Analysis of optimization and numerical approaches to solve the linear least square problem

Emanuele Cosenza Riccardo Massidda

***Abstract*—The linear least square problem can be tackled using a wide range of optimization or numerical methods. The saddle-free Newton method of the class of limited-memory quasi-Newton algorithms has been chosen for the former, whilst the thin QR factorization with Householder reflectors for the latter. Both these algorithms have been implemented from scratch using Python language, to finally experiment over their performances in terms of precision, stability, speed and the accordance of the implementations with the underlying theoretical models.**

Introduction

Given a dataset composed by a matrix $\hat{X} \in \mathbb{R}^{m \times n}$ with $m \geq n$ and a vector $y \in \mathbb{R}^n$, the linear least square (LLS) problem is used to determine the linear function, parameterized by a vector $w \in \mathbb{R}^n$, that best fits the data. (Nocedal and Wright 2006) This can be formalized as the following minimization problem:

$$w_* = \min_w \|\hat{X}w - y\|_2$$

The LLS problem can be dealt both with iterative methods or with direct numerical methods. One algorithm has been implemented for each of these fields to discuss their experimental results.

Saddle-free Newton method

The presence of numerous saddle-points constitutes an issue to both Newton and quasi-Newton traditional iterative methods, the saddle-free Newton method (SFN) is aimed to replicate Newton dynamics yet repulsing saddle-points. (Dauphin et al. 2014)

The original implementation uses a fixed number of steps to approximate the solution of the problem, in the present implementation is instead considered valuable a stopping criterion with accuracy ϵ over the norm of the gradient.

As a quasi-Newton method the SFN does not directly use the Hessian matrix H to overcome the issues related to the positive definitess, instead the matrix $|\mathbf{H}|$ obtained by taking the absolute value of each eigenvalue is used. Also the exact computation of $|\mathbf{H}|$ is avoided thus qualifying the SFN as a limited memory method. This latter feature is obtained by optimizing the function in a lower-dimensional Krylov subspace, exploiting the Lanczos algorithm that produces the k biggest eigenvectors of the Hessian matrix and using them as a base for the subspace. Even inside the Lanczos algorithm the Hessian can be implicitly by using the so called Pearlmutter trick. (Pearlmutter 1994)

The resulting matrix $|\hat{\mathbf{H}}|$ can then be re-used for multiple steps, assuming that the very same won't change much from one iteration to another. The best number of steps t without updating $|\hat{\mathbf{H}}|$ is not trivially determinable and it is so treated as an hyperparameter.

The damping coefficient λ that maximizes the effectiveness of the step is not chosen with a rigorous sub-optimization task, instead as in the original paper a set of discrete values of different order of magnitude is tried.

Thin QR factorization

For the numerical counterpart the thin QR factorization with Householder reflectors has been implemented as described in (Trefethen and Bau 1997).

By using the Householder QR factorization the matrix R is constructed in place of $\hat{X}$, also the n reflection vectors $v_1, \ldots, v_n$ are stored. The reduced matrix $\hat{R}$ is trivially obtainable by slicing as in $\hat{R} = R_{1:n,1:n}$, given that $\hat{X}$ is already stored in memory and fully needed there would not be any advantage in directly constructing the reduced matrix.

By using the Householder vectors it is also possible to implicitly compute $\hat{Q}^T b$ to finally obtain w_* by back substitution over the upper-triangular system $\hat{R}w = \hat{Q}^T b$.

Bibliography

Dauphin, Yann, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. 2014. "Identifying and Attacking the Saddle Point Problem in High-Dimensional Non-Convex Optimization." *arXiv:1406.2572 [Cs, Math, Stat]*, June. http://arxiv.org/abs/1406.2572.

Nocedal, Jorge, and Stephen J. Wright. 2006. *Numerical Optimization.* 2nd ed. Springer Series in Operations Research. New York: Springer.

Pearlmutter, Barak A. 1994. "Fast Exact Multiplication by the Hessian." *Neural Computation* 6 (1): 147–60. https://doi.org/10.1162/neco.1994.6.1.147.

Trefethen, Lloyd N., and David Bau. 1997. *Numerical Linear Algebra.* Philadelphia: Society for Industrial; Applied Mathematics.