Stony Brook University
CSE592 – Convex Optimization – Spring 18
Homework 3, Due: April, 10, 2018, 11:59PM

March 27, 2018

Instructions

- The homework is due on April 10, 2018. Anything that is received after the deadline will not be considered.
- The write-up **must** be prepared in Latex or Word and converted to pdf. No scanned hand-written notes!
- We can use any Latex class you like, just report question number and your answer.
- If the question requires you to implement a Python function, you must also submit a file with the implementation. Make sure it is sufficiently well documented that the TAs can understand what it is happening.

Description

In this assignment, you will experiment with the log-barrier method. The included archive contains partial Python code, which you must complete. Areas that you will fill in are marked with "TODO" comments.

You should turn in an archive containing all of your source code, and a document containing all plots, and answers to underlined questions.

Please remember to turn in a complete and readable document which contains your plots, and in which you answer the questions (in particular, talk about what your plots mean)! Your grade will be based both on the document and on your source code.

1 Log-barrier function

Most of the functions in the file "algorithms.py" are (filled-in) versions of the functions in the assignment 2. The function in "objective_log_barrier" in "algorithms.py" takes as parameters an objective function f_0, a constraint function ϕ (which has the same calling convention as the objective function), a vector x, and a scaling parameter t, and returns the value, gradient and Hessian of the log-barrier objective:

$$f(x,t) \;=\; tf_0(x) + \phi(x)$$

Make sure that you understand the contents of all of these files.

1.1 Implementation

The functions "quadratic_log_barrier" in "hw3_functions.py" implements an objective function, and computes the value, gradient and Hessian of:

$$f(x,t) \;=\; t\left(\frac{1}{2}x^\top Qx + v^\top x\right) - \sum_i \log(a_i x + b_i)$$

that is the log-barrier objective function $f(x,t) = tf_0(x) + \phi(x)$ for a quadratic objective f_0, and log-barrier function ϕ for the linear constraints $Ax + b \succeq 0$, where a_i are the rows of A and b_i the elements of b. Note that Q is assumed symmetric.

Fill in this function. You must use matrix/vector operations.

1.2 Experiments

The function "main_quadratic" in "hw3_functions.py" minimizes a quadratic objective subject to three linear constraints, with $t = m$, m being the number of constraints, in this case 3 (this is an extremely small value of t), and plots the iterates of gradient descent and Newton's method. What do you observe? Remember that you can zoom in on the plots!

The function "main_quadratic2" in "hw3_functions.py" script plots the number of iterations required by gradient descent and Newton's method for t between $\frac{1}{8}$ and 2 (once more, these are extremely small values of t, but the performance of gradient descent should give you a clue as to why we didn't use larger t). What do you observe?

2 Log barrier method

2.1 Implementation

One of the most appealing properties of the log-barrier method is that, with constraints of the form $f_i(x) \geq 0$, and given oracle access to each constraint function f_i and its first and second derivatives, we may calculate the value, gradient and Hessian of the log-barrier function $g_i(x) =$

$-\log f_i(x)$. The function "objective_scalar_constraints" in "algorithms.py" implements this. It takes as parameters a *list* of constraint functions f_i and a point x, and returns $\sum_i g_i(x)$, $\sum_i \nabla g_i(x)$ and $\sum_i \nabla^2 g_i(x)$. Each constraint function takes x as a parameter, and returns $f_i(x)$, $\nabla f_i(x)$ and $\nabla^2 f_i(x)$.

A list is a python data structure which may be treated similarly to an array. The main difference is that it may contain any object (not just scalars), including, as we have already seen, function handles.

Fill in the function "objective_scalar_constraints".

The function "quadratic" implements a constraint function which returns the value, gradient and Hessian of:

$$f_i(x) = \frac{1}{2}x^\top Qx + v^\top x + c$$

which will be used to implement the constraint $f_i(x) \geq 0$. This constraint function is almost identical to the quadratic objective function–the only difference is the presence of a constant term c.

The functions "log_barrier" in "algorithms.py" contains a mostly-complete implementation of the log-barrier method, algorithm 11.1 of Boyd and Vandenberghe. This function takes (among other things) a list of constraint functions as a parameter, and constructs the log-barrier objective $f(x,t) = tf_0(x) - \sum_i \log f_i(x)$ using "objective_log_barrier" and your filled-in version of "objective_scalar_constraints".

Fill in "log_barrier".

2.2 Experiments

The function "main_linear" in "hw3_functions.py" optimizes a linear program:

$$\begin{aligned} \text{minimize} &: v^\top x \\ \text{subject to} &: a_i x \geq b_i \end{aligned}$$

The parameter μ controls how quickly the log-barrier scaling parameter t changes between inner Newton optimizations. The generated plot shows the total number of Newton iterations performed for a run of the log-barrier method for various values of μ, and $t_0 = 1$.

Describe the dependence which you expect to see here.

Does the plot meet your expectations?

What is the solution which is found for this LP?