Assignment 5 of MATP4820

Xinshi Wang

Problem 1

Consider the constrained quadratic minimization problem:

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

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 $\mathbf{A} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}$, $l_1 = l_2 = 0$, 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 $\alpha_k = 0.5$, $\forall k$

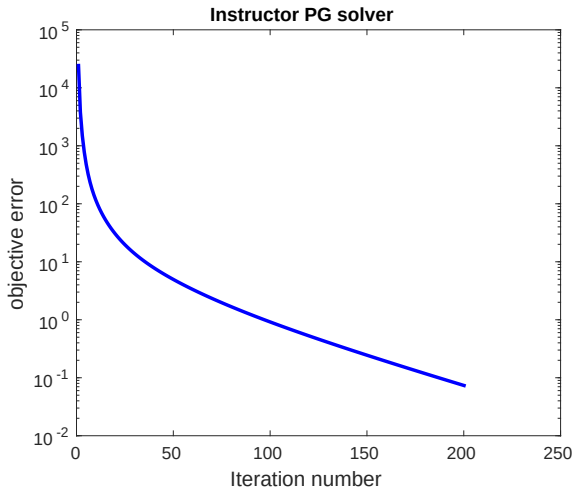
$$X_1 = \underset{x}{Proj}(X_0 - \alpha_k X_0) = max(0, min(1, \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.5 \begin{bmatrix} 0 \\ -3 \end{bmatrix})) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

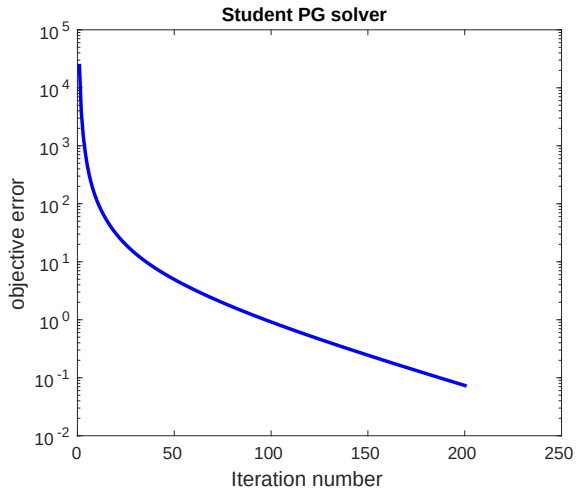
$$grad_1 = Ax - b = \begin{bmatrix} -1 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 3 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$X_2 = \underset{x}{Proj}(X_1 - \alpha_k X_1) = max(0, min(1, \begin{bmatrix} 0 \\ 1 \end{bmatrix} - 0.5 \begin{bmatrix} -1 \\ -1 \end{bmatrix})) = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

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.

```
function [x, hist_obj] = quadMin_pg(A,b,x0,maxit,lb,ub)

% projected gradient method for solving
% min_x 0.5*x'*A*x - b'*x
% s.t. lb <= x <= ub

x = x0;

% compute gradient of the objective
grad = A*x-b;

% compute the Lipschitz constant of grad
L = norm(A);

hist_obj = .5*(x'* (grad - b));

for iter = 1:maxit

    % update x by projected gradient with step size 1/L
    x = max(lb,min(ub,x-(1/L*grad)));

    % compute gradient of the objective
    grad = A*x-b;

    % save objective value
    hist_obj = [hist_obj; .5*(x'* (grad - b))];

end
end
```





Student code: Total running time is 0.1093

Final objective value is -23.2941

Intructor code: Total running time is 0.0933

Final objective value is -23.2941

Problem 2

Let $g(\mathbf{x}) = \|\mathbf{x}\|_2$ for $\mathbf{x} \in \mathbb{R}^n$.

1. **(Only for 4820 students:)** Let $n = 2$. Evaluate the proximal mapping $\mathbf{prox}_g$ at $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$$
\begin{aligned}
prox_g(x) &= \underset{y \in \mathbb{R}^n}{argmin} \frac{1}{2}||y - x||_2^2 + g(y) \\
&= \underset{y \in \mathbb{R}^n}{argmin} \frac{1}{2}||y - x||_2^2 + \frac{y}{||y||_2} \\
\text{Let } f(y) &= \frac{1}{2}||y - x||_2^2 + ||y||_2 \\
\frac{\partial f(y)}{\partial y} &= y - x + \frac{y}{||y||_2} \\
prox_g(x) &= y \text{ at } \frac{\partial f(y)}{\partial y} = 0 \\
y - x + \frac{y}{||y||_2} &= 0 \\
y(1 + \frac{1}{||y||_2}) &= x \\
y\theta(y) &= x \\
y &= \frac{x}{\theta(y)} = x\sigma(y)
\end{aligned}
$$

