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

Lecture 3: January 24

Lecturer: Siva Balakrishnan

3.1 Gradient Descent

For the next couple of lectures we'll focus on a basic unconstrained optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x).$$

For most of today we'll also assume that f is differentiable everywhere. A classical method to solve such optimization problems is gradient descent, i.e. we initialize at some guess x^0 and execute iterations of the form,

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

for some choice of the step-size $\eta > 0$, and until we reach some stopping condition.

There are many ways to motivate this algorithm. One is to notice that if we were at a point x and moved in a direction v with step-size $\eta > 0$

$$f(x + \eta v) \geq f(x) + \eta v^T \nabla f(x).$$

So at the very least we'd like to ensure that the second term is negative, i.e. $v^T \nabla f(x) \leq 0$ (otherwise we're moving to a strictly worse point). Such directions (which make a larger than 90-degree angle with the gradient) are typically called *descent* directions (for f at x). It should be clear that the negative gradient always gives us a descent direction (and in some sense gives us one for which the term $v^T \nabla f(x)$ is most negative – amongst vectors v with a given norm).

3.1.1 Gradient Descent as Minimizing the Local Linear Approximation

A more interesting way to motivate GD (which will also be subsequently useful to motivate mirror descent, the proximal method and Newton's method) is to consider minimizing a linear approximation to our function (locally). A picture will be helpful.

With a picture in mind, we can view GD as solving the following local minimization problem:

$$x^{t+1} = \arg\min_{y \in \mathbb{R}^d} f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\eta} \|y - x^t\|_2^2,$$

where the second term behaves as a regularizer to ensure that (for small η) our update remains close to our current iterate x^t. This local optimization problem has a closed form solution (take the derivative and set it to 0), and this precisely gives us our familiar GD update:

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

3.2 Choosing the Step-Size

In practice, the most important choice to be made is that of the step-size. We'll see various theoretical rules/schedules that one might follow based on what we know about the objective function. Here are some natural possibilities:

1. **Fixed Step-Size:** Here we simply select a fixed step-size η and run the algorithm with that fixed step-size. An immediate problem that you will encounter (in practice) even for very benign problems is that if you select the step-size too large then GD can diverge, and if you select it too small it might take a very long time to converge.

   You will find pictures of this in the BV textbook, but here is a typical analytical example to keep in mind.

   (a) Suppose we have $f(x) = x^2/2$, initialize at $x^0 = 1$ we take our step size to be 3 (too large). Then the iterates will be $x^t = -2, 4, -8, \ldots$ (i.e. GD will diverge).

   (b) For the same function, initialization, if we take our step size to be 0.00001 then GD would take 10^5 steps to converge.

   (c) On the other hand if we picked the "correct" step-size of 1, we would converge in 1 step.

   In theory, we'd like to understand this issue better (i.e. what properties of a function make certain step-sizes "too big", "too small", or "correct"). The correct step-size in many cases may depend on properties of the function that we don't know. In practice, it will often be useful to have at our disposal a few different ways to tune the step-size (and some understanding of how we might diagnose issues with the step-size choice).

2. **Exact Line-Search:** Once we've committed to a direction (in GD this is the direction of the negative gradient), one might consider solving the following 1D optimization problem to determine the best step-size:

   $$\eta^t = \arg\min_{\widetilde{\eta} \geq 0} f(x^t - \widetilde{\eta} \nabla f(x^t)).$$

   It's often computationally cumbersome to solve this optimization problem exactly, so we resort to some approximation of this idea.

3. **Backtracking Line-Search:** The idea of backtracking line-search very roughly, is to try an aggressive (large) step-size, and reduce it by some factor if it's too big.

   Here is the algorithm: we pick two parameters $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$. At iteration t: initialize $\eta = 1$,

   (a) If $f(x^t - \eta \nabla f(x^t)) > f(x^t) - \alpha\eta\|\nabla f(x^t)\|_2^2$, then reduce $\eta := \beta \times \eta$ and go back to step (a).

   (b) Otherwise, take a step, i.e. set $x^{t+1} = x^t - \eta \nabla f(x^t)$.

