Abstract

Recently, the online optimization attracts much attention in the big data science since it is an effective optimization tool to deal with data in real time. It is an iterative method to solve optimization problems whose objective function f is the sum of numerous functions f^t $(t = 1, \dots, T)$. It exploits information of a single function f^t at each iteration, and hence its computational time of each iteration is much fewer than that of the batch optimization, such as the steepest descent method.

In this paper, we consider the online convex optimization problem, where each function f^t is convex. For the problem, several gradient descent methods have been proposed. However, they still take much time at each iteration when the number of decision variables is huge. To overcome the difficulty, we propose a randomized block-coordinate descent method, which is an extension of Nesterov's random coordinate descent method for the batch convex optimization. We show that the expected value of the regret of the proposed method is $O(\sqrt{T})$. In some sense, this result includes the convergence results for the greedy projection method by Zinkevich and the random coordinate descent method by Nesterov as special cases. We also report some numerical results for problems of the online learning with real data.

Contents

1 Introduction

The big data science has attracted much attention since the evolving ICT induces us to pile huge data. Statistics and machine learning are key tools in the big data science. They sometimes need to solve huge optimization problems whose size is proportional to the amount of data. Moreover, some applications in the big data science require to get an appropriate solution of the optimization problem in real time.

There are two frameworks to solve optimization problems with big data. They are the batch optimization and the online optimization. The batch optimization solves the problems using all of the information, that is, all data. The well known batch optimization methods are the gradient descent method and the Newton method. On the other hand, the online optimization, which is an iterative method, gets a piece of data one by one and uses only a few data at each iteration. It is suitable for the applications mentioned above, because it has the following advantages.

- It uses lower memory since it needs not to store all of the data.
- Since it updates decision variables every time it gets a piece of data, it responds quickly and is suitable to treat data in real time.
- Even if a property of upcoming data are varying, it works flexibly.

In this paper, we consider the following problem.

$$\min \quad \sum_{t=1}^{T} F^t(x) \equiv \sum_{t=1}^{T} \Big\{ f^t(x) + \sum_{i=1}^{n} r_i(x_i) \Big\}, \tag{1}$$

where each function $f^t : \mathbb{R}^n \to \mathbb{R}\,(t = 1, \dots, T)$ is a differentiable convex function and each function $r_i : \mathbb{R} \to \mathbb{R}\,(i = 1, \dots, n)$ is a continuous convex function. The function f^t represents a certain statistic function of the tth data. Note that r_i is not necessarily differentiable. The problem (1) is convex, and it appears in statistics, signal processing, neural network, and so on. A typical example of the second term $\sum_{i=1}^{n} r_i(x_i)$ of F^t is the l_1 regularization and the indicator function of the box set. When r_i is the indicator function of some interval for each i, the problem (1) is essentially the convex problem with the box constraint. On the other hand, the l_1 regularization is frequently used in machine learning and signal processing since it induces sparse solutions ignoring small noises.

For solving the problem (1), the gradient based methods are suitable for applications where a reasonable solution is required as soon as a piece of data is obtained. For the problem (1) with the indicator function r_i, the greedy projection method [9] based on a gradient ∇f^t has been proposed. For more general r_i, the proximal gradient method is very popular [4, 8]. The proximal gradient method is sometimes called the forward-backward splitting [4]. When $r_i(x_i)$ is l_1 regularization term, that is $r_i(x_i) = |x_i|$, the method updates the next iteration as

$$\begin{aligned} x_i^{t+1} &= S_{\lambda\eta_t}(x_i^t - \eta_t \nabla_i f^t(x^t)) \quad (t = 1, \dots, T) \quad (i = 1, \dots, n) \qquad (2)\\ S_\alpha(z) &= [|z| - \alpha]_+ \operatorname{sgn}(z)\,, \end{aligned}$$

where $\eta_t > 0$ is called a stepsize or a learning rate, $[a]_+ = \max\{0, a\}$, and $\mathrm{sgn}(a)$ is defined as follows.

$$\mathrm{sgn}(a) = \begin{cases} 1 & (a > 0) \\ 0 & (a = 0) \\ -1 & (a < 0). \end{cases}$$

It is known that the proximal gradient method has global convergence if $\eta_t = O(\frac{1}{\sqrt{t}})$. However, when the number of the decision variables n is large, the method would take much time at every iteration since it updates all components of x^t at each iteration.

In this paper, we extend the idea of the block-coordinate descent method to the online convex optimization problem. The block-coordinate descent method chooses a set of coordinates (that is to say a block) I_t at each iteration t, and then, updates only the components which belong to the set I_t. It does not update the other components. Therefore, it can reduce the calculation time of each iteration, and it is effective to solve problems which have a lot of decision variables. The typical rules to choose the block I_t are the cyclic rule (Gauss-Seidel rule) [6] or the randomized rule [5]. Nesterov proposed the interesting randomized rule where the probability of choosing the block I_t is related with Lipschitz constants of $\nabla_{I_t} f$ [5]. He shows that the method with the randomized rule can find a solution quickly in comparison with that with the simple uniform distribution. However, the method is developed for the batch optimization, hence it can not be applied directly to the problems (1) with huge T.

