Optimization Theory and Algorithms

Final Project: Solving a QCQP

(Due on December 19, 2018, 5pm)

A QCQP problem:

Let $A \in \mathbb{R}^{n\times n}$ be symmetric and $b, c \in \mathbb{R}^n$ for $n \geq 2$. Consider the following quadratic-constrained, quadratic program (QCQP):

$$\min_{x,y\in\mathbb{R}^n} \quad f(x,y) = \frac{1}{2}\left(x^TAx + y^TAy\right) - b^Tx - c^Ty \tag{1}$$

$$\text{s.t.} \quad h_i(x,y) = 0,\ i = 1,2,3 \tag{2}$$

where A usually has structures such as sparsity or low-rankness, and

$$h_1(x,y) = \frac{1}{2}(x^Tx - 1),\ \ h_2(x,y) = \frac{1}{2}(y^Ty - 1),\ \ h_3(x,y) = x^Ty. \tag{3}$$

In this project, we will assume that A is sparse and negative definite (which is actually not restrictive).

Project Description

Write one (or more) Matlab function to solve the above QCQP problem. You may try any method of your choice, but one of the following 3 options is suggested (each allowing further variants):

1. an alternating direction method of multipliers (ADMM),
2. an augmented Lagrangian multiplier method (ALMM),
3. a quadratic penalty method.

Each approach has the flexibility of allowing different strategies in formulating and solving unconstrained or constrained subproblems.

Test scripts and codes by the instructor, such as `test_qcqp.m` and `yz_qcqp_admm.p`, will be posted on the course website. You can run these test scripts with or without your code. As you can tell from the name, `yz_qcqp_admm` implements an ADMM algorithm which can solve large-scale problems with, for example, $n = 250000$ or even over a million if your computer is capable (for one thing, this code only use A in matrix-vector multiplications). Once your code is in place, you will be able to use the test scripts to compare your code with the the instructor's on either small-scale or, hopefully, relatively large-scale problems.

Let $\mathcal{L}(x, y, \lambda)$ be the Lagrangian function of the QCQP problem for $\lambda \in \mathbb{R}^3$. The stopping criterion is

$$\max\left\{\frac{\|\nabla_x\mathcal{L}(x,y,\lambda)\|}{1+\|b\|}, \frac{\|\nabla_y\mathcal{L}(x,y,\lambda)\|}{1+\|c\|}, \|h(x,y)\|\right\} \leq \texttt{tol}.$$

Therefore, your code need to calculate and check the above 3 residual quantities at each iterations. The instructor's code `yz_qcqp_admm` prints out the 3 quantities at the frequency of every 10 iterations.

Program Requirements

The header of your function should be of the form:

```
function [x,y,lamb,out] = my_qcqp_xxxx(A,b,c,tol)
%
% Solving the QCQP problem:
% ----------------------------------------
%     min x'*A*x/2 - b'*x + y'*A*y/2 - c'*y
%    st. (x'*x-1)/2=0, (y'*y-1)/2=0, x'*y=0
% ----------------------------------------
%
% input:
%        (A,b,c) = QCQP problem data
%        tol = tolerance for stationarity
%
% output:
%        (x,y) = computed solutions
%        lamb  = [1 by 3] 3 multipliers
%         out  = a struct with fields:
%           iter : number of iterations taken
%           converged : true or false
%           rres : [iter by 3] 3 relative residual
%                  quantities at every iterations
```

where `xxxx` stands for one of the 3 indicators: `admm`, `almm`, `qpen`, or some other four-letter indicators showing the method you use.

Your code should be *properly modularized and briefly documented*. For example, a separate function should be created for solving a significant subproblem that will be solved multiple times.

Project Specifications

1. Normally, students need to write their own solvers and run the test script `test_qcqp.m` for different combinations of n and s. Three levels of scores will be given depending on the capacities and performance of a submitted code. Along with a complete submission and a well-written project report, a student can receive up to 80 points (out of 100) for a level-1 code, 90 points for a level-2 code, and 100 points for a level-3 code (which will be quite challenging).

    - Level-1: Within a reasonable amount of time, you code is able to correctly solve medium-sized problems with n up to a few thousands and for s values ranging from 0.1 to 10. To test your code, run `test_qcqp` with $n = 3600$ and $s = [0.1, 1, 10]$ (all 3 values).
    - Level-2: In addition to level 1, within a reasonable amount of time you code is able to correctly solve larger-sized problems with n up to tens of thousands. Run `test_qcqp` with $n = 10000$ and $s = [0.1, 1, 10]$, $n = 40000$ and $s = [1, 10]$. Also try $n = 90000$ and $s = [1, 10]$.

- Level-3: In addition to level 2, within a reasonable amount of time you code is able to correctly solve really large problems with n up to a million. Run `test_qcqp` with $n = 250000$ and $s = [1, 10]$. Also try n equal to a million and $s = 10$.

Before running the test scripts on your codes, you may need to modify the scripts to specify the correct name in the definitions of the `Solvers` array for your to-be-tested code.

2. At the very least, one might write a function `my_qcqp_fmin.m` that uses the Matlab function `fmincon` in the Optimization Toolbox to solve small-sized problems. For this purpose, one can run the test script `test_qcqp0` for $n = 400$ and $n = 900$, which also calls the instructor's code `yz_qcqp_fmin.p` for comparison. However, using a Matlab solver or a solver from any outside source will only entitle a student to receive at most 50 points (out of 100).

3. It is important that *all test results should be generated by a single unchanged code*, without hand-tuning parameters each time for a different test run.

4. Submit your code, the printout from the required test runs as specified above (you may truncate a middle portion of the iteration output if it is too long), and selected output graphs. In addition, upload your code to the Black Board system online.

5. Write (typeset) a Project Report, of no more than 2 pages, to describe what you have done, summarize your results, and offer relevant comment on the project. The report should be an honest assessment of your work on this project. The quality of the report will be reflective of your quality of learning from the project and in the course. It will weigh quite heavily on your grade for the project.

Individual Effort Requirements

Parts of the project must be *individual effort done by each student independently*.

1. Codes and project report must be written individually. Copying, borrowing or sharing any part of these two items is strictly prohibited.

2. Before and on December 7th, you may exchange ideas and discuss methodologies with each other in the class. After December 7th, no discussions on the project should be allowed, except possibly with the instructor and the TA.

3. Anyone found in violation of the above rules will be subject to heavy penalty.