Accelerated Stochastic Block Coordinate Descent with Optimal Sampling

Aston Zhang
Dept. of Computer Science
University of Illinois at Urbana-Champaign
IL, USA 61801
lzhang74@illinois.edu

Quanquan Gu
Dept. of Systems and Information Engineering
University of Virginia
VA, USA 22904
qg5w@virginia.edu

ABSTRACT

We study the composite minimization problem where the objective function is the sum of two convex functions: one is the sum of a finite number of strongly convex and smooth functions, and the other is a general convex function that is non-differentiable. Specifically, we consider the case where the non-differentiable function is block separable and admits a simple proximal mapping for each block. This type of composite optimization is common in many data mining and machine learning problems, and can be solved by block coordinate descent algorithms. We propose an accelerated stochastic block coordinate descent (ASBCD) algorithm, which incorporates the incrementally averaged partial derivative into the stochastic partial derivative and exploits optimal sampling. We prove that ASBCD attains a linear rate of convergence. In contrast to uniform sampling, we reveal that the optimal non-uniform sampling can be employed to achieve a lower iteration complexity. Experimental results on different large-scale real data sets support our theory.

CCS Concepts

•Information systems → Data mining; •Computing methodologies → Machine learning;

Keywords

Stochastic block coordinate descent; Sampling

1. INTRODUCTION

We consider the problem of minimizing a composite function, which is the sum of two convex functions:

$$\mathbf{w}^* = \underset{\mathbf{w}\in\mathbb{R}^d}{\operatorname{argmin}} \quad P(\mathbf{w}) = F(\mathbf{w}) + R(\mathbf{w}), \tag{1.1}$$

where $F(\mathbf{w}) = n^{-1}\sum_{i=1}^{n} f_i(\mathbf{w})$ is a sum of a finite number of strongly convex and smooth functions, and $R(\mathbf{w})$ is a block separable non-differential function. To explain block separability, let $\{\mathcal{G}_1, \ldots, \mathcal{G}_k\}$ be a partition of all the d coordinates where $\mathcal{G}_j$ is a block of coordinates. A subvector $\mathbf{w}_{\mathcal{G}_j}$ is $[w_{k_1}, \ldots, w_{k_{|\mathcal{G}_j|}}]^\top$, where $\mathcal{G}_j = \{k_1, \ldots, k_{|\mathcal{G}_j|}\}$ and $1 \le j \le m$. The fact that $R(\mathbf{w})$ is block separable is equivalent to

$$R(\mathbf{w}) = \sum_{j=1}^{m} r_j(\mathbf{w}_{\mathcal{G}_j}). \tag{1.2}$$

The above problem is common in data mining and machine learning, such as the regularized empirical risk minimization, where $F(\mathbf{w})$ is the empirical loss function averaged over the training data sets, and $R(\mathbf{w})$ is a regularization term. For example, suppose that for a data mining problem there are n instances in a training data set $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$. By choosing the squared loss $f_i(\mathbf{w}) = (\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)^2/2$ and $R(\mathbf{w}) = 0$, a least square regression is obtained. If $R(\mathbf{w})$ is chosen to be the sum of the absolute value of each coordinate in $\mathbf{w}$, it becomes a lasso regression [46]. In general, the problem in (1.1) can be approximately solved by proximal gradient descent algorithms [32] and proximal coordinate descent algorithms [23].

Coordinate descent algorithms have received increasing attention in the past decade in data mining and machine learning due to their successful applications in high dimensional problems with structural regularizers [12, 11, 28, 2, 47]. Randomized block coordinate descent (RBCD) [31, 36, 26, 39, 4, 14, 21] is a special block coordinate descent algorithm. At each iteration, it updates a block of coordinates in vector $\mathbf{w}$ based on evaluation of a random feature subset from the entire training data instances. The iteration complexity of RBCD was established and extended to composite minimization problems [31, 36, 26]. RBCD can choose a constant step size and converge at the same rate as gradient descent algorithms [31, 36, 26]. Compared with gradient descent, the per-iteration time complexity of RBCD is much lower. This is because RBCD computes a partial derivative restricted to only a single coordinate block at each iteration and updates just a single coordinate block of vector $\mathbf{w}$. However, it is still computationally expensive because at each iteration it requires evaluation of the gradient for all the n component functions f_i: the per-iteration computational complexity scales linearly with the training data set size n.

In view of this, stochastic block coordinate descent was proposed recently [8, 51, 48, 35]. Such algorithms compute the stochastic partial derivative restricted to one coordinate block with respect to one component function, rather than the full partial derivative with respect to all the component functions. Essentially, these algorithms employ sampling of both features and data instances at each iteration. However, they can only achieve a sublinear rate of convergence.

We propose an algorithm for stochastic block coordinate descent using optimal sampling, namely *accelerated stochastic block coordinate descent with optimal sampling* (ASBCD). On one hand, ASBCD employs a simple gradient update with optimal non-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD'16, August 13–17, 2016, San Francisco, CA, USA

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ISBN 978-1-4503-4232-2/16/08…$15.00

DOI: http://dx.doi.org/10.1145/2939672.2939819

Algorithm 1 ASBCD: Accelerated Stochastic Block Coordinate Descent with Optimal Sampling

1: **Inputs:** step size η and sampling probability set $\mathcal{P} = \{p_1, \ldots, p_n\}$ of component functions $f_1, \ldots, f_n$
2: **Initialize:** $\boldsymbol{\phi}_i^{(0)} = \mathbf{w}^{(0)} \in \mathbb{R}^d$
3: **for** $t = 1, 2, \ldots$ **do**
4: Sample a component function index i from $\{1, \ldots, n\}$ at probability $p_i \in \mathcal{P}$ with replacement
5: $\boldsymbol{\phi}_i^{(t)} \leftarrow \mathbf{w}^{(t-1)}$
6: Sample a coordinate block index j from $\{1, \ldots, m\}$ uniformly at random with replacement
7: $\mathbf{w}_{\mathcal{G}_j}^{(t)} \leftarrow \operatorname{prox}_{\eta,j}\big(\mathbf{w}_{\mathcal{G}_j}^{(t-1)} - \eta\big[(np_i)^{-1}\nabla_{\mathcal{G}_j} f_i(\boldsymbol{\phi}_i^{(t)}) - (np_i)^{-1}\nabla_{\mathcal{G}_j} f_i(\boldsymbol{\phi}_i^{(t-1)}) + n^{-1}\sum_{k=1}^n \nabla_{\mathcal{G}_j} f_k(\boldsymbol{\phi}_k^{(t-1)})\big]\big)$
8: $\mathbf{w}_{\setminus\mathcal{G}_j}^{(t)} \leftarrow \mathbf{w}_{\setminus\mathcal{G}_j}^{(t-1)}$
9: **end for**

uniform sampling, which is in sharp contrast to the aforementioned stochastic block coordinate descent algorithms based on uniform sampling. On the other hand, we incorporate the incrementally averaged partial derivative into the stochastic partial derivative to achieve a linear rate of convergence rather than a sublinear rate.

To be specific, given error ϵ and number of coordinate blocks m, for strongly convex $f_i(\mathbf{w})$ with the convexity parameter μ and the Lipschitz continuous gradient constant L_i ($L_M = \max_i L_i$), the iteration complexity of ASBCD is

$$\mathcal{O}\left[m\left(\frac{1}{n}\sum_{i=1}^n \frac{L_i}{\mu} + n\right)\log\frac{1}{\epsilon}\right].$$

Notation. Here we define and describe the notation used through this paper. Let w_k be the k^{th} element of a vector $\mathbf{w} = [w_1, \ldots, w_d]^\top \in \mathbb{R}^d$. We use $\|\mathbf{w}\| = \|\mathbf{w}\|_2 = \big(\sum_{k=1}^d w_k^2\big)^{1/2}$ to denote the ℓ_2 norm of a vector $\mathbf{w}$ and $\|\mathbf{w}\|_1 = \sum_{k=1}^d |w_k|$. The subvector of $\mathbf{w}$ excluding $\mathbf{w}_{\mathcal{G}_j}$ is denoted by $\mathbf{w}_{\setminus\mathcal{G}_j}$. The simple proximal mapping for each coordinate block, also known as the proximal operator, is defined as

$$\operatorname{prox}_{\eta,j}(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{u}\in\mathbb{R}^d} \frac{1}{2\eta}\|\mathbf{w}-\mathbf{u}\|^2 + r_j(\mathbf{u}). \tag{1.3}$$

2. THE PROPOSED ALGORITHM

We propose ASBCD (Algorithm 1), an accelerated algorithm for stochastic block coordinate descent with optimal sampling. It starts with known initial vectors $\boldsymbol{\phi}_i^{(0)} = \mathbf{w}^{(0)} \in \mathbb{R}^d$ for all i.

