Homework S

- **Due: Tuesday, April 25, before class**
- Please refer to the syllabus for homework policies. (Points of interest include late policies, typesetting, collaboration, extra-credit stuff...)
- If you explicitly do not want your submission to be considered as a solution key, please state so clearly at the top of your submission. If you explicitly want to remain anonymous (if selected), please state so clearly at the top of your submission.
- If you use dynamic programming in your solution, please address the steps outlined in the typed notes.
- There are also some multiple-choice/short-answer problems on Gradescope.
- There is no need to include your PUID on the submission thanks to Gradescope.

Problems

1. This problem has 2 parts.

   (a) Let $f : \mathbb{R} \to \mathbb{R}$ be a convex and continuously differentiable function over $\mathbb{R}$. (Note that the input to f is 1-dimensional.) You may assume that querying a value $f(x)$, or a gradient $f'(x)$, each takes $O(1)$ time.

   Consider the problem of minimizing f. Suppose we are promised that there is a unique global minimizer $x^\star$ that lies in the open interval $(0, 1)$. Design and analyze an algorithm that, given an additional parameter $\epsilon > 0$, finds a point such that $x^\star - x \leq \epsilon$. (The running time should depend on ϵ (or rather $1/\epsilon$). As usual, the better the dependency on ϵ, the better.)

   (b) Suppose we want to minimize an L-smooth and convex function $f : \mathbb{R}^n \to \mathbb{R}$, but did not actually know the value of the parameter L. Suppose instead that you knew that L was a value between $1/\text{poly}(n)$ and $\text{poly}(n)$ (You may assume for simplicity that L is between $1/n^2$ and n^2). Design and analyze a variation of the gradient descent algorithm in the notes that does not explicitly use L, but still obtains a similarly iteration count depending on (the unknown value) L. (You may also incur a mild dependence on n.) In particular, one needs to figure out how to choose an appropriate step size since the algorithm we discussed choose the step size as a function of L.