In this paper, we propose a randomized block-coordinate descent method for online convex optimization problem applying the idea of Nesterov [5]. At each iteration t, the proposed method uses the information of a single function f^t only, and also updates only components corresponding to the chosen block I_t. It is particularly effective in problems where almost all of the components of $\nabla f^t(x^t)$ are 0. Indeed, if we choose a set $I_t = \{i \,|\, \nabla_i f^t(x^t) \neq 0\}$, we may update only critical components in a short time. On the other hand, the full proximal gradient method (2) updates all of the components x_i^t even though $\nabla_i f^t(x^t) = 0$. As such an example, we present the Multi-Domain Sentiment Datase [2] in Section 4. This dataset has the huge number of the decision variables ($n = 332440$), while a gradient of each function ∇f^t has at most 5412 non-zero elements.

We show a kind global convergence property of the proposed method under the same conditions in [9] and [5]. When each coordinate is chosen with probability 1, that is to say when the proposed method is reduced to the greedy projection method, the result is the same as the regret analysis of [9]. This shows that the result is a natural extension of [9]. In addition, we discuss how to set probabilities to get rapid convergence as Nesterov [5] from the convergence theory.

The paper is organized as follows. In Section 2, we introduce the important concepts of the coordinate descent method, the online convex optimization problem, and the proximal gradient method. Then, we propose the randomized block-coordinate descent method for online convex optimization and analyze its convergence in Section 3. In Section 4, we report some results of numerical experiments with the some datasets, and compare the proposed method with the existing method. Finally, we give concluding remarks in Section 5.

Throughout this paper, we use the following notations. For a vector $x \in \mathbb{R}^n$, x' denotes the transposed vector of x. For a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla_i f(x)$ denotes the ith component of $\nabla f(x)$.

In a similar way, for a block $I \subseteq \{1,\dots,n\}$, $\nabla_I f(x)$ denotes a $|I|$-dimensional vector of components of $\nabla_i f(x)$, $i \in I$. For a vector $x \in \mathbb{R}^n$, $\|x\|$ and $|x|_1$ are the l_2-norm and l_1-norm of x, respectively. In addition, for a vector $x \in \mathbb{R}^n$ and a positive definite symmetric matrix $G \in \mathbb{R}^{n\times n}$, the G-norm is defined as $\|x\|_G = (\sqrt{x^T G x})$.

2 Preliminaries

In this section, we introduce the block-coordinate descent method proposed by Nesterov. We also describe basics of the online optimization problem and the proximal gradient method.

2.1 Coordinate Descent Method

Firstly, we introduce Nesterov's random coordinate descent method [5]. For simplicity, we consider the following unconstrained minimization problem.

$$\min \quad f(x)\,, \tag{3}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a differentiable convex function.

We first define blocks $J_k \subseteq \{1,\dots,n\}$ $(k = 1,\dots,m)$, and let $J = \{J_1,\dots,J_m\}$ as the set of the blocks. Then, for given $x^t \in \mathbb{R}^n$, the block-coordinate descent method chooses a block I_t from the set J, and generates x^{t+1} as

$$x_i^{t+1} = \begin{cases} x_i^t - \eta_t \frac{\partial f(x)}{\partial x_i} & (i \in I_t) \\ 0 & (i \notin I_t)\,, \end{cases} \tag{4}$$

where η_t is a positive stepsize.

Nesterov's random coordinate descent method [5] assumes that J is a decomposition of $\{1,\dots,n\}$, that is, J_k $(k = 1,\dots,m)$ satisfy

$$\begin{aligned} &\bigcup_{k=1}^{m} J_k = \{1,\dots,n\}\,, \\ &J_i \cap J_j = \phi \quad \text{for all} \quad i,j \in \{1,\dots,m\}\,,\, i \neq j\,, \end{aligned}$$

where ϕ denotes the empty set. Furthermore, the method assumes that the gradient of the function f is block-wise Lipschitz continuous, that is, there exist positive constants L_k $(k = 1,\dots,m)$ such that

$$\|\nabla_{J_k} f(x) - \nabla_{J_k} f(y)\| \le L_k \|x - y\|\,, \quad \forall x, y \in \mathbb{R}^n \quad \text{such that} \quad x_i = y_i \quad \text{for all} \quad i \notin J_k. \tag{5}$$

Then, the method chooses a block I_t from J randomly at each iteration t. The probability that $I_t = J_k$ is set to

$$q_k = \frac{L_k^\alpha}{\sum_{j=1}^m L_j^\alpha} \quad (k = 1,\dots,m).$$

Then, the next iterate x^{t+1} is updated by (4) with $\eta_t = \frac{1}{L_k}$ where k denotes the index of J_k chosen at the tth iteration. Nesterov shows that the expected value of $f(x^t)$ converges to optimal value as follows.

