Elizabeth Solis Díaz

113032

Convex Optimization for Big Data.

Convex formulations have increased due to new models like support vector machines, the explosion of this tool is mainly due to more efficient algorithms for optimal solutions in computing and in geometry. This is used for convexity to prove solutions.

Convex algorithms are being more necessarily as data sizes are too large.

Fundamentals of Big Data Optimization is defined:

$$F \triangleq min_x \{F(x) \triangleq f(x) + g(x) : x \in R^p\}$$

with g and f as convex functions.

There are three pillars for understanding Big Data Optimization algorithms:

a) First Order Conditions:

Has low accuracy for numerical solutions as this method is based on information only from objective and its first order conditions.

On the other hand, is useful for handle non-smooth variants of an optimal solution. These methods rely on computational primitives, which are ideal for distributed and parallel computation.

b) Randomization:

This technique stands out among other approximation methods as behavior can be controlled. This, technique replaces deterministic gradient with estimators, which speeds up basic linear algebra.

c) Parallel and Distributed Computation:

First Order Scalability can be increased from synchronous parallel algorithms with centralized communications to asynchronous algorithms with decentralized communications.

These three techniques are complementary for Big Data Optimization.

One of main Big Data Problems in many disciplines is:

$$y = \phi x_0 + z$$

$$x_0 = unknown\ parameter;\ \phi \in R^{nxp} =\ is\ known\ matrix;\ z \in R^n =\ unknown\ perturbations/noise$$

With zero-mean iid Gaussian entries with variance σ^2.

It is worth mentioning that linear observation could approximate a more complicated problem of nonlinear phenomena.

The classical convex formulation for signal processing is the least square estimator (LS)

$$\hat{x}_{Ls} = arg\ min_{x \in R^p}\{F(x) : \tfrac{1}{2} \| y - \phi x \|_2^2\}$$

Which can be solved by Krylon subspace method. However, there is a variation of the (LS) which consists in:

$$\hat{x}_{LASSO} = arg\ min_{x \in R^p}\{F(x) : \tfrac{1}{2} \| y - \phi x \|_2^2 + \lambda \| x \|_1\}$$

where λ controls the regularization.

The LASSO operator has the advantage that produces sparse solutions, it is to say that $\hat{x}_{LASSO}$ has mostly zero entries, thus fewer variables. However, its numerical solution is harder since the λ term is non-smooth. Thus, the regularization in sparse signal recovery with the linear model imparts a denoising effect to solution when $n \geq p$.

I. <u>First Order Methods for Non-Smooth Convex Optimization.</u>

The gradient method is low per iteration cost, despite there are faster algorithms that requires fewer iterations to reach a target, those other methods requires additional information of the objective function F or requires a more expensive computations.

The gradient method uses a gradient $\nabla f(x)$ and iteratively performs.

Moreover, many gradient iterations may be performed for the cost of a single iteration of more complicated method, to reach the same level of accuracy ϵ.

Furthermore, it can be analyzed how many iterations gradient would need to reach a certain level of $\epsilon - accuracy$; this analysis is possible by making some assumptions about f. An example is assuming gradient is Lipschitz continuous.

$$\forall x, y \in R^p, \ \| \nabla f(x) - \nabla_f(y) \|_2 \leq L \| x - y \|_2$$

for some constant L.

By setting this step-size $\alpha_k = \frac{1}{L}$ or decreasing f the most, the gradient method needs $O(\frac{1}{\varepsilon})$ - iterations for an accurate solution in the worst case.

However, this convergence does not attain the complexity lower bound, which holds for all the f functions with Lipschitz continuous gradients.

On the other hand, Nesterov achieves an Optimal converge with a modification by $\alpha_k = \frac{1}{L}$ and an extra-momentum step.

Thus, the best possible worst-case rate given by gradient method is commonly referred as an Optimal First-Order method.

There are other functions that has other structures like strong convexity, which offers more efficiency as there is a more unique minimizer.

As strongly convex is worthy to work with, there is a transformation from any convex problem to strongly convex by adding a squared regularization term:

$$\hat{x}_{RIDGE} = arg\ min_{x \in R^p} \{F(x) : \tfrac{1}{2} \| y - \phi x \|_2^2 + \tfrac{\lambda}{2} \| x \|_2^2\}$$

when f is twice differentiable, if eigenvalues of its hessian $\nabla^2 f(x)$ are bounded below.

Also, for LS requires that ϕ to have independent columns.

Moreover, the accelerated-gradient method is benefited from a strong convexity model if there is an appropriate choice of the momentum term, as the method obtains a near-optimal convergence rate.

Proximal-gradient methods takes advantage of composite structure to retain convergence rates similar to gradient method for smooth classes.

Proximal-gradient methods use the same approximation of f but including non-smooth term in an explicit form.

It is worth mentioning that not all the first-order methods are applicable, however by reformulating the linear model by:

$$min_{x,z\in R^p}\{F(x,z)≝h(x)+g(z):\phi z=z\}$$

Can address non-smooth and non-Lipschitz objective functions, also it may be applied the alternating direction method of multipliers. This method requires a penalty parameter γ to produce a sequence of iterations that approaches feasible and produce optimal objective value in the limit. Nonetheless, this method has no convergence guarantee with problems of more than two objectives terms. This can be solved through dual decomposition to treat multiples terms in the objective as separate problems, by solving them simultaneously in parallel.

II. Big Data Scaling Via Randomization

First-order methods may result in infeasible as dimensions grow because of iterations. Nevertheless, randomized approximations increase the reach of first-order methods, as an example, there is Google's PageRank problem, which measures the importance of nodes via its incidence matrix; assuming more important nodes have more connections.

In order to calculate full gradient for Page Rank problem, it requires a matrix operation at each iteration. Moreover, coordinate descent methods are relevant as with this method it could be chosen the coordinate at each iteration.

The difficulty in finding best coordinate requires computational effort as high as gradient. Convergence is slower than gradient method. Nonetheless, by randomization, the coordinate choice results in a cost independent of the set chosen and achieves the same convergence rate.

On the other hand, stochastic gradient methods update all coordinates simultaneously, but uses approximate gradients. Similarly, to coordinate descent methods, the crucial problem is the selection of data points at each iteration. Moreover, results obtained are better at randomly choosing rather than cycling through data.

Stochastic gradient descent methods have shown that using large step sizes and weighted averaging of the iterates allows achieving optimal convergence rates while being robust to the setting of the step size.

III. The Role of Parallel and Distributed Computation.

There are two main issues that block the use of distributed and heterogeneous hardware:

- Deficient communication links between computers and within local hierarchy memory reduces efficiency of first-order methods. It could be addressed by designing algorithms that minimizes communications.
- First-order method must coordinate the activities of different computers whose primitives depend on same vector at each iteration. This slows down the process.

To address this problem, asynchronous algorithms allow update using outdated versions of their parameters.

The ideal scenario for parallelization is where the job is split into independent calculations that can be done simultaneously. To reduce communications, coordinate several descent updates at same time in parallel would help each processor to communicate a single coordinate update.

Models with lock-free stochastic gradient keeps a global vector, processors are free to update it without any notice to other processors.