INMA 2471 – Optimization models and methods II

Homework II – Solving ℓ_1-regularized least-squares

[v1.0]

This homework consists in investigating a certain number of optimization methods designed to solve the following problem, called ℓ_1-*regularized least-squares*, featuring a nonnegative parameter $\lambda \geq 0$

$$\min_x \Big\{ f(x) = \|Ax - b\|^2 + \lambda \|x\|_1 \Big\} . \qquad (\ell_1\text{−LS})$$

1 Motivation

Consider an $m \times n$ matrix A and a column vector b of size m. Very often one wishes to solve the linear system $Ax = b$, but one can face several difficulties:

(a) The linear system could be unsolvable, especially when the number of columns n is less than the number of rows m.

One possible answer is to solve the system in the *least-squares* sense, which means finding a solution x that minimizes the (squares) residual $\|Ax - b\|^2$.

(b) Even when $m = n$, the linear system can be ill-conditioned, potentially leading to inaccurate solutions.

A way to tackle that is to *regularize* the problem, which can be done by adding a penalty term $\|x\|$ to the objective function. The norm will penalize solutions with large entries, and its strength is usually controlled with a nonnegative parameter λ.

In this homework, we choose to solve a least-squares problems regularized with a ℓ_1-norm, which leads to the above (ℓ_1−LS) formulation. The reason behind the choice of an ℓ_1-norm (instead of the more usual ℓ_2- norm) is that it known to induce *sparse* solutions, i.e. solution vectors x which feature only a small number of nonzero coefficients. This will be useful for the image processing application described below.

Question 1.1 Establish whether problem (ℓ_1−LS) with dimensions $m > n$ and parameter $\lambda \geq 0$ is

(a) convex ?

(b) smooth, i.e. admits a gradient that is L-Lipschitz (and, in that case, what is the value of L) ?

(c) strongly convex (and, in that case, what is the value of the strong convexity parameter μ) ?

Hint: your answer can depend on the problem data such as matrix A and parameter λ ; in particular the cases $\lambda = 0$ and $\lambda > 0$ can differ.

Question 1.2 Write optimality conditions for the problem

(a) in the case $\lambda = 0$,

(b) in the case $\lambda > 0$.

Simplify the obtained conditions as much as possible.

2 Methods

In this section, a variety of methods are considered to solve the above problem. For each method

(a) answer the questions present in the text,

(b) provide a pseudo-code version of the algorithm specialized for the problem (ℓ_1-LS) (i.e. do not write code for a general function f), except for method M6

(c) use theory see during lectures or news reasonings to establish a convergence rate for the error on the objective function after N iterations (i.e. establish a bound of the type $f(x_N) - f(x^*) \leq$ something(N) ; note that the bound can sometimes involve the best iterate instead of the last one),

(d) deduce from the previous bound the number of iterations that is needed to guarantee that the error on the objective function becomes smaller than some $\epsilon > 0$.

M1. Subgradient Method. It can be shown (accept this result without proof) that the iterates of the subgradient method with constant step length α, started from iterate x_0, satisfy the following convergence bound

$$\min_{0 \leq i \leq N} f(x_i) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\alpha N} + \frac{\alpha G^2}{2}$$

where G is an upper bound on the size of any subgradient.

Is the method converging to an optimal solution ? Provide an explanation based on the above bound, as well as an intuitive explanation (based on the notion of subgradient). What type of step-length would be necessary to hope for convergence to a solution ?

The method can be used to compute solutions with some prescribed accuracy. Given a target accuracy ϵ, what is the optimal constant step-length α ? (use that step-length in the sequel)

Compute a subgradient for the objective function of (ℓ_1-LS). *Hint*: one can study each term in the objective function separately and combine the obtained results.

Compute an upper bound G on the size of a subgradient valid for problem (ℓ_1-LS).

M2. Gradient Method with Smoothing. For which values of λ can the standard gradient method (with fixed step length) be used to solve problem (ℓ_1-LS) ?

In order to be able to apply the gradient method to the problem for all values of λ, one can *smooth* the problem, i.e. replace the nonsmooth component of the objective function by a smooth

approximation. Here we are going to use the Huber function for that approximation (see Exercise session 4, problem 3), with parameter μ.

For a given parameter μ, what is the maximal error introduced on the objective function by the Huber approximation.

How should the gradient method be applied to the smoothed problem ? Establish the convergence rate for the error (do not forget to add the error due to the use of an approximation).

Given a target accuracy ϵ for the objective function, propose an optimal choice for the smoothing parameter μ (i.e. such that the expected number of iterations is minimized).

M3. Accelerated Gradient Method with Smoothing. Apply the accelerated gradient method to using the same smoothing technique as above. Answer the same questions as above.

M4. Proximal Gradient Method. The proximal gradient method is an extension of the projected gradient method, designed to minimize an objective function of the form $f(x) = f_1(x) + f_2(x)$ where f_1 is a smooth convex function, and f_2 is a convex function that admits a computable proximal operator.

At each iteration, a standard gradient step is taken for the smooth part f_1 of the objective function and then, instead of applying the projection operator as in the projected , a *proximal operator* is used according to

$$x_{k+1} = \mathrm{prox}\big(x_k - \frac{1}{L}\nabla f_1(x_k)\big)$$

where the proximal operator is defined as

$$\mathrm{prox}(x) = \arg\min_u \Big(f_2(u) + \frac{L}{2}\|u - x\|^2 \Big) .$$

Identify functions f_1 and f_2 in problem (ℓ_1-LS).

Give an explicit analytic formula for the proximal operator corresponding to this choice of f_2.

Prove that the proximal gradient method converges with the *same rate* as the projected gradient method (*hint*: the proof seen in the lectures can be adapted to this situation with only a few modifications).

M5. Accelerated Proximal Method. The proximal gradient method can be accelerated in the same way as the standard projected gradient method. Write down the corresponding algorithm. You can accept (without proof) that the convergence rate of the accelerated projected gradient method also holds for the proximal case.

M6. Interior-point method with a conic formulation. Reformulate problem (ℓ_1-LS) with a conic formulation. What is the total barrier parameter for the problem ? What is the convergence rate for a short-step path-following interior-point algorithm ? For this method, instead of writing your own code, use an external solver (such as SeDuMi).

3 Implementation, tests and observations

The goal of the section is to test on an image processing problem. The data (matrix A and vector b) will be provided on Dropbox. Implement (in the language of your choice) each method described

in Section 2 and test it on the provided problem data with the suggested parameters (λ penalty, ϵ target accuracy). Use the provided starting point if possible.

Compare for each method the observed convergence behavior to the theoretical bounds mentioned in the previous section. Use graphs, where the abscissa is the number of iterations.

Observe the effect of the penalty parameter λ. Does it induce sparsity ?

Provide a general comparison between all methods. Use graphs. In order for the comparison to be fair, abscissa should be CPU time, not number of iterations. What method would you finally recommend for a given accuracy target ϵ ?

Work in pairs (in particular, you should exploit the fact that work on the different methods can be done in parallel). You can organize the contents of your report as you wish (i.e. you can for example merge the answers to sections 2 and 3 for each method). The total length of your report should not exceed twelve pages.

The deadline for submitting your report and all accompanying files (source code, data files, etc.) in a single zip file on Moodle is **Friday December 23rd.**.