In sharp contrast to stochastic block coordinate descent with uniform sampling, ASBCD selects a component function according to non-uniform probabilities (Line 4 of Algorithm 1).

In Algorithm 1, we define the gradient of any function $f(\boldsymbol{\phi})$ with respect to a coordinate block $\mathcal{G}_j$ of $\boldsymbol{\phi}$ as $\nabla_{\mathcal{G}_j} f(\boldsymbol{\phi}) = [\nabla f(\boldsymbol{\phi})]_{\mathcal{G}_j} = [\partial f(\boldsymbol{\phi})/\partial\phi_{k_1}, \ldots, \partial f(\boldsymbol{\phi})/\partial\phi_{k_{|\mathcal{G}_j|}}]^\top$, where $\mathcal{G}_j = \{k_1, \ldots, k_{|\mathcal{G}_j|}\}$.

Algorithm 1 has a lower computational cost than either proximal gradient descent or RBCD at each iteration. The update at each iteration of Algorithm 1 is restricted to only a sampled component function (Line 4) and a sampled block of coordinates (Line 6).

The key updating step (Line 7) with respect to a stochastic block of coordinates incorporates the incrementally averaged partial derivative into the stochastic partial derivative with the third term $n^{-1}\sum_{k=1}^n \nabla_{\mathcal{G}_j} f_k(\boldsymbol{\phi}_k^{(t-1)})$ within the square bracket. At each iteration with i and j sampled, this summation term $\sum_{k=1}^n \nabla_{\mathcal{G}_j} f_k(\boldsymbol{\phi}_k^{(t-1)})$ is efficiently updated by subtracting $\nabla_{\mathcal{G}_j} f_i(\boldsymbol{\phi}_i^{(t-2)})$ from itself while adding $\nabla_{\mathcal{G}_j} f_i(\boldsymbol{\phi}_i^{(t-1)})$ to itself.

REMARK 2.1. *For many empirical risk minimization problems with each training data instance $(\mathbf{x}_i, y_i)$ and a loss function ℓ, the gradient of $f_i(\mathbf{w})$ with respect to $\mathbf{w}$ is a multiple of $\mathbf{x}_i$: $\nabla f_i(\mathbf{w}) = \ell'(\langle\mathbf{w}, \mathbf{x}_i\rangle, y_i)\mathbf{x}_i$. Therefore, $\nabla f_i(\boldsymbol{\phi}_i)$ can be compactly saved in memory by only saving scalars $\ell'(\langle\boldsymbol{\phi}_i, \mathbf{x}_i\rangle, y_i)$ with the same space cost as those of many other related algorithms MRBCD, SVRG, SAGA, SDCA, and SAG described in Section 5.*

REMARK 2.2. *The sampling probability of component functions f_i in Line 4 of Algorithm 1 is according to a given probability set $\mathcal{P} = \{p_1, \ldots, p_n\}$. The uniform sampling scheme employed by stochastic block coordinate descent methods fits under this more generalized sampling framework as a special case, where $p_i = 1/n$. We reveal that the optimal non-uniform sampling can be employed to lower the iteration complexity in Section 3.*

When taking the expectation of the squared gap between the iterate $\mathbf{w}^{(t)}$ and the optimal solution $\mathbf{w}^*$ in (1.1) with respect to the stochastic coordinate block index, the obtained upper bound does not depend on such an index or the proximal operator. This property may lead to additional algorithmic development and here it is important for deriving a linear rate of convergence for Algorithm 1. We prove the rate of convergence bound in Appendix A after presenting and discussing the main theory in Section 3.

3. MAIN THEORY

In this section, we present and discuss the main theory of our proposed algorithm (Algorithm 1). The proof of the main theory is presented in the appendix.

We begin with the following assumptions on $F(\mathbf{w})$ and $R(\mathbf{w})$ in the composite objective optimization problem as characterized in (1.1). These assumptions are mild and can be verified in many regularized empirical risk minimization problems in data mining and machine learning.

ASSUMPTION 3.1 (LIPSCHITZ CONTINUOUS GRADIENT). *Each gradient $\nabla f_i(\mathbf{w})$ is Lipschitz continuous with the constant L_i,* i.e.*, for all $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{u} \in \mathbb{R}^d$ we have*

$$\|\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{u})\| \le L_i\|\mathbf{w}-\mathbf{u}\|.$$

ASSUMPTION 3.2 (STRONG CONVEXITY). *Each function $f_i(\mathbf{w})$ is strongly convex,* i.e.*, there exists a positive constant μ such that for all $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{u} \in \mathbb{R}^d$ we have*

$$f_i(\mathbf{u}) - f_i(\mathbf{w}) - \langle\nabla f_i(\mathbf{w}), \mathbf{u}-\mathbf{w}\rangle \ge \frac{\mu}{2}\|\mathbf{u}-\mathbf{w}\|^2.$$

Assumption 3.2 implies that $F(\mathbf{w})$ is also strongly convex, i.e.*, there exists a positive constant μ such that for all $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{u} \in \mathbb{R}^d$ we have*

$$F(\mathbf{u}) - F(\mathbf{w}) - \langle\nabla F(\mathbf{w}), \mathbf{u}-\mathbf{w}\rangle \ge \frac{\mu}{2}\|\mathbf{u}-\mathbf{w}\|^2.$$

ASSUMPTION 3.3 (BLOCK SEPARABILITY). *The regularization function $R(\mathbf{w})$ is convex but non-differentiable, and a closed-form solution can be obtained for the proximal operator defined in* (1.3)*. Importantly, $R(\mathbf{w})$ is block separable as defined in* (1.2)*.*

With the above assumptions being made, now we establish the linear rate of convergence for Algorithm 1, which is stated in the following theorem.

THEOREM 3.4. *Let* $L_M = \max_i L_i$ *and* $p_I = \min_i p_i$. *Suppose that Assumptions 3.1—3.3 hold. Based on Algorithm 1 and with* $\mathbf{w}^*$ *defined in* (1.1), *by setting* $\eta = \max_i np_i/[2(n\mu + L_i)]$, $\zeta = np_I/(L_M\eta) - 1 > 0$, $\kappa = L_M^2 m/[2n\eta(L_M - \mu + L_M\eta\mu\zeta)] > 0$, *and* $0 < \alpha = 1 - \eta\mu/m < 1$, *it holds that*

$$\begin{aligned}&\mathbb{E}_{i,j}[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2]\\ &\leq \alpha^t\big[\|\mathbf{w}^{(0)} - \mathbf{w}^*\|^2 + \frac{1}{\kappa}[F(\mathbf{w}^{(0)}) - F(\mathbf{w}^*)\\ &\quad - \langle\nabla F(\mathbf{w}^*), \mathbf{w}^{(0)} - \mathbf{w}^*\rangle]\big].\end{aligned}$$

REMARK 3.5. *Theorem 3.4 justifies the linear rate of convergence for Algorithm 1. Parameter* α *depends on the number of coordinate blocks* m. *It may be tempting to set* $m = 1$ *for faster convergence. However, this is improper due to lack of considerations for the computational cost at each iteration. When* $m = 1$, *at each iteration the gradient is updated with respect to all coordinates. When* $m > 1$, *at each iteration of Algorithm 1 the gradient is updated with respect to only a sampled coordinate block among all coordinates, so the computational cost is lower than that of* $m = 1$ *per iteration. Therefore, comparing algorithms that update the gradient with respect to different numbers of coordinates per iteration should be based on the same number of entire data passes (the least possible iterations for passing through the entire data instances with respect to all coordinates). We perform experiments to compare such different algorithms in Section 4.*

REMARK 3.6. *Theorem 3.4 implies a more generalized iteration complexity of Algorithm 1, which is*

$$\mathcal{O}\bigg[m\bigg(\min_i \frac{L_i/\mu + n}{np_i}\bigg)\log\frac{1}{\epsilon}\bigg] \tag{3.1}$$

given the error $\epsilon > 0$. *The uniform sampling scheme fits this more generalized result with* $p_i = 1/n$. *With* $L_M = \max_i L_i$, *by setting* $p_i = 1/n$, $\eta = 1/[2(L_M + n\mu)] > 0$, $\zeta = (L_M + 2n\mu)/L_M > 0$, $\kappa = m/[2n\eta(1 - \eta\mu)] > 0$, *and* $0 < \alpha = 1 - \mu/[2m(L_M + n\mu)] < 1$, *Theorem 3.4 still holds. The iteration complexity of ASBCD with uniform sampling is*

$$\mathcal{O}\bigg[m\bigg(\frac{L_M}{\mu} + n\bigg)\log\frac{1}{\epsilon}\bigg]. \tag{3.2}$$

