Assignment 3 of MATP6610/4820

(Due on Feb-21-2020 in class)

Problem 1

Let $g(\mathbf{x}) = \|\mathbf{x}\|_\infty$ for $\mathbf{x} \in \mathbb{R}^n$.

1. Let $n = 2$. Evaluate the proximal mapping $\mathbf{prox}_g$ at $\mathbf{y} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\mathbf{y} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$.
2. **(Only for 6610 students:)** For general $n \geq 1$, give the explicit formula of $\mathbf{prox}_g(\mathbf{y})$.

Problem 2

For ill-conditioned problems, the conjugate gradient (CG) method may converge very slowly, and a better method is the preconditioned conjugate gradient (PCG) method. Consider the quadratic minimization problem:

$$\underset{\mathbf{x}}{\text{minimize}}\, f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{b}^\top \mathbf{x}, \tag{QuadMin}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, and $\mathbf{b} \in \mathbb{R}^n$. Write a solver for (QuadMin) by the PCG method. We will use a non-singular triangular matrix $\mathbf{C}$ as the preconditioner.

1. Let $\mathbf{C} \in \mathbb{R}^{n\times n}$ be a non-singular upper triangular matrix. Write a solver `pcg_linsolv` for the linear system $\mathbf{C}^\top\mathbf{C}\mathbf{y} = \mathbf{r}$. Treat $\mathbf{C}$ and $\mathbf{r}$ as the input of the solver. Recall that to solve the linear system, we can first solve $\mathbf{C}^\top\mathbf{z} = \mathbf{r}$ by forward substitution and then solve $\mathbf{C}\mathbf{y} = \mathbf{z}$ by back substitution.

2. Write a Matlab function `quadMin_pcg` with input $\mathbf{A}$, $\mathbf{C}$, and $\mathbf{b}$, where $\mathbf{C}$ is a non-singular upper triangular matrix, used as the conditioning matrix. Within your PCG solver, use `pcg_linsolv` as the linear solver for solving linear systems in the form of $\mathbf{C}^\top\mathbf{C}\mathbf{y} = \mathbf{r}$. Also test your function by running the provided test file `test_pcg.m` and compare to the instructor's function. Print your code and the results you get.

Problem 3

Consider the constrained quadratic minimization problem:

$$\underset{\mathbf{x}}{\text{minimize}}\, f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}\mathbf{x} - \mathbf{b}^\top \mathbf{x}, \text{ s.t. } l_i \le x_i \le u_i, \ \forall i = 1, \ldots, n, \qquad \text{(C-QuadMin)}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix, $\mathbf{b} \in \mathbb{R}^n$, and $\mathbf{l} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^n$ are lower and upper bounds.

1. Let $\mathbf{A} = \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $l_1 = l_2 = 0$, and $u_1 = u_2 = 1$. Set the initial vector $\mathbf{x}^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. Do by hand two iterations the projected gradient method with constant step size $\alpha_k = 0.5$, $\forall k$

2. Use the instructor's provided file `quadMin_pg.m` to write a Matlab function `quadMin_pg` with input $\mathbf{A}$, $\mathbf{b}$, $\mathbf{l}$, $\mathbf{u}$, initial vector $\mathbf{x}0$, and maximum iteration number `maxit`. Also test your function by running the provided test file `test_pg.m` and compare to the instructor's function. Print your code and the results you get.

Problem 4 (Only for 6610 students)

Suppose we have two iterative algorithms $\mathcal{A}$ and $\mathcal{B}$ for solving the problem

$$\operatorname*{minimize}_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}).$$

Denote $\mathbf{x}^k$ as the k-th iterate and $\mathbf{x}^*$ one optimal solution. For the algorithm $\mathcal{A}$, its per-iteration complexity is $2n$, and its convergence rate is

$$f(\mathbf{x}^k) - f(\mathbf{x}^*) \le \frac{C_0}{k}, \ \forall k \ge 1, \tag{1}$$

where C_0 is a constant depending on the initial point. For the algorithm $\mathcal{B}$, its per-iteration complexity is $4n^2$, and its convergence rate is

$$f(\mathbf{x}^k) - f(\mathbf{x}^*) \le C_0\eta^k, \ \forall k \ge 1,$$

where C_0 is the same constant as that in (1), and $\eta \in (0,1)$ is a constant.

1. Given $\varepsilon > 0$, suppose we want to have a point $\mathbf{x}^k$ such that $f(\mathbf{x}^k)-f(\mathbf{x}^*) \le \varepsilon$. Estimate from the convergence rate results how many iterations each algorithm needs to run.

2. We measure efficiency by the total complexity (i.e., per-iteration complexity multiplying by iteration number). Discuss how small ε should be such that the algorithm $\mathcal{B}$ is more efficient.