Project 2 for "Convex Optimization"

Zaiwen Wen
Beijing International Center for Mathematical Research
Peking University

December 18, 2016

1 Semismooth Newton Algorithms for LP

Consider the standard form of LP

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & c^T x \\ s.t. \quad & Ax = b, \\ & x \geq 0, \end{aligned} \tag{1.1}$$

where $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ are given. The dual problem is

$$\begin{aligned} \max_{y\in\mathbb{R}^m, s\in\mathbb{R}^n} \quad & b^T y \\ s.t. \quad & A^T y + s = c, \\ & s \geq 0. \end{aligned} \tag{1.2}$$

1. Write down and implement an augmented Lagrangian method for solving (1.2). Item (a) is required. Choose to do either (b) or (c). Extra credit will be awarded if (c) is chosen.
   (a) Write down an augmented Lagrangian method for solving the dual problem (1.2), where the variable s is eliminated (i.e., the variable s should not appear in the update of the algorithm).
   (b) Method 1: Apply a gradient-type method to minimize each augmented Lagrangian function. It can be a method from Homework 5.
   (c) Method 2: Write down a semi-smooth Newton method for minimizing each augmented Lagrangian function. A reference is:
   Zhao, Xin-Yuan, Defeng Sun, and Kim-Chuan Toh. "A Newton-CG augmented Lagrangian method for semidefinite programming." SIAM Journal on Optimization 20.4 (2010): 1737-1765.
   `http://epubs.siam.org/doi/abs/10.1137/080718206`.
2. Semi-smooth Newton method based on solving a fixed-point equation. Items (a) and (b) are required. Item (c) is for extra credit.
   (a) Write down and implement the DRS for (1.1) and ADMM for the dual problem of (1.2).

(b) Derive the explicit relationship between the variables of DRS and ADMM mentioned above.

(c) Write down and implement a regularized semi-smooth Newton method for solving (1.1). A reference is Xiao, Xiantao, et al. "A Regularized Semi-Smooth Newton Method With Projection Steps for Composite Convex Programs." arXiv preprint arXiv:1603.07870 (2016) `https://arxiv.org/abs/1603.07870`.

3. Requirement:

(a) The interface of each method should be written in the following format

```
[x, out] = method_name(c, A, b, opts, x0);
```

Here, c, A and b are given data, opts is a struct which stores the options of the algorithm, out is a struct which saves all other output information. The parameter x0 is an optional given input initial solution. In other words, x0 is not necessarily required as an input.

(b) Test problems:

- Rand data:

```
n = 100;
m = 20;
A  = rand(m,n);
xs = full(abs(sprandn(n,1,m/n)));
b  = A*xs;
y  = randn(m,1);
s  = rand(n,1).*(xs==0);
c  = A'*y + s;
```

- Extra Credit (not required to test this set of problems): Netlib test problems. A matlab version of these data can be found at:
  `http://bicmr.pku.edu.cn/~wenzw/code/MPS-presolve-mat.zip`

(c) Compare the efficiency (cpu time) and accuracy (checking optimality condition) with the LP solvers in Mosek or Gurobi.

(d) Prepare a report including

- detailed answers to each question
- numerical results and their iterpretation

(e) Pack all of your codes in one file named as "proj2-name-ID.zip" and send it to both me and TA:
wendouble@gmail.com
pkuopt@163.com

(f) If you get significant help from others on one routine, write down the source of references at the beginning of this routine.