A DELAYED PROXIMAL GRADIENT METHOD WITH LINEAR CONVERGENCE RATE

Hamid Reza Feyzmahdavian, Arda Aytekin, and Mikael Johansson

School of Electrical Engineering and ACCESS Linnaeus Center
KTH - Royal Institute of Technology, SE-100 44 Stockholm, Sweden

ABSTRACT

This paper presents a new incremental gradient algorithm for minimizing the average of a large number of smooth component functions based on delayed partial gradients. Even with a constant step size, which can be chosen independently of the maximum delay bound and the number of objective function components, the expected objective value is guaranteed to converge linearly to within some ball around the optimum. We derive an explicit expression that quantifies how the convergence rate depends on objective function properties and algorithm parameters such as step-size and the maximum delay. An associated upper bound on the asymptotic error reveals the trade-off between convergence speed and residual error. Numerical examples confirm the validity of our results.

Index Terms— Incremental gradient, asynchronous parallelism, machine learning.

1. INTRODUCTION

Machine learning problems are constantly increasing in size. More and more often, we have access to more data than can be conveniently handled on a single machine, and would like to consider more features than are efficiently dealt with using traditional optimization techniques. This has caused a strong recent interest in developing optimization algorithms that can deal with problems of truly huge scale.

A popular approach for dealing with huge feature vectors is to use coordinate descent methods, where only a subset of the decision variables are updated in each iteration. A recent breakthrough in the analysis of coordinate descent methods was obtained by Nesterov [1] who established global non-asymptotic convergence rates for a class of randomized coordinate descent methods for convex minimization. Nesterov's results have since been extended in several directions, for both randomized and deterministic update orders, (see, e.g. [2–5]).

To deal with the abundance of data, it is customary to distribute the data on multiple (say M) machines, and consider the global loss minimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x)$$

as an average of M losses,

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \frac{1}{M} \sum_{m=1}^M f_m(x). \quad (1)$$

The understanding is here that machine m maintains the data necessary to evaluate $f_m(x)$ and to estimate $\nabla f_m(x)$. Even if a single server maintains the current iterate of the decision vector and orchestrates the gradient evaluations, there will be an inherent delay in querying the machines. Moreover, when the network latency and work load on the machines change, so will the query delay. It is therefore important that techniques developed for this set-up can handle time-varying delays [6–8]. Another challenge in this formulation is to balance the delay penalty of waiting to collect gradients from all machines (to compute the gradient of the total loss) and the bias that occurs when the decision vector is updated incrementally (albeit at a faster rate) as soon as new partial gradient information from a machine arrives (cf. [7]).

In this paper, we consider this second set-up, and propose and analyze a new family of algorithms for minimizing an average of loss functions based on delayed partial gradients. Contrary to related algorithms in the literature, we are able to establish *linear rate* of convergence for minimization of strongly convex functions with Lipschitz-continuous gradients *without any additional assumptions on boundedness* of the gradients (e.g. [6, 7]). We believe that this is an important contribution, since many of the most basic machine learning problems (such as least-squares estimation) do not satisfy the assumption of bounded gradients used in earlier works.

Our algorithm is shown to converge for all upper bounds on the time-varying delay that occurs when querying the individual machines, and explicit expressions for the convergence rate are established. While the convergence rate depends on the maximum delay bound, the constant step-size of the algorithm does not. Similar to related algorithms in the literature, our algorithm does not converge to the optimum unless additional assumptions are imposed on the individual loss functions (e.g. that gradients of component functions all vanish at the optimum, cf. [9]). We also derive an explicit bound on the asymptotic error that reveals the trade-off between convergence speed and residual error. Extensive sim-

ulations show that the bounds are reasonably tight, and highlight the strengths of our method and its analysis compared to alternatives from the literature.

The paper is organized as follows. In the next section, we motivate our algorithm from observations about delay sensitivity of various alternative implementations of delayed gradient iterations. The algorithm and its analysis is then presented in Section 3. Section 4 presents experimental results while Section 5 concludes the paper.

