A Simple Practical Accelerated Method for Finite Sums:Point-SAGA

Menglong Li

May 12, 2017

Abstract

This report will give an intuitive explanation of the point-SAGA algorithm by comparing with SAGA algorithm. We will show that point-SAGA is the combination of SAGA and proximal point algorithm, which means it has the advantage of both algorithms. It is shown that point-SAGA is faster than SAGA when the sample number is smaller than condition number and is as fast as SAGA when sample number is larger than condition number.

Contents

1 Introduction

In ECE543, the ERM algorithm needs to solve the problem:

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} l(f(x_i), y_i).$$

Here, $\mathcal{F}$ is typically a closed convex subset of a Hilbert space $\mathcal{H}$, l is the loss function, $(x_1, y_1), ..., (x_n, y_n)$ is the sample data. Typically, l is convex, for example, hinge loss $l(f; x_i, y_i) = (1 - y_i f(x_i))_+$, logistic loss $\log(1 + \exp(-y_i f(x_i)))$, square loss $(y_i - f(x_i))^2$. Therefore, general finite sum problem has been studied:

$$\frac{1}{n}\sum_{i=1}^{n} f_i(x) = F(x).$$

And many efficient algorithms have been proposed such as stochastic gradient descent(SGD), SAGA. It is known that SGD achieves sublinear convergence rate. In order to accelerate SGD to linear convergence rate, many techniques have been developed. The most recently developed methods are called fast incremental gradient methods(FIG), and SAGA is one example of these methods. Paper[1] gives another accelerated method called point-SAGA by combining the proximal operator with SAGA, and achieves a faster algorithm than SAGA. In the following, we will illustrate point-SAGA in an intuitive way by introducing proximal operator and SAGA and finally compare the complexity of these algorithms.

2 Proximal operator

In gradient descent algorithm, we update the next point by

$$x^{k+1} = x^k - \alpha_k \partial f(x^k),$$

$\alpha_k > 0$ is the stepsize. That means $x^{k+1} - x^k = -\alpha_k \partial f(x^k)$ is the directions such that $f(x^{k+1}) - f(x^k) \leq 0$, i.e. descent direction.
Note that if f is convex, by the definition of subgradient, $f(x^k) \geq f(x^{k+1}) + \partial f(x^{k+1})^T(x^k - x^{k+1})$, we have

$$x^k - x^{k+1} = \alpha_k \partial f(x^{k+1}) \Rightarrow f(x^k) \geq f(x^{k+1}) + \alpha_k \partial ||f(x^{k+1})||^2 \Rightarrow f(x^k) - f(x^{k+1}) \geq 0.$$

Therefore, we can also view $\partial f(x^{k+1})$ as descent direction. If we update the next point as

$$x^{k+1} = x^k - \alpha_k \partial f(x^{k+1}). \tag{1}$$

Then we come to the so-called proximal gradient descent algorithm.
Note that the RHS of (1) depends on x^{k+1}, we introduce the proximal operator:

$$Prox_f(x) = argmin_{y \in \mathbb{R}^n} \{f(y) + \frac{1}{2}||y - x||^2\}.$$

If f is convex,

$$y = Prox_f(x) \Leftrightarrow 0 \in \partial f(y) + y - x \Leftrightarrow y \in x - \partial f(y).$$

Therefore, we can write the proximal gradient descent as:

$$x^{k+1} = Prox_{\alpha_k f}(x^k).$$

In many cases, the proximal operator is easy to compute.

