COS 597G: Toward Theoretical Understanding of Deep Learning **Fall 2018**

Lecture 6: 04 October 2018

Lecturer: Sanjeev Arora *Scribe: Pranay Manocha, Mohammad Tariqul Islam*

Recall that gradient descent makes progress so long as gradient is nonzero. But progress slows down when gradient becomes close to 0. Note that for nonconvex objectives —which occur in deep learning—zero gradient does not guarantee that we're at a minimum — even a local minimum. Fig-6.1 illustrates this. It can lead us to either of a local mimina, a local maxima or a saddle point, more on which we will elaborate in this lecture.

This lecture discusses different solution concepts for nonconvex optimization and how gradient descent (with minor change) can help us reach these solutions fairly efficiently.

6.1 Gradient Descent

Optimization using gradient descent (GD) involves minimizing the loss function $f : \mathbb{R}^d \to \mathbb{R}$ iteratively by

$$x_{t+1} = x_t - \eta \nabla f(x_t) \tag{6.1}$$

where $\eta > 0$ is called the step size as well as the learning rate. If f is strongly convex, (6.1) converges quickly to a first order stationary point ($\nabla f = 0$) subject to the step size parameter η.

6.1.1 The Descent Lemma

We recall that if f is a twice-differentiable function with continuous derivatives and smoothness β (recall that smoothness is the maximum eigenvalue, in magnitude, of the Hessian) then gradient descent makes good progress at each step if we set learning rate small enough. We'll assume for this lecture that β is known approximately.

Lemma 1. ***Descent Lemma*** *If the learning rate η is set to $\frac{1}{\beta}$, then for every step $t \geq 0$ of gradient descent we have that*

$$f(x_t) - f(x_{t+1}) \geq \frac{1}{2\beta} \|\nabla f|_{x_t}\|^2 \tag{6.2}$$

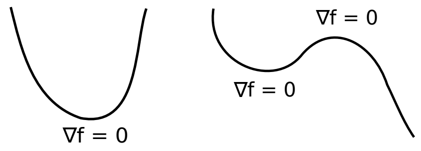
where $x_{t+1} = x_t - \eta \nabla f|_{x_t}$

Thus, so long as $\|\nabla f\|$ is not too small, some amount of descent happens. A corollary to the descent lemma is given in terms of distance moved.

Corollary 1. *If $\eta \leq \frac{1}{\ell}$ then taking a GD step from x_t to x_{t+1} results in $f(x_{t+1}) - f(x_t) \leq -\frac{1}{2\eta}\|x_{t+1} - x_t\|^2$.*

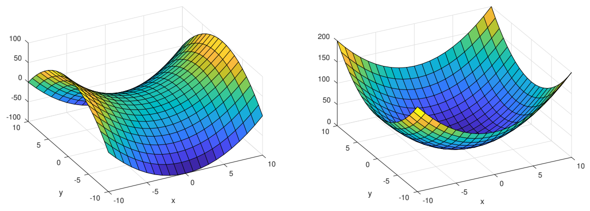
Proof. We apply the ℓ-smoothness of f to get

$$\begin{aligned}
f(x_{t+1}) &\leq f(x_t) + \nabla f(x_t) \cdot (x_{t+1} - x_t) + \frac{\ell}{2}\|x_{t+1} - x_t\|^2 \\
&= f(x_t) - \eta\|\nabla f(x_t)\|^2 + \frac{\eta^2 \ell}{2}\|\nabla f(x_t)\|^2 \\
&\leq f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 \\
&= f(x_t) - \frac{1}{2\eta}\|x_{t+1} - x_t\|^2
\end{aligned}$$



(a) (b)

Figure 6.1: Illustration of (a) convex function and (b) non-convex function in 1 dimension. A convex function has one first order stationary point (the minima) whereas in the non-convex function, the first order stationary point might be a local minima or local maxima.



(a) (b)

Figure 6.2: Illustration of (a) a saddle point ($x^2 - y^2$) and (b) a global minima ($x^2 + y^2$) in 2 dimensions. The saddle point is minima from the direction of x but a maxima from a direction of y. The global minima is a minima in both direction.

