**Convex Optimization**

Exercise 2 (110/500)

Report Delivery Date: April 17, 2018

Instructor: Athanasios P. Liavas

---

In this exercise, we will implement and study the gradient descent and the Newton methods to solve convex optimization problems without constraints.

A. In the first part of the exercise, we shall condider a simple quadratic optimization problem.

   (a) (10) Let $f : \mathbb{R}^n \to \mathbb{R}$. For fixed $\mathbf{x} \in \mathbb{R}^n$, let $g : \mathbb{R}^n \to \mathbb{R}$ be defined as

$$g_{\mathbf{x}}(\mathbf{y}) := f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{m}{2}\|\mathbf{y} - \mathbf{x}\|_2^2. \tag{1}$$

   i. (5) Compute $\nabla g_{\mathbf{x}}(\mathbf{y})$.

   ii. (5) Compute $\mathbf{y}_* = \underset{\mathbf{y}}{\operatorname{argmin}}\, g_{\mathbf{x}}(\mathbf{y})$ and $g_{\mathbf{x}}(\mathbf{y}_*)$. Compare your finding with those of the notes (relation (1.33))

B. In the second part of the exercise, we shall solve convex quadratic problems. Our goal is to study the behavior of the gradient method, with exact and backtracking line search.

   (a) (40) Consider the problem

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ f(\mathbf{x}) = \tfrac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x}, \tag{2}$$

   where $\mathbf{P} \in \mathbb{R}^{n\times n}$, $\mathbf{P} = \mathbf{P}^T \succ \mathbf{O}$ and $\mathbf{q} \in \mathbb{R}^n$ (indicative values of $n$ are $n = 2, 50, 500, 1000$.)

   i. A random positive definite matrix $\mathbf{P}$ can be constructed in many ways. To fully control the condition number of the matrix, we can act as follows. Every positive definite $\mathbf{P}$ can be expressed as

$$\mathbf{P} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T \tag{3}$$

with $\mathbf{U}, \mathbf{\Lambda} \in \mathbb{R}^{n \times n}$, $\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}_n$,[1] and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n)$, with $\lambda_i > 0$, for $i = 1, \ldots, n$. The columns of the $\mathbf{U}$ are the eigenvectors of $\mathbf{P}$ and the elements of the diagonal of $\mathbf{\Lambda}$ are the eigenvalues of $\mathbf{P}$.

(a) To create a random orthonormal $\mathbf{U}$, create a random $(n \times n)$ matrix $\mathbf{A}$ and then calculate its singular value decomposition

$$[\texttt{U}, \texttt{S}, \texttt{V}] = \texttt{svd}(\texttt{A});$$

Compute $\texttt{U} * \texttt{U}'$ and $\texttt{U}' * \texttt{U}$. What do you observe?

(b) To construct the eigenvalues $\lambda_i$, for $i = 1, \ldots, n$, select the smallest and largest, $\lambda_{\text{min}}$ and $\lambda_{\text{max}}$, respectively. You can create the other eigenvalues in any way you wish. For example, you can create $(n-2)$ random uniformly distributed numbers in the interval $[\lambda_{\text{min}}, \lambda_{\text{max}}]$ using the command

$$\texttt{z} = \lambda_{\text{min}} + (\lambda_{\text{max}} - \lambda_{\text{min}}) * \texttt{rand}(\texttt{n} - 2, 1); \tag{4}$$

and the vector of the $n$ eigenvalues as

$$\texttt{eig\_P} = [\lambda_{\text{min}}; \lambda_{\text{max}}; \texttt{z}]; \tag{5}$$

Matrix $\mathbf{\Lambda}$ may be constructed as

$$\mathbf{\Lambda} = \texttt{diag}(\texttt{eig\_P}); \tag{6}$$

The condition number of the problem is $\mathcal{K} := \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$.

ii. (5) Construct random $\mathbf{q}$ and random $\mathbf{P}$, with condition number $\mathcal{K} = 10, 100, 1000$.

iii. (5) Solve problem (2) using the closed-form solution.

iv. (15) Solve problem (2) using the gradient algorithm (with exact and backtracking line search).

v. (5) For $n = 2$, plot the trajectories of $\{\mathbf{x}_k\}$ produced by the two algorithms.

vi. (5) Plot quantity $\log(f(\mathbf{x}_k) - p_*)$, produced by the two algorithms, versus $k$. What do you observe?

vii. (5) Using the convergence analysis results for strongly convex functions, and the values of $f(\mathbf{x}_0)$, $p_*$ and $\epsilon$, compute the minimum number of iterations

[1] Matrices that satisfy this property are called **orthonormal**.

that guarantees solution within accuracy $\epsilon$, i.e., $f(\mathbf{x}_k) - p_* \le \epsilon$, and compare it with the number of repetitions performed by the algorithms. What do you observe?