We then substitute y into the original equation

$$
\begin{aligned}
x\sigma(y) - x + \frac{x\sigma(y)}{||x\sigma(y)||_2} &= 0 \\
x\sigma(y) - x + \frac{x\sigma(y)}{\sigma(y)||x||_2} &= 0 \\
\sigma(y) &= 1 - \frac{1}{||x||_2} \\
\text{Therefore } prox_g(x) &= x\sigma(y) = (1 - \frac{1}{||x||_2})x
\end{aligned}
$$

At $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$, the proximal mapping is $\begin{bmatrix} 1 - \sqrt{2}/2 \\ 1 - \sqrt{2}/2 \end{bmatrix}$.

2. **(Only for 6610 students:)** Let $n = 2$. Evaluate $\mathbf{prox}_g$ at $\begin{bmatrix} -1 \\ 2 \end{bmatrix}$.

3. **(Bonus question:)** Derive the formula of $\mathbf{prox}_g$ at any point $\mathbf{x} \in \mathbb{R}^n$.

As we have derived above, the general formula is:

$$prox_g(x) = x\sigma(y) = (1 - \frac{1}{||x||_2})x$$

Problem 3

In the lecture, we derived the proximal gradient method for solving the Lasso problem

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\, \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|^2 + \lambda\|\mathbf{x}\|_1, \tag{Lasso}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a sensing matrix, $\mathbf{b}$ is the measurement vector, and $\lambda > 0$ is the regularization parameter.

1. Write a solver for (Lasso) by the proximal gradient method. Use the instructor's provided file `PG_Lasso.m` to write a Matlab function `PG_Lasso` with input $\mathbf{A}$, $\mathbf{b}$, initial vector $\mathbf{x}0$, parameter `lam`, and tolerance `tol`, and with a provided stopping condition.

```
function [x,hist_res] = PG_Lasso(A,b,x0,lam,tol)

% proximal gradient method for solving the Lasso
% min_x .5*||A*x-b||^2 + lam*||x||_1

% compute Lipschitz constant
L = norm(A*A');

% compute gradient at x0
grad = A'*(A*x0-b);

% perform one proximal gradient step with stepsize 1/L to get a new x
x = sign(x0 - grad/L) .* max(abs(x0 - grad/L) - lam/L, 0);

% evaluate norm of the proximal gradient mapping
res = L * norm(x-x0);

hist_res = res;

while res > tol
    x0 = x;

    % compute gradient at x0
    grad = A'*(A*x0-b);

    % perform one proximal gradient step with stepsize 1/L to get a new x
```

```
    x = sign(x0 - grad/L) .* max(abs(x0 - grad/L) - lam/L, 0);

    % evaluate norm of the proximal gradient mapping
    res = L * norm(x-x0);
    hist_res = [hist_res; res];
end

end

% define the proximal operator for the L1 norm
function y = prox_l1(x, t)
    y = sign(x) .* max(abs(x) - t, 0);
end
```

2. Write a solver for (Lasso) by the accelerated proximal gradient method. Use the instructor's provided file `APG_Lasso.m` to write a Matlab function `APG_Lasso` with input $\mathbf{A}$, $\mathbf{b}$, initial vector $\mathbf{x}0$, parameter `lam`, and tolerance `tol`, and with a provided stopping condition.

```
function [x,hist_res] = APG_Lasso(A,b,x0,lam,tol)

% accelerated proximal gradient method for solving the Lasso
% min_x .5*||A*x-b||^2 + lam*||x||_1

% compute Lipschitz constant
L = norm(A*A');

% compute gradient at x0
grad = A'*(A*x0-b);

% perform one proximal gradient step with stepsize 1/L to get a new x
x = sign(x0 - grad/L) .* max(abs(x0 - grad/L) - lam/L, 0);

% evaluate norm of the proximal gradient mapping
res = L * norm(x-x0);

hist_res = res;
```

```
x0 = x;

% y used to denote the extrapolated point
y = x;

% used to compute the extrapolation weight
t0 = 1;

while res > tol

    % compute a gradient at y
    grad = A'*(A*y-b);

    % perform one proximal gradient step with stepsize 1/L to get a new x
    x_new = sign(y - grad/L) .* max(abs(y - grad/L) - lam/L, 0);

    % update t value to compute extrapolation weight
    t1 = (1 + sqrt(1 + 4*t0^2))/2;

    % update the extrapolated point y
    y = x_new + (t0 - 1)/t1 * (x_new - x);

    x = x_new;

    % evaluate norm of the proximal gradient mapping
    res = L * norm(x-x0);
    hist_res = [hist_res; res];

    t0 = t1; x0 = x;
end

end
```

3. Test your two solvers by running the provided test file `test_Lasso.m` and compare to the instructor's function. Print your code and the results you get. What do you observe on the convergence speed of the two methods?