Theorem 1. *[5, Theorem 1] Suppose that there exists a positive number X such that $\|x^t - x^*\|_\Lambda \leq X\,(t = 1, 2, \dots)$, where $\Lambda \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $\Lambda_{ii} = \frac{1}{q_i}\,(i \in J_k)$. Then, we have*

$$E[f(x^t)] - f^* \leq \frac{2}{t+4} X^2,$$

where f^* is an optimal value of f. $\square$

In this paper, we will apply the random coordinate descent method to the online convex optimization problem (1).

2.2 Online Convex Optimization Problem

We first show a simple online algorithm for explanation.

Algorithm 1 Basic online algorithm

```
Input: Set the initial point $x^1 \in \mathbb{R}^n$ and the initial stepsize $\eta_1 > 0$
  for $k = 1, 2, \dots$ do
    1. Choose $t_k \in \{1, \dots, T\}$
    2. Update $x^{k+1} = x^k - \eta_t \nabla f^{t_k}(x^k)$.
    3. Update a stepsize $\eta_{t+1}$.
  end for
```

Note that k is the iteration number of the algorithm, and the algorithm generates x^{k+1} by using the function f^{t_k} only. This basic framework assumes that all functions f^t and the number T of functions in advance. However, when we apply the online algorithm in real time, we may not know T. Moreover, T might be infinity for some applications, such as the stochastic optimization. Therefore, in the following of this paper, we suppose that T can vary, and the index of the functions t_k is same as the number of steps k, and we use the same notation t for the step and the index. It means that a new function f^t is obtained at each iteration t, and the number T of functions can get larger and larger.

When T tends to vary, the criterion for batch optimization problems can not be applied for online algorithms. For this reason, we often use a regret of $f^t(x^t)$ compared to $f^t(x^*)$ for some point x^* as a criterion of online algorithms.

The regret $R(T)$ at the T step is defined as the difference of function values between the sequence $\{x^1, x^2, \dots, x^T\}$ and a fixed optimal point $x^* \in \operatorname{argmin} \sum_{t=1}^{T} F^t(x)$, that is,

$$R(T) = \sum_{t=1}^{T} \Big\{ F^t(x^t) - F^t(x^*) \Big\}. \tag{6}$$

In general, $R(T)$ tends to be ∞ as $T \to \infty$. The growth rate of $R(T)$ is useful for evaluating the approximate solutions $\{x^t\}$ of an online algorithm. In particular, the growth rate is required to be less than $O(T)$, which implies that $\frac{R(T)}{T} \to 0$ as $T \to \infty$. Since $\frac{R(T)}{T}$ denotes the average of $F^t(x^t) - F^t(x^*)$, this means the average of $\{F^t(x^t)\}$ converges to $F^t(x^*)$ in some sense.

Zinkevich [9] presented the regret of the greedy projection method. For simplicity, let each function r_i be the indicator function of a box set $C_i = \{z \in \mathbb{R} \,|\, l_i \le z \le u_i\}$. The method updates

$$x^{t+1} = P_C(x^t - \eta_t \nabla f^t(x^t))\,,$$

where $P_C(\boldsymbol{\cdot})$ denotes a projection onto the box set $C = C_1 \times C_2 \times \ldots, C_n$, and $\eta_t > 0$ is a stepsize. The regret of the method is given in the following theorem.

Theorem 2. *[9, Theorem 1] Let $\eta_t = \frac{c}{\sqrt{t}}$ with a constant $c > 0$. Then, the regret $R(T)$ of the greedy projection method satisfies*

$$R(T) \le \frac{\sqrt{T}}{2c} X^2 + \frac{c(2\sqrt{T}-1)}{2} G^2\,, \tag{7}$$

where $X = \max_{x,y \in C} \|x - y\|,\ \ G = \max_{x \in C, t \in 1,\ldots,T} \|\nabla f^t(x)\|$. □

Since the order of the regret of the greedy projection method is $O(\sqrt{T})$ from (7), we have $\frac{R(T)}{T} = O(\frac{1}{\sqrt{T}})$.

2.3 Proximal Gradient Method

In this section, we consider the following problem.

$$\min \quad f(x) + \sum_{i=1}^{n} r_i(x_i)\,, \tag{8}$$

where f is a convex function. When the function $f(x) = \sum_{t=1}^{T} f^t(x)$, the problem (8) is reduced to the problem (1).

