Convex Optimization Homework #3

Due: Friday June 26, 2020, 6pm.

Version 1.0 was Posted on June 7, 2020.

The current version is Version 2.0, Updated June 16, 2020. (Click here to report any typo/error you may find.)

1. (100% + Bonus 40%) In this problem, we aim to apply the barrier method we learned in class to solve the linear constrained problem

$$\begin{array}{ll} \text{minimize} & f_0(x) = \sum_{i=1}^{k} e^{a_i^T x - b_i} \\ \text{subject to} & Cx \preceq d \end{array} \tag{1}$$

where $d \in \mathbf{R}^m$, $C = \begin{bmatrix} c_1 & \cdots & c_m \end{bmatrix}^T \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^k$, and $A = \begin{bmatrix} a_1 & \cdots & a_k \end{bmatrix}^T \in \mathbf{R}^{k\times n}$, and $k > n$.

A special case of this is when we choose $k = 3, n = 2$, and

$$A = \begin{bmatrix} 1 & 3 \\ 1 & -3 \\ -1 & 0 \end{bmatrix}, b = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \end{bmatrix}, \tag{2}$$

then it reduces to Eq. (9.20) in the textbook:

$$f_0(x) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}.$$

(a) (5%) Let $f_0(x) = \sum_{i=1}^{k} e^{a_i^T x - b_i}$, $f_i(x) = c_i^T x - d_i$, and $\phi(x) = -\sum_{i=1}^{m} \log(-f_i(x))$. Let $f(x) = tf_0(x) + \phi(x)$ for any $t > 0$. Find **dom** f and derive $\nabla f(x)$ and $\nabla^2 f(x)$.

(b) (5%) Write a matlab function which takes inputs A, b, C, d, x, and t, and evaluates the function f at the point x, as well as the gradient and the Hessian.
Hint: the function can have a header that looks like the following.

```
function [f,g,H] = my_objective_with_log_barrier(x, C, d, A, b, t)
```

Then, prepare to write an m-file as the main function for your program to solve Problem (1). Let $m = 2$, and choose $C = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix}$, $d = \begin{bmatrix} 0.3 \\ 0.2 \end{bmatrix}$.

(c) (5% in total for (c)-(k)) Set the initial point $x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and let $t = 1$. Let $l = 0$ denoting the Newton iteration index.

(d) Use the Newton step's formula: $\Delta x_{\text{nt}} = -(\nabla^2 f(x))^{-1} \nabla f(x)$ and calculate the Newton step $\Delta x_{\text{nt}}^{(l)}$.

(e) Calculate the Newton decrement $\lambda^{(l)}(x) = (-\nabla f(x^{(l)})^T \Delta x_{\text{nt}}^{(l)})^{1/2}$. Set the duality gap as $g_{\text{dual}}^{(l)} = m/t$.

(f) Perform backtracking line search along search direction using $\beta = 0.7$ starting from $s = 1$ until $x^{(0)} + s\Delta x_{\text{nt}}^{(0)} \in$ **dom** f. (Note: $s^+ := \beta s$) [1]

(g) Continue the backtracking line search until $f(x + s\Delta x_{\text{nt}}^{(0)}) \leq f(x) - \alpha s \lambda(x)^2$, where $\alpha = 0.1$.

(h) Perform the update $x^{(l+1)} = x^{(l)} + s\Delta x_{\text{nt}}^{(l)}$. Determine whether $\lambda^2/2 \leq \epsilon_{\text{inner}}$ where $\epsilon_{\text{inner}} = 10^{-5}$.

(i) Let $l = l + 1$ and repeat (d)-(h) for the next Newton iteration. Wrap up these steps in an inner loop. For the lth iteration, record the following items for future use: (1) $f(x^{(l)})$, (2) $\lambda(x^{(l)})$, (3) $s^{(l)}$, (4) $g_{\text{dual}}^{(l)}$. Let $l := l + 1$ at the end of each Newton iteration until $\lambda^2/2 \leq \epsilon_{\text{inner}}$ as in (h).

(j) Write an outer loop that include 1) steps of the inner loop obtained from (d) to (i); 2) Updating $t^+ := \mu t$ where $\mu = 20$; 3) Checking if the stopping criterion $m/t \leq \epsilon_{\text{outer}}$ is met, where $\epsilon_{\text{outer}} = 10^{-10}$. The Newton iteration index l continues to grow and does not reset to 0 in the event of a new outer iteration.