Now we show that the iteration complexity in (3.2) *can be further improved by optimal sampling. To begin with, minimizing* α *can be achieved by maximizing* η *with respect to* p_i. *It is easy to show that* η *is maximized when* $p_i = (n + L_i/\mu)/\sum_{k=1}^n (n + L_k/\mu)$. *Then, by setting* $\eta = n/\big[2\sum_{i=1}^n (n\mu + L_i)\big] > 0$ *we obtain the iteration complexity of ASBCD with optimal sampling:*

$$\mathcal{O}\bigg[m\bigg(\frac{1}{n}\sum_{i=1}^n \frac{L_i}{\mu} + n\bigg)\log\frac{1}{\epsilon}\bigg]. \tag{3.3}$$

COROLLARY 3.7. *Let* $L_M = \max_i L_i$. *Suppose that Assumptions 3.1—3.3 hold. Based on Algorithm 1 and with* $\mathbf{w}^*$ *defined in* (1.1), *by setting* $p_i = (n + L_i/\mu)/\sum_{k=1}^n (n + L_k/\mu)$, $\zeta = \sum_{i=1}^n L_i^{-1}/\sum_{i=1}^n (2n\mu + 2L_i)^{-1} - 1 > 0$, *and* $0 < \alpha = 1 - n\mu/[2m\sum_{i=1}^n (n\mu + L_i)] < 1$, *we chose* $\eta = n/[2\sum_{i=1}^n (n\mu + L_i)] > 0$ *and it holds that*

$$\begin{aligned}&\mathbb{E}_{i,j}[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2]\\ &\leq \alpha^t\Big[\|\mathbf{w}^{(0)} - \mathbf{w}^*\|^2 + \frac{n}{m(L_M + n\mu)}[F(\mathbf{w}^{(0)}) - F(\mathbf{w}^*)\\ &\quad - \langle\nabla F(\mathbf{w}^*), \mathbf{w}^{(0)} - \mathbf{w}^*\rangle]\Big].\end{aligned}$$

Comparing the iteration complexity of ASBCD in (3.3) and (3.2), it is clear that the optimal sampling scheme results in a lower iteration complexity than uniform sampling.

4. EVALUATION

We conduct experiments to evaluate the performance of our proposed ASBCD algorithm in comparison with different algorithms on large-scale real data sets.

4.1 Problems and Measures

We define the problems and measures used in the empirical evaluation. Classification and regression are two corner-stone data mining and machine learning problems. We evaluate the performance of the proposed ASBCD algorithm in solving these two problems.

4.1.1 Classification and Regression Problems

As a case study, the classification problem is $\ell_{1,2}$-regularized logistic regression:

$$\begin{aligned}\mathbf{w}^* &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} P(\mathbf{w})\\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \log\big[1 + \exp(-y_i\langle\mathbf{w}, \mathbf{x}_i\rangle)\big]\\ &\quad + \frac{\lambda_2}{2}\|\mathbf{w}\|^2 + \lambda_1\|\mathbf{w}\|_1.\end{aligned}$$

For the the regression problem in this empirical study, the elastic net is used:

$$\begin{aligned}\mathbf{w}^* &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} P(\mathbf{w})\\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \frac{(\langle\mathbf{w}, \mathbf{x}_i\rangle - y_i)^2}{2} + \frac{\lambda_2}{2}\|\mathbf{w}\|^2 + \lambda_1\|\mathbf{w}\|_1.\end{aligned}$$

The regularization parameters λ_1 and λ_2 in both problems are tuned by proximal gradient descent using five-fold cross-validation on the training data sets.

4.1.2 Measures for Convergence and Testing Accuracy

Recall the problem of composite function minimization as formalized in (1.1). In evaluation of the algorithm performance on the convergence effect, we use the measure of objective gap value: $P(\mathbf{w}) - P(\mathbf{w}^*)$.

To further study model prediction capabilities that are trained by different algorithms, we evaluate testing accuracy using two different measures:

- **AUC**: For the classification problem, area under receiver operating characteristic curve (AUC) is measured [13]. Note that a higher testing accuracy can be reflected by a higher AUC.
- **MSE**: For the regression problem, mean squared error (MSE) is compared. Note that a higher testing accuracy can be reflected by a lower MSE.

4.2 Large-Scale Real Data Sets

The empirical studies are conducted on the following four real data sets that are downloaded using the LIBSVM software [3]:

- **KDD 2010**: Bridge to Algebra data set from KDD Cup 2010 Educational Data Mining Challenge [44].
- **COVTYPE**: Data set for predicting forest cover type from cartographic variables [22].

Table 1: Summary statistics of four large-scale real data sets in the experiments. These data sets are used for evaluating performance of algorithms in solving two corner-stone data mining and machine learning problems: classification and regression.

Data Set	#Training Instances	#Testing Instances	#Features	Problem	Measure for Testing Accuracy (↑)
KDD 2010	19,264,097	748,401	29,890,095	Classification	AUC (↑)
COVTYPE	290,506	290,506	54	Classification	AUC (↑)
RCV1	20,242	677,399	47,236	Classification	AUC (↑)
E2006-TFIDF	16,087	3,308	150,360	Regression	MSE (↓)

- **RCV1**: Reuters Corpus Volume I data set for text categorization research [20].
- **E2006-TFIDF**: Data set for predicting risk from financial reports from thousands of publicly traded U.S. companies [16].

Each of these real data sets has a large size in either its instance count or feature size, or both. For instance, the KDD 2010 data set has over 19 million training instances with nearly 30 million features. Summary statistics of these data sets are provided in Table 1.

4.3 Algorithms for Comparison

We evaluate the performance of ASBCD in comparison with recently proposed competitive algorithms. To comprehensively evaluate ASBCD, we also compare variants of ASBCD with different sampling schemes.

Below are the seven algorithms for comparison.

- **SGD (SG)**: Proximal stochastic gradient descent. This algorithm has a sublinear rate of convergence. To ensure the high competitiveness of this algorithm, the implementation is based on a recent work [1].
- **SBCD (SB)**: Stochastic block coordinate descent. It is the same as SGD except that SBCD updates the gradient with respect to a randomly sampled block of coordinates at each iteration. SBCD also converges at a sublinear rate.
- **SAGA (SA)**: Advanced stochastic gradient method [9]. This algorithm is based on uniform sampling of component functions. It updates the gradient with respect to all coordinates at each iteration. SAGA has a linear rate of convergence.
- **SVRG (SV)**: (Proximal) stochastic variance reduced gradient [15, 50]. This algorithm is based on uniform sampling of component functions. It updates the gradient with respect to all coordinates at each iteration. Likewise, SVRG converges to the optimum at a linear rate.
- **MRBCD (MR)**: Mini-batch randomized block coordinate descent [54]. This algorithm uses uniform sampling of component functions. MRBCD converges linearly to the optimum.
- **ASBCD-U (U)**: The proposed ASBCD algorithm with uniform sampling of component functions. The sampling probability p_i for component function f_i is $p_i = 1/n$. The sampling probability p_i for component function f_i: $p_i = L_i / \sum_{k=1}^{n} L_k$.
- **ASBCD-O (O)**: The proposed ASBCD algorithm with optimal sampling as described in Corollary 3.7. The sampling probability p_i for component function f_i is $p_i = (n+L_i/\mu)/\sum_{k=1}^{n}(n+L_k/\mu)$.

4.4 Experimental Setting

Note that algorithms SBCD, MRBCD, and ASBCD update the gradient with respect to a sampled block of coordinates at each iteration. In contrast, SGD, SAGA, and SVRG update the gradient with respect to all the coordinates per iteration. Recalling Remark 3.5, comparison of these algorithms is based on the same entire data passes.

4.4.1 Equipment Configuration

We evaluate convergence and testing accuracy with respect to training time. The experiments on the KDD 2010 data set are conducted on a computer with two 14-core 2.4GHz CPUs and a 256GB RAM while the experiments on the other data sets are conducted on a computer with an 8-core 3.4GHz CPU and a 32GB RAM.

