Math 4650/MSSC 5650 - Homework 6

Instructor: Greg Ongie

Spring 2023

Problem 1 (5 pts)**.** Let $f : \mathbb{R}^2 \to \mathbb{R}$ be the function $f(x, y) = x^2 + xy + y^2$. Find *two different* descent directions for f at the point $(x, y) = (1, -1)$.

Problem 2 (5 pts)**.** Let $f : \mathbb{R}^2 \to \mathbb{R}$ be the function $f(x, y) = \frac{1}{2}x^2 + y$. Suppose we take $(x_0, y_0) = (1, -1)$ to be our initial iterate, and $(d_x, d_y) = -\nabla f(x_0, y_0) = (-1, -1)$ to be our descent direction. Find the next iterate (x_1, y_1) computed using the descent direction method with exact line search. Verify that $f(x_1, y_1) < f(x_0, y_0)$ by direct computation.

Problem 3 (5 pts)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be a general quadratic function $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + 2\mathbf{b}^\top \mathbf{x} + c$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is positive definite, and $\mathbf{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$ are arbitrary. Fix an $\mathbf{x} \in \mathbb{R}^n$ and suppose $\mathbf{d} \in \mathbb{R}^d$ is a descent direction of f at $\mathbf{x}$, i.e., $\mathbf{d}^\top \nabla f(\mathbf{x}) < 0$. Show that the exact line search step-size t^*, defined by

$$t^* = \arg\min_{t\geq 0} f(\mathbf{x} + t\mathbf{d}),$$

has the explicit form

$$t^* = -\frac{\mathbf{d}^\top(\mathbf{A}\mathbf{x} + \mathbf{b})}{\mathbf{d}^\top \mathbf{A}\mathbf{d}}.$$

In your derivation, where is the assumption that $\mathbf{A}$ is positive definite being used?
(Hint: If you get stuck, this is worked out in Example 4.4 in Beck, Ch. 4; but please put the proof in your own words).

Problem 4 (MATLAB, 5 pts)**.** Consider the quadratic function $f : \mathbb{R}^2 \to \mathbb{R}$ defined by $f(\mathbf{x}) = \frac{1}{2}(\gamma x_1^2 + x_2^2)$ with $\gamma = 0.05$, which has a unique global minimum at $(0, 0)$. Using the provided MATLAB script `gd.m` as a starting point[1], implement the following two versions of the decent direction method to minimize this function:

(a) descent direction $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$ (steepest descent), constant stepsize $t_k = 0.05$

(b) descent direction $\mathbf{d}_k = -\begin{bmatrix} 1/\gamma & 0 \\ 0 & 1 \end{bmatrix} \nabla f(\mathbf{x}_k)$, constant stepsize $t_k = 0.05$.

[1] Note: you will need to modify the definition of `f` and `grad` in the script to match this new function.

In both cases, start with the initial guess $\mathbf{x}_0 = \begin{bmatrix} -1 \\ 0.05 \end{bmatrix}$ and run 100 iterations. What do you observe? How well did each method do in finding the minimizer? How do the trajectories of the iterates differ for the two choices of descent directions? In your write-up, include a print-out of your code and plots of the trajectories of the iterates in both cases.

Problem 5 (MATLAB, 5 pts)**.** Repeat Problem 4 but instead of using a constant stepsize, use *exact line search* to update the stepsize t_k for both versions of the descent direction method. What do you observe? How well did the two versions of the descent direction method do in finding the minimizer? How do the trajectories of the iterates differ for the two versions? In your write-up, include a print-out of your code and plots of the trajectories of the iterates in both cases.
(Hint: Note that $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x}$ where $\mathbf{A} = \begin{bmatrix} \gamma/2 & 0 \\ 0 & 1/2 \end{bmatrix}$; now use the result in Problem 3 to compute the exact line search stepsize t_k)

Problem 6 (**MSSC**, MATLAB, 5 pts)**.** Repeat Problem 4 but instead of using a constant stepsize, use *backtracking line search* to update the stepsize t_k for both versions of the descent direction method. Use the parameters $s = 2$, $\alpha = 0.25$, $\beta = 0.5$. What do you observe? How well did the two versions of the descent direction method do in finding the minimizer? How do the trajectories of the iterates differ for the two versions? In your write-up, include a print-out of your code and plots of the trajectories of the iterates in both cases. (Hint: See Example 4.9 in Beck Ch. 4 for an example implementation of backtracking line search)