The batch type proximal gradient method [1, 7] for the problem (8) updates the variable x^{t+1} by solving the following subproblem.

$$x^{t+1} \in \operatorname*{argmin}_{x} \Big\{ f(x^t) + \langle x - x^t, \nabla f(x^t) \rangle + \frac{1}{\eta_t} d(x, x^t) + \sum_{i=1}^{n} r_i(x_i) \Big\}\,, \tag{9}$$

where $\eta_t > 0$ denotes a stepsize, and the function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is the Bregman function $d(y,z) = \psi(y) - \psi(z) - \langle \nabla\psi(z), y - z \rangle$ with a differentiable strongly convex function $\psi : \mathbb{R}^n \to \mathbb{R}$. In the following, we set $d(y,z) = \frac{1}{2}\|y - z\|^2$ for simplicity.

The subproblem (9) can be decomposed into the following n subproblem since $\sum_{i=1}^{n} r_i(x_i)$ is separable and $f(x^t)$ is constant.

$$x_i^{t+1} \in \operatorname*{argmin}_{x_i} \Big\{ \langle x_i - x_i^t, \nabla_i f(x^t) \rangle + \frac{1}{2\eta_t}(x_i - x_i^t)^2 + r_i(x_i) \Big\} \quad (i = 1, \ldots, n). \tag{10}$$

When we use the l_1 regularization function, that is, $r_i(x_i) = \lambda |x_i|$ with a positive constant λ, the update rule (10) can be written as

$$x_i^{t+1} = S_{\lambda\eta_t}(x_i^t - \eta_t \nabla_i f(x^t))\,,$$

where $S_\alpha : \mathbb{R} \to \mathbb{R}$ is defined by

$$S_\alpha(z) = [|z| - \alpha]_+ \operatorname{sgn}(z).$$

The following property is related to the subproblem (10), and it is important for convergence analysis of the proximal gradient method [3]. This property is known as 3-Point Property, and we will use it for regret analysis of the proposed method in Section 3.2.

Lemma 1. *[3, Lemma 3.2] For any proper lsc convex function $\Psi : \mathbb{R} \to \mathbb{R}$, any $a \in \mathbb{R}$, positive constant δ and $a^+ = \operatorname*{argmin}_c \{\Psi(c) + \delta(a-c)^2\}$, the following inequality holds.*

$$\Psi(b) \geq \Psi(a^+) + \delta(a^+ - a)^2 + \delta(b - a^+)^2 - \delta(b-a)^2 \quad \forall b \in \mathbf{dom}\,\Psi.$$

□

3 Randomized Block-Coordinate Descent Method

In this section, we propose the randomized block-coordinate descent method for the online convex optimization problem (1). It is based on the random coordinate descent method presented in Section 2.1. Furthermore, we give a bound of the regret of the proposed algorithm.

3.1 Algorithm

Let $J_k \subseteq \{1,\dots,n\}\,(k=1,\dots,m)$, and let $J = \{J_1,\dots,J_m\}$. Note that the each block J_k can have intersection of other blocks. We apply the block coordinate descent method to the problem (1). For given x^t and a coordinate block I_t, the next iterate is updated by

$$x_i^{t+1} = \begin{cases} \operatorname*{argmin}_{x_i} \Big\{ \langle x_i - x_i^t, \nabla_i f^t(x^t)\rangle + \frac{1}{2\gamma_t\beta_i}\|x_i - x_i^t\|^2 + r_i(x_i)\Big\} & (i \in I_t) \\ x_i^t & (i \notin I_t)\,, \end{cases}$$

where $\gamma_t\beta_i$ correspond to the stepsize η_t. Note that the parameter γ_t depends on an iteration t, and the parameter β_i depends on a coordinate i, which allows us to have a different stepsize for each coordinate. Now, we describe the proposed method.

Algorithm 2 Randomized Block-Coordinate Descent Method

```
Input: Set coordinate blocks $J_k \subseteq \{1,\dots,n\}\,(k=1,\dots,m)$, probabilities $q_k\,(k=1,\dots,m)$ of choosing
  the block $J_k$, stepsizes $\beta_i\,(i=1,\dots,n)$ and $\gamma_1$, and the initial point $x^1 \in \mathbb{R}^n$.
  for $t = 1,2,\dots$ do
    1. Choose $I_t \in \{J_1,\dots,J_m\}$ with probabilities $q_1,\dots,q_m$.
    2. Update $x^{t+1}$ as follows.
       $$x_i^{t+1} = \begin{cases} \operatorname*{argmin}_{x_i} \Big\{ \langle x_i - x_i^t, \nabla_i f^t(x^t) + \frac{1}{2\gamma_t\beta_i}\|x_i - x_i^t\|^2 + r_i(x_i)\Big\} & (i \in I_t) \\ x_i^t & (i \notin I_t). \end{cases} \quad (11)$$
    3. Update a stepsize $\gamma_{t+1}$.
  end for
```

3.2 Regret Analysis

In this subsection, we give a bound of the regret of Algorithm 2. Since Algorithm 2 uses random elements for updating, the sequence $\{x^t\}$ is stochastic variables. Therefore, we evaluate the expected value of the regret.