4.4.2 Parameter Setting

Different from the other algorithms in comparison, the SVRG and MRBCD algorithms both have multiple stages with two nested loops. The inner-loop counts in SVRG and MRBCD are set to the training data instance counts as suggested in a few recent studies [15, 50, 54].

For each algorithm, its parameters, such as the step size (η in this paper), are chosen around the theoretical values to give the fastest convergence under the five-fold cross validation. Here we describe the details. The training data set is divided into five subsets of approximately the same size. One validation takes five trials on different subsets: in each trial, one subset is left out and the remaining four subsets are used. The convergence effect in one cross-validation is estimated by the averaged performance of the five trials.

4.5 Experimental Results

All the experimental results are obtained from 10 replications. Both the mean and standard deviation values are reported in Tables 2—6. For clarity of exposition, Figures 1—4 plot the mean values of the results from all these replications.

4.5.1 Results on KDD 2010

For classification on KDD 2010, Figure 1 compares convergence of all the algorithms for the same number of entire data passes. In general, among all the seven algorithms in comparison, ASBCD with optimal sampling converges fastest to the optimum for the same number of entire data passes. Sublinearly-convergent algorithms SGD and SBCD converge much more slowly than the other linearly-convergent algorithms.

We also observe from Figure 1 that, stochastic block coordinate descent algorithms generally converge faster than those algorithms without using stochastic block coordinates. For instance, SBCD converges faster than SGD while MRBCD converges faster than SVRG for the same number of entire data passes.

The varied convergence effects across all the seven algorithms can be visualized more clearly when they are compared for the same training time. Figure 2 exhibits such performance variations. Clearly, ASBCD with optimal sampling achieves the fastest convergence for the same training time. Similar to the results in Figure 1, for the same training time, stochastic block coordinate descent algorithms still generally converge faster than those algorithms without using stochastic block coordinates.

It is not surprising that the convergence effects influence the testing accuracy. Tables 2 and 3 report the AUC comparison of algorithms for the same entire data passes and training time. Consist-

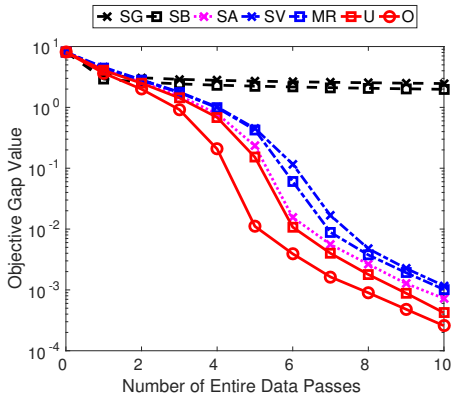


Figure 1: Classification on KDD 2010: Convergence comparison of algorithms for the same number of entire data passes. In general, ASBCD with optimal sampling (O) converges fastest to the optimum for the same number of entire data passes.

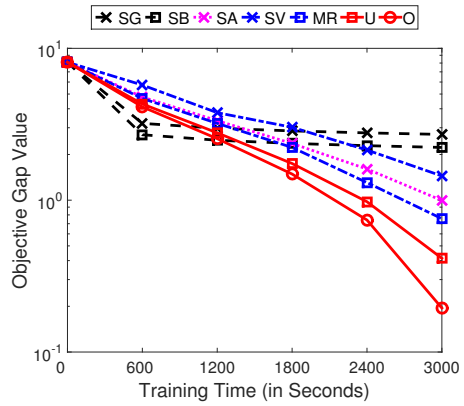


Figure 2: Classification on KDD 2010: Convergence comparison of algorithms for the same training time. In general, ASBCD with optimal sampling (O) converges fastest to the optimum for the same training time.

ent with the observed convergence performance in Figures 1 and 2, ASBCD with optimal sampling generally achieves the highest testing accuracy for both the same number of entire data passes and the same training time.

4.5.2 Results on More Data Sets

We further compare the algorithms on three more data sets COVTYPE, RCV1, and E2006-TFIDF as described in Section 4.2 and summarized in Table 1. COVTYPE and RCV1 are used for the classification problem, while E2006-TFIDF is for the regression problem. All the results are reported in Tables 4—6, Figure 3, and Figure 4. To begin with, we describe the convergence effects. Figures 3 and 4 compare convergence of algorithms for the same entire data passes and for the same training time. In general, ASBCD with optimal sampling (O) converges fastest to the optimum for both the same number of entire data passes and the same training time.

Tables 4—6 present the testing accuracy comparison results for the same training time. Note that a lower MSE for regression on E2006-TFIDF indicates a higher testing accuracy. These testing accuracy results agree with the varied convergence effects of different algorithms on the same data set. Among all the algorithms, ASBCD with optimal sampling generally achieves the highest testing accuracy for the same training time.

5. RELATED WORK

The first line of research in modern optimization is randomized block coordinate descent (RBCD) algorithms [11, 49, 25, 39, 36]. These algorithms exploit the block separability of regularization function $R(\mathbf{w})$. With separable coordinate blocks, such algorithms only compute the gradient of $F(\mathbf{w})$ with respect to a randomly selected block at each iteration rather than the full gradient with respect to all coordinates: they are faster than the full gradient descent at each iteration [11, 49, 25, 39, 36]. However, such algorithms still compute the exact partial gradient based on all the n component functions per iteration, though accessing the entire component functions is computationally more expensive when the training data set has a larger number of instances [52].

Recently, an MRBCD algorithm was proposed for randomized block coordinate descent using mini-batches [54]. At each iteration, both a block of coordinates and a mini-batch of component functions are sampled but there are multiple stages with two nested loops. For each iteration of the outer loop, the exact gradient is computed once; while in the follow-up inner loop, gradient estimation is computed multiple times to help adjust the exact gradient. MRBCD has a linear rate of convergence for strongly convex and smooth $F(\mathbf{w})$ only when the batch size is "large enough" although batches of larger sizes increase the per-iteration computational cost [54] (Theorem 4.2). Similar algorithms and theoretical results to those of MRBCD were also proposed [48, 18]. Chen and Gu further considered related but different sparsity constrained non-convex problems and studied stochastic optimization algorithms with block coordinate gradient descent [6].

Our work departs from the related work in the above line of research by attaining a linear convergence using optimally and non-uniformly sampling of a single data instance at each of iterations.

The second line of research in modern optimization is proximal gradient descent. In each iteration, a proximal operator is used in the update, which can be viewed as a special case of splitting algorithms [24, 5, 35]. Proximal gradient descent is computationally expensive at each iteration, hence proximal stochastic gradient descent is often used when the data set is large. At each iteration, only one of the n component functions f_i is sampled, or a subset of f_i are sampled, which is also known as mini-batch proximal stochastic gradient [43]. Advantages for proximal stochastic gradient descent are obvious: at each iteration much less computation of the gradient is needed in comparison with proximal gradient descent. However, due to the variance in estimating the gradient by stochastic sampling, proximal stochastic gradient descent has a sublinear rate of convergence even when $P(\mathbf{w})$ is strongly convex and smooth.

To accelerate proximal stochastic gradient descent, variance reduction methods were proposed recently. Such accelerated algorithms include stochastic average gradient (SAG) [38], stochastic dual coordinate ascent (SDCA) [42], stochastic variance reduced gradient (SVRG) [15], semi-stochastic gradient descent (S2GD) [19], permutable incremental gradient (Finito) [10], minimization by incremental surrogate optimization (MISO) [27], and advanced stochastic gradient method (SAGA) [9]. There are also some more recent extensions in this line of research, such as proximal SDCA (ProxSDCA) [40], accelerated mini-batch SDCA (ASDCA) [41], adaptive variant of SDCA (AdaSDCA) [7], randomized dual coordinate ascent (Quartz) [34], mini-batch S2GD (mS2GD) [17], and proximal SVRG (ProxSVRG) [50].

Besides, several studies show that non-uniform sampling can be used to improve the rate of convergence of stochastic optimization

Table 2: Classification on KDD 2010: AUC comparison of algorithms for the same entire data passes. The boldfaced results with symbol • denote the highest AUC among all the algorithms for the same number of entire data passes.