- Indicator function

$$f(x) = \begin{cases} 0, & x \in C \\ \infty, & x \notin C \end{cases}$$

$$Prox_f(x) = argmin_y \{f(y) + \tfrac{1}{2}||x - y||^2\} = argmin_{y \in C} \{||y - x||^2\} = \Pi_C(x).$$

- hinge loss $f(z) = l(z; x_i, y_i) = (1 - y_i z^T x_i)_+$. The proximal operator has a closed form expression:

$$prox_{\gamma f}(z) = z - \gamma y_i \nu x_i,$$

where $s = \frac{1 - y_i z^T x_i}{\gamma ||x_i||^2}$,

$$\nu = \begin{cases} -1, & s \geq 1 \\ 0, & s \leq 0 \\ -s, & otherwise \end{cases}$$

- Squared loss $f(z) = l(z; x_i, y_i) = \frac{1}{2}(y_i - z^T x_i)^2$. Let $\gamma' = \gamma ||x_i||^2$, $a = z^T x_i$ and $c = \frac{a + \gamma' y}{1 + \gamma'}$. Then

$$Prox_{\gamma f}(z) = z - (a - c)\frac{x_i}{||x_i||^2}.$$

We introduce some properties of the proximal operator.

- **Co-coercive** $||Prox_f(x) - Prox_f(x')||^2 \leq (x - x')^T (Prox_f(x) - Prox_f(x'))$.
  Let $u = Prox_f(x), u' = Prox_f(x')$, then $x - u \in \partial f(u), x' - u' \in \partial f(u')$. Note that $f(u') \geq f(u) + \partial f(u)^T (u' - u)$ and $f(u') \geq f(u) + \partial f(u')^T (u - u')$ implies $0 \geq (u - u')^T (\partial f(u') - \partial f(u)) = (u - u')^T (x' - u' - x + u) = ||u - u'||^2 + (u - u')^T (x' - x)$. Therefore, $(u - u')^T (x - x') \geq ||u - u'||^2$.

- **Contraction** $||Prox_f(x) - Prox_f(x')|| \leq ||x - x'||$.
  Co-coercive and Cauchy-schwarz inequality implies $||u - u'||^2 \leq (u - u')^T (x - x') \leq ||u - u'|| \times ||x - x'||$, then $||u - u'|| \leq ||x - x'||$. This is a generalization of projection mapping.

- **Orthogonal decomposition** $x = Prox_f(x) + Prox_{f^*}(x)$, where f^* is the conjugate function of f.

- **Affine transformation** $Prox_h(x) = \frac{1}{t}(Prox_{t^2 f}(tx + a) - a)$, where $h(x) = f(tx + a)$, $t \neq 0$.

- **Conjugate** $Prox_{th}(x) = x - tProx_{h/t}(x/t)$.

- **L2 regularization** $Prox_{\gamma F}(z) = Prox_{\alpha\gamma f}(\alpha z)$. Where $F(x) = f(x) + \frac{\mu}{2}||x||^2$ and $\alpha = 1 - \frac{\mu\gamma}{1 + \mu\gamma}$.

3 Gradient aggregation

In SGD, we only use the information of the current calculated gradient $\partial f_i(x^k)$. If the current iterate has not been displaced too far previous iterates, then stochastic gradient information from previous iterates may still be useful. More specifically, although $E[f_i(x^k)] = \frac{1}{n}\sum_{j=1}^n f_i(x^k)$, that is the average of $f_i(x^k)$ is the deepest descent direction $\frac{1}{n}\sum_{j=1}^n f_i(x^k)$, but the variance may be very large, which means randomly choose the direction $f_i(x^k)$ will not descent fast. However, we can use the previous gradient information to reduce the variance and thus accelerate the descent speed.

3.1 SGD+GA=SAGA

If we combine SGD and GA, we have the SAGA algorithm:

1. Initialize x^1 and stepsize $\alpha > 0$.
2. For $i = 1, ..., n$, compute $\partial f_i(x^1)$, store $\partial f_i(x_{[i]}) \leftarrow \partial f_i(x^1)$.
3. For $k = 1, 2, ...$, choose j uniformly in $\{1, ..., n\}$, compute $\partial f_j(x^k)$. Set the direction $g_k \leftarrow \partial f_j(x^k) - \partial f_j(x_{[j]}) + \frac{1}{n}\sum_{i=1}^n \partial f_i(x_{[i]})$. $x^{k+1} \leftarrow x^k - \alpha g_k$. Store $\partial f_j(x_{[j]}) \leftarrow \partial f_j(x^k)$.