Recall that the probability of $I_t = J_k$ at each iteration is q_k. Thus, the probability p_i that the ith coordinate is updated is given by

$$p_i = \sum_{i \in J_k} q_k \quad (i = 1, \dots, n).$$

Now, we require the following assumption on p_i.

Assumption 1. *The probability $p_i > 0$ for $i = 1, \dots, n$.*

If $p_i = 0$, then the ith coordinate is not updated by the algorithm 2. Thus, Assumption 1 is quite natural. We use the following notations for the subsequent discussion. Let B be a diagonal matrix consisting of the stepsizes β_i $(i = 1, \dots, n)$, that is,

$$B = \begin{pmatrix} \beta_1 & & & 0 \\ & \beta_2 & & \\ & & \ddots & \\ 0 & & & \beta_n \end{pmatrix}.$$

Similarly, let D be a diagonal matrix consisting of the probabilities p_i $(i = 1, \dots, n)$, that is,

$$D = \begin{pmatrix} p_1 & & & 0 \\ & p_2 & & \\ & & \ddots & \\ 0 & & & p_n \end{pmatrix}.$$

Note that the both matrix B and D are symmetric positive definite.

We first show the following key inequality from 3-Point Property (Lemma 1).

Lemma 2. *Suppose that Assumption 1 holds. For a given x^t, let x^{t+1} be the point generated by Algorithm 2. Then, for any $x \in \mathbb{R}^n$, we have*

$$\begin{aligned} & r_i(x_i) + \langle x_i - x_i^t, \nabla_i f^t(x^t) \rangle \\ & \geq r_i(x_i^t) + \frac{1}{p_i} E(r_i(x_i^{t+1})) - \frac{1}{p_i} r_i(x_i^t) \\ & \quad + \frac{1}{p_i} \langle E(x_i^{t+1}) - x_i^t, \nabla_i f^t(x^t) \rangle + \frac{1}{2\gamma_t \beta_i p_i} E((x_i^{t+1} - x_i^t)^2) \\ & \quad + \frac{1}{2\gamma_t \beta_i p_i} E((x_i - x_i^{t+1})^2) - \frac{1}{2\gamma_t \beta_i p_i} (x_i^t - x_i)^2 \qquad (i = 1, \dots, n). \end{aligned} \tag{12}$$

Proof. Firstly, note that only x^{t+1} is a stochastic variable in this lemma.

Suppose that the ith coordinate is chosen at the iteration t, and $\bar{x}_i^{t+1}$ is updated by (11), that is

$$\bar{x}_i^{t+1} = \operatorname*{argmin}_{x_i} \Big\{ f^t(x^t) + \lambda r_i(x_i) + \langle x_i - x_i^t, \nabla_i f(x^t) \rangle + \frac{1}{2\gamma_t \beta_i} (x_i - x_i^t)^2 \Big\}.$$

Letting $\Psi(x_i) = r_i(x_i) + \langle x_i - x_i^t, \nabla_i f(x^t)\rangle$ and $\delta = \frac{1}{2\gamma_t \beta_i}$ in Lemma 1, we have

$$\begin{aligned}
&r_i(x_i) + \langle x_i - x_i^t, \nabla_i f^t(x^t)\rangle \\
&\geq r_i(\bar{x}_i^{t+1}) + \langle \bar{x}_i^{t+1} - x_i^t, \nabla_i f(x^{t+1})\rangle \\
&\quad + \frac{1}{2\gamma_t\beta_i}(\bar{x}_i^{t+1} - x_i^t)^2 + \frac{1}{2\gamma_t\beta_i}(x_i - \bar{x}_i^{t+1})^2 - \frac{1}{2\gamma_t\beta_i}(x_i - x_i^t)^2. \qquad (13)
\end{aligned}$$

Since the coordinate i is chosen in accordance with the probability p_i, the expected value of x_i^{t+1} is written as

$$E(x_i^{t+1}) = p_i \bar{x}_i^{t+1} - (1-p_i)x_i^t.$$

Similarly, we have

$$\begin{aligned}
\langle E(x_i^{t+1}) - x_i^t, \nabla_i f^t\rangle &= p_i\langle \bar{x}_i^{t+1} - x_i^t, \nabla_i f^t(x^t)\rangle\,, \\
E(r_i(x_i^{t+1})) &= p_i r_i(\bar{x}_i^{t+1}) + (1-p_i) r_i(x_i^t)\,, \\
E((x_i^{t+1} - x_i)^2) &= p_i(\bar{x}_i^{t+1} - x_i)^2 + (1-p_i)(x_i^t - x_i)^2\,, \\
E((x_i^{t+1} - x_i^t)^2) &= p_i(\bar{x}_i^{t+1} - x_i^t)^2.
\end{aligned}$$