Thus,

$$f(x_t) - f(x_{t+1}) \geq \frac{1}{2\eta}\|x_{t+1} - x_t\|^2$$

We use this on non-convex functions, as there is no restriction on the convexity of the function for the quantum of descent. We will use this fact to prove the main lemma 1

□

6.1.2 Different solution concepts for nonconvex objectives

Clearly, so long as the objective has a lower bound on its value, gradient descent must ultimately get us to a point where the gradient is zero. We are interested in how fast, and what are the properties of this final point. The following is a taxonomy of different types of points one could hope to reach:

1. $\nabla f(x) = 0$. This is called a *stationary point*.
2. $\nabla f(x) = 0$ and $\nabla^2 f(x) \succeq 0$ (i.e., Hessian is positive semidefinite). This is called a *2nd order local minimum.* Note that for a convex f, the Hessian is a psd matrix at any point x; so every stationary point in such function is also a 2nd order local minimum.
3. x that minimizes f (in a compact set). This is called the global minimum. Clearly, it must also be a local minimum. For convex functions the stationary point is already a global minimum (assuming strict convexity) whereas for nonconvex functions there could be many local optima that are not global optima.

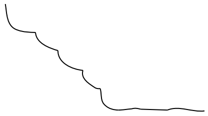


Figure 6.3: Gradient descent in a high-dimensional surface. The algorithm walks walks and walks and arrives at a point where $|\nabla f|$ is small and $\lambda_{min}(\nabla^2 f) <\leq -\delta$. And the algorithm stops at some point.

4. $\nabla f(x) = 0$ and $\nabla^2 f(x) \prec 0$ (i.e., Hessian has a negative eigenvalue). These are called *saddle points*, and obviously these don't exist for convex f.

Figure 6.1 shows that for a non-convex function, we may end up as at a local maxima while finding the stationary point. Figure 6.2 illustrates one example of a saddle point. A saddle point is a first order stationary point which is typically a local minima for all variables except one, for which it is a local maxima. In Fig 6.2, there is a local minimum along x axis, but a local maximum along y. For non-convex optimization, saddle-points happen quite often [Dauphin et al., 2014] and can cause training to seemingly come to a halt because gradient is zero. Unbeknownst to the algorithm (Figure 6.3), which lacks information about the Hessian (which is too expensive to compute in today's settings), there is a direction such that moving according to it would cause gradient to become nonzero again, but finding that direction either requires computing the eigenvectors of the Hessian (too expensive) or walking around aimlessly around the point until this direction is accidentally discovered. But exploration in $\Re^d$ may take $\exp(d)$ steps which we discussed earlier in lecture 2.

6.2 Escaping saddle points: perturbed gradient descent

Next, based on the motivation from the previous section on how following second order methods (calculating the Hessian) is computationally expensive, we aim to design approximate second order methods which are not as computationally expensive as Hessian but are successfully able to evade saddle points.

6.2.1 Perturbed Gradient Descent

An *approximate 2nd order minimum* is x such that (i) $\nabla f(x)$ is small (ii) $\nabla^2 f(x)$ does not have a very negative singular value. We will show that Perturbed Gradient Descent (PGD) can find such solutions. This is the version of gradient descent which, whenever it detects lack of progress for a few iterations, takes a step of a certain length in a random direction.

1. Do T steps of GD. (*Deterministic steps*)
2. If there is insufficient descent in the line 1, perform one step $x_{t+1} \leftarrow x_t + \epsilon$, where $\epsilon \underset{U(\mathbb{R}^d)}{\in} B(0, r)$ where $B(0, r)$ is the ball around the origin of radius r. (*Random jump.*)

We'll show that PGD efficiently ends up at x such that $\nabla f|_x$ is small (i.e. $||\nabla f|| \leq \epsilon$) and $\nabla^2 f|_x \succeq -\delta I$ (i.e. all eigenvalues are greater than $-\delta$: $\lambda_{min}(\nabla^2 f(x)) \geq -\delta$).

This line of work was started in [Ge et al., 2015] who used a version of gradient descent which adds gaussian noise to the gradient, and finds approximate 2nd order minima in polynomial time. Convergence was improved in [Levy, 2016] who employed normalized gradients, and the analysis presented here is from [Jin et al., 2017].

Theorem 1. *PGD with $\eta = \frac{1}{l}$ and $r = \frac{1}{\chi^3 \sqrt{\kappa}} \frac{\sqrt{\epsilon}}{\sqrt{\rho}}$ will pass through an ϵ-approximate second order stationary point with $1 - \epsilon$ probability in $\mathcal{O}(\frac{l \Delta_f}{\epsilon^3})$ iterations. Here x_k is the number of steps between perturbations and Δ_f is the* *difference between the function value at the first point of the iteration and the optimum value such that* $\Delta_f = f(x_0 - f_{x^*})$

Such papers often have this kind of a formulation with a lot of symbols. While such statements are a necessary evil in the larger scheme of things (peer review, etc.) they tend to make the result forbidding for the students. We break down the proof into easy to understand pieces, while being a bit loose with details. The overall proof idea is as follows.