   Often in practice, taking $\alpha = 0.3$ and $\beta = 0.5$ work reasonably well.

   You will develop some better intuition when we study the main descent lemma for GD, but roughly if your function is nice (the Hessian term in a Taylor series is ignorable), you should expect to make about $\eta\|\nabla f(x^t)\|_2^2$ amount of progress in one step of GD if η is small enough. The backtracking line search simply says if you're making upto an α factor of this amount of progress you should be content and take a step.

3.3 Two Canonical Examples

It is worth studying gradient descent in two simple analytical examples to understand the type of behavior we might expect.

1. Suppose we are solving a least squares problem:

$$\min \frac{1}{2}\|Ax - b\|_2^2,$$

   where $S := A^T A$ has finite condition number, i.e.

$$\kappa(S) = \frac{\lambda_{\max}(S)}{\lambda_{\min}(S)} < \infty.$$

   This is equivalent to saying our problem is both smooth and strongly convex (the most favorable case for GD).

   Here we know the solution in closed form:

$$\widehat{x} = (A^T A)^{-1} A^T b,$$

   and in particular we can write $\widehat{x}$ as the (only) solution to the linear system $(A^T A)\widehat{x} = A^T b$. However, we might wish to avoid computing and inverting the covariance matrix, and instead simply use GD on the least squares objective.

Now, observe that the gradient of the objective, is $\nabla f(x) = -A^T(b - Ax)$ so that, the gradient descent iteration is simply,

$$x^{t+1} = x^t + \eta A^T(b - Ax^t).$$

Re-arranging this and using the characterization of $\widehat{x}$ above we can see that,

$$x^{t+1} - \widehat{x} = \left[I - \eta(A^T A)\right](x^t - \widehat{x}).$$

We can unroll this to see that after k time steps x^k satisfies,

$$x^k - \widehat{x} = \left[I - \eta(A^T A)\right]^k (x^t - \widehat{x}),$$

as a direct consequence we see that,

$$\|x^k - \widehat{x}\|_2 \leq \|I - \eta(A^T A)\|_{\text{op}}^k \|x^0 - \widehat{x}\|_2.$$

So if we can ensure that the operator norm term < 1 we will have rapid (geometric) decay of the distance between our iterate and the optimal solution.

Let us denote $A^T A := S$. Now, one can check the following fact: if we choose $\eta = \frac{2}{\lambda_{\max}(S)+\lambda_{\min}(S)}$ (this is some ideal choice that we won't have access to in practice, but will help us in theory) then $\|I - \eta(A^T A)\|_{\text{op}} = (\lambda_{\max}(S) - \lambda_{\min}(S))/(\lambda_{\max}(S) + \lambda_{\min}(S)) = (\kappa(S) - 1)/(\kappa(S) + 1) < 1$ and we see that,

$$\|x^k - \widehat{x}\|_2 \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|x^0 - \widehat{x}\|_2.$$

Some notes about this result:

(a) We might sometimes (often) care instead about the value of the objective function at our iterates i.e. we would like to upper bound $f(x^k) - f(\widehat{x})$. For nice quadratics its easy to obtain a bound on this error from a bound on $\|x^k - \widehat{x}\|_2$. Some algebra will show that,

$$\begin{aligned} f(x^k) - f(\widehat{x}) &= \frac{(x^k - \widehat{x})^T A^T A (x^k - \widehat{x})}{2} \\ &\leq \frac{\lambda_{\max}(S)}{2} \|x^k - \widehat{x}\|_2^2 \\ &\leq \frac{\lambda_{\max}(S)}{2} \left(\frac{\kappa - 1}{\kappa + 1}\right)^{2k} \|x^0 - \widehat{x}\|_2^2. \end{aligned}$$

(b) This type of convergence is often called *linear* convergence in optimization (and sometimes called geometric convergence). A consequence of the above statement is that if I want my error to be $\leq \epsilon$ then it suffices to take $k \sim \log(1/\epsilon)$ steps (ignoring constants which depend on how far you initialize, and the condition number of S).