Note that

$$E[g_k] = E_j[\partial f_j(x^k) - \partial f_j(x_{[j]}) + \frac{1}{n}\sum_{i=1}^n \partial f_i(x_{[i]})] = \frac{1}{n}\sum_{i=1}^n \partial f_i(x^k)$$

$\Rightarrow g_k$ is unbiased estimate of deepest descent direction. Note that $Var_j[g_k] = E[(\partial f_j(x^k) - \partial f_j(x_{[j]}) + E[\partial f_j(x_{[j]})] - E[\partial f_j(x^k)])^2] = Var_j(\partial f_j(x^k)) + Var_j(\partial f_j(x_{[j]})) - 2Cov(\partial f_j(x^k), \partial f_j(x_{[j]}))$. Therefore, if $\partial f_j(x^k)$ and $\partial f_j(x_{[j]})$ are highly correlated, then $Var(g_k)$ is smaller than $Var(\partial f_j(x^k))$. Indeed, we have the following theorem.

Theorem 3.1.
- *If f_i are μ-strongly convex and L-smooth and SAGA runs with stepsize $\alpha = \frac{1}{2(\mu n + L)}$, then*

$$E||x^k - x^*||^2 \leq (1 - \frac{\mu}{2(\mu n + L)})^k [||x^0 - x^*||^2 + \frac{n}{\mu n + L}[f(x^0) - (x^0 - x^*)^T \partial f(x^*) - f(x^*)]].$$

- *If only have $\frac{1}{n}\sum_{i=1}^n f_i(x)$ is μ-strongly convex, set $\alpha = \frac{1}{2(\mu n + L)}$, then*

$$E||x^k - x^*||^2 \leq (1 - \frac{\mu}{6(\mu n + L)})^k [||x^0 - x^*||^2 + \frac{n}{\mu n + L}[f(x^0) - (x^0 - x^*)^T \partial f(x^*) - f(x^*)]].$$

- *In the non-strongly convex case, let $\overline{x}^k = \frac{1}{k}\sum_{i=1}^k x^i, \alpha = \frac{1}{3L}$, then*

$$E[F(\overline{x}^k)] - F(x^*) \leq \frac{4n}{k}[\frac{2L}{n}||x^0 - x^*||^2 + f(x^0) - (x^0 - x^*)^T \partial f(x^*) - f(x^*)].$$

Therefore, we can see SAGA achieves linear convergence rate.

3.2 SAGA+Proximal operator=point-SAGA

In the SAGA algorithm, we update the next point by

$$x^{k+1} = z_j^k - \alpha \partial f_j(x^k),$$

where $z_j^k = x^k + \alpha(\partial f_j(x_{[j]}) - \frac{1}{n}\sum_{i=1}^n \partial f_i(x_{[i]}))$, which is a unbiased estimate of x^k. Therefore, SAGA is a revised version of SGD. Point-SAGA utilize the proximal operator in the second step, that is $x^{k+1} = Prox_{\gamma f_j}(z_j^k)$. We write the full point-SAGA algorithm below:
Point-SAGA

1. Initialize x^1 and stepsize $\gamma > 0$.

2. For $i = 1, ..., n$, compute $\partial f_i(x^1)$, store $\partial f_i(x_{[i]}) \leftarrow \partial f_i(x^1)$.

3. For $k = 1, 2, ...$, choose j uniformly in $\{1, ..., n\}$, compute $\partial f_j(x^k)$. Let

$$z_j^k = x^k + \gamma(\partial f_j(x_{[j]}) - \frac{1}{n}\sum_{i=1}^{n} \partial f_i(x_{[i]})),$$

$$x^{k+1} = Prox_{\gamma f_j}(z_j^k).$$

Store $\partial f_j(x_{[j]}) \leftarrow \frac{1}{\gamma}(z_j^k - x^{k+1})$.