1. So long as gradient is large, the algorithm makes significant progress in the T deterministic steps.
2. If gradient descent stops making significant progress (i.e., reduction in objective value) for many iterations, then gradient must be small. Furthermore we show it must be stuck in a small geometric ball during these iterations. (Lemma 2.). We call this *trap*
3. Since the algorithm is trapped in a small ball, the gradient is not changing much and neither is the Hessian (due to bounds on derivative norms). If this Hessian doesn't have a significantly negative eigenvalue, our mission is accomplished.
4. Else (i.e., if Hessian has a significant negative eigenvalue) taking a reasonably large step in a random direction is quite likely to bring us to a new point, where gradient descent again starts making progress and is guaranteed to decrease the objective by a certain prescribed amount. We call this the *release*.

Why do these four ideas suffice to prove the theorem? Recall that we assume that the function has a lower bound on its value. Each time the above trap-and-release happens, the function value drops by a prescribed amount. Thus the trap-and-release can only happen a bounded number of times, at the end of which the PGD algorithm must end up up at a point where the gradient is small and the Hessian does not have a significantly negative eigenvalue.

6.2.2 Not going anywhere lemma

Now we formalize the notion of being trapped: if GD is not making progress (i.e. reducing the objective significantly) for many steps, then algorithm must be stuck inside a ball of small radius.

Lemma 2. ***Not Going Anywhere Lemma:*** *If* $f(x_T) - f(x_0) \geq -\mathscr{F}$ *then* $\forall\, t \leq T$ *we have* $||x_t - x_0|| \leq \sqrt{2\eta T \mathscr{F}}$.

Proof: We apply Cauchy-Schwartz followed by the inequality from the descent lemma:

$$||x_t - x_0|| = \sum_{\tau=1}^{t} ||x_\tau - x_{\tau-1}|| \leq \sqrt{T \sum_{\tau=1}^{t} ||x_\tau - x_{\tau-1}||^2} \leq \sqrt{2\eta T(f(x_0) - f(x_T))} \leq \sqrt{2\eta T \mathscr{F}}$$

Thus effectively, GD is stuck in a ball of a radius upper bounded by $\sqrt{2\eta T \mathscr{F}}$. PGD has to escape the ball the GD is stuck in.

In terms of our proof, $\mathscr{F}$ is the "quantum"of descent.

6.2.2.1 Release: Intuitive Difficulty

If the Hessian has a negative eigenvalue, there is one direction in $\mathbb{R}^n$ corresponding to eigenvector u that increases the gradient. If the algorithm moves in a random direction by distance r then it also moves around $r/\sqrt{d}$ in the direction corresponding to u (see Figure 6.4). But analysing progress this way gets complicated and the running time is not so good (see the original paper of Ge et al.).

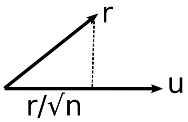


Figure 6.4: A random jump r in a direction will cause only $r/\sqrt(n)$ movement in the direction of bad eigenvector u.

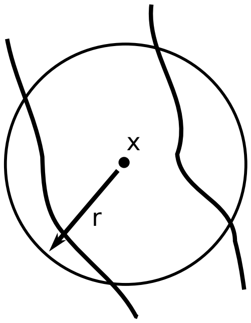


Figure 6.5: If a step of GD is performed in this ball the algorithm will not descend. This is a small volume of region where the algorithm is not making progress.

The current proof will take a different tack: it will show that progress happens for a large fraction of points in the ball of radius r, and therefore the perturbed GD gets released with high probability when it makes the random step. We show the following where "stuck region"is the name for points in ball $B(x,r)$ of radius r around the stuck point x where T steps of deterministic GD doesn't lead to a good quantum of descent. We show:

$$\frac{vol_{stuck-region}}{vol_{B(0,r)}} < small \tag{6.3}$$

6.2.2.2 Bounding the width of stuck region

To prove the above bound on the volume of the stuck region of the ball of radius r around the current point, we will show that the width of the stuck region is small.

To do so we consider any pair of points x_1, x_1' which differ by a small multiple of u (recall, u is the Eigenvector corresponding to λ_{min}; we assume third derivative is small enough that u does not change much in this neighborhood). The idea is to imagine starting two copies of GD, one at x_1 and the other at x_1'. We show one of these copiesis guaranteed to do well (that is do descend) and this lets us conclude that the stuck region in this ball is small. The process is illustrated in Figure 6.6.