1.1. Notation and preliminaries

Here, we introduce the notation and review the key definitions that will be used throughout the paper. We let $\mathbb{R}$, $\mathbb{N}$, and $\mathbb{N}_0$ denote the set of real numbers, natural numbers, and the set of natural numbers including zero, respectively. The Euclidean norm $\|\cdot\|_2$ is denoted by $\|\cdot\|$. A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is called L -smooth on $\mathbb{R}^n$, if ∇f is Lipschitz-continuous with Lipschitz constant L , defined as

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathbb{R}^n.$$

A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is μ -strongly convex over $\mathbb{R}^n$ if

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2.$$

2. CONVERGENCE RATE OF DELAYED GRADIENT ITERATIONS

The structure of our delayed proximal gradient method is motivated by some basic observations about the delay sensitivity of different alternative implementations of delayed gradient iterations. We summarize these observations in this section.

The most basic technique for minimizing a differentiable convex function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is to use the gradient iterations

$$x(t+1) = x(t) - \alpha \nabla f(x(t)). \quad (2)$$

If f is L -smooth, then these iterations converge to the optimum if the positive step-size α is smaller than $2/L$. If f is also μ -strongly convex, then the convergence rate is linear, and the optimal step-size is $\alpha = 2/(\mu + L)$ [10].

In the distributed optimization setting that we are interested in, the gradient computations will be made available to the central server with a delay. The corresponding gradient iterations then take the form

$$x(t+1) = x(t) - \alpha \nabla f(x(t - \tau(t))), \quad (3)$$

where $\tau : \mathbb{N}_0 \rightarrow \mathbb{N}_0$ is the query delay. Iterations that combine current and delayed states are often hard to analyze, and are known to be sensitive to the time-delay. As an alternative, one could consider updating the iterates based on a delayed prox-step, *i.e.*, based on the difference between the delayed state and the (scaled) gradient evaluated at this delayed state:

$$x(t+1) = x(t - \tau(t)) - \alpha \nabla f(x(t - \tau(t))). \quad (4)$$

One advantage of iterations (4) over (3) is that they are easier to analyze. Indeed, while we are unaware of any theoretical results that guarantee linear convergence rate of the delayed gradient iteration (3) under time-varying delays, we can give the following guarantees for the iteration (4):

Proposition 1 *Assume that $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is L -smooth and μ -strongly convex. If $0 \leq \tau(t) \leq \tau_{\max}$ for all $t \in \mathbb{N}_0$, then the sequence of vectors generated by (4) with the optimal step-size $\alpha = 2/(\mu + L)$ satisfies*

$$\|x(t) - x^*\| \leq \left(\frac{Q-1}{Q+1} \right)^{\frac{t}{1+\tau_{\max}}}, \quad t \in \mathbb{N}_0,$$

where $Q = L/\mu$.

Note that the optimal step-size is independent of the delays, while the convergence rate depends on the upper bound on the time-varying delays. We will make similar observations for the delayed prox-gradient method described and analyzed in Section 3. Another advantage of the iterations (4) over (3) is that they tend to give a faster convergence rate. The following simple example illustrates this point.

Example 1 *Consider minimizing the quadratic function*

$$f(x) = \frac{1}{2}(Lx_1^2 + \mu x_2^2) = \frac{1}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^T \begin{bmatrix} L & 0 \\ 0 & \mu \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

and assume that the gradients are computed with a fixed one-step delay, *i.e.* $\tau(t) = 1$ for all $t \in \mathbb{N}_0$. The corresponding iterations (3) and (4) can then be re-written as linear iterations in terms of the augmented state vector

$$\bar{x}(t) = (x_1(t), x_2(t), x_1(t-1), x_2(t-1))^T,$$

and studied by using the eigenvalues of the corresponding four-by-four matrices. Doing so, we find that

$$\|\bar{x}(t)\| \leq \lambda^t \|\bar{x}(0)\|,$$

where $\lambda = Q/(Q+1)$ for the delayed gradient iterations (3), while $\lambda = \sqrt{(Q^2-1)/(Q+1)}$ for the iterations (4). Clearly, the latter iterations have a smaller convergence factor, and hence, converge faster than the former, see Figure 1.

