Homework 8

Requirements:

1. Digital format (can be typeset or photos, ought to write clearly if written by hand), upload to `https://course.pku.edu.cn/`.
2. Submit by next class
3. A problem is not counted if nobody can work it out
4. Each homework 10 points; 1 point deducted for each week's delay

Problems:

1. Suppose that f is strongly convex with $m\mathbf{I} \preceq \nabla^2 f(\mathbf{x}) \preceq M\mathbf{I}$. Let $\Delta\mathbf{x}$ be a descent direction at $\mathbf{x}$. Show that the backtracking stopping condition holds for

$$0 < t \leq -\frac{\nabla f(\mathbf{x})^T \Delta\mathbf{x}}{M\|\Delta\mathbf{x}\|_2^2}.$$

Use this to give an upper bound on the number of backtracking iterations.

2. Given $f : \mathbb{R}^n \to \mathbb{R}$, consider the general iterative algorithm

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)},$$

where $\mathbf{d}^{(1)}, \mathbf{d}^{(2)}, \cdots$ are given vectors in $\mathbb{R}^n$ and α_k is chosen to minimize $f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})$; that is,

$$\alpha_k = \operatorname*{argmin}_{\alpha} f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}).$$

Show that for each k, the vector $\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$ is orthogonal to $\nabla f(\mathbf{x}^{(k+1)})$ (assuming that the gradient exists).

3. Consider the problem:

$$\min_{\mathbf{x}} f(\mathbf{x}) = \sum_{i=1}^{m} \left[\exp(\mathbf{a}_i^T\mathbf{x}) + \exp(-\mathbf{a}_i^T\mathbf{x})\right]$$

with variable $\mathbf{x} \in \mathbb{R}^n$.

We can choose $\mathbf{x}^{(0)} = \mathbf{1}$ as our initial point. You can generate instances of this problem by choosing $\mathbf{a}_i$ from some distribution on $\mathbb{R}^n$. Use the gradient method to solve the problem, using reasonable choices for the backtracking parameters, and a stopping criterion of the form $\|\nabla f(\mathbf{x})\|_2 \leq \eta$. Plot $\log(f - p^*)$ and step length versus iteration number (For this problem, the minimum is obviously achieved at $\mathbf{x} = \mathbf{0}$.). Experiment with the backtracking parameters α and β to see their effect on the total number of iterations required. Hand in your code and a report showing how the parameters are chosen and the figures.

4. Explain how to find a steepest descent direction in the ℓ_∞-norm, and write down its pseudo-code. Apply it to solve Problem 3 (i.e., redo the problem by replacing the gradient descent with your steepest descent).

5. Use the Damped Newton method and the Gauss-Newton method to minimize Rosenbrock's function:

$$f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

Use an initial condition of $\mathbf{x}^{(0)} = [-2, 2]^T$. Terminate the algorithm when the norm of the gradient of f is less than 10^{-5}. Write a report to show the figures of $\log(f(\mathbf{x}^{(k)}) - p^*)$ vs. iteration number k and running time, respectively, and your choice of parameters, and what your observation is. p^* is clearly 0. Hand in both your code and report.

6. Let $f : \mathbb{R}^n \to \mathbb{R}$ be given by $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{x}^T\mathbf{b}$, where $\mathbf{b} \in \mathbb{R}^n$, and $\mathbf{Q}$ is a real symmetric positive definite $n \times n$ matrix. Show that in the conjugate gradient method for this f, $\mathbf{d}^{(k)T}\mathbf{Q}\mathbf{d}^{(k)} = -\mathbf{d}^{(k)T}\mathbf{Q}\mathbf{g}^{(k)}$.

7. Consider a conjugate gradient algorithm applied to a quadratic function. Show that the gradients associated with the algorithm are $\mathbf{Q}$-conjugate if separated by at least two iterations. Specifically, show that $\mathbf{g}^{(k+1)T}\mathbf{Q}\mathbf{g}^{(i)} = 0$ for all $0 \le k \le n-1$ and $0 \le i \le k-1$.

8. Use conjugate gradient to minimize extended Rosenbrock function

$$f(\mathbf{x}) = \sum_{i=1}^{n/2} \left[\alpha(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2\right],$$

where α is a parameter that you can vary (for example, 1 or 100). The solution is $\mathbf{x}^* = (1, 1, \cdots, 1)^T$, $f^* = 0$. Choose $n = 100$ and the starting point as $(-1, -1, \cdots, -1)^T$. Report your choice of other parameters and compare the performance using the Hestenes-Stiefel formula, the Polak-Ribiere formula, and the Fletcher-Reeves formula.

Note: When comparing optimization algorithms, remember to restart your computer before running the program. This is to minimize the influence of other programs.