Since $p_i \neq 0$ by Assumption 1, they are rewritten as

$$\begin{aligned}
\langle \bar{x}_i^{t+1} - x_i^t, \nabla_i f^t(x^t)\rangle &= \frac{1}{p_i}\langle E(x_i^{t+1}) - x_i^t, \nabla_i f^t\rangle \\
r_i(\bar{x}_i^{t+1}) &= \frac{1}{p_i}E(r_i(x_i^{t+1}) - \frac{1}{p_i}r_i(x_i^t) + r_i(x_i^t) \\
(\bar{x}_i^{t+1} - x_i)^2 &= \frac{1}{p_i}E((x_i^{t+1} - x_i)^2) - \frac{1}{p_i}(x_i^t - x_i)^2 + (x_i^t - x_i)^2 \\
(\bar{x}_i^{t+1} - x_i^t)^2 &= \frac{1}{p_i}E((x_i^{t+1} - x_i)^2).
\end{aligned}$$

Substituting these equations into the inequality (13), we obtain

$$\begin{aligned}
&r_i(x_i) + \langle x_i - x_i^t, \nabla_i f^t(x^t)\rangle \\
&\geq r_i(x_i^t) + \frac{1}{p_i}E(r_i(x_i^{t+1})) - \frac{1}{p_i}r_i(x_i^t) + \frac{1}{p_i}\langle E(x_i^{t+1}) - x_i^t, \nabla_i f^t(x^t)\rangle \\
&\quad + \frac{1}{2\gamma_t\beta_i p_i}E((x_i^{t+1} - x_i^t)^2) + \frac{1}{2\gamma_t\beta_i p_i}E((x_i - x_i^{t+1})^2) - \frac{1}{2\gamma_t\beta_i p_i}(x_i^t - x_i)^2\,,
\end{aligned}$$

which is the desired inequality. □

Using this lemma, we get an important property of Algorithm 2.

Proposition 1. *Suppose that Assumption 1 holds. For a given x^t, let x^{t+1} be the point generated by Algorithm 2. Then, for any $x \in \mathbb{R}^n$, we have*

$$\begin{aligned}
&F^t(x^t) - F^t(x) \\
&\leq \sum_{i=1}^n \frac{1}{p_i} r_i(x_i^t) - E(\sum_{i=1}^n \frac{1}{p_i} r_i(x_i^{t+1})) \\
&\quad + \frac{1}{2\gamma_t}\|x - x^t\|^2_{D^{-1}B^{-1}} - \frac{1}{2\gamma_t}E(\|x - x^{t+1}\|^2_{D^{-1}B^{-1}}) + \frac{\gamma_t}{2}\|\nabla f^t(x^t)\|^2_{D^{-1}B^{-1}}. \qquad (14)
\end{aligned}$$

Proof. Summing up the inequality (12) in Lemma 2 for i and adding a constant $f^t(x^t)$ to both sides, we have

$$\begin{aligned}
&f^t(x^t) + \sum_{i=1}^n r_i(x_i) + \langle x - x^t, \nabla f^t(x^t)\rangle \\
&\geq f^t(x^t) + \sum_{i=1}^n r_i(x_i^t) + E(\sum_{i=1}^n \frac{1}{p_i} r_i(x_i^{t+1})) - \sum_{i=1}^n \frac{1}{p_i} r_i(x_i^t) + \langle E(x^{t+1}) - x^t, D^{-1}\nabla f^t(x^t)\rangle \\
&\quad + \frac{1}{2\gamma_t} E(\|x^{t+1} - x^t\|^2_{D^{-1}B^{-1}}) + \frac{1}{2\gamma_t} E(\|x - x^{t+1}\|^2_{D^{-1}B^{-1}}) - \frac{1}{2\gamma_t}\|x - x^t\|^2_{D^{-1}B^{-1}}.
\end{aligned} \tag{15}$$

Since the function f^t is convex, we have

$$f^t(x) + \sum_{i=1}^n r_i(x_i) \geq f^t(x^t) + \sum_{i=1}^n r_i(x_i) + \langle x - x^t, \nabla f^t(x^t)\rangle. \tag{16}$$