The combination of a more tractable analysis and potentially faster convergence rate leads us to develop a distributed optimization method based on these iterations next.

3. A DELAYED PROXIMAL GRADIENT METHOD

In this section, we leverage on the intuition developed for delayed gradient iterations in the previous section and develop an optimization algorithm with objective function f of the form (1) under the assumption that M is large. In this case, it

Fig. 1. Comparison of the convergence factor λ of the iterations (3) and (4) for different values of the parameter $Q \in [1, \infty)$.

is natural to use randomized incremental method that operates on a single component f_m at each iteration, rather than on the entire objective function.

Specifically, we consider the following iteration

$$\begin{aligned} i(t) &= \mathcal{U}[1, M], \\ s(t) &= x(t - \tau(t)) - \alpha \nabla f_{i(t)}(x(t - \tau(t))), \\ x(t+1) &= (1 - \theta)x(t) + \theta s(t), \end{aligned} \quad (5)$$

where $\theta \in (0, 1]$, and $i(t) = \mathcal{U}[1, M]$ means that $i(t)$ is drawn from a discrete uniform distribution with support $\{1, 2, \dots, M\}$.

We impose the following basic assumptions:

- A1)** Each $f_m : \mathbb{R}^n \rightarrow \mathbb{R}$, for $m = 1, \dots, M$, is L_m -smooth on $\mathbb{R}^n$.
- A2)** The overall objective function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is μ -strongly convex

Note that Assumption A1) guarantees that f is L -smooth with a constant $L \leq L_{\max}$, where

$$L_{\max} = \max_{1 \leq m \leq M} L_m.$$

Under these assumptions, we can state our main result:

Theorem 2 Suppose in iteration (5) that $\tau(t) \in [0, \tau_{\max}]$ for all $t \in \mathbb{N}_0$, and that the step-size α satisfies

$$\alpha \in \left(0, \frac{\mu}{L_{\max}^2}\right).$$

Then the sequence $\{x(t)\}$ generated by iteration (5) satisfies

$$\mathbb{E}_{t-1}[f(x(t))] - f^* \leq \rho^t (f(x(0)) - f^*) + e, \quad (6)$$

where $\mathbb{E}_{t-1}$ is the expectation over all random variables $i(0), i(1), \dots, i(t-1)$, f^* is the optimal value of problem (1),

$$\rho = \left(1 - 2\alpha\mu\theta \left(1 - \alpha \frac{L_{\max}^2}{\mu}\right)\right)^{\frac{1}{1+\tau_{\max}}}, \quad (7)$$

and

$$\epsilon = \frac{\alpha L_{\max}}{2M(\mu - \alpha L_{\max}^2)} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2. \quad (8)$$

Theorem 2 shows that even with a constant step size, which can be chosen independently of the maximum delay bound $\tau_{\max}$ and the number of objective function components M , the iterates generated by (5) converge linearly to within some ball around the optimum. Note the inherent trade-off between ρ and ϵ : a smaller step-size α yields a smaller residual error ϵ but also a larger convergence factor ρ .

Our algorithm (5) is closely related to Hogwild! [7], which uses randomized delayed gradients as follows:

$$x(t+1) = x(t) - \alpha \nabla f_{i(t)}(x(t - \tau(t))).$$

As discussed in the previous section, iterates combining the current state and delayed gradient are quite difficult to analyze. Thus, while Hogwild! can also be shown to converge linearly to an ϵ -neighborhood of the optimal value, the convergence proof requires that the gradients are bounded, and the step size which guarantees the convergence depends on $\tau_{\max}$, M as well as the maximum bound on $\|\nabla f(x)\|$.

4. EXPERIMENTAL RESULTS

To evaluate the performance of our delayed proximal gradient method, we have focused on unconstrained quadratic programming (QP) optimization problems, since they are frequently encountered in machine learning applications. We are thus interested in solving optimization problems on the form

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad f(x) = \frac{1}{M} \sum_{m=1}^M \left(\frac{1}{2} x^T P_m x + q_m^T x \right). \quad (9)$$

We have chosen to use randomly generated instances, where the matrices P_m and the vectors q_m are generated as explained in [11]. We have considered a scenario with $M = 20$ machines, each with a loss function defined by a randomly generated positive definite matrix $P_m \in \mathbb{R}^{20 \times 20}$, whose condition numbers are linearly spaced between $[1, 10]$, and a random vector $q_m \in \mathbb{R}^{20}$. We have simulated our algorithm with different $\tau_{\max}$ and α values for 1000 times, and present the expected error versus the iteration count.

