Assignment 1
CS60B: Algorithmic Machine Learning, Spring 2022
Due: April 13, 10PM

1. For a function $f : \mathbb{R}^d \to \mathbb{R}$, a point x_0 is a *local minimum* if there exists $\delta > 0$ such that for all x with $\|x - x_0\| < \delta$, $f(x_0) \leq f(x)$. Similarly, a point x_0 is a *local maximum* if the opposite holds: if there exists $\delta > 0$ such that for all x with $\|x - x_0\| < \delta$, $f(x_0) \geq f(x)$.

   There exist smooth functions for which finding a *local minimum* is NP-hard. However, if f is differentiable, then one *necessary* condition for x_0 to be a local minimum is that $\nabla f(x_0) = 0$.

   (a) This is necessary but not sufficient! Give an example of a function in two dimensions and a point x_0 such that $\nabla f(x_0) = 0$ but x_0 is neither a local minimum nor a local maximum of f. [2 points]
   You can take $f(a, b) = a^2 - b^2$ and $x_0 = (1, 1)$ (for example). Clearly, the gradient vanishes at x_0, but $(1, 1)$ is not a local minimum; for any δ, $(1, 1 - \delta)$ has a lower value than $f(x_0)$. Nor is it a local maximum.

   (b) Inspite of the above, vanishing gradients is often desired. Suppose we have a β-smooth function $f : \mathbb{R}^d \to \mathbb{R}$. Show that gradient descent can be used to find a point w such that $\|\nabla f(w)\| \leq \varepsilon$. How many iterations of GD do you need for finding such a point? Your bound can depend on the starting point, $f(w_0)$, β, ε, and the value of the glolbal optimum. [3 points]
   [Hint: Your point w could be any of the iterations of GD. Use our claim on monotonicity of GD.]
   Recall monotonic-decrease inqeuality from class: $f(w_{k+1}) \leq f(w_k) - (1/2\beta)\|\nabla f(w_k)\|^2$. Rearranging this, we get $\nabla f(w_k)^2\| \leq 2\beta(f(w_k) - f(w_{k+1})$. If you now add this inequality over all iterations, we get some cancellations and

   $$\sum_{i=0}^{k-1} \|\nabla f(w_i)\|^2 \leq 2\beta(f(w_0) - f(w_k)) \leq 2\beta(f(w_0) - f(w_*)).$$

   So if we pick the index j that minimizes $\|\nabla f(w_i)\|^2$ among $i = 0, \ldots, k-1$, we get $\|\nabla f(w_j)\|^2 \leq 2\beta(f(w_0) - f(w_*))/k$.
   So it suffices to take $k = 2\beta(f(w_0) - f(w_*))/\varepsilon^2$ so that we get the guarantee of the problem.

2. For $\alpha > 0$, a smooth function $f : \mathbb{R}^n \to \mathbb{R}$ is *α-convex* if for all $x, y \in \mathbb{R}^n$,

   $$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\alpha}{2}\|y - x\|_2^2.$$

In this exercise we will show a better bound on convergence rate of gradient descent (GD) for such functions.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a α-convex β-smooth function. Let x^* be an optimal minimizer for f. Consider the GD algorithm with starting point x_0 and step-size $t = 1/\beta$. Show that after k iterations we have

$$\|x_k - x^*\|_2^2 \leq \|x_0 - x^*\|_2^2 \left(1 - \frac{\alpha}{\beta}\right)^k.$$

In other words, the distance of the k'th iterate to the optimal decreases exponentially in the number of iterations. [5 points]

[Hint: You may use the fact that $\nabla f(x^*) = 0$.]

Remark: Note that the number of iterations to get error ε for such functions is $O(\log(1/\varepsilon))$ for fixed x_0, α, β; this is exponentially better than the bound for general covex functions which was $O(1/\varepsilon)$.

We first start with monotonic decrease

$$f(x_{k+1}) \leq f(x_k) - (1/2\beta)\|\nabla f(x_k)\|^2.$$

Further, by strong convexity, we also have

$$f(x_k) + \langle \nabla f(x_k), x_* - x_k \rangle + (\alpha/2)\|x_k - x_*\|^2 \leq f(w_*).$$

Add the above two inequalities and rearranging some terms, we get

$$0 \leq f(x_{k+1}) - f(x_*) \leq \langle \nabla f(x_k), x_k - x_* \rangle - (1/2\beta)\|\nabla f(x_k)\|^2 - (\alpha/2)\|x_k - x_*\|^2.$$

Further, we also have

$$\|x_{k+1} - x_*\|^2 = \|x_k - x_* - (1/\beta)\nabla f(x_k)\|^2 = \|x_k - x_*\|^2 + (1/\beta^2)\|\nabla f(x_k)\|^2 - (2/\beta)\langle \nabla f(x_k), x_k - x_* \rangle.$$

Multiplying the previous inequality by a factor of $(2/\beta)$ and adding to the above (so that we can cancel the second two terms above), we get

$$\|x_{k+1} - x_*\|^2 \leq (1 - \alpha/\beta)\|x_k - x_*\|^2.$$

Applying the above inequality for all iterations, we get $\|x_k - x_*\|^2 \leq (1 - \alpha/\beta)^k \|x_0 - x_*\|^2$.