Moreover, the fifth and sixth terms in the right hand side of (15) are

$$\begin{aligned}
&\langle E(x^{t+1}) - x^t, D^{-1}\nabla f^t(x^t)\rangle + \frac{1}{2\gamma_t} E(\|x^{t+1} - x^t\|^2_{D^{-1}B^{-1}}) \\
&= E\left(\langle x^{t+1} - x^t, D^{-1}\nabla f^t(x^t)\rangle + \frac{1}{2\gamma_t}\|x^{t+1} - x^t\|^2_{D^{-1}B^{-1}}\right) \\
&= E\left(\frac{1}{2}\left\|\frac{1}{\sqrt{\gamma_t}} B^{-\frac{1}{2}}(x^{t+1} - x^t) + \sqrt{\gamma_t} B^{\frac{1}{2}} \nabla f^t(x^t)\right\|^2_{D^{-1}} - \frac{\gamma_t}{2}\|\nabla f^t(x^t)\|^2_{D^{-1}B}\right) \\
&\geq -\frac{\gamma_t}{2}\|\nabla f^t(x^t)\|^2_{D^{-1}B}.
\end{aligned} \tag{17}$$

It follows from (15) - (17) that

$$\begin{aligned}
&f^t(x) + \sum_{i=1}^n r_i(x_i) \\
&\geq f^t(x^t) + \sum_{i=1}^n r_i(x_i) + \langle x - x^t, \nabla f^t(x^t)\rangle \\
&\geq f^t(x^t) + \sum_{i=1}^n r_i(x_i^t) + E(\sum_{i=1}^n \frac{1}{p_i} r_i(x_i^{t+1})) - \sum_{i=1}^n \frac{1}{p_i} r_i(x_i^t) + \langle E(x^{t+1}) - x^t, D^{-1}\nabla f^t(x^t)\rangle \\
&\quad + \frac{1}{2\gamma_t} E(\|x^{t+1} - x^t\|^2_{D^{-1}B^{-1}}) + \frac{1}{2\gamma_t} E(\|x - x^{t+1}\|^2_{D^{-1}B^{-1}}) - \frac{1}{2\gamma_t}\|x - x^t\|^2_{D^{-1}B^{-1}} \\
&\geq f^t(x^t) + \sum_{i=1}^n r_i(x_i^t) + E(\sum_{i=1}^n \frac{1}{p_i} r_i(x_i^{t+1})) - \sum_{i=1}^n \frac{1}{p_i} r_i(x_i^t) \\
&\quad - \frac{\gamma_t}{2}\|\nabla f^t(x^t)\|^2_{D^{-1}B} + \frac{1}{2\gamma_t} E(\|x - x^{t+1}\|^2_{D^{-1}B^{-1}}) - \frac{1}{2\gamma_t}\|x - x^t\|^2_{D^{-1}B^{-1}}.
\end{aligned}$$

Therefore, we get the following inequality.

$$\begin{aligned}
&F^t(x^t) - F^t(x) \\
&\leq \sum_{i=1}^n \frac{1}{p_i} r_i(x_i^t) - E(\sum_{i=1}^n \frac{1}{p_i} r_i(x_i^{t+1})) \\
&\quad + \frac{1}{2\gamma_t}\|x - x^t\|^2_{D^{-1}B^{-1}} - \frac{1}{2\gamma_t} E(\|x - x^{t+1}\|^2_{D^{-1}B^{-1}}) + \frac{\gamma_t}{2}\|\nabla f^t(x^t)\|^2_{D^{-1}B}.
\end{aligned}$$

□

Using this proposition, we directly get a bound of the regret of Algorithm 2.

Theorem 3. *Suppose that Assumption 1 holds. Suppose also that there exist a positive constant G and X such that $\|\nabla f^t(x)\|_{D^{-1}B} \leq G\,(t = 1, \dots, T)$ for any $x \in \mathbb{R}^n$ and $\|x^t - x^*\|_{D^{-1}B^{-1}} \leq X\,(t = 1, \dots, T)$. If $\gamma_t = \frac{c}{\sqrt{t}}\,(c > 0)$, then the expected value of the regret of Algorithm 2 satisfies*

$$E[R(T)] \leq \sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^1) + \frac{\sqrt{T}}{2c} X^2 + \frac{c(2\sqrt{T} - 1)}{2} G^2. \tag{18}$$

Proof. For $t \geq 2$, the sequence $\{x^t\}$ generated by Algorithm 2 depends on the sequence of the stochastic variables $\xi_t = \{I_1, \dots, I_{t-1}\}$. We define $E_{\xi_t}(x^t)$ as the expected value of x^t with the stochastic variables ξ_t. Then, for $t = 1, \dots, T$, we get the following inequality by Proposition 1.

$$\begin{aligned}
&E_{\xi_t}(F^t(x^t)) - F^t(x^*) \\
&\leq E_{\xi_t}\Big(\sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^t)\Big) - E_{\xi_{t+1}}\Big(\sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^{t+1})\Big) \\
&\quad + \frac{1}{2\gamma_t} E_{\xi_t}(\|x^* - x^t\|^2_{D^{-1}B^{-1}}) - \frac{1}{2\gamma_t} E_{\xi_{t+1}}(\|x^* - x^{t+1}\|^2_{D^{-1}B^{-1}}) + \frac{\gamma_t}{2} E_{\xi_t}\|\nabla f^t(x^t)\|^2_{D^{-1}B}.
\end{aligned}$$

