MS&E 211X/MS&E 111X/ENGR 62X
Introduction to Optimization (Accelerated) HW6
Course Instructor: Yinyu Ye
Due Date: 5th December 2019, 5pm

Please submit your homework through Canvas. Note: late homework will not be accepted. Each problem below is worth 10 points.

We allow teamwork for coding-related Problems 3∼5, with team size of up to 3 students. If you choose to work in teams for Problems 3∼5, please indicate all your teammates in your submission.

For Problems 3∼5, you may use whatever programming language you like, but you cannot use any off-the-shelf package to directly solve it. You are allowed to use and modify any code provided in the lecture slides or problem sessions. Please attach/print your code for Problems 3∼5 at the end of your submission.

1 Gradient ascent

Consider a problem of the form

$$\begin{aligned} \max \quad & f(\mathbf{x}) \\ s.t. \quad & \mathbf{x} \geq 0 \end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbf{R}^n$ and we assume f is differentiable on $\mathbf{x} \geq 0$ in this problem. A gradient-type procedure has been suggested for this kind of problem that accounts for the inequality constraint. At a given point $\mathbf{x} = (x_1, x_2, \cdots, x_n)$, we can calculate the gradient $\nabla f(\mathbf{x}) = \mathbf{g} = (g_1, g_2, \cdots, g_n)$ and then decide the direction $\mathbf{d} = (d_1, d_2, \cdots, d_n)$ by:

$$d_i = \begin{cases} g_i & x_i > 0 \text{ or } g_i > 0 \\ 0 & x_i = 0 \text{ and } g_i \leq 0 \end{cases} \tag{2}$$

This direction is then used as a direction of search in the usual manner.

(a) What are the first-order KKT conditions for a maximum point of this problem?

(b) Show that $\mathbf{d}$, as determined by the algorithm, is zero only at a point satisfying the first-order conditions. (recall that $\mathbf{d} = 0$ means every coordinate of $\mathbf{d}$ is 0)

(c) Show that if $\mathbf{d} \neq 0$ (i.e. $\mathbf{d}$ has at least one coordinate not equal 0), it is possible to increase the value of $f(\mathbf{x})$ by movement along $\mathbf{d}$.

2 Central path

Recall we can solve an LP by adding log-barrier with $\mu > 0$:

$$\begin{array}{ll} \min & LP(\mathbf{x}) = c^T\mathbf{x} \\ s.t. & A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0 \end{array} \quad \Rightarrow \quad \begin{array}{ll} \min & BLP_\mu(\mathbf{x}) = c^T\mathbf{x} - \mu \sum_{j=1}^{n} \log(\mathbf{x}_j) \\ s.t. & A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0 \end{array} \tag{3}$$

If $\mathbf{x}(\mu)$ is the minimizer of $BLP_\mu(\mathbf{x})$, the central path can be constructed by solving a sequence of $BLP_{\mu_i}(\mathbf{x})$ with decreasing $\mu_1 > \cdots > \mu_i > \mu_{i+1} > \cdots > 0$.

Using this method, we now consider solving a slightly simpler variant of Question 4 in Problem Session 2. In particular, let us consider a 4-variable version of the original Question 4:

$$\begin{array}{ll} \min & -x_1 - 2x_2 \\ s.t. & x_1 + x_3 = 1 \\ & x_2 + x_4 = 1 \\ & x_i \geq 0, i = 1, 2, 3, 4 \end{array} \tag{4}$$

(a) Find the central path point $\mathbf{x}(\mu)$ for a general $\mu > 0$. Notice that $\mathbf{x}(\mu) \in \mathbf{R}^4$ since we have 4 decision variables.

(b) What is the analytic center of the feasible region with the logarithmic barrier function? [Hint: the techniques for solving this subproblem are covered in Lecture 13, page 10]

(c) As stated before, we can use BLP to solve LP. Compute central path points $\mathbf{x}(\mu_i)$ for $i = 1, 2, 3, 4$ with $\mu_1 = 10$, $\mu_2 = 1$, $\mu_3 = 0.5$, $\mu_4 = 0.1$ and compute the LP objective values $LP(\mathbf{x}(\mu_i)) = -x_1(\mu_i) - 2x_2(\mu_i)$ for $i = 1, 2, 3, 4$. Report your results up to 0.001 accuracy by filling in the following table:

μ	x_1	x_2	x_3	x_4	$LP(\mathbf{x}(\mu))$
10					
1					
0.5					
0.1					

Observe and then interpret the pattern you see in the last column, i.e. how the objective values of the original LP changes as μ changes and why this happens.

