Software Implementation Project for “Convex Optimization”

Zaiwen Wen
Beijing International Center for Mathematical Research
Peking University

November 15, 2020

1 Proximal Gradient Method for Composite Program

Consider the composite optimization problem

$$\min_x \quad f(x) + h(x), \tag{1.1}$$

where $f(x)$ is differentiable and $h(x)$ is a function whose proximal operator is easily available. Both $f(x)$ and $h(x)$ may be nonconvex. Let $h(x)$ be a proper and close function, and $\inf_{x \in \mathrm{dom} h} h(x) > -\infty$. The proximal operator of $h(x)$ is defined as

$$\mathrm{prox}_h(x) = \arg\min_x h(u) + \frac{1}{2}\|u - x\|^2.$$

Then starting from a suitable initial point x^0, the gradient method is performed as

$$x^{k+1} = \mathrm{prox}_h(x^k - t_k \nabla f(x^k)),$$

where t_k is a chosen step size.

1. Design a software package for the proximal gradient method using the C++ language. The linear algebra should be based on the package “Eigen”:
   `http://eigen.tuxfamily.org`
2. Strategies for choosing the step size t_k:
   - backtracking line search to achieve the Armijo condition
   - backtracking line search to achieve a non-monotone condition using the BB step size
3. A few typical scenarios of $f(x)$ are listed as follows. Your code should support at least two of them. Calculate and written down their gradient.
   - Least squares: $f(x) = \frac{1}{2}\|Ax - b\|_2^2$ or $f(x) = \frac{1}{2}\|Ax - b\|_F^2$.
   - Logistic regression:
   $$f(x) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + \exp(-b_i a_i^T x)).$$

Of course, the choices of $f(x)$ also depends on the selection of $h(x)$. Other functions not in the list are also welcome.

4. A few typical scenarios of $h(x)$ are listed as follows. Your code should support at least three of them. Other functions not in the list are also welcome. Note that the variable x can either be a vector or a matrix. Furthermore, x may also be divided into a few blocks. Calculate and written down the corresponding proximal operator explicitly.

   - General functions
     - vectors: ℓ_0, ℓ_1, ℓ_2, ℓ_∞-norm
     - matrices: $\ell_{1,2}$, $\ell_{2,1}$-norm, generalized group lasso
     - matrices: nuclear norm
     - 1D, 2D TV-norm
     - elastic-net, sum of logarithms, log-barrier
   - indicator functions $1_C(x)$
     - vectors: ℓ_0, ℓ_1, ℓ_2, ℓ_∞-ball
     - vectors: simple box (including the special cases: nonnegative and nonpositive orthant)
     - matrices: nuclear norm ball, positive definite cone, rank constraints

5. A few Matlab package for references:

   - UnLocBox
     `https://epfl-lts2.github.io/unlocbox-html/`
   - ForBES
     `http://kul-forbes.github.io/ForBES/`

6. Requirement:

   (a) Generate both random data and collect a few real data to test the codes.

   (b) Prepare a report including
   - detailed description of the design of the package
   - detailed answers to each question
   - tables of numerical results (including the total number of iterations, the optimality measures, the CPU time and etc) and their interpretation

   (c) Pack the report and all of your codes in one file named as “pg-StudentID-date.zip” and send it to TA: pkuopt@163.com

   (d) Optional: talk to Liu Haoyang (liuhaoyang@pku.edu.cn) and implement things based on his framework.

   (e) If you get significant help from others on one routine, write down the source of references at the beginning of this routine.