Homework Assignment 3 – Math 118, Winter 2021

Dhruv Chakraborty

February 28, 2021

Problem 1

Prove that if $g(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ is convex and $f(x) : \mathbb{R} \to \mathbb{R}$ is convex and non-decreasing then $f(g(\mathbf{x}))$ is convex.

Solution: Consider $f(g(t\mathbf{x} + (1-t)\mathbf{y}))$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ and $t \in (0, 1)$. We know that g is convex which means that $g(t\mathbf{x} + (1-t)\mathbf{y}) \leq tg(\mathbf{x}) + (1-t)g(\mathbf{y})$ and that f is non-decreasing so we have $f(x) \leq f(y)$ for all $x, y \in \mathbb{R}$ such that $x \leq y$. In particular then, we get that $f(g(t\mathbf{x} + (1-t)\mathbf{y})) \leq f(tg(\mathbf{x}) + (1-t)g(\mathbf{y}))$. But note that f is also convex and $g(\mathbf{x}), g(\mathbf{y}) \in \mathbb{R}$ so we have $f(tg(\mathbf{x}) + (1-t)g(\mathbf{y})) \leq tf(g(\mathbf{x})) + (1-t)f(g(\mathbf{y}))$.
$\therefore f(g(t\mathbf{x} + (1-t)\mathbf{y})) \leq tf(g(\mathbf{x})) + (1-t)f(g(\mathbf{y})) \implies f(g(\mathbf{x}))$ is convex.

Problem 2

Show that $f(x) = x^3$ is not Lipshitz differentiable.

Solution: We know that $f(x)$ is Lipshitz differentiable with constant L if $||\nabla f(x) - \nabla f(y)||_2 \leq L||x - y||_2$. Suppose for a contradiction that $f(x) = x^3$ is Lipshitz differentiable with some constant L. Fix $y = 0$ and note that $\nabla f(x) = 3x^2$. Then we must have for any x that $||\nabla f(x)||_2 \leq L||x||_2 \implies 3x^2 \leq L||x||_2$. Squaring and dividing both sides by x^2 gives us $9x^2 \leq L$. This is not true for infinitely many $x \geq X$ for a sufficiently large X. In particular, $f(x)$ fails to hold for the given inequality as $x \to \infty$ and thus x^3 is not Lipshitz differentiable.

Problem 3

Recall the least squares problem $\mathbf{x}_* = \text{argmin}\|A\mathbf{x} - \mathbf{b}\|_2^2$ and let $f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|_2^2$, so that the problem becomes $\mathbf{x}_* = \text{argmin} f(\mathbf{x})$.

1. Show that $\nabla f(\mathbf{x}) = 2A^\top (A\mathbf{x} - \mathbf{b})$.

2. Show that $f(\mathbf{x})$ is Lipschitz differentiable with constant $L = 2\sigma_1(A)^2$.

3. Now, write code for solving the least squares problem using gradient descent. For A, use a random 100×50 matrix generated using the Python command `np.random.randn(100,50)`. For $\mathbf{b}$, use a random vector generated using `numpy.random.randn(100)`. Experiment with different step-sizes, α. How large can you take α before the algorithm fails to converge?

Solution:

1. From problem 4 below, we know that $f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{x}^\top Q\mathbf{x} - 2\mathbf{x}^\top \mathbf{c} + \|\mathbf{b}\|^2$ where $Q = A^\top A$ and $\mathbf{c} = A^\top \mathbf{b}$. Moreover from Section 5.4: "Differentiation Rules" in Chong and Zak's *Introduction to Optimization*, we have that $D(\mathbf{x}^T\mathbf{y}) = \mathbf{y}$ and $D(\mathbf{x}^T Q\mathbf{x}) = 2Q\mathbf{x}$ for symmetric Q and constant y where D is the derivative w.r.t. $\mathbf{x}$. Applying this, we have $\nabla f(\mathbf{x}) = \nabla[\mathbf{x}^\top Q\mathbf{x} - 2\mathbf{x}^\top \mathbf{c} + \|\mathbf{b}\|^2] = 2Q\mathbf{x} - 2\mathbf{c} = 2(A^TA\mathbf{x} - A^T\mathbf{b}) = 2A^T(A\mathbf{x} - \mathbf{b})$ as wanted.