3 Solving robust portfolio management problem

Recall in Homework 4 Problem 5, we turned a robust portfolio management problem into a standard quadratic programming problem as a joint single layer minimization. Now, let us think of a close variant of Homework 4 Problem 5 by keeping all the constraints, but changing the objective to:

$$\begin{aligned}
\min_{x_1,x_2} \quad & [x_1^2 + 2x_2^2 - x_1x_2 + \max_{\mu_1,\mu_2}(-x_1\mu_1 - x_2\mu_2)] \\
s.t. \quad & x_1 + x_2 = 1 \\
& 3x_1 - x_2 \geq 0 \\
& \mu_1 + 3\mu_2 = 2 \\
& |\mu_1 - \mu_2| \leq 1
\end{aligned} \tag{5}$$

In other words, we only change the cross-product term from $-2x_1x_2$ to $-x_1x_2$ in the objective function.

(a) Find the joint single layer minimization problem.

(b) With starting point $(x_1, x_2) = (0.5, 0.5)$ and $\beta = 10$, write code to perform gradient-solution projection for 100 iterations. Report your final (x_1, x_2) up to 0.001 accuracy. State whether you think the algorithm has converged and briefly explain why.

(c) Interpret the meaning of your final (x_1, x_2) value calculated in Part (b), under the setting of robust portfolio management.

4 Solving Regularized SVM

Recall in Homework 3 Problem 4, we put an l_2 regularization on SVM:

$$\begin{aligned}
\min \quad & f(\delta, x_1, x_2, x_0) = \delta + 0.001\|\mathbf{x}\|^2 \\
s.t. \quad & \mathbf{a}_i^T\mathbf{x} + x_0 - 1 + \delta \geq 0, i = 1, 2 \\
& -\mathbf{b}_j^T\mathbf{x} - x_0 - 1 + \delta \geq 0, j = 1, 2 \\
& \delta \geq 0
\end{aligned} \tag{6}$$

We are still given the exactly same set of data $\mathbf{a}_1 = (1; 0); \mathbf{a}_2 = (0; 1); \mathbf{b}_1 = (-1; 0); \mathbf{b}_2 = (0; -1)$. Now, we try solving it using gradient-solution projection.

(a) Give gradient-solution projection update iteration for the problem in (6). You do not need to plug in the data for $\mathbf{a}_i$'s and $\mathbf{b}_j$'s into your iterations, but you can use the data to simplify the expressions.

Hint: the gradient-solution projection introduced in class can make sure gradient descent remains feasible for $\mathbf{x} \geq 0$ type of inequality constraints. Here in the regulated SVM, we have a more complex set of inequality constraints, so you need to modify the lower-bound projection part of the gradient-solution projection algorithm to suit our problem.

(b) Implement the gradient-solution projection for the problem in (6). To facilitate accuracy, we use a dynamic step scheme in which $\beta = 2 + \frac{t}{100}$ where t is the iteration count. By making β larger, we will be making smaller steps along the gradient direction when t is becoming larger.

With starting point $(\delta, x_1, x_2, x_0) = (3, 0.0001, 0.0001, 0)$, run the algorithm for 100 iterations so that each variable is updated 100 times and plot the objective $f(\delta, x_1, x_2, x_0)$ against the iterations. State whether you think the algorithm has converged and briefly explain why.

5 Solving Fisher price equilibrium social optimization

Recall that in Lecture 14, we learned the following Fisher Price equilibrium social optimization and saw that it can be numerically solved by ADMM:

$$\begin{aligned} \min \quad & F(x_1, x_2, x_3, x_4) = -5\log(2x_1 + x_3) - 8\log(3x_2 + x_4) \\ s.t. \quad & x_1 + x_2 = 1 \\ & x_3 + x_4 = 1 \\ & x_i \geq 0, i = 1, 2, 3, 4 \end{aligned} \tag{7}$$

In this problem, we strengthen our understanding of this problem by solving a slight more complex utility function:

$$\begin{aligned} \min \quad & G(x_1, x_2, x_3, x_4, x_5, x_6) = -5\log(2x_1 + x_3) - 8\log(3x_2 + x_4) - 2\log(x_5 + x_6) \\ s.t. \quad & x_1 + x_2 + x_5 = 1 \\ & x_3 + x_4 + x_6 = 1 \\ & x_i \geq 0, i = 1, 2, 3, 4 \end{aligned} \tag{8}$$

In class, we saw that in order to use 2-block ADMM for the problem in (7), we need to first set $u_1 = 2x_1 + x_3$, $u_2 = 3x_2 + x_4$ and then add slacks $s_1 \sim s_4$.

Similarly, in order to apply 3-block ADMM for the new problem in (8), we need to set another intermediate $u_3 = x_5 + x_6$, and add slacks s_5 and s_6. Then, we pick $(x_1, x_2, x_3, x_4, x_5, x_6)$ as the first block X_1, (u_1, u_2, u_3) as the second block X_2 and $(s_1, s_2, s_3, s_4, s_5, s_6)$ as the third block X_3.

Implement 3-block ADMM for the new problem and run it for 100 rounds (i.e. each variable is updated 100 times). Produce the plot with horizontal axis: number of iterations; vertical axis: the l_2 norm of error. (recall the l_2 norm of error is $\|A_1X_1 + A_2X_2 + A_3X_3 - b\|_2$, and indicates how much we deviate from the equality constraints)