Figure 2 shows how our algorithm converges to an ϵ -neighborhood of the optimum value, irrespective of the upper bound of the delays. The simulations, shown in light colors, confirm that the delays affect the convergence rate, but not

the remaining error. The theoretical upper bounds derived in Theorem 2, shown in darker color, are clearly valid.

To decrease the residual error, one needs to decrease the step size, α . However, as observed in Figure 3, there is a distinct trade-off: decreasing the step size reduces the remaining error, but also yields slower convergence.

Fig. 2. Convergence for different values of maximum delay bound $\tau_{\max}$ and for fixed step size, α . The dark blue curves represent the theoretical upper bounds on the expected error, whereas the light blue curves represent the averaged experimental results for $\tau_{\max} = 1$ (full lines) and $\tau_{\max} = 7$ (dashed lines).

Fig. 3. Convergence for two different choices of the step size, α . Dashed dark blue curve represents the averaged experimental results for a bigger step size, whereas the solid light blue curve corresponds to a smaller step size.

We have also compared the performance of our method to that of Hogwild!, with the parameters suggested by the theoretical analysis in [7]. To compute an upper bound on the gradients, required by the theoretical analysis in [7], we assumed that the Hogwild! iterates never exceeded the initial value in norm. We simulated the two methods for $\tau_{\max} = 7$. Figure 4 shows that the proposed method converges faster than Hogwild! when the theoretically justified step-sizes were used. In the simulations, we noticed that the step-size for Hogwild! could be increased (yielding faster convergence)

on our quadratic test problems. However, for these step-sizes, the theory in [7] does not give any convergence guarantees.

Fig. 4. Convergence of the two algorithms for $\tau_{\max} = 7$. Solid dark blue curve represents the averaged experimental results of our method, whereas dashed light blue curve represents that of Hogwild! With theoretically justified step-sizes, our algorithm converges faster.

5. CONCLUSIONS

We have proposed a new method for minimizing an average of a large number of smooth component functions based on partial gradient information under time-varying delays. In contrast to previous works in the literature, which require that the gradients of the individual loss functions be bounded, we have established linear convergence rates for our method assuming only that the total objective function is strongly convex and the individual loss functions have Lipschitz-continuous gradients. Using extensive simulations, we have verified our theoretical bounds and shown that they are reasonably tight. Moreover, we observed that with the theoretically justified step-sizes, our algorithm tends to converge faster than Hogwild! on a set of quadratic programming problems.

6. APPENDIX

Before proving Theorem 2, we state a key lemma that is instrumental in our argument. Lemma 3 allows us to quantify the convergence rates of discrete-time iterations with bounded time-varying delays.

Lemma 3 *Let $\{V(t)\}$ be a sequence of real numbers satisfying*

$$V(t+1) \leq pV(t) + q \max_{t-\tau(t) \leq s \leq t} V(s) + r, \quad t \in \mathbb{N}_0, \quad (10)$$

for some nonnegative constants p , q , and r . If $p + q < 1$ and

$$0 \leq \tau(t) \leq \tau_{\max}, \quad t \in \mathbb{N}_0, \quad (11)$$

then

$$V(t) \leq \rho^t V(0) + \epsilon, \quad t \in \mathbb{N}_0, \quad (12)$$

where $\rho = (p+q)^{\frac{1}{1+\tau_{\max}}}$ and $\epsilon = r/(1-p-q)$.