Method	#Data Passes = 2		#Data Passes = 4		#Data Passes = 6		#Data Passes = 8		#Data Passes = 10	
	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.
SGD	0.8341	±0.0214	0.8343	±0.0188	0.8348	±0.0153	0.8350	±0.0123	0.8351	±0.0179
SBCD	0.8352	±0.0278	0.8353	±0.0264	0.8354	±0.0315	0.8355	±0.0288	0.8357	±0.0352
SAGA	0.8360	±0.0103	0.8401	±0.0105	0.8481	±0.0305	0.8528	±0.0154	0.8542	±0.0097
SVRG	0.8349	±0.0111	0.8393	±0.0146	0.8438	±0.0178	0.8514	±0.0097	0.8531	±0.0087
MRBCD	0.8352	±0.0301	0.8395	±0.0232	0.8449	±0.0270	0.8517	±0.0162	0.8535	±0.0183
ASBCD-U	0.8367	±0.0153	0.8407	±0.0127	0.8494	±0.0212	0.8530	±0.0165	0.8551	±0.0168
ASBCD-O	**0.8374** •	±0.0112	**0.8429** •	±0.0133	**0.8525** •	±0.0109	**0.8542** •	±0.0098	**0.8562** •	±0.0076

*Std.: Standard Deviation

Table 3: Classification on KDD 2010: AUC comparison of algorithms for the same training time. The boldfaced results with symbol • denote the highest AUC among all the algorithms for the same training time.

Method	Time = 600s		Time = 1200s		Time = 1800s		Time = 2400s		Time = 3000s	
	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.
SGD	0.8339	±0.0163	0.8341	±0.0153	0.8342	±0.0146	0.8344	±0.0154	0.8344	±0.0112
SBCD	0.8348	±0.0265	0.8350	±0.0233	0.8350	±0.0245	0.8352	±0.0256	0.8353	±0.0356
SAGA	0.8306	±0.0223	0.8337	±0.0241	0.8351	±0.0153	0.8365	±0.0151	0.8379	±0.0099
SVRG	0.8293	±0.0287	0.8320	±0.0198	0.8340	±0.0166	0.8353	±0.0083	0.8368	±0.0087
MRBCD	0.8309	±0.0280	0.8339	±0.0296	0.8356	±0.0146	0.8370	±0.0170	0.8390	±0.0153
ASBCD-U	0.8311	±0.0148	0.8346	±0.0150	0.8367	±0.0153	0.8385	±0.0134	0.8415	±0.0113
ASBCD-O	**0.8314** •	±0.0122	**0.8351** •	±0.0097	**0.8371** •	±0.0103	**0.8396** •	±0.0087	**0.8432** •	±0.0081

*Std.: Standard Deviation

algorithms [45, 31, 29, 50, 34, 53, 37, 33]. However, the proposed sampling schemes in these studies cannot be directly applied to our algorithm, because they are limited in at least one of the following two aspects: (1) the algorithm does not apply to composite objectives with a non-differentiable function; (2) it does not support randomized block coordinate descent.

6. CONCLUSION

Research on big data is increasingly important and common. Training data mining and machine learning models often involve minimizing empirical risk or maximizing likelihood over the training data set, especially in solving classification and regression problems. Thus, big data research may rely on optimization algorithms, such as proximal gradient descent algorithms. At each iteration, proximal gradient descent algorithms have a much higher computational cost due to updating gradients based on all the data instances and features. Randomized block coordinate descent algorithms are still computationally expensive at each iteration when the data instance size is large. Therefore, we focused on stochastic block coordinate descent that samples both data instances and features at every iteration.

We proposed the ASBCD algorithm to accelerate stochastic block coordinate descent. ASBCD incorporates the incrementally averaged partial derivative into the stochastic partial derivative. For smooth and strongly convex functions with non-differentiable regularization functions, ASBCD is able to achieve a linear rate of convergence. The optimal sampling achieves a lower iteration complexity for ASBCD. The empirical evaluation with both classification and regression problems on four large-scale real data sets supported our theory.

Acknowledgement. We would like to thank the anonymous reviewers for their helpful comments. This research was partially supported by Quanquan Gu's startup funding at Department of Systems and Information Engineering, University of Virginia.

7. REFERENCES

[1] L. Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer, 2012.

[2] P. Breheny and J. Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The annals of applied statistics*, 5(1):232, 2011.

[3] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[4] K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. Coordinate descent method for large-scale l2-loss linear support vector machines. *The Journal of Machine Learning Research*, 9:1369–1398, 2008.

[5] G. H. Chen and R. Rockafellar. Convergence rates in forward–backward splitting. *SIAM Journal on Optimization*, 7(2):421–444, 1997.

[6] J. Chen and Q. Gu. Accelerated stochastic block coordinate gradient descent for sparsity constrained nonconvex optimization. In *Conference on Uncertainty in Artificial Intelligence*, 2016.

[7] D. Csiba, Z. Qu, and P. Richtárik. Stochastic dual coordinate ascent with adaptive probabilities. *arXiv:1502.08053*, 2015.

[8] C. D. Dang and G. Lan. Stochastic block mirror descent methods for nonsmooth and stochastic optimization. *SIAM Journal on Optimization*, 25(2):856–881, 2015.

[9] A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

[10] A. J. Defazio, T. S. Caetano, and J. Domke. Finito: A faster, permutable incremental gradient method for big data

Table 4: Classification on COVTYPE: AUC comparison of algorithms for the same training time. The boldfaced results with symbol • denote the highest AUC among all the algorithms for the same training time.

Method	Time = 100s		Time = 200s		Time = 300s		Time = 400s		Time = 500s	
	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.
SGD	0.6998	±0.0211	0.7004	±0.0309	0.7007	±0.0160	0.7010	±0.0187	0.7010	±0.0255
SBCD	0.7021	±0.0528	0.7024	±0.0352	0.7026	±0.0312	0.7027	±0.0319	0.7028	±0.0334
SAGA	0.7259	±0.0165	0.7321	±0.0323	0.7318	±0.0298	0.7332	±0.0075	0.7351	±0.0135
SVRG	0.7234	±0.0176	0.7265	±0.0233	0.7288	±0.0236	0.7298	±0.0182	0.7305	±0.0208
MRBCD	0.7241	±0.0313	0.7268	±0.0254	0.7295	±0.0216	0.7306	±0.0249	0.7313	±0.0185
ASBCD-U	**0.7302** •	±0.0243	**0.7329** •	±0.0249	0.7372	±0.0156	0.7393	±0.0175	0.7424	±0.0123
ASBCD-O	0.7296	±0.0258	0.7325	±0.0223	**0.7390** •	±0.0149	**0.7438** •	±0.0104	**0.7483** •	±0.0112

*Std.: Standard Deviation

Table 5: Classification on RCV1: AUC comparison of algorithms for the same training time. The boldfaced results with symbol • denote the highest AUC among all the algorithms for the same training time.

Method	Time = 100s		Time = 200s		Time = 300s		Time = 400s		Time = 500s	
	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.
SGD	0.9192	±0.0143	0.9194	±0.0213	0.9194	±0.0202	0.9195	±0.0143	0.9196	±0.0138
SBCD	0.9193	±0.0198	0.9194	±0.0214	0.9196	±0.0224	0.9197	±0.0188	0.9199	±0.0167
SAGA	0.9215	±0.0204	0.9231	±0.0280	0.9240	±0.0198	0.9257	±0.0143	0.9278	±0.0321
SVRG	0.9201	±0.0251	0.9210	±0.0250	0.9221	±0.0132	0.9224	±0.0049	0.9228	±0.0099
MRBCD	0.9206	±0.0203	0.9223	±0.0144	0.9226	±0.0139	0.9233	±0.0199	0.9242	±0.0168
ASBCD-U	0.9216	±0.0166	0.9228	±0.0072	0.9242	±0.0069	0.9259	±0.0045	0.9282	±0.0057
ASBCD-O	**0.9218** •	±0.0199	**0.9232** •	±0.0063	**0.9247** •	±0.0048	**0.9268** •	±0.0031	**0.9291** •	±0.0040

*Std.: Standard Deviation

Table 6: Regression on E2006-TFIDF: MSE comparison of algorithms for the same training time. The boldfaced results with symbol • denote the lowest MSE among all the algorithms for the same training time.