A different approach with a slightly weaker bound (that will also get full-credit):

Recall the inequality we showed in lecture 4:

$$f(x_{k+1}) - f(x_*) \leq \frac{\beta}{2}\left(\|x_k - x_*\|^2 - \|x_{k+1} - x_*\|^2\right).$$

On the other hand, by strong-convexity at the point x_*, where $\nabla f(x_*) = 0$,

$$\frac{\alpha}{2}\|x_{k+1} - x_*\|^2 \leq f(x_{k+1}) - f(x_*).$$

Therefore,

$$\frac{\alpha}{2}\|x_{k+1} - x_*\|^2 \leq \frac{\beta}{2}\left(\|x_k - x_*\|^2 - \|x_{k+1} - x_*\|^2\right).$$

Simplifying the above we get,

$$\|x_{k+1} - x_*\|^2 \leq (1 + \alpha/\beta)^{-1}\|x_k - x_*\|^2.$$

Iterating the above, we get an exponential decrease again:

$$\|x_k - x_*\|^2 \leq (1 + \alpha/\beta)^{-k}\|x_0 - x_*\|^2.$$

3. Show that for a differentiable convex function $f : \mathbb{R}^d \to \mathbb{R}$, every local minimum, i.e., any point x with $\nabla f(x) = 0$, is also a global minimum. [3 points]

   Let x be a local minimum, then by the definition of convexity, (as the tangent line at x must be below the curve), we get that for any y

   $$f(x) + \langle \nabla f(x), y - x \rangle \leq f(y).$$

   As $\nabla f(x) = 0$, we immediately get $f(x) \leq f(y)$ for all y; hence, x must be a global minimum.

4. The goal of this exercise is to implement and compare GD, and Nesterov's accelerated gradient descent (NAGD) for *logistic regression*.

   Suppose we have binary data, i.e., the labels are 0 or 1. When trying to *fit* binary labels with linear predictors, a commonly used loss function is the *logit loss* function: $\ell(a, b) = -a \log b - (1 - a)\log(1 - b)$ (also known as *cross-entropy* loss etc.).

   One commonly used parameterized predictor family is $h_w(x) = \sigma(w \cdot x)$, where $\sigma(t) = 1/(1 + e^{-t})$ is the sigmoid function. Combining these two, given a dataset $(x_1, y_1), \ldots, (x_n, y_n)$, the corresponding ERM optimization problem is to minimize

   $$L(w) = (1/n)\sum_{i=1}^{n} \ell(y_i, \sigma(\langle w, x_i \rangle)).$$

   The above formulation is called *logistic regression*. Your goal is to implement variants of GD for the above loss function and test it some random dataset.

   Generating dataset. Set $n = 1000, d = 20$ (or higher too if you want). Pick a random 'hidden vector' $w_* \in \mathbb{R}^d$ of unit-norm. As in the workspace examples, generate random data as follows: For $i = 1, \ldots, n$: Generate a random Gaussian vector $x_i \in \mathbb{R}^d$ and after that, independently set $y_i = 1$ with probability $\sigma(\langle w_*, x_i \rangle)$ and 0 with probability $1 - \sigma(\langle w_*, x_i \rangle)$. Sample code for this is provided at the bottom of GD workspace on edStem.

(a) Write down a formula for the gradient of $L(w)$. [1 point]

We can compute the gradient by repeated application of the chain rule:

$$\nabla_w L(w) = (1/n) \sum_{i=1}^{n} \nabla_w \ell(y_i, \sigma(\langle w, x_i \rangle)) = (1/n) \sum_{i=1}^{n} \partial_b \ell(y_i, \sigma(\langle w, x_i \rangle)) \sigma'(\langle w, x_i \rangle) x_i.$$

(b) Implement GD, NAGD with various step-sizes (say 3 step-sizes each) for several hundred iterations to get convergence and plot the loss values. Also plot the distance to the hidden vector w_* of the iterates under the three methods. [4 points]

(Don't ask why this is the **right** statistical model, but you can check the wiki page for details. We are only interested in optimization and not on the statistical aspects.)

Your submission on gradescope should be the following.

i. Screenshots of the implementations of the three algorithms (the gradient subroutine and other important parts).

ii. Plot of the values of f against the number of iteration for GD and NAGD on the same figure.

iii. Plot of the distance of the current iterate w_k to w_* under GD and NAGD on the same figure.

You may use/copy any of the code from the workspaces as you need (especially for plotting and generating data). Make sure your plots are well-labeled (the axes, and with an appropriate legend).