2. For any $\mathbf{x}, \mathbf{y}$, we have $||\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})||_2 = ||2A^T(A\mathbf{x} - \mathbf{b}) - 2A^T(A\mathbf{y} - \mathbf{b})||_2 = ||2A^TA(\mathbf{x} - \mathbf{y})||_2$. But we know that $||A^TA||_2 = \sigma_1(A)^2$ so in particular, $||2A^TA(\mathbf{x} - \mathbf{y})||_2 \leq ||2A^TA||_2||(\mathbf{x} - \mathbf{y})||_2 = 2\sigma_1(A)^2||(\mathbf{x} - \mathbf{y})||_2$. Therefore by definition, $f(\mathbf{x})$ is Lipschitz differentiable with constant $L = 2\sigma_1(A)^2$.

3. When considering the difference between $A\mathbf{x}_*$ (where $\mathbf{x}_*$ is the solution found by the gradient descent algorithm) and a randomly generated $\mathbf{b}$, a step size around 0.001 seems to perform the best. Step sizes larger 0.005 than result in extremely poor approximations. While reducing the step size further doesn't worsen performance as much as increasing it, error does increase significantly for step sizes smaller than 0.0005. Even if we generate a $\mathbf{b}$ linearly by taking a random $\mathbf{y}$ and setting $\mathbf{b} = A\mathbf{y}$, a step size of approximately 0.001 does best. The required code is reproduced below.

```
import numpy as np

A = np.random.randn(100, 50)
# y = np.random.randn(50)
b = np.random.randn(100)
# b = A@y

def grad_descent(alpha = 0.1, k = 1000):
    x = np.random.randn(50)
    for i in range(k):
        grad = 2*A.T@((A@x)-b)
        x = x - alpha*grad
    return x
```

Problem 4

Let $Q = A^\top A$ and $\mathbf{c} = A^\top \mathbf{b}$ Show that:

$$\|A\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{x}^\top Q\mathbf{x} - 2\mathbf{x}^\top \mathbf{c} + \|\mathbf{b}\|^2$$

Hint: Use the fact that, for any vector $\mathbf{v}$ *we have that* $\|\mathbf{v}\|_2^2 = \mathbf{v}^\top \mathbf{v}$.

Solution:

$$\begin{aligned}
||A\mathbf{x} - \mathbf{b}||_2^2 &= (A\mathbf{x} - \mathbf{b})^T (A\mathbf{x} - \mathbf{b}) \\
&= (\mathbf{x}^T A^T - \mathbf{b}^T)(A\mathbf{x} - \mathbf{b}) \\
&= \mathbf{x}^T A^T A\mathbf{x} - \mathbf{b}^T (A\mathbf{x}) - (A\mathbf{x})^T \mathbf{b} + \mathbf{b}^T \mathbf{b} \\
&= \mathbf{x}^T A^T A\mathbf{x} - 2\mathbf{x}^T A^T \mathbf{b} + \mathbf{b}^T \mathbf{b} \\
&= \mathbf{x}^T Q\mathbf{x} - 2\mathbf{x}^T \mathbf{c} + ||\mathbf{b}||^2 \text{ as required.}
\end{aligned}$$

Problem 5

In class we mentioned that, under the right circumstances, Newton's method converges a lot faster than Gradient Descent. In this question we will show that, for the least squares problem, Newton's method converges in a single step!

1. Using your work, and the same notation, from Question 4, argue that:

$$\arg\min \|A\mathbf{x} - \mathbf{b}\|_2^2 = \arg\min f(\mathbf{x}) \quad \text{where } f(\mathbf{x}) = \mathbf{x}^\top Q\mathbf{x} - 2\mathbf{x}^\top \mathbf{c}$$

   The new formulation will make computing the Hessian easier.