(k) Run your code, and record the following items for future use: 1) the number of outer iterations; 2) for each outer iteration, the number of inner Newton steps; 3) for each outer iteration, record t and the function value $f(x)$ at the end of the outer iteration. 4) the total number of Newton steps; 5) The optimal point $x^*(t)$ obtained at the end of the last outer iteration.

(l) (10%) Draw the central path on the $\mathbf{R}^2$ plane by connecting all the central points $x^*(t)$ for every $t = \mu^l t^{(0)}, l = 0, 1, ...$ that you collected from the end of each outer iterations. Draw also on the same plot the m hyperplanes defined by $Cx = d$ (as m lines), which represent the boundary of the feasible set of the original problem (1). Obtain two pictures of the plot with different zooms according to the following rules.

i. Zoom to a proper scale so that the whole path from $t = 1$ to $t \to \infty$ can be observed.

[1] Here we choose to use the letter s as the line search parameter instead of t, with the consideration to avoid confusion between the parameter t associated with the logarithm barrier function.

ii. Zoom in into a detailed scale around the optimal point x^* so that at least the last three central points you found can be visually distinguished on the plot.

(m) (5%) Make a comment on the relationship between the boundary and the central path. In particular, determine the number of inequality constraints that are active according to the plot you generated.

(n) (10%) Draw the curves of duality gap versus the number of Newton iterations in the log scale (e.g., Use `semilogy` instead of `plot` in Matlab), for $\mu = 2, 20, 200, 600, 2000$. Comments on the convergence performance for various values of μ.

(o) (Bonus, 10%) Run the `cvx` toolbox to solve the same problem. Compare your results (i.e., optimal value, optimal point) with those you obtain from `cvx`.

(p) (Bonus[2], 10%) For the case $\mu = 20$, plot the value $\lambda(x^{(l)})^2/2$ versus l in the log scale. On the same plot, draw the duality gap versus l. Comment on the relationship of these two curves in the plot.

(q) (Bonus, 10%) For the case $\mu = 20$, plot the backtracking parameter $s^{(l)}$ versus l in the log scale. On the same plot, draw the duality gap versus l. Comment on the relationship of these two curves in the plot.

(r) (Bonus, 10%) For the case $\mu = 20$, plot the objective function value $f_0(x^{(l)})$ versus l. Does the function value $f_0(x^{(l)})$ always decrease as l grows? If not, can we still argue that the method we use is a descent method? Make your comments to justify your observations.

(s) (30%) Let C still be $C = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix}$, but change d to $d = \begin{bmatrix} 0.3 \\ 0.3 \end{bmatrix}$. Repeat (c)-(n).

(t) (30%) Let C still be $C = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix}$, but change d to $d = \begin{bmatrix} 0.4 \\ 0.4 \end{bmatrix}$. Repeat (c)-(n). (*Hint: In this case, the inequality constraints will be* **inactive**, *and so the optimal point is supposed to coincide with that of the unconstrained problem in HW# 2*)

Homework submission guidelines:

- Submit your answer online as a set of m-files (in *.m) and a document file (in *.pdf) that contains all answers (including plots) in this problem set.
- If you choose to do the homework in Python, then submit *.py files instead of m-files.
- Submit your files online at the Ceiba website. No paper shall be handed in.
- Late submissions will be treated under the principle elaborated as follows.
  (1) Homework received by 6pm, June 26 (t_1) will be counted fully.
  (2) Homework received after 9pm, June 26 (t_2) will not be counted.
  (3) Homework received between t_1 and t_2 will be counted with a discount rate
  $$\frac{t_2 - t}{t_2 - t_1}$$
  where t is the received time. Note that $t_2 - t_1$ is three hours.
- Plagiarism is strongly prohibited. While discussions among classmates are allowed (and encouraged), you shall not ask anyone else to share his/her codes with you, nor should you attempt to share with anyone your codes. If any part of your submission is found to be copied from someone else's submission, then both of your homework submissions will be counted zero.

[2]Bonus problems (p)(q)(r) will be graded only when the results you get match that of the `cvx` toolbox in subproblem (o).