Proof: Since $p+q < 1$, it holds that

$$1 \leq (p+q)^{-\frac{\tau_{\max}}{1+\tau_{\max}}},$$

which implies that

$$\begin{aligned} p + q\rho^{-\tau_{\max}} &= p + q(p+q)^{-\frac{\tau_{\max}}{1+\tau_{\max}}} \\ &\leq (p+q)(p+q)^{-\frac{\tau_{\max}}{1+\tau_{\max}}} \\ &= (p+q)^{\frac{1}{1+\tau_{\max}}} \\ &= \rho. \end{aligned} \quad (13)$$

We now use induction to show that (12) holds for all $t \in \mathbb{N}_0$. It is easy to verify that (12) is true for $t = 0$. Assume that the induction hypothesis holds for all t up to some $\bar{t} \in \mathbb{N}_0$. Then,

$$\begin{aligned} V(\bar{t}) &\leq \rho^{\bar{t}} V(0) + \epsilon, \\ V(s) &\leq \rho^s V(0) + \epsilon, \quad s = \bar{t} - \tau(\bar{t}), \dots, \bar{t}. \end{aligned} \quad (14)$$

From (10) and (14), we have

$$\begin{aligned} V(\bar{t}+1) &\leq p\rho^{\bar{t}} V(0) + p\epsilon + q \left(\max_{\bar{t}-\tau(\bar{t}) \leq s \leq \bar{t}} \rho^s \right) V(0) + q\epsilon + r \\ &\leq p\rho^{\bar{t}} V(0) + p\epsilon + q\rho^{\bar{t}-\tau(\bar{t})} V(0) + q\epsilon + r \\ &\leq p\rho^{\bar{t}} V(0) + p\epsilon + q\rho^{\bar{t}-\tau_{\max}} V(0) + q\epsilon + r \\ &= (p+q\rho^{-\tau_{\max}})\rho^{\bar{t}} V(0) + \epsilon, \end{aligned}$$

where we have used (11) and the fact that $\rho \in [0, 1]$ to obtain the second and third inequalities. It follows from (13) that

$$V(\bar{t}+1) \leq \rho^{\bar{t}+1} V(0) + \epsilon,$$

which completes the induction proof.

6.1. Proof of Theorem 2

First, we analyze how the distance between $f(x(t))$ and f^* changes in each iteration. Since f is convex and $\theta \in (0, 1]$,

$$\begin{aligned} f(x(t+1)) - f^* &= f((1-\theta)x(t) + \theta s(t)) - f^* \\ &\leq (1-\theta)f(x(t)) + \theta f(s(t)) - f^* \\ &= (1-\theta)(f(x(t)) - f^*) + \theta(f(s(t)) - f^*). \end{aligned} \quad (15)$$

As f is $L_{\max}$ -smooth, it follows from [10, Lemma 1.2.3] that

$$\begin{aligned} f(s(t)) &\leq f(x(t-\tau(t))) \\ &\quad + \langle \nabla f(x(t-\tau(t))), s(t) - x(t-\tau(t)) \rangle \\ &\quad + \frac{L_{\max}}{2} \|s(t) - x(t-\tau(t))\|^2. \end{aligned}$$

Note that $s(t) - x(t-\tau(t)) = -\alpha \nabla f_{i(t)}(t - x(\tau(t)))$. Thus,

$$\begin{aligned} f(s(t)) &\leq f(x(t-\tau(t))) \\ &\quad - \alpha \langle \nabla f(x(t-\tau(t))), \nabla f_{i(t)}(x(t-\tau(t))) \rangle \\ &\quad + \frac{\alpha^2 L_{\max}}{2} \|\nabla f_{i(t)}(x(t-\tau(t)))\|^2 \\ &\leq f(x(t-\tau(t))) \\ &\quad - \alpha \langle \nabla f(x(t-\tau(t))), \nabla f_{i(t)}(x(t-\tau(t))) \rangle \\ &\quad + \alpha^2 L_{\max} \|\nabla f_{i(t)}(x(t-\tau(t))) - \nabla f_{i(t)}(x^*)\|^2 \\ &\quad + \alpha^2 L_{\max} \|\nabla f_{i(t)}(x^*)\|^2, \end{aligned}$$