2. Compute $\nabla f(\mathbf{x})$ and $\nabla^2 f(\mathbf{x})$, then write down and simplify the Newton update:

$$\mathbf{x}_1 = \mathbf{x}_0 - \left(\nabla^2 f(\mathbf{x}_0)\right)^{-1} \nabla f(\mathbf{x}_0).$$

3. Assume that Q is positive definite. Use the condition that for $\mathbf{x}_*$ to be a local minimizer, $\nabla f(\mathbf{x}_*) = 0$, to solve for the unique local minimizer of $f(\mathbf{x})$. *(Because $f(\mathbf{x})$ is strongly convex, this is also the unique global minimizer).*

4. Now verify that, no matter what $\mathbf{x}_0$ is, $\mathbf{x}_1$ is equal to $\mathbf{x}_*$.

Solution:

1. In problem 3 we found that $\nabla\|A\mathbf{x} - \mathbf{b}\|_2^2 = \nabla f(\mathbf{x})$ since $\|A\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{x}^\top Q\mathbf{x} - 2\mathbf{x}^\top \mathbf{c} + \|\mathbf{b}\|^2$ and the constant term vanishes when taking the gradient. Since $\|A\mathbf{x} - \mathbf{b}\|_2^2$ and $f(\mathbf{x})$ are convex, we can find their minimizers by finding their stationary points i.e. setting the gradients to 0. $\therefore \arg\min \|A\mathbf{x}-\mathbf{b}\|_2^2 = \arg\min f(\mathbf{x})$.

2. From problem 3 and part 1 above we have that $\nabla f(\mathbf{x}) = 2Q\mathbf{x} - 2\mathbf{c}$. Then note that $\nabla^2 f(\mathbf{x}) = 2Q$. So we have the Newton update

$$\mathbf{x}_1 = \mathbf{x}_0 - (2Q)^{-1}(2Q\mathbf{x}_0 - 2\mathbf{c}) = \mathbf{x}_0 - Q^{-1}(Q\mathbf{x}_0 - \mathbf{c}).$$

3. We want $\nabla f(\mathbf{x}_*) = 0$ i.e. $2Q\mathbf{x}_* - 2\mathbf{c} = 0 \implies Q\mathbf{x}_* = \mathbf{c} \implies \mathbf{x}_* = Q^{-1}\mathbf{c}$.

4. $\mathbf{x}_1 = \mathbf{x}_0 - Q^{-1}(Q\mathbf{x}_0 - \mathbf{c}) = \mathbf{x}_0 - (Q^{-1}Q\mathbf{x}_0 - Q^{-1}\mathbf{c}) = \mathbf{x}_0 - (I\mathbf{x}_0 - Q^{-1}\mathbf{c}) = \mathbf{x}_0 - \mathbf{x}_0 + Q^{-1}\mathbf{c} = Q^{-1}\mathbf{c} = \mathbf{x}_*$. $\therefore$ Newton's method converges in a single step for the least squares problem.

Problem 6

Convert the following linear programming problem to standard form:

$$\begin{aligned} \text{minimize } & 2x_1 + x_2 \\ \text{subject to: } & x_1 + x_2 \leq 3 \\ & x_1 + 2x_2 \leq 5 \\ & x_1 \geq 0 \text{ and } x_2 \geq 0 \end{aligned}$$

Solution: We first introduce slack variables $x_3 = 3 - x_1 - x_2$ and $x_4 = 5 - x_1 - 2x_2$ such that $x_3 \geq 0$ when $x_1 + x_2 \leq 3$ and $x_4 \geq 0$ when $x_1 + 2x_2 \leq 5$. Our problem then becomes minimizing $2x_1 + x_2$ with $\mathbf{x} > 0$ and $A\mathbf{x} = \mathbf{b}$ where $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$, $A = \begin{bmatrix} -1 & -1 & -1 & 0 \\ -1 & -2 & 0 & -1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} -3 \\ -5 \end{bmatrix}$.