2. Another prototypical example is applying (sub)GD to the (univariate) function $f(x) = |x|$. Suppose that we initialize at some point x_0, and use some constant step-size η (ignoring for now the non-differentiability at 0). We can see that in general the GD iterates will bounce around the optimum, and will not converge. A picture is easier to follow.

   In this case, the only way to "force" GD to converge will be to use a decaying step-size (or if we want to get to within ϵ of the optimum we should use a step-size that is smaller than that), and this will result in much slower convergence.

   This is one of the main problems of trying to optimize functions which are not smooth.

3.4 GD on Smooth Functions

For the rest of this lecture, we'll assume that our objective function f is twice-differentiable and β-smooth. Our goal will be to try to understand the behaviour of GD in three settings which are increasingly "nicer":

1. Arbitrary (possibly non-convex) function f which is twice-differentiable and β-smooth.
2. Convex function f which is twice-differentiable and β-smooth.
3. Convex function f which is twice-differentiable and β-smooth, and is additionally α-strongly convex.

Most of these results don't require twice-differentiability but the proofs are sometimes a bit more transparent when you do have twice-differentiability.

3.4.1 Smooth Possibly Non-Convex Functions

For a not necessarily convex problem, we should not expect to be able to find a point which is a global optimum. Instead we'll settle for finding a point with small gradient norm, i.e. a point x for which $\|\nabla f(x)\|_2 \leq \epsilon$ (say). These points are called approximate saddle points (points where the gradient is 0 are called saddle points).

Here is a simple fact (it's just the multivariate analogue of Taylor's theorem) that holds for functions which are twice differentiable:

Lemma 3.1 *For any $x, y \in \mathbb{R}^d$, there is a z on the line joining x to y such that,*

$$f(y) = f(x) + \nabla f(x)^T(y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x).$$

Proof: Try to apply the usual (univariate) Taylor theorem to the function $g(t) = f((1-t)x + ty)$. You can for instance use this lemma to see why the second-order characterization of convexity implies the first-order characterization. ■

As a side note, you can just stare at the above expression and try to convince yourselves that (for twice differentiable functions) the second-order characterization of convexity implies the first-order one.

The main "descent" lemma:

Lemma 3.2 *For any step-size $\eta \leq 2/\beta$, the GD algorithm is a descent algorithm. For any $\eta \leq 1/\beta$ it further satisfies,*

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

Worth noting that some (pretty miraculous) facts are true:

1. If $\|\nabla f(x^t)\|_2 > 0$ then we have strict descent, i.e. $f(x^{t+1}) < f(x^t)$.
2. Furthermore, if the gradient is large (in norm) then an iteration of GD decreases the function by a large amount.
3. Just by smoothness (no convexity), we already see that GD doesn't suffer from the "bouncing around" problem it encounters when applied to the (non-smooth) function $|x|$, even with a fixed step-size.

Proof: Our β-smoothness condition implies that,

$$\frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x) \leq \frac{\beta}{2}\|y-x\|_2^2.$$

Applying the above Lemma we now see that,

$$\begin{aligned} f(x^{t+1}) &\leq f(x^t) - \eta\|\nabla f(x^t)\|_2^2 + \frac{\eta^2\beta}{2}\|\nabla f(x^t)\|_2^2 \\ &\leq f(x^t) - \left(\eta - \frac{\eta^2\beta}{2}\right)\|\nabla f(x^t)\|_2^2. \end{aligned}$$

From this we directly obtain our two conclusions. ■

3.4.1.1 Descent Lemma Without Twice Differentiability

The descent lemma follows from smoothness directly. Here is an alternate proof. By smoothness,

$$f(x^{t+1}) \leq f(x^t) + \nabla f(x^t)^T(x^{t+1} - x^t) + \frac{\beta}{2}\|x^{t+1} - x^t\|_2^2,$$

so using the fact that, $x^{t+1} = x^t - \eta \nabla f(x^t)$, we see that,

$$f(x^{t+1}) \leq f(x^t) - \eta \|\nabla f(x^t)\|_2^2 + \frac{\eta^2 \beta}{2} \|\nabla f(x^t)\|_2^2,$$

which is exactly the claim we had above.