Method	Time = 300s		Time = 600s		Time = 900s		Time = 1200s		Time = 1500s	
	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.	Mean	±Std.
SBCD	0.2852	±0.0219	0.2806	±0.0118	0.2794	±0.0132	0.2790	±0.0143	0.2773	±0.0164
SGD	0.2850	±0.0142	0.2829	±0.0136	0.2824	±0.0136	0.2815	±0.0187	0.2808	±0.0238
SAGA	0.2386	±0.0136	0.1892	±0.0139	0.1629	±0.0109	0.1611	±0.0129	0.1598	±0.0109
SVRG	0.2723	±0.0123	0.2324	±0.0134	0.2008	±0.0043	0.1643	±0.0085	0.1620	±0.0048
MRBCD	0.2402	±0.0144	0.1854	±0.0098	0.1635	±0.0130	0.1607	±0.0092	0.1594	±0.0096
ASBCD-U	0.2183	±0.0043	0.1653	±0.0050	0.1628	±0.0087	0.1605	±0.0054	0.1589	±0.0042
ASBCD-O	**0.2158** •	±0.0078	**0.1647** •	±0.0064	**0.1621** •	±0.0047	**0.1597** •	±0.0057	**0.1583** •	±0.0054

*Std.: Standard Deviation

problems. In *Proceedings of the International Conference on Machine Learning*, 2014.

[11] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

[12] W. J. Fu. Penalized regressions: the bridge versus the lasso. *Journal of computational and graphical statistics*, 7(3):397–416, 1998.

[13] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[14] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th international conference on Machine learning*, pages 408–415. ACM, 2008.

[15] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

[16] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 272–280. Association for Computational Linguistics, 2009.

[17] J. Konečnỳ, J. Liu, P. Richtárik, and M. Takáč. ms2gd: Mini-batch semi-stochastic gradient descent in the proximal setting. *arXiv:1410.4744*, 2014.

[18] J. Konečnỳ, Z. Qu, and P. Richtárik. Semi-stochastic coordinate descent. *arXiv preprint arXiv:1412.6293*, 2014.

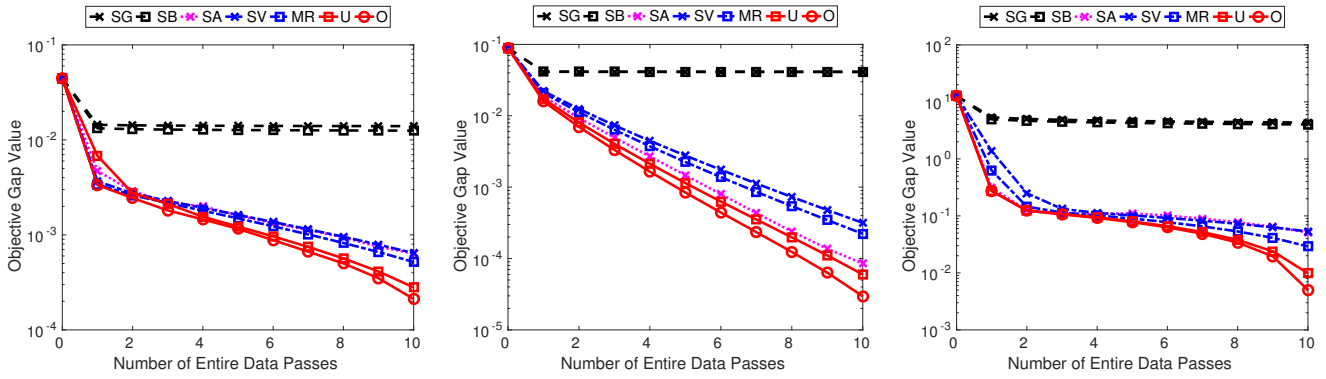
[19] J. Konečnỳ and P. Richtárik. Semi-stochastic gradient descent methods. *arXiv:1312.1666*, 2013.

[20] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.

[21] Y. Li and S. Osher. Coordinate descent optimization for âĎŞ 1 minimization with application to compressed sensing; a greedy algorithm. *Inverse Problems and Imaging*, 3(3):487–503, 2009.

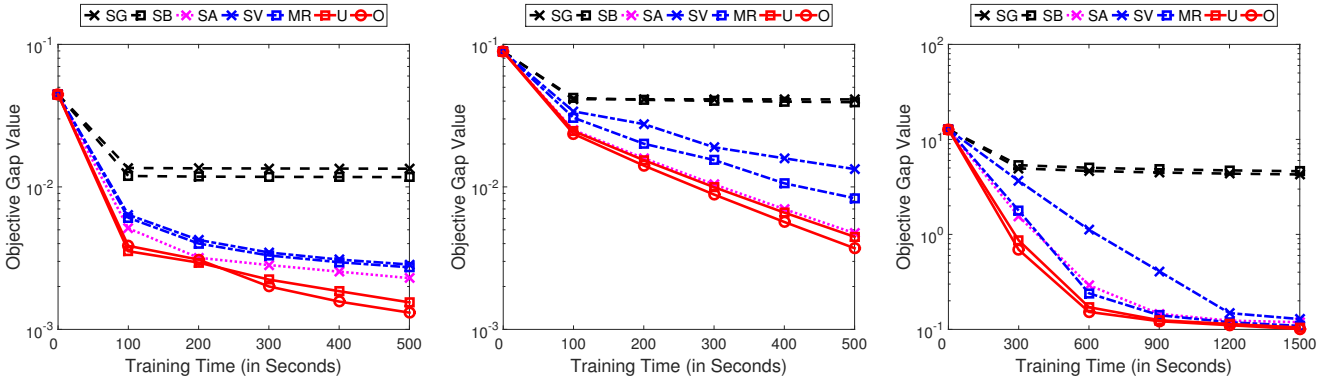
[22] M. Lichman. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2013.

[23] Q. Lin, Z. Lu, and L. Xiao. An accelerated proximal



(a) Classification on *COVTYPE* (b) Classification on *RCV1* (c) Regression on *E2006-TFIDF*

Figure 3: Convergence comparison of algorithms for the same number of entire data passes for classification and regression on three data sets. In general, ASBCD with optimal sampling (O) converges fastest to the optimum for the same number of entire data passes.



(a) Classification on *COVTYPE* (b) Classification on *RCV1* (c) Regression on *E2006-TFIDF*

Figure 4: Convergence comparison of algorithms for the same training time for classification and regression on three data sets. In general, ASBCD with optimal sampling (O) converges fastest to the optimum for the same training time.

coordinate gradient method and its application to regularized empirical risk minimization. *arXiv preprint arXiv:1407.1296*, 2014.

[24] P.-L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979.

[25] H. Liu, M. Palatucci, and J. Zhang. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 649–656. ACM, 2009.

[26] Z. Lu and L. Xiao. On the complexity analysis of randomized block-coordinate descent methods. *Mathematical Programming*, 152(1-2):615–642, 2015.

[27] J. Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *arXiv:1402.4419*, 2014.

[28] R. Mazumder, J. H. Friedman, and T. Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 2012.

[29] D. Needell, R. Ward, and N. Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2014.

[30] Y. Nesterov. *Introductory lectures on convex optimization: A Basic Course*. Springer Science & Business Media, 2004.

[31] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[32] Y. Nesterov et al. *Gradient methods for minimizing composite objective function*. Technical report, Center for Operations Research and Econometrics, 2007.

[33] Z. Qu and P. Richtárik. Coordinate descent with arbitrary sampling i: Algorithms and complexity. *arXiv preprint arXiv:1412.8060*, 2014.

[34] Z. Qu, P. Richtárik, and T. Zhang. Randomized dual coordinate ascent with arbitrary sampling. *arXiv preprint arXiv:1411.5873*, 2014.

[35] S. Reddi, A. Hefny, C. Downey, A. Dubey, and S. Sra. Large-scale randomized-coordinate descent methods with non-separable linear constraints. *arXiv preprint arXiv:1409.2617*, 2014.

[36] P. Richtárik and M. Takáč. Iteration complexity of

randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

[37] M. Schmidt, R. Babanezhad, M. O. Ahmed, A. Defazio, A. Clifton, and A. Sarkar. Non-uniform stochastic average gradient method for training conditional random fields. *arXiv preprint arXiv:1504.04406*, 2015.

[38] M. Schmidt, N. L. Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *arXiv:1309.2388*, 2013.

[39] S. Shalev-Shwartz and A. Tewari. Stochastic methods for l 1-regularized loss minimization. *The Journal of Machine Learning Research*, 12:1865–1892, 2011.

[40] S. Shalev-Shwartz and T. Zhang. Proximal stochastic dual coordinate ascent. *arXiv:1211.2717*, 2012.

[41] S. Shalev-Shwartz and T. Zhang. Accelerated mini-batch stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems*, pages 378–385, 2013.

[42] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss. *The Journal of Machine Learning Research*, 14(1):567–599, 2013.

[43] S. Sra, S. Nowozin, and S. J. Wright. *Optimization for machine learning*. Mit Press, 2012.

