MS&E 211X/MS&E 111X/ENGR 62X
Introduction to Optimization (Accelerated) HW5
Course Instructor: Yinyu Ye
Due Date: 21st November 2019, 5pm

Please submit your homework through Canvas. We allow teamwork for coding-related Problems 4 and 5, with team size of up to 3 students. If you choose to work in teams for those questions, please indicate all your teammates in your submission. Please note: late homework will not be accepted. Each problem below is worth 10 points.

1 True or False

State whether the following propositions are True or False. If True, please provide your brief justification. If false, please provide your reasoning or a counterexample.

(a) If we only follow simplex method, we will never pick the same variable twice as incoming variable for any LP.

(b) If we start at the point $(0, 0)$, Newton's method converges within 1 step for the following optimization:

$$\min \quad f(x_1, x_2) = x_1^2 - 8x_1 + x_2^2 + 16 \tag{1}$$

(c) If we start at the point $(0, 0)$, steepest gradient descent with line search converges within 1 step for the optimization problem in part (b).

(d) $g(x) = \sqrt[p]{x}$ defined on $x \in [0, \infty)$ might be First-Order Lipschitz for some integer p if $p > 1$.

2 Simplex method for transportation problem

Solve the transportation problem described in page 14, Lecture 10 using simplex method. Specifically, write down the shadow prices, reduced costs, the termination test and the new BFS (if exists) for each iteration. Clearly state your optimal solution at the end.
[Hint: You can begin with the BFS provided in page 21, Lecture 10.]

3 Newton's method

Newtons's method is fast and conceptually straightforward, but it might not work because it relies on the objective function to have "nice" derivatives. In this problem, we probe the areas where Newton's method may fail.

(a) (Root finding) Consider the following function:

$$f(x) = x^3 - 2x + 2, \quad x \in \mathbb{R} \tag{2}$$

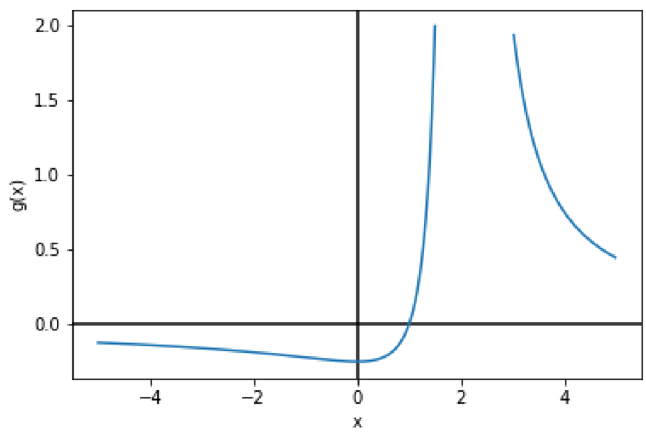
Choosing $x_0 = 0$ as starting point, can you find the root of $f(x) = 0$ by Newton's method? Show your iterations of Newton's method. If you think the algorithm succeeds, report your root. Otherwise, briefly state why it fails.

(b) (Optimization) Consider the following problem:

$$\begin{aligned} \min \quad & g(x) = \frac{x-1}{(x-2)^2} \\ s.t. \quad & x \neq 2 \end{aligned} \tag{3}$$

For your reference, $g'(x) = -\frac{x}{(x-2)^3}$ and $g''(x) = \frac{2(x+1)}{(x-2)^4}$. As can be seen from the following plot, there is a unique minimizer for $g(x)$:

Figure 1: Plot of $g(x)$



Without seeing the plot of $g(x)$ or knowing the derivatives $g'(x)$ and $g''(x)$, your friend suggested 2 potential candidates for starting points x_0. Can you find the minimizer using Newton's method, if you begin with these initial points?

(i) $x_0 = -1$

(ii) $x_0 = 3$

For each of these initial points: if you think the algorithm succeeds, report your iterations and optimizer; otherwise, briefly state why Newton's method fails. You can use the plot of g and the derivatives directly in your arguments.

4 Coding up Newton's method

Now we implement the Newton's method for optimization. You are free to use whatever programming language you prefer, but you cannot use any off-the-shelf Newton's method implementation such as `scipy.optimize.newton` in `python`.

We stop iterating at the earliest round when the objective value improvements drop below 10^{-3} or we reach 1000^{th} round[1]:

$$k_{stop} = \min\{k : |g(x_{k+1}) - g(x_k)| < 0.001, k \leq 1000\} \tag{4}$$

(a) (Implementation) Implement Newton's method for optimization in Problem 3 Part (b). Run your program using the second suggested initial point $x_0 = 3$. Draw and report the following 2 plots:

(1) horizontal axis: the iterations k, vertical axis: x_k;

(2) horizontal axis: the iterations k, vertical axis: $g(x_k)$;

(b) (Interpretations) Reading the plots, which of the stopping criterion was triggered? Do the plots confirm your answer in Problem 3 Part (b)?

(c) (Try other initial points) For Problem 3 Part (b), propose your own initial point that is not $x = 0$, such that stopping criterion are reached within 5 iterations. Produce and report the same 2 plots as Problem 4 Part (a).

(d) Attach/print your code for implementations of this problem.

[1] Remark: In real applications, people can tweak these stopping criterion based on their precision needs, but we set the criterion in (4) for you in this exercise for convenience.

5 Gradient method

Recall the logistic regression in Homework 3 Problem 3: we have already calculated its gradient and explored some of this problem's property. Now, we consider numerically solving it using gradient descent. You are free to use whatever programming language you prefer, but you cannot use any off-the-shelf gradient descent implementation such as `scipy.optimize` in `python`.

(a) (Implementation) We fix step size as 0.1 and we stop iterating at the earliest round when the Euclidean norm of gradient drops below 10^{-3} or we reach 1000^{th} round:

$$k_{stop} = \min\{k : \|\nabla_{\mathbf{x}^{(k)}, x_0^{(k)}} f(\mathbf{x}^{(k)}, x_0^{(k)})\|_2 < 0.001, k \leq 1000\} \tag{5}$$

Write code for fixed step size gradient descent for the problem in Homework 3 Problem 3 Part (a). You are free to choose your starting point of $(\mathbf{x}^{(0)}, x_0^{(0)})$. After running the algorithm, please produce and report the following 2 plots:

(1) horizontal axis: the iterations k, vertical axis: Euclidean norm of $[\mathbf{x}, x_0]$;

(2) horizontal axis: the iterations k, vertical axis: the value of negative log-likelihood function $f(\mathbf{x}^{(k)}, x_0^{(k)})$;

Do you think these 2 plots agree with with your theoretical results in Homework 3 Problem 3 Part (a)?

(b) (Regularization) Modify your code to add the regularization proposed by Homework 3 Problem 3 Part (b), which is

$$\min f(\mathbf{x}, x_0) + \rho\|\mathbf{x}\|_2^2 \tag{6}$$

Produce the same plots as Part (a) with $\rho_1 = 0.01$ and $\rho_2 = 1$ (i.e. please report 4 plots for this subsection, 2 plots for each regularization scale).

(c) (Interpretation) Compare the results in Parts (a) and (b) and summarize what effects of regularization you notice.

(d) Attach/print your code for implementations of this problem.