where the second inequality holds since for any vectors x and y , we have

$$\|x+y\|^2 \leq 2(\|x\|^2 + \|y\|^2).$$

Each f_m , for $m = 1, \dots, M$ is convex and L_m -smooth. Therefore, according to [10, Theorem 2.1.5], it holds that

$$\begin{aligned} &\|\nabla f_{i(t)}(x(t-\tau(t))) - \nabla f_{i(t)}(x^*)\|^2 \\ &\leq L_{i(t)} \langle f_{i(t)}(x(t-\tau(t))) - \nabla f_{i(t)}(x^*), x(t-\tau(t)) - x^* \rangle, \\ &\leq L_{\max} \langle f_{i(t)}(x(t-\tau(t))) - \nabla f_{i(t)}(x^*), x(t-\tau(t)) - x^* \rangle, \end{aligned}$$

implying that

$$\begin{aligned} f(s(t)) &\leq f(x(t-\tau(t))) \\ &\quad - \alpha \langle \nabla f(x(t-\tau(t))), \nabla f_{i(t)}(x(t-\tau(t))) \rangle \\ &\quad + \alpha^2 L_{\max}^2 \langle f_{i(t)}(x(t-\tau(t))) - \nabla f_{i(t)}(x^*), x(t-\tau(t)) - x^* \rangle \\ &\quad + \alpha^2 L_{\max} \|\nabla f_{i(t)}(x^*)\|^2. \end{aligned}$$

We use $\mathbb{E}_{t|t-1}$ to denote the conditional expectation in term of $i(t)$ given $i(0), i(1), \dots, i(t-1)$. Note that $i(0), i(1), \dots, i(t)$ are independent random variables. Moreover, $x(t)$ depends on $i(0), \dots, i(t-1)$ but not on $i(t')$ for any $t' \geq t$. Thus,

$$\begin{aligned} \mathbb{E}_{t|t-1}[f(s(t))] - f^* &\leq f(x(t-\tau(t))) - f^* \\ &\quad - \alpha \langle \nabla f(x(t-\tau(t))), \frac{1}{M} \sum_{m=1}^M \nabla f_m(x(t-\tau(t))) \rangle \\ &\quad + \alpha^2 L_{\max}^2 \left\langle \frac{1}{M} \sum_{m=1}^M \nabla f_m(x(t-\tau(t))), x(t-\tau(t)) - x^* \right\rangle \\ &\quad + \frac{\alpha^2 L_{\max}}{M} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2 \\ &= f(x(t-\tau(t))) - f^* \\ &\quad - \alpha \|\nabla f(x(t-\tau(t)))\|^2 \\ &\quad + \alpha^2 L_{\max}^2 \langle \nabla f(x(t-\tau(t))), x(t-\tau(t)) - x^* \rangle \\ &\quad + \frac{\alpha^2 L_{\max}}{M} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2. \end{aligned}$$

As f is μ -strongly convex, it follows from [10, Theorem 2.1.10] that

$$\langle \nabla f(x(t-\tau(t))), x(t-\tau(t)) - x^* \rangle \leq \frac{1}{\mu} \|\nabla f(x(t-\tau(t)))\|^2,$$

which implies that

$$\begin{aligned} \mathbb{E}_{t|t-1}[f(s(t))] - f^* &\leq f(x(t - \tau(t))) - f^* \\ &\quad - \alpha \left(1 - \frac{\alpha L_{\max}^2}{\mu}\right) \|\nabla f(x(t - \tau(t)))\|^2 \\ &\quad + \frac{\alpha^2 L_{\max}}{M} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2. \end{aligned}$$

Moreover, according to [2, Theorem 3.2], it holds that

$$2\mu(f(x(t - \tau(t))) - f^*) \leq \|\nabla f(x(t - \tau(t)))\|^2.$$

Thus, if we take

$$\alpha \in \left(0, \frac{\mu}{L_{\max}^2}\right), \quad (16)$$

we have

$$\begin{aligned} \mathbb{E}_{t|t-1}[f(s(t))] - f^* &\leq \left(1 - 2\mu\alpha\left(1 - \frac{\alpha L_{\max}^2}{\mu}\right)\right) (f(x(t - \tau(t))) - f^*) \\ &\quad + \frac{\alpha^2 L_{\max}}{M} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2. \quad (17) \end{aligned}$$