[44] J. Stamper, A. Niculescu-Mizil, S. Ritter, G. Gordon, and K. Koedinger. Bridge to algebra dataset from kdd cup 2010 educational data mining challenge. http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp, 2010.

[45] T. Strohmer and R. Vershynin. A randomized kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(2):262–278, 2009.

[46] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[47] R. Tibshirani, J. Bien, J. Friedman, T. Hastie, N. Simon, J. Taylor, and R. J. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2):245–266, 2012.

[48] H. Wang and A. Banerjee. Randomized block coordinate descent for online and stochastic optimization. *arXiv preprint arXiv:1407.0107*, 2014.

[49] T. T. Wu and K. Lange. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, pages 224–244, 2008.

[50] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

[51] Y. Xu and W. Yin. Block stochastic gradient iteration for convex and nonconvex optimization. *SIAM Journal on Optimization*, 25(3):1686–1716, 2015.

[52] A. Zhang, A. Goyal, R. Baeza-Yates, Y. Chang, J. Han, C. A. Gunter, and H. Deng. Towards mobile query auto-completion: An efficient mobile application-aware approach. In *Proceedings of the 25th International Conference on World Wide Web*, pages 579–590, 2016.

[53] P. Zhao and T. Zhang. Stochastic optimization with importance sampling. *arXiv preprint arXiv:1401.2753*, 2014.

[54] T. Zhao, M. Yu, Y. Wang, R. Arora, and H. Liu. Accelerated mini-batch randomized block coordinate descent method. In *Advances in Neural Information Processing Systems*, pages 3329–3337, 2014.

APPENDIX

A. PROOF OF THE MAIN THEORY

We provide the proof for the main theory delivered in Section 3. Note that all the expectations are taken conditional on $\mathbf{w}^{(t-1)}$ and each $\boldsymbol{\phi}_i^{(t-1)}$ unless otherwise stated. For brevity, we define

$$\mathbf{g}_i = \frac{1}{np_i}\nabla f_i(\boldsymbol{\phi}_i^{(t)}) - \frac{1}{np_i}\nabla f_i(\boldsymbol{\phi}_i^{(t-1)}) + \frac{1}{n}\sum_{k=1}^{n}\nabla f_k(\boldsymbol{\phi}_k^{(t-1)}). \tag{A.1}$$

Let us introduce several important lemmas. To begin with, since Algorithm 1 leverages randomized coordinate blocks, the following lemma is needed for taking the expectation of the squared gap between the iterate $\mathbf{w}^{(t)}$ and the optimal solution $\mathbf{w}^*$ in (1.1) with respect to the coordinate block index j.

LEMMA A.1. *Suppose that Assumption 3.3 holds. Let j be a coordinate block index. With $\mathbf{g}_i$ defined in* (A.1) *and $\mathbf{w}^*$ defined in* (1.1)*, based on Algorithm 1 we have*

$$\mathbb{E}_j[\|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2] \leq \frac{1}{m}\big[(m-1)\|\mathbf{w}^{(t-1)} - \mathbf{w}^*\|^2 + \|\mathbf{w}^{(t-1)} - \eta\mathbf{g}_i - \mathbf{w}^* + \eta\nabla F(\mathbf{w}^*)\|^2\big].$$

Lemma A.1 takes the expectation of the squared gap between the iterate $\mathbf{w}^{(t)}$ and the optimal solution $\mathbf{w}^*$ in (1.1) with respect to the randomized coordinate block index. The obtained upper bound does not have a randomized coordinate block index or the proximal operator. Block separability and non-expansiveness of the proximal operator are both exploited in deriving the upper bound. This upper bound is used for deriving a linear rate of convergence for Algorithm 1.

LEMMA A.2. *Based on Algorithm 1 and as defined in* (A.1)*, we have $\mathbb{E}_i[\mathbf{g}_i] = \nabla F(\mathbf{w}^{(t-1)})$.*

Lemma A.2 guarantees that $\mathbf{g}_i$ is an unbiased gradient estimator of $F(\mathbf{w})$. The proof is strictly based on the definition of $\mathbf{g}_i$ in (A.1).

LEMMA A.3. *With $\mathbf{g}_i$ defined in* (A.1) *and $\mathbf{w}^*$ defined in* (1.1)*, based on Algorithm 1 and for all $\zeta > 0$ we have*

$$\begin{aligned}\mathbb{E}_i\big[\,\|\mathbf{g}_i - \nabla F(\mathbf{w}^*)\|^2\,\big] \leq (1+\zeta)\mathbb{E}_i\bigg[\Big\|\frac{1}{np_i}\nabla f_i(\mathbf{w}^{(t-1)}) \\ - \frac{1}{np_i}\nabla f_i(\mathbf{w}^*)\Big\|^2\bigg] - \zeta\|\nabla F(\mathbf{w}^{(t-1)}) - \nabla F(\mathbf{w}^*)\|^2 \\ + (1+\zeta^{-1})\mathbb{E}_i\bigg[\Big\|\frac{1}{np_i}\nabla f_i(\boldsymbol{\phi}_i^{(t-1)}) - \frac{1}{np_i}\nabla f_i(\mathbf{w}^*)\Big\|^2\bigg].\end{aligned}$$

Lemma A.3 makes use of the property that $\mathbb{E}[\|\mathbf{x}\|^2] = \mathbb{E}[\|\mathbf{x} - \mathbb{E}[\mathbf{x}]\|^2] + \|\mathbb{E}[\mathbf{x}]\|^2$ for all $\mathbf{x}$ and the property that $\|\mathbf{x}+\mathbf{y}\|^2 \leq (1+\zeta)\|\mathbf{x}\|^2 + (1+\zeta^{-1})\|\mathbf{y}\|^2$ for all $\mathbf{x}, \mathbf{y}$, and $\zeta > 0$.

LEMMA A.4. *Let f be strongly convex with the convexity parameter μ and its gradient be Lipschitz continuous with the constant L. For all $\mathbf{x}$ and $\mathbf{y}$, it holds that*

$$\begin{aligned}\langle\nabla f(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle \leq f(\mathbf{x}) - f(\mathbf{y}) - \frac{1}{2(L-\mu)}\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2 \\ - \frac{\mu}{L-\mu}\langle\nabla f(\mathbf{x}) - \nabla f(\mathbf{y}), \mathbf{y} - \mathbf{x}\rangle - \frac{L\mu}{2(L-\mu)}\|\mathbf{y} - \mathbf{x}\|^2.\end{aligned}$$

Lemma A.4 leverages properties of strongly convex functions with Lipschitz continuous gradient.

LEMMA A.5. *Algorithm 1 implies that*

$$\mathbb{E}_i\Big[\frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}f_i(\boldsymbol{\phi}_i^{(t)})\Big] = \frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{n}f_i(\mathbf{w}^{(t-1)}) + \frac{1}{n}\sum_{i=1}^{n}\frac{(1-p_i)L_i}{np_i}f_i(\boldsymbol{\phi}_i^{(t-1)}).$$

Lemma A.5 is obtained according to the non-uniform sampling of component functions in Algorithm 1.

REMARK A.6. *Similar to Lemma A.5, we have*

$$\begin{aligned}\mathbb{E}_i\Big[\frac{1}{n}\sum_{i=1}^{n}\Big\langle\frac{L_i}{np_i}\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t)}-\mathbf{w}^*\Big\rangle\Big] &= \frac{1}{n}\sum_{i=1}^{n}\Big\langle\frac{L_i}{n}\nabla f_i(\mathbf{w}^*),\mathbf{w}^{(t-1)}-\mathbf{w}^*\Big\rangle \\ &\quad+\frac{1}{n}\sum_{i=1}^{n}\Big\langle\frac{(1-p_i)L_i}{np_i}\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t-1)}-\mathbf{w}^*\Big\rangle. \end{aligned} \tag{A.2}$$

Now we develop the main theorem of bounding the rate of convergence for Algorithm 1.

