**Assignment 4**  
Due: June 4th, 11pm. Upload everything to CANVAS.

**Question 1.** Consider the same Lasso problem in Homework 3:

\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} \|Ax - b\|_2^2 + \tau \|x\|_1,
\]

(1)

where \( \tau > 0 \) is a weighting parameter, \( A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m \) are given data. Use the following Matlab code to generate the data:

```matlab
m = 300; n = 500; s = 2;
A = randn(m,n);
xs = zeros(n,1); picks = randperm(n); xs(picks(1:s)) = 100*rand(s,1);
b = A*xs;
tau = 1;
```

Note that \( xs \) is the true solution that you want to find.

Write codes as requested below. For all codes, choose \( x = 0 \) as the starting point, and terminate your codes when your iterate \( x^k \) satisfies

\[
\frac{\|x^k - xs\|_2}{\|xs\|_2} < \epsilon,
\]

(2)

for \( \epsilon = 10^{-3} \).

(a) In homework 3 you have seen one way to reformulate the problem as a two-block problem and solve it using ADMM. The following is another way to reformulate it:

\[
\min_{x \in \mathbb{R}^n, r \in \mathbb{R}^m} \frac{1}{2} \|r\|_2^2 + \tau \|x\|_1, \\
\text{s.t.} \quad Ax - b = r.
\]

(3)

When you use ADMM to solve this reformulation, you will find that you don’t have closed-form solution for the \( x \)-subproblem. So, implement the linearized ADMM to solve it. That is, you need to linearize the \( x \)-subproblem. Compare the result (cpu time, iteration number etc.) with the ADMM you implemented in homework 3. Which one is better?

(b) Implement Coordinate Gradient Descent algorithm to solve (1). Implement both the cyclic rule and randomized rule. Use fixed step size. Do you achieve the targeted accuracy? Compare your algorithms with proximal gradient method (you implemented this before) for solving (1). Which one is the best?