Letting $\mathbf{c}^T = [2 \quad 1 \quad 0 \quad 0]$, our problem in standard form is then to minimize $\mathbf{c}^T\mathbf{x}$ subject to $\mathbf{x} \geq 0$ and $A\mathbf{x} = \mathbf{b}$.

Problem 7

Use the simplex method to solve the following problem:

$$\begin{aligned} \text{minimize } & -x_1 - x_2 - 3x_3 \\ \text{subject to: } & x_1 + x_3 = 1 \\ & x_2 + x_3 = 2 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

starting with the basis $B = \{1, 2\}$. For each iteration of the method, clearly display B, $\mathbf{x}_B$, all δ_j used, $\epsilon_{\max}$ and i^*. *(See the lecture notes for definitions of these quantities.)*

Solution: We first write the problem in standard form i.e. we want to minimize $\mathbf{c}^T\mathbf{x}$ subject to $A\mathbf{x} = \mathbf{b}$ and $\mathbf{x} \geq 0$, where $\mathbf{c} = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix}, A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \mathbf{x} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$.

For the **first iteration** of the method we use the given basis $B = \{1, 2\}$. So $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = I \times \mathbf{b} = [1 \quad 2]^T$. Since $j \in D = B^c = \{3\}$, we have one edge $\mathbf{y} = \mathbf{x} + \epsilon\delta_3$ where $\delta_3 = \begin{bmatrix} -\mathbf{B}^{-1}\mathbf{a}_3 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}$. The reduced unit cost along this edge is then $\mathbf{c}^T\delta_3 = [-1 \quad -1 \quad -3] \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} = 1 + 1 - 3 = -1$. Since the unit cost is negative, this is a viable edge for us to choose. We hence proceed by calculating $\epsilon_{\max}$ and i^* to find the new basic feasible solution.

$$\begin{aligned} \epsilon_{\max} &= \max\{\epsilon : x_i - \epsilon(\mathbf{B}^{-1}\mathbf{a}_3)_i > 0 \text{ for all } i \in B\} \\ &= \max\{\epsilon : x_i - \epsilon(\begin{bmatrix} 1 \\ 1 \end{bmatrix})_i > 0 \text{ for all } i \in B\} \\ &= \max\{\epsilon : x_i - \epsilon > 0 \text{ for all } i \in B\} \\ &= 1 \text{ since } x_1 = 1 \text{ and } 1 \in B. \end{aligned}$$

$$\begin{aligned} i^* &= \arg\min\{\frac{x_i}{(\mathbf{B}^{-1}\mathbf{a}_3)_i}, i \in B \text{ and } -(\mathbf{B}^{-1}\mathbf{a}_3)_i < 0\} \\ &= 1 \text{ since } \frac{x_1}{(\mathbf{B}^{-1}\mathbf{a}_3)_1} = 1 < 2 = \frac{x_2}{(\mathbf{B}^{-1}\mathbf{a}_3)_2} \end{aligned}$$

So our new basis $B = \{2, 3\}$ and the new corner $\mathbf{x}^{\text{new}} = \begin{bmatrix} \mathbf{x}_B - \epsilon_{\max}\mathbf{B}^{-1}\mathbf{a}_3 \\ \epsilon_{\max} \times 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$.

For the **second iteration** we have $\mathbf{B} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ so that $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. We only have one edge $\mathbf{y} = \mathbf{x} + \epsilon\delta_1$ where $\delta_1 = \begin{bmatrix} 1 \\ -\mathbf{B}^{-1}\mathbf{a}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$. The reduced unit cost along this edge is then $\mathbf{c}^T\delta_1 = [-1 \quad -1 \quad -3] \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} = -1 - 1 + 3 = 1$. Since the reduced unit cost along the only possible edge is positive, $\mathbf{x} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ is the optimal solution to this problem and the simplex method terminates.