PROOF OF THEOREM 3.4. By applying Lemma A.1, A.2, and Lemma A.3,

$$\begin{aligned}&\mathbb{E}_{i,j}\big[\|\mathbf{w}^{(t)}-\mathbf{w}^*\|^2\big] \\ &\le\frac{1}{m}\Bigg[m\|\mathbf{w}^{(t-1)}-\mathbf{w}^*\|^2+2\eta\langle\nabla F(\mathbf{w}^*),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle \\ &\quad-2\eta\langle\nabla F(\mathbf{w}^{(t-1)}),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle \\ &\quad+\eta^2(1+\zeta)\mathbb{E}_i\Big[\Big\|\frac{1}{np_i}\nabla f_i(\mathbf{w}^{(t-1)})-\frac{1}{np_i}\nabla f_i(\mathbf{w}^*)\Big\|^2\Big] \\ &\quad+\eta^2(1+\zeta^{-1})\mathbb{E}_i\Big[\Big\|\frac{1}{np_i}\nabla f_i(\boldsymbol{\phi}_i^{(t-1)})-\frac{1}{np_i}\nabla f_i(\mathbf{w}^*)\Big\|^2\Big] \\ &\quad-\eta^2\zeta\|\nabla F(\mathbf{w}^{(t-1)})-\nabla F(\mathbf{w}^*)\|^2\Bigg]. \end{aligned} \tag{A.3}$$

Substituting $\mathbf{x},\mathbf{y}$, and f with $\mathbf{w}^*,\mathbf{w}^{(t-1)}$, and f_i in Lemma A.4, and taking average on both sides of the inequality in Lemma A.4, we obtain

$$\begin{aligned}&-2\eta\langle\nabla F(\mathbf{w}^{(t-1)}),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle \\ &\le\frac{2\eta}{n}\sum_{i=1}^{n}\frac{L_i-\mu}{L_i}[f_i(\mathbf{w}^*)-f_i(\mathbf{w}^{(t-1)})] \\ &\quad-\frac{\eta}{n}\sum_{i=1}^{n}\frac{1}{L_i}\|\nabla f_i(\mathbf{w}^*)-\nabla f_i(\mathbf{w}^{(t-1)})\|^2 \\ &\quad-\frac{2\eta\mu}{n}\sum_{i=1}^{n}\frac{1}{L_i}\langle\nabla f_i(\mathbf{w}^*),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle-\eta\mu\|\mathbf{w}^*-\mathbf{w}^{(t-1)}\|^2. \end{aligned} \tag{A.4}$$

Recall the property of any function f that is convex and has a Lipschitz continuous gradient with the constant L: $f(\mathbf{y})\ge f(\mathbf{x})+\langle\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle+\|\nabla f(\mathbf{x})-\nabla f(\mathbf{y})\|^2/(2L)$ for all $\mathbf{x}$ and $\mathbf{y}$ [30] (Theorem 2.1.5). Taking average on both sides, we have

$$\begin{aligned}&\mathbb{E}_i\Big[\Big\|\frac{1}{np_i}\nabla f_i(\boldsymbol{\phi}_i^{(t-1)})-\frac{1}{np_i}\nabla f_i(\mathbf{w}^*)\Big\|^2\Big] \\ &\le\frac{2}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}\big[f_i(\boldsymbol{\phi}_i^{(t-1)})-f_i(\mathbf{w}^*)-\langle\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t-1)}-\mathbf{w}^*\rangle\big] \end{aligned} \tag{A.5}$$

after substituting $\mathbf{y},\mathbf{x}$, and f with $\boldsymbol{\phi}_i^{(t-1)},\mathbf{w}^*$, and f_i while rearranging terms.

Before further proceeding with the proof, we define

$$H^{(t)}=\frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}\big[f_i(\boldsymbol{\phi}_i^{(t)})-f_i(\mathbf{w}^*)-\langle\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t)}-\mathbf{w}^*\rangle\big]+\kappa\|\mathbf{w}^{(t)}-\mathbf{w}^*\|^2. \tag{A.6}$$

Following the definition in (A.6), for all $\alpha>0$,

$$\begin{aligned}&\mathbb{E}_{i,j}[H^{(t)}]-\alpha H^{(t-1)} \\ &=\mathbb{E}_{i,j}\Big[\frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}f_i(\boldsymbol{\phi}_i^{(t)})\Big]-\frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}f_i(\mathbf{w}^*)-\mathbb{E}_{i,j}\Big[\frac{1}{n}\sum_{i=1}^{n}\frac{L_i}{np_i}\cdot \\ &\quad\langle\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t)}-\mathbf{w}^*\rangle\Big]+\mathbb{E}_{i,j}\big[\kappa\|\mathbf{w}^{(t)}-\mathbf{w}^*\|^2\big]-\alpha H^{(t-1)}. \end{aligned}$$

Recall the property of any strongly convex function f with the convexity parameter μ that $f(\mathbf{y})\le f(\mathbf{x})+\langle\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle+\|\nabla f(\mathbf{x})-\nabla f(\mathbf{y})\|^2/(2\mu)$ for all $\mathbf{x}$ and $\mathbf{y}$ [30] (Theorem 2.1.10). We can obtain $-\|\nabla f_i(\mathbf{w}^{(t-1)})-\nabla f_i(\mathbf{w}^*)\|^2\le -2\mu\big[f_i(\mathbf{w}^{(t-1)})-f_i(\mathbf{w}^*)-\langle\nabla f_i(\mathbf{w}^*),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle\big]$.

Combining (A.3) with a positive constant κ, (A.4), and (A.5), after simplifying terms, by Lemma A.5 and (A.2), with defining $L_M=\max_i L_i$ and $p_I=\min_i p_i$ we have

$$\mathbb{E}_{i,j}[H^{(t)}]-\alpha H^{(t-1)}\le\sum_{k=1}^{4}c_kT_k, \tag{A.7}$$

where the four constant factors are

$$\begin{aligned}c_1&=\frac{\kappa\eta}{mn}\Big(\frac{\eta(1+\zeta)}{np_I}-\frac{1}{L_M}\Big),\\ c_2&=\frac{1}{n}\Big(\frac{L_M}{n}-\frac{2\kappa\eta(L_M-\mu)}{L_Mm}-\frac{2\beta\kappa\eta^2\mu}{m}\Big),\\ c_3&=\kappa\Big(1-\frac{\eta\mu}{m}-\alpha\Big),\\ c_4&=\frac{L_M}{n^2}\Big(\frac{2\kappa\eta^2(1+\zeta^{-1})}{mp_I}+\frac{1-\alpha}{p_I}-1\Big),\end{aligned}$$

and the four corresponding terms are

$$\begin{aligned}T_1&=\sum_{i=1}^{n}\|\nabla f_i(\mathbf{w}^{(t-1)})-\nabla f_i(\mathbf{w}^*)\|^2,\\ T_2&=\sum_{i=1}^{n}\big[f_i(\mathbf{w}^{(t-1)})-f_i(\mathbf{w}^*)-\langle\nabla f_i(\mathbf{w}^*),\mathbf{w}^{(t-1)}-\mathbf{w}^*\rangle\big],\\ T_3&=\|\mathbf{w}^{(t-1)}-\mathbf{w}^*\|^2,\\ T_4&=\sum_{i=1}^{n}\big[f_i(\boldsymbol{\phi}_i^{(t-1)})-f_i(\mathbf{w}^*)-\langle\nabla f_i(\mathbf{w}^*),\boldsymbol{\phi}_i^{(t-1)}-\mathbf{w}^*\rangle\big].\end{aligned}$$

There are four constant factors associated with four terms on the right-hand side of (A.7). Among the four terms, obviously $T_1\ge0$ and $T_3\ge0$. By the convexity property of f_i, we have $T_2\ge0$ and $T_4\ge0$. We choose $\eta=\max_i np_i/[2(n\mu+L_i)]$. By setting $c_1=0$ with $\zeta=np_I/(L_M\eta)-1>0$, $c_2=0$ with $\kappa=L_M^2m/\big[2n\eta(L_M-\mu+L_M\eta\mu\zeta)\big]>0$, and $c_3=0$ with $0<\alpha=1-\eta\mu/m<1$, it can be verified that $c_4\le0$.

With the aforementioned constant factor setting, $\mathbb{E}_{i,j}[H^{(t)}]-\alpha H^{(t-1)}\le0$, where the expectation is conditional on information from the previous iteration $t-1$. Taking expectation with this previous iteration gives $\mathbb{E}_{i,j}[H^{(t)}]\le\alpha\mathbb{E}_{i,j}[H^{(t-1)}]$. By chaining over t iteratively, $\mathbb{E}_{i,j}[H^{(t)}]\le\alpha^tH^{(0)}$. Since the sum of the first three terms in (A.6) is non-negative by the convexity of F, we have $\kappa\|\mathbf{w}^{(t)}-\mathbf{w}^*\|^2\le H^{(t)}$. Together with the aforementioned results by chaining over t, the proof is complete. □