Assignment 6: Robust PCA

Irina Gaynanova

4/9/2019

HW5: Robust Principal Component Analysis (PCA)

Introduction

In this homework, you will be asked to implement ADMM algorithm to solve robust PCA problem. The idea is that given a matrix $M \in \mathbb{R}^{n \times p}$, you want to decompose the matrix into low-rank and sparse components. The reason it's called robust PCA is because of the sparse components. The standard PCA can be viewed as a low-rank matrix decomposition. The robust PCA uses sparsity to allow "few" elements to deviate from low rank assumptions (hence sparse component).

Given matrix M, Robust PCA is formulated as optimization problem

$$\text{minimize}_{L,M} \{\|L\|_* + \gamma \|S\|_1\} \quad \text{s.t.} \quad L + S = M.$$

Here $\|L\|_*$ is the nuclear norm of L (discussed in class), $\|S\|_1 = \sum_{i,j} |s_{i,j}|$ is the sum of absolute values of the elements of matrix S (ℓ_1 norm applied to vectorises S, i.e. lasso type penalty). The parameter $\gamma > 0$ controls the relative weights between nuclear norm and sparsity penalties, and is chosen by the user.

The Robust PCA problem is already in the form required by ADMM, so the implementation should consist of three updates: update of L, update of S and update of dual variable η (we consider scaled ADMM for simplicity). You should discover that the first two updates correspond to proximal operators of nuclear norm and ℓ_1 norm correspondingly evaluated at particular points.

Remark: While all ADMM updates in lecture were given for vectors, they generalize easily to matrices by substituting squared Frobenius norm $\|\cdot\|_F^2$ instead of squared euclidean norm, and trace operator $\text{tr}(A^\top B)$ as matrix inner-product instead of vector inner-product $a^\top b$.

Update derivation

Your first task is to derive explicit form of all three updates for ADMM for Robust PCA. Please fill your resulting updates below.

Augmented Lagrangian: given $\tau > 0$

$$L_\tau(L, S, \nu) = \|L\|_* + \gamma \|S\|_1 + \text{tr}(\nu^T (L + S - M)) + \frac{1}{2\tau} \|L + S - M\|_F^2$$

Using the scaled version of ADMM updates, let $\eta = \nu \cdot \tau$. Derive explicit updates below.

Update of L:

$$L^{t+1} = \arg\min_L \{\|L\|_* + \frac{1}{2\tau} \|L + S^t - M + \eta^t\|_F^2\},$$

which is equivalent to (derive the explicit update and fill in below, use more than one line if necessary)

$$L^{t+1} = \text{prox}_{\tau, f}(M - S^t - \eta^t)$$

Update of S:

$$S^{t+1} = \arg\min_S \{\gamma \|S\|_1 + \frac{1}{2\tau}\|L^{t+1} + S - M + \eta^t\|_F^2\},$$

which is equivalent to (derive the explicit update and fill in below)

$$S^{t+1} = \text{prox}_{\tau,g}(-M + L^{t+1} + \eta^t)$$

Update of η: (fill in, note that here η is a matrix)

$$\eta^{t+1} = \eta^t + L^{t+1} - S^{t+1} - M$$

Functions Instructions

You will need to implement 4 functions as specified within **ADMMfunctions.R**. The first three are helper functions (soft-thresholding operators + objective value), whereas the last function is actual ADMM algorithm. Feel free to create additional functions if needed, however please adhere to specified format of the 4 provided functions.

You are also provided with **TestADMM.R** to test your functions on a simple example. Please note that this test is for exploratory purposes only, that is you need to extensively test your functions beforehand to make sure they are running correctly. The provided test is an illustration of how the sparsity part allows to extract low-rank component even when the original matrix is far from low-rank. You don't need to add any additional code to this script.

Data instructions

Grading criteria

Your assingment will be graded based on

- correctness (50%)

Take advantage of objective function values over iterations as a way to indirectly check the correctness of your function.

- speed (30%)

There is not a lot of degrees of freedom in this assignment, but you will get +5 points if your code is significantly faster than mine (the code for ADMM given in TestADMM takes as median 27 sec to run on my laptop, you get +5 if yours is around 25 sec or less on my laptop)

- code style/documentation (10%)

You need to comment different parts of the code so it's clear what they do, have good identation, readable code with names that make sense.

- version control/commit practices (10%)

I expect you to start early on this assignment, and work gradually. You want to commit often, have logically organized commits with short description that makes sense.