IE510 Applied Nonlinear Programming Homework 2

Instructor name: Ruoyu Sun Student name: _____

Remark: The work your are submitting for this homework assignment must be your own, and suspiciously similar homework submitted by multiple individuals may be reported to the University for investigation.

1 Reading

- Reading: Textbook Section 1.1 - 1.2
- Appendix A.

2 Problems

1. (20 points)

   1) Tom is using gradient descent with constant stepsize to solve a problem $\min_{x\in\mathbb{R}^n} f(x)$ where f is continous differentiable with a finite lower bound, and the gradient does not converge to 0. Besides the bug in the code, what are the possible reasons? Name at least two, and explain by optimization theory.

   2) Jane is using gradient descent with constant stepsize to solve a problem $\min_{x\in\mathbb{R}^n} f(x)$ where f is strongly convex, twice differentiable, with Lipschitz continuous gradient, and has a finite lower bound 0 (i.e. $f(x) \geq 0, \forall x$). She found that during the process, the objective value is decreasing and gradually approaches 0.6. She thought about two possiblities: first, the global minimal value is very close to 0.6; second, the algorithm is converging too slowly due to a large condition number. She could not compute the condition number for now because that requires computing the eigenvalues of the Hessian which takes too much time. She asked for your help: could you provide a suggestion on how to verify or eliminate the first possibility?

   3) Following the previous question. Assume this is a linear regression problem, i.e., the Hessian matrix is a constant positive-definite matrix. Your suggested method eliminates the first possiblity that the global minimal value is very close to 0.6. Jane wants to verify whether the second possibility is indeed true. She finally got a supercomputer and calculated the condition number which turns out to be quite small: only 10. This is very small compared to the 10,000 iterations she has run, which implies that the second possibility cannot be true. She asked for your help: what could be the possible reason of the slowly decreasing function values around 0.6 then (again, besides a bug in code)?

Remark: In this problem, the setting is that your friend is running the algorithm, and you are a consultant providing suggestions. This means that you shall not assume your friend has followed standard practice, and you shall consider all possibilities.

2. (20 points) (texbook exercise 1.2.2) Consider using gradient descent with constant stepsize α to solve the problem $\min_{x\in\mathbb{R}^d} \|x\|^{2+\beta}$ where $\beta \geq 0$ is a fixed constant.

   (i) Prove that for any $L > 0$, the Lipschitz condition $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \forall x, y$ does not hold.

   (ii) For an arbitrary initial point $x^0 \neq 0$, find all values of $\alpha > 0$ such that GD converges to $x^* = 0$.

   Hint: Consider the stepsize that makes $\|x^1\| < \|x^0\|$.

3. (20 points) (textbook exercise 1.2.8) To solve the problem $\min_{x\in\mathbb{R}^d} f(x)$ where f is continuously differentiable, we use an iterative descent method $x^{k+1} = x^k + \alpha^k d^k$, where α^k is chosen by the Armijo rule, and

$$d^k = -\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \frac{\partial f(x^k)}{\partial x_i} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

   where i is the index for which $|\frac{\partial f(x^k)}{\partial x_j}|$ is maximized over $j \in \{1, 2, \ldots, n\}$. In other words, the index i is defined by $i = \text{argmax}_j |\frac{\partial f(x^k)}{\partial x_j}|$. Note that for different iteration k, the index i also differs. Prove that every limit point of $\{x^k\}$ is a stationary point.

4. (40 points) Consider a non-singular matrix $A = (A_1, A_2, \ldots, A_n)$ of size $d \times n$, where $A_j \in \mathbb{R}^{d\times 1}, j = 1, \ldots, n$. We want to solve

$$\min_{x\in\mathbb{R}^d} ||A^T x - b||^2. \tag{1}$$

   Assume $b = 0$ for simplicity.

   Let $n = 100$ and $d = 100$. Suppose Unif[0,c] represents a uniform distribution on the interval $[0, c]$. Consider the setting $A_{ij} \sim \text{Unif}[0, m_j]$, where

$$m_j = \begin{cases} 5, & j > n/2; \\ 1, & \text{otherwise.} \end{cases}$$

   After generating A, normalize each column A_j to get unit norm for $\forall j$, i.e. define $\widetilde{A_{ij}} = \frac{A_{ij}}{||A_j||}$, $\forall i, j$. Consider the problem

$$\min_{x\in\mathbb{R}^d} ||\widetilde{A}^T x - b||^2. \tag{2}$$

Define a new matrix B as follows:

$$B_{ij} = \widetilde{A}_{ij} - \mu_j, \quad \forall i, j,$$

where $\widetilde{A}$ is the matrix with normalized columns defined in (b) and

$$\mu_j = \sum_{i=1}^{d} \widetilde{A}_{ij}/d, \quad j = 1, \ldots, n.$$

In other words, in the new matrix B, the sum of each column should be 0. Consider the problem

$$\min_{x \in \mathbb{R}^d} ||B^T x - b||^2. \tag{3}$$

(a) Suppose the Hessians of the two problems (1) and (2) are H and $\widetilde{H}$ respectively. Compare GD with constant stepsize $1/\lambda_{\max}(H)$ for the previous problem and GD with constant stepsize $1/\lambda_{\max}(\widetilde{H})$ for the normalized problem. Which one is faster?

(b) Suppose the Hessian of the problem (3) is H_B. Compare GD with constant stepsize $1/\lambda_{\max}(H_B)$ for solving this problem and GD with constant stepsize $1/\lambda_{\max}(\widetilde{H})$ for solving the problem (2). Which one is faster?

(c) Compute the condition numbers of H, $\widetilde{H}$ and H_B and compare them. Is this comparison consistent with the comparison of the convergence speed of gradient descent for the three problems?

(d) Excluding all eigenvalues of M that are smaller than $\epsilon_0 = 10^{-10}$, and suppose the smallest among the remaining eigenvalues is $\lambda_{\min,>\epsilon_0}(M)$. Define the modified condition number of a matrix M as $\lambda_{\max}(M)/\lambda_{\min,>\epsilon_0}(M)$. Compute the modified condition numbers of H, $\widetilde{H}$ and H_B and compare them. Is this comparison consistent with the comparison of the convergence speed of gradient descent for the three problems?

Remark 1: We often use random initial point. One possible choice is $x^0 = (x_1^0, \ldots, x_n^0)$, where $x_i^0 \sim \mathcal{N}(0, 1)$, $i = 1, \ldots, n$. Here $\mathcal{N}(0, 1)$ represents standard Gaussian distribution. Do not use zero initial point, since it is the optimal solution for this problem.

Remark 2: In class, we discuss the normalization of each feature; to normalize each feature, you need to normalize each row. Nevertheless, in this problem we normalize each column (each data point);