max(0, y_i + \lambda)$ where λ is chosen such that $\sum_{i=1}^{d} x_i = 1$.

This can be interpreted as a rigid shift of the elements of the original point y, where we keep only those elements who are positive after the shift and the rest become 0.

11.3 Analyzing the Projected Subgradient Method

We'll do an example analyzing the projected subgradient method, and you'll work out a couple of other cases (analyzing projected GD) on your HW.

Theorem 11.8. *Suppose that f is convex and G-Lipschitz, and define x^{best} to be the best iterate seen so far and choose step-size η_t in each round, then we have the guarantee:*

$$f(x^{best}) - f(x^*) \leq \frac{\|x^0 - x^*\|_2^2 + G^2 \sum_{t=0}^{k-1} \eta_t^2}{2 \sum_{t=0}^{k-1} \eta_t}.$$

This is exactly the same guarantee that we had before, and more generally projected methods often inherit properties of their unconstrained counterparts. The major caveat is that each iteration of the projected algorithm needs to solve a projection step.

Proof: The proof will almost exactly mirror our proof for the subgradient method. Lets first observe that,

$$\begin{aligned}
\|y^{t+1} - x^*\|_2^2 &= \|x^t - \eta_t g_{x^t} - x^*\|_2^2 \\
&= \|x^t - x^*\|_2^2 + \eta_t^2 \|g_{x^t}\|_2^2 - 2\eta_t g_{x^t}^T (x^t - x^*) \\
&\leq \|x^t - x^*\|_2^2 + \eta_t^2 G^2 + 2\eta_t (f(x^*) - f(x^t)),
\end{aligned}$$

where the last step uses convexity. Now, we can re-arrange, and sum from $t = 0, \ldots, k$ to obtain that,

$$\sum_{t=0}^{k} 2\eta_t(f(x^t) - f(x^*)) \leq \sum_{t=0}^{k} \eta_t^2 G^2 + \sum_{t=0}^{k} \left(\|x^t - x^*\|_2^2 - \|y^{t+1} - x^*\|_2^2 \right).$$

Now we arrive at our main problem, which is that the second term on the right no longer telescopes.

The key observation is something we proved earlier, the projection onto a convex set is a contraction, i.e. recall we showed that if we have two point x, y then it is always the case that,

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

Now, observe that $x^* \in C$ (so projecting it onto C gives us the same point) so we have the immediate consequence:

$$\|P_C(y^{t+1}) - x^*\|_2 \leq \|y^{t+1} - x^*\|_2.$$

where $P_C(y^{t+1})$ is just x^{t+1}. The important consequence of this inequality is that after projecting a point back into the set it will be closer to x^*.

Returning to our bound we then see that,

$$\begin{aligned}\sum_{t=0}^{k} 2\eta_t(f(x^t) - f(x^*)) &\leq \sum_{t=0}^{k} \eta_t^2 G^2 + \sum_{t=0}^{k} \left(\|x^t - x^*\|_2^2 - \|x^{t+1} - x^*\|_2^2 \right) \\ &\leq \sum_{t=0}^{k} \eta_t^2 G^2 + \|x^0 - x^*\|_2^2,\end{aligned}$$

and replacing $f(x^t)$ with $f(x^{\text{best}})$ and re-arranging gives us our main result. ■

In essence, the projection operation at each step only pushes us closer to the optimal point x^* which can only help the algorithm.

11.3.1 Observations

There are lots of interesting things about the projected gradient descent method (some of them should provoke you to think about the proximal

method we'll talk about next). Suppose I was interested in solving the constrained form of the LASSO, i.e.

$$\min_{\|x\|_1 \leq L} \frac{1}{2}\|Ax - b\|_2^2,$$

then provided I can project efficiently onto the L1 ball (you can), then we might expect (this is true) that the projected gradient descent will behave like gradient descent on the least squares objective. When $A^T A$ is well-conditioned (i.e. the objective is smooth and strongly convex) this means that projected GD will have a (fast) linear rate of convergence, $1/k$.

As you might know (and as we'll see more formally later on) minimizing this objective is very similar to solving the penalized form of the LASSO, i.e.

$$\min_{x \in \mathbb{R}^d} \frac{1}{2}\|Ax - b\|_2^2 + \lambda \|x\|_1,$$

where λ and L have some correspondence (i.e. roughly, large values of $L > 0$ correspond to small values of $\lambda > 0$).

However, this latter problem is an example of an (unconstrained) non-smooth optimization problem. Our best bet (so far) would be to use the subgradient method (which we know has a very slow $1/\sqrt{k}$ rate of convergence at least in the worst-case).

It's a bit mysterious why folding the non-smoothness into a constraint helps us, and this motivates us to ask if there is an analogous way of dealing with "nice" non-smooth objective functions. It is worth noting that the projected GD algorithm is no longer a simple first-order algorithm (i.e. it operates in a very different "oracle" model, since it is not just using function values and gradients) and so is not subject to the rates of convergence lower bounds we discussed before.