Results by student solver
PG method: time = 0.0933, Relative Error = 1.2742e-02
APG method: time = 0.0261, Relative Error = 1.2313e-02
Results by Instructor solver
PG method: time = 0.1053, Relative Error = 1.2742e-02
APG method: time = 0.0221, Relative Error = 1.2313e-02

APG achieved even a lower Error compared to PG method in roughly a quarter of the time PG takes. APG converges much quicker compared to the PG method.

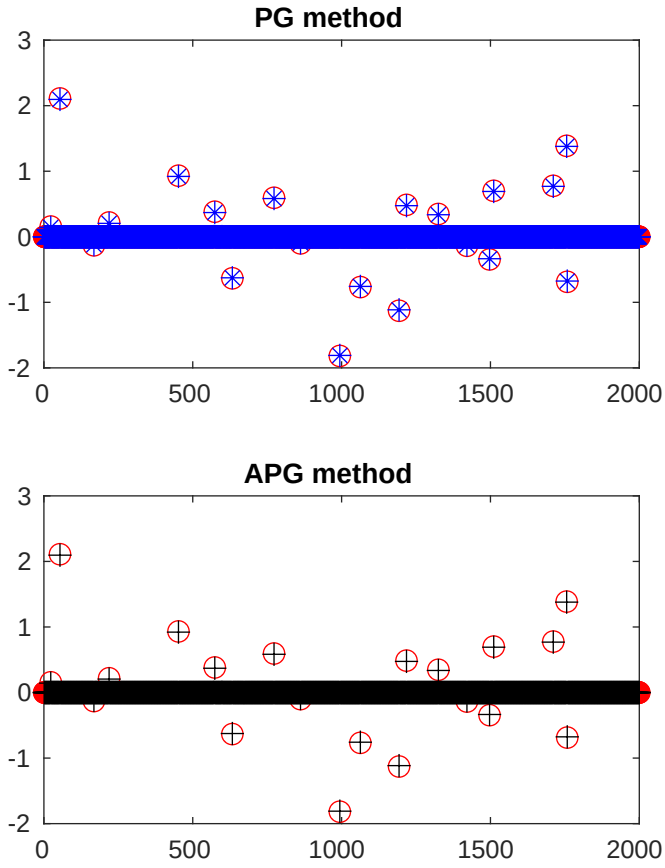


Figure 1: Student Solution

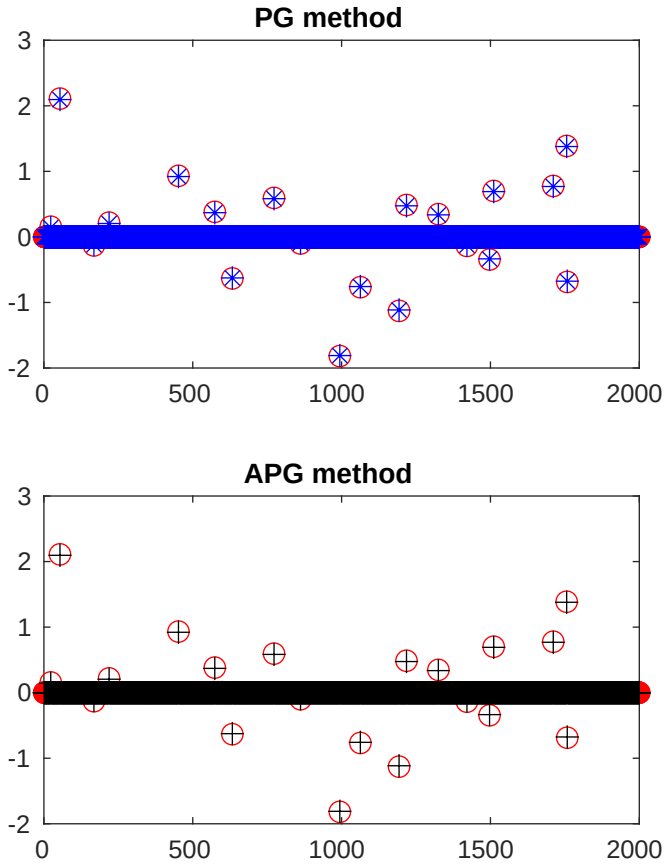


Figure 2: Instructor Solution