C. In the third part of the exercise, we will solve general convex unconstrained problems.

(60) We consider the function

$$f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} - \sum_{i=1}^{m} \log(b_i - \mathbf{a}_i^T\mathbf{x}) = \mathbf{c}^T\mathbf{x} - \texttt{sum}\left(\texttt{log}(\mathbf{b} - \mathbf{A}\mathbf{x})\right), \tag{7}$$

with $\mathbf{x} \in \mathbb{R}^n$ and $m > n$ (or $m \gg n$) (indicative value pairs $(n, m) = (2, 20), (50, 200)$. If you have patience and enough memory in the computer, you can set $(n, m) = (300, 800)$ or larger values).

Some observations are as follows (see Boyd–Vandenberghe, pages 141, 419–422, 458–459, 472, 492):

- The set $\mathbf{dom} f$ contains only the points $\mathbf{x} \in \mathbb{R}^n$ for which the arguments of the logarithms are positive.

  Prove that

  (a) (5) the set $\mathbf{dom} f$ is convex.

  (b) (5) Function $f$ is convex.

- Observe that if $b_i > 0$, for $i = 1, \ldots, m$, then $\mathbf{dom} f \neq \emptyset$ because $\mathbf{x} = \mathbf{0}$ is a feasible point (in the experiments, it makes sense to always use this convention).
- Function $f$ may be unbounded from below. In this case, the problem has no solution (no need to do something for it - `cvx` will help you to identify these cases).
- If a solution $\mathbf{x}_*$ exists, then it lies in the interior of $\mathbf{dom} f$, because when we approach the boundary of $\mathbf{dom} f$ the value of the function increases without bound. This means that a necessary and sufficient condition for $\mathbf{x}_*$ to be an optimal point is $\nabla f(\mathbf{x}_*) = \mathbf{0}$.

Our study will proceed as follows.

(a) Minimise $f$ using the `cvx`. If the problem has a solution, then `cvx` will compute it, otherwise it will display a message saying that the problem has no solution.

(b) (5) If $n = 2$, then plot $f$ and its level sets in the neighborhood of the optimum point. You can check whether a point $\mathbf{x}$ belongs to the domain of $f$ by checking the argument of the logarithm at this point. If a point $\mathbf{x}$ belongs to $\mathbf{dom} f$, then you can compute the value of $f$ at this point. Otherwise, you can give an arbitrarily large value to $f$ at this point (for example, $f(\mathbf{x}) = 10^3$).

(c) (20) Assuming that $\mathbf{x} = \mathbf{0}$ is a feasible point, minimize $f$ using the gradient algorithm with backtracking line search, starting from $\mathbf{x}_0 = \mathbf{0}$. The implementation will have the following main difference from the baseline implementation.

- In step $(k + 1)$, given the vectors $\mathbf{x}_k$ and $\Delta\mathbf{x}_k$, and having set $t = 1$, before starting the backtracking, you should check whether the point $\mathbf{x}_k + t\Delta\mathbf{x}_k$ belongs to $\mathbf{dom} f$ or not. If it does not belong to $\mathbf{dom} f$, then you must put $t := \beta t$ ($\beta$ is the backtracking parameter) and repeat this process until you find a point that belongs to $\mathbf{dom} f$.[2] When you arrive at a point that belongs to $\mathbf{dom} f$, then you can proceed to the typical backtracking line search.

(d) (20) Following the analogous procedure, minimize function $f$ using the Newton algorithm.

(e) (5) Plot, using `semilogy`, quantities $(f_{\text{gradient}}(\mathbf{x}_k) - p_*)$ and $(f_{\text{newton}}(\mathbf{x}_k) - p_*)$, as a function of step $k$. What do you observe for small and large values of the pair $(n, m)$?

[2] It may help you in the understanding of the behavior of the method if you print a message every time point $\mathbf{x} + t\Delta\mathbf{x}$ does not belong in $\mathbf{dom} f$. You may observe different behavior far away and close to the solution.