6.2.3 Main Lemma

Lemma 3. ***Main Lemma:*** *x is such that $\nabla f|_x$ is small and $\lambda_{min}(\nabla^2 f|_x) \leq -\delta$. Here $x_1, x_1' \in Ball(x,r)$, such that $x_1 - x_1' = \beta u$, where u is the bad eigenvector and β is a constant with reasonably large absolute value.*
Then for appropriate parameter values, if we do T steps of plain GD from x_1 and x_1' respectively, then at least one of them achieves prescribed quantum descent.

Important comments about the proof technique: *The algorithm does not know u, and thus cannot find two points like x_1, x_1' where $x_1 - x_1'$ is a multiple of u. This is a thought experiment! But this thought experiment shows that if x_1 is stuck, it has a twin x_1' which is not stuck and vice versa subject to $x_1 - x_1' \geq \beta u$. So we have shown that the*

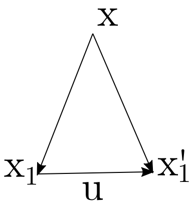


Figure 6.6: u is the direction of the eigenvector, called bad eigenvector, in which the GD algorithm is stuck in. u corresponds to $\lambda_{min}(\nabla^2 f)$. Two different points x_1 and x_1' differ by a multiple of the vector u. The proof shows that, gradient descent from one of them must make progress.

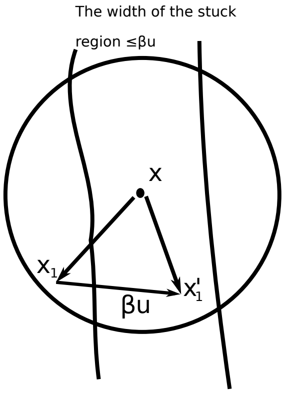


Figure 6.7: The region stuck in the direction of bad eigenvector u is finite ($\leq \beta u$). Thus with two random jumps, x_1 and x_1' from the stuck region with a relative direction $x_1 - x_1' \geq \beta u$, at least one of them will escape the stuck region.

width of the stuck region is small, and thus (by simple integration) the volume of the stuck region is a small portion of the ball.

The key insight is the following. Here x_t, x_t' denote respectively the points reached by (deterministic) GD starting from x_1, x_1'.

Lemma 4. *[Repulsion Lemma] The distance $|x_t - x_t'|$ grows geometrically with t.*

Proof. Let $w_t = x_t - x_t'$. It satisfies:

$$\begin{aligned} w_{t+1} &= x_{t+1} - x'_{t+1} \\ &= x_t - \eta \nabla f|_{x_t} - (x_t' - \eta \nabla f|_{x_t'}) \\ &\approx (x_t - x_t') - \eta H (x_t - x_t') \\ &= (I - \eta H)(x_t - x_t') \\ &= (I - \eta H)^t (x_1 - x_1') \qquad (6.4) \\ &= (1 + \eta \beta \delta)^t u. \qquad (6.5) \end{aligned}$$

where the last line uses that $(x_1 - x_1') = \beta u$ where u is an eigenvector of H with eigenvalue $-\delta$. □

Next, we prove Theorem 1 in an informal way. The quantum of descent $\mathscr{F}$ is going to be set small enough and the parameter δ in the theorem statement is going to be set large enough so that the radius of the ball $2\eta T \mathscr{F}$ in the "trap"(see Lemma 2) is much smaller than the amount of repulsion $(1 + \eta\beta\delta)^T$ in the Repulsion Lemma.

This guarantees that at least one of x_1, x_1' in Repulsion Lemma must lead to a standard quantum of descent in T steps. This follows by a simple proof by contradiction. If both x_1, x_1' did not lead to a quantum of descent, then the GD is in a trap around x_1, x_1' of a small radius, whereas the Repulsion Lemma says that the two GD trajectories diverge. Thus at least one of the points gets released. This shows by our earlier argument that the volume of the stuck region is small, and the random-step in perturbed GD will with high probability land at a point where descent resumes.

Hence, the main result 1 results from the fact that at least one of the points make a descent and from Corollary 1 which proves that GD decreases even for non convex functions proves the theorem.

References

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

Kfir Y Levy. The power of normalization: Faster evasion of saddle points. *arXiv preprint arXiv:1611.04831*, 2016.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.