Define the sequence $\{V(t)\}$ as

$$V(t) = \mathbb{E}_{t-1}[f(x(t))] - f^*, \quad t \in \mathbb{N}.$$

Note that

$$\begin{aligned} V(t+1) &= \mathbb{E}_t[f(x(t+1))] - f^* \\ &= \mathbb{E}_{t-1}[\mathbb{E}_{t|t-1}[f(x(t+1))]] - f^*. \end{aligned}$$

Using this fact together with (15) and (17), we obtain

$$\begin{aligned} V(t+1) &\leq \underbrace{(1 - \theta)}_p V(t) \\ &\quad + \underbrace{\theta \left(1 - 2\mu\alpha\left(1 - \frac{\alpha L_{\max}^2}{\mu}\right)\right)}_q V(t - \tau(t)) \\ &\quad + \underbrace{\frac{\theta\alpha^2 L_{\max}}{M} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2}_r. \end{aligned}$$

One can verify that for any α satisfying (16),

$$p + q = 1 - 2\theta\mu\alpha\left(1 - \frac{\alpha L_{\max}^2}{\mu}\right) \in \left[1 - \frac{\theta\mu^2}{2L_{\max}^2}, 1\right).$$

It now follows from Lemma 3 that

$$V(t) \leq \rho^t V(0) + \epsilon, \quad t \in \mathbb{N}_0,$$

where

$$\rho = (p + q)^{\frac{1}{1+\tau_{\max}}} = \left(1 - 2\alpha\mu\theta\left(1 - \alpha\frac{L_{\max}^2}{\mu}\right)\right)^{\frac{1}{1+\tau_{\max}}},$$

and

$$\epsilon = \frac{r}{1 - p - q} = \frac{\alpha L_{\max}}{2M(\mu - \alpha L_{\max}^2)} \sum_{m=1}^M \|\nabla f_m(x^*)\|^2.$$

The proof is complete.

7. REFERENCES

- [1] Yurii Nesterov, “Efficiency of coordinate descent methods on huge-scale optimization problems,” *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 341–362, 2012.
- [2] Amir Beck and Luba Tretuashvili, “On the convergence of block coordinate descent type methods,” *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2037–2060, 2013.
- [3] Olivier Fercoq and Peter Richtárik, “Accelerated, parallel and proximal coordinate descent,” *arXiv preprint arXiv:1312.5799*, 2013.
- [4] Zhaosong Lu and Lin Xiao, “On the complexity analysis of randomized block-coordinate descent methods,” *arXiv preprint arXiv:1305.4723*, 2013.
- [5] Ji Liu and Stephen J Wright, “Asynchronous stochastic coordinate descent: Parallelism and convergence properties,” *arXiv preprint arXiv:1403.3862*, 2014.
- [6] Alekh Agarwal and John C. Duchi, “Distributed delayed stochastic optimization,” in *IEEE Conference on Decision and Control*, 2012, pp. 5451–5452.
- [7] Benjamin Recht, Christopher Ré, Stephen J. Wright, and Feng Niu, “Hogwild!: A lock-free approach to parallelizing stochastic gradient descent,” in *NIPS*, 2011, pp. 693–701.
- [8] Mu Li, Li Zhou, Zichao Yang, Aaron Li, Fei Xia, David G Andersen, and Alexander Smola, “Parameter server for distributed machine learning,” in *Big Learning Workshop, NIPS*, 2013.
- [9] Mark Schmidt and Nicolas Le Roux, “Fast Convergence of Stochastic Gradient Descent under a Strong Growth Condition,” *arXiv preprint number 1308.6370*, 2013.
- [10] Yurii Nesterov, *Introductory lectures on convex optimization : a basic course*, Kluwer Academic Publ., Boston, Dordrecht, London, 2004.
- [11] Melanie L. Lenard and Michael Minkoff, “Randomly generated test problems for positive definite quadratic programming,” *ACM Trans. Math. Softw.*, vol. 10, no. 1, pp. 86–96, Jan. 1984.