3.4.1.2 The main theorem

Theorem 3.3 *Let x^* be any minimizer of f, then GD with step-size $\frac{1}{\beta}$ has the property that within k iterations it will reach a point x such that*

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

1. **Dimension-free:** It is worth noticing an amazing fact about the above result, and more generally about many of the results about optimization algorithms you will see in this course. The result is completely dimension-free, i.e. the error goes doesn't depend at all on the ambient dimension d.

Proof: For contradiction, assume that in all iterations from $t \in \{0, 1, \ldots, k\}$ we have that, $\|\nabla f(x^t)\|_2 \geq \sqrt{\frac{2\beta}{k}(f(x^0) - f(x^*))}$. Then on each iteration $t \in \{0, \ldots, k-1\}$ we conclude that we must have,

$$f(x^{t+1}) \leq f(x^t) - \frac{f(x^0) - f(x^*)}{k},$$

and re-arranging we see that,

$$f(x^{t+1}) - f(x^t) \leq -\frac{f(x^0) - f(x^*)}{k}.$$

Summing (telescoping) from $t = 0$ through $k - 1$, we have,

$$f(x^k) - f(x^0) \leq f(x^*) - f(x^0),$$

which means that x^k must either be a global minimizer of the function (in which case the Theorem is certainly true) or it cannot be the case that the gradient was large on each iteration.

■

3.4.2 Gradient Descent on Smooth Convex Functions

Before we prove a result it's worth understanding how convexity might help us. In the previous section, we already showed GD will find a point with small gradient norm. We know that for convex functions we have the upper bound:

$$f(x) - f(x^*) \leq \nabla f(x)^T(x - x^*) \leq \|\nabla f(x)\|\|x - x^*\|,$$

by Cauchy-Schwarz. Suppose that we initialize in some finite neighborhood of x^*, i.e. that $\|x - x^*\| \leq R$. Intuitively, just by convexity we already know that if the gradient is small we must be close to the optimum (in function value) – this is one of the key properties of convex functions. Our subsequent proof will be a refinement of this basic intuition.

Theorem 3.4 *Let x^* be any minimizer of f, then GD with step-size $\frac{1}{\beta}$ has the property that after k iterations it will reach a point x^k such that*

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

1. It is worth noting that now we obtain a global guarantee (i.e. GD will find a point as good as the best point x^*). However, the guarantee is still much slower than the one we derived earlier for quadratics. To obtain ϵ-error we need to take roughly $1/\epsilon$ steps.

2. This proof – and many proofs in convex optimization will follow a few elementary steps. It might be a bit mysterious at first, but you'll get the hang of it. Usually, the steps are playing with quadratics (i.e. some form of the Pythagorean theorem) and then using the conditions (convexity, smoothness, strong convexity) in a clever way.

Proof: Notice that, for any $t \in \{1, \ldots, k\}$

$$\|x^t - x^*\|_2^2 = \|x^{t-1} - \eta\nabla f(x^{t-1}) - x^*\|_2^2 \tag{3.1}$$

$$= \|x^{t-1} - x^*\|_2^2 - 2\eta\nabla f(x^{t-1})^T(x^{t-1} - x^*) + \eta^2\|\nabla f(x^{t-1})\|_2^2. \tag{3.2}$$

By our main descent lemma (which holds even without convexity) we know that for our choice of step-size,

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

By convexity, we know that,

$$f(x^{t-1}) - f(x^*) \leq \nabla f(x^{t-1})^T(x^{t-1} - x^*).$$

So we obtain from (3.1),

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

This gives us the fact that,

$$f(x^t) - f(x^*) \leq \frac{\beta}{2} \left(\|x^{t-1} - x^*\|_2^2 - \|x^t - x^*\|_2^2 \right).$$

Summing from $t = 1, \ldots, k$ (and dividing by k), and dropping the remaining negative term we obtain that,

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

Now, we can conclude the proof by noticing that for our choice of step-size, $f(x^k) \leq f(x^t)$ for $t = \{1, \ldots, k\}$ (i.e. GD is a descent algorithm). ■

In the next lecture, we'll continue our discussion of gradient descent with an eye toward trying to bridge the gap between the result we obtained above and the result we began this lecture with for quadratics.