In the following, we illustrate the convergence performace of point-SAGA.
Let x^* be the optimal solution. g_j^* is subgradient of f_j at x^* such that $\sum_{i=1}^n g_j^* = 0$. Let $v_j = x^* + \gamma g_j^*$, then $x^* = pror_{\gamma f_j}(v_j)$. Denote g_j^k to be the subgradient of f_j at iteration k. Then

Theorem 3.2 ([1],Theorem 5)**.** *Let*

$$T^k = \frac{c}{n}\sum_{i=1}^{n} ||g_i^k - g_i^*||^2 + ||x^k - x^*||^2,$$

for $c = \frac{1}{\mu L}$. *Then using step size* $\gamma = \frac{\sqrt{(n-1)^2 + 4n\frac{L}{\mu}}}{2Ln} - \frac{1-\frac{1}{n}}{2L}$, *the expectation of* T^{k+1}, *over the random choice of* j, *conditioning on* x^k *and each* g_i^k, *is*

$$E[T^{k+1}] \leq (1-\kappa)T^k$$

for $\kappa = \frac{\mu\gamma}{1+\mu\gamma}$, *when each* $f_i : \mathbb{R}^d \to \mathbb{R}$ *is* L*-smooth and* μ*-strongly convex and* $0 < \mu < L$.

Corollary 3.3 ([1],Corollary 6)**.** *If each* $f_i : \mathbb{R}^d \to \mathbb{R}$ *is* L*-smooth and* μ*-strongly convex, then*

$$E[||x^k - x^*||^2] \leq (1-\kappa)^k \frac{\mu + L}{\mu}||x^0 - x^*||^2.$$

Theorem 3.4 ([1], Theorem 7)**.** ***(Non-smooth case)*** *If each* $f_i : \mathbb{R}^d \to \mathbb{R}$ *is* μ*-strongly convex,* $||g_i^0 - g_i^*|| \leq B$ *and* $||x^0 - x^*|| \leq R$. *Then after* k *iterations of Point-SAGA with step size* $\gamma = \frac{R}{B\sqrt{n}}$*:*

$$E[||\overline{x}^k - x^*||^2] \leq 2\frac{\sqrt{n}(1 + \frac{\mu R}{B\sqrt{n}})}{\mu k} RB,$$

where $\overline{x}^k = \frac{1}{k}E[\sum_{i=1}^k x^i]$.

4 Comparation of complexity

This section, we compare the total complexity of algorithms: gradient descent, SGD, SAGA and Point-SAGA.

	GD	SGD	SAGA	Point-SAGA
number of iterations	$O(\kappa log(1/\epsilon))$	$O(\frac{L}{\mu^2\epsilon})$	$O((\kappa+n)log(1/\epsilon))$	$O((\sqrt{n\kappa}+n)log(1/\epsilon))$
iteration cost	$O(n)$	$O(1)$	$O(1)$	$O(1)$
total complexity	$O(n\kappa log(1/\epsilon))$	$O(\frac{L}{\mu^2\epsilon})$	$O((\kappa+n)log(1/\epsilon))$	$O((\sqrt{n\kappa}+n)log(1/\epsilon))$

Number of iterations is the number of iterations needed to achieve ϵ accuracy. Here, κ is the condition number L/μ. From the above table we found that SAGA and Point-SAGA is much faster than SGD and GD. And Point-SAGA is faster than SAGA when $n << \kappa$, and is as fast as SAGA when $n > \kappa$.

References

[1] Defazio, Aaron. "A simple practical accelerated method for finite sums." Advances In Neural Information Processing Systems. 2016.

[2] Bottou, Léon, Frank E. Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning." arXiv preprint arXiv:1606.04838 (2016).