Week 6 Report - Jesus Ruvalcaba

- Week of 2/23 to 3/1

Introduction

- We are still working on developing a fully functional proximal gradient descent method that takes in an already derived proximal operator for a given function as a parameter.
- The main focus this week was to create a function that prints out the relevant data values produced by the method
    - Allows us to observe the behavior of the method when it is acting on different parameters such as matrix dimensions and step size values.

Summary of Work Completed

- Implemented a function called **print_relevant_data** that takes in the following parameters:
    - g
        - This is the function $g(x)$, which has a computable sub-gradient
    - sub_grad_g
        - The sub-gradient of $g(x)$
    - h
        - The function $h(x)$ that is not necessarily differentiable
    - prox_l1_norm
        - Function computing the proximal operator of an L1 norm
    - lamb

        - Lambda value
    - alphas
        - A list of all the step sizes that we want to test
            - Used: 1e-3, 1e-4, 1e-5, 1e-6
    - dimen
        - A list of objects, where each object has the number of rows and columns (dimensions of the matrix A)
            - Used: 20×25, 40×50 and 80×100
- **How does it work?**
    1. For every dimension in the list of objects (dimen)
        a. Randomly generate the matrix A and the vectors b and x
        b. Print out the dimension that is currently being tested
        c. For every step size in alphas
            i. Print out the step size being used
            ii. Call the proximal gradient descent method
            iii. Print out the total time it took to converge/not converge
            iv. Construct a table and print out all the function values computed and the norm of the difference between x_k and x_{old}

Analysis

- $20x25$
    - Converged for every step size except 1e-3
    - For step size 1e-3, when observing the difference in x-values, towards the end, the values were getting closer to 0 but then in the last two iterations, the difference increased. Thus, it started drifting away from the optimal function value.
- $40x50$

    - Converged for step size's 1e-5 and 1e-6
    - For step size 1e-3, the difference in x-values was decreasing but it seems like it needed more iterations, however, the difference was still quite large (far from our tolerance).
    - For step size 1e-4, the difference in x-values was decreasing and increasing in every iteration, thus, it was oscillating.
- $80x100$
    - Converged for step size's 1e-5 and 1e-6
    - For step size 1e-3, the difference in x-values was exactly the same for what it seems like all 20,000 iterations. It doesn't seem to be a mistake with the algorithm as it didn't do this with other dimensions with the same step size.
    - For step size 1e-4, similarly, the difference in x-values gets to around 0.01 at the third iteration and it remains that way for the rest of the iterations (20,000 max. iterations)
- It seems like as the dimension gets noticeably larger for A, the smaller step sizes (1e-3 and 1e-4) provide very little progress towards the minimizer. The "step taken" might just be too small for any real progress to be made in a reasonable amount of time.