Summing the inequalities, we have

$$\begin{aligned}
E[R(T)] &= \sum_{t=1}^{T} \Big\{ E_{\xi_t}(F^t(x^t)) - F^t(x^*) \Big\} \\
&\leq \sum_{t=1}^{T} \Big\{ E_{\xi_t}\Big(\sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^t)\Big) - E_{\xi_{t+1}}\Big(\sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^{t+1})\Big) \\
&\quad + \frac{1}{2\gamma_t} E_{\xi_t}(\|x^* - x^t\|^2_{D^{-1}B^{-1}}) - \frac{1}{2\gamma_t} E_{\xi_{t+1}}(\|x^* - x^{t+1}\|^2_{D^{-1}B^{-1}}) + \frac{\gamma_t}{2} E_{\xi_t}\|\nabla f^t(x^t)\|^2_{D^{-1}B} \Big\} \\
&= \sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^1) - E_{\xi_{T+1}}\Big(\sum_{i=1}^{n} \frac{1}{p_i} (r_i(x_i^{T+1})\Big) \\
&\quad + \frac{1}{2\gamma_1} \|x - x^1\|^2_{D^{-1}B^{-1}} - \frac{1}{2\gamma_T} E_{\xi_{T+1}}(\|x - x^{T+1}\|^2_{D^{-1}B^{-1}}) \\
&\quad + \sum_{t=2}^{T} \Big\{ \Big(\frac{1}{2\gamma_t} - \frac{1}{2\gamma_{t-1}}\Big) E_{\xi_t}(\|x^t - x^*\|^2_{D^{-1}B^{-1}}) \Big\} + \sum_{t=1}^{T} \frac{\gamma_t}{2} E_{\xi_t}(\|\nabla f^t(x^t)\|^2_{D^{-1}B}) \\
&\leq \sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^1) + \frac{1}{2\gamma_T} X^2 + \sum_{t=1}^{T} \frac{\gamma_t}{2} G^2.
\end{aligned}$$

Since $\gamma_T = \frac{c}{\sqrt{T}}$ and $\sum_{t=1}^{T} \gamma_t \leq c(2\sqrt{T} - 1)$, we get

$$E[R(T)] \leq \sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^1) + \frac{\sqrt{T}}{2c} X^2 + \frac{c(2\sqrt{T} - 1)}{2} G^2.$$

□

Theorem 3 shows that the expected value of the regret of Algorithm 2 has $O(\sqrt{T})$ bound under the appropriate condition on the stepsize. Therefore, $\frac{E[R(T)]}{T} = O(\frac{1}{\sqrt{T}})$. Theorem 3 is more general than Theorem 2 of the greedy projection method. The reason is as follows. Set $p_i = 1\,(i = 1, \dots, n)$ and

$\beta_i = 1\,(i = 1,\dots,n)$. Then, Algorithm 2 is reduced to the greedy projection method. Moreover, the matrix D and B are equivalent to the unit matrix I, and hence $\|\cdot\|_{D^{-1}B^{-1}} = \|\cdot\|_{D^{-1}B} = \|\cdot\|$. Then, the formula (18) accords with the formula (7) in Theorem 2.

Theorem 3 induces the relation between the regret bound of the algorithm 2 and confidence level. Using Markov inequality and Theorem 3, we immediately get the next corollary.

Corollary 1. *Suppose that Assumption 1 holds. Suppose also that there exist a positive constant G and X such that $\|\nabla f^t(x)\|_{D^{-1}B} \le G\,(t = 1,\dots,T)$ for any $x \in \mathbb{R}^n$ and $\|x^t - x^*\|_{D^{-1}B^{-1}} \le X\,(t = 1,\dots,T)$. For any confidence level $0 < \rho < 1$, if $\gamma_t = \frac{c}{\sqrt{t}}\,(c > 0)$, the regret of Algorithm 2 satisfies the following inequality at least $1 - \rho$ probability.*

$$R(T) \le \frac{1}{\rho}\Big\{\sum_{i=1}^{n} \frac{1}{p_i} r_i(x_i^1) + \frac{\sqrt{T}}{2c}X^2 + \frac{c(2\sqrt{T}-1)}{2}G^2\Big\}.$$

□

4 Numerical Experiment

In this section, we report some numerical results. We compare the proposed method with the existing proximal gradient method in terms of their regrets with respect to the calculation time and the number of iterate. All computations were carried out on a machine with Intel(R)Core(TM)2Quad 2.83GHz CPU and 4.00 GB memory, and we implement all codes in MATLAB7.6.0(R2008a).

4.1 Experiment with Random Data

We apply the proposed method and the existing proximal gradient method to the least square problem. The test problems are generated as follows. We first generate 1000 data $(a^k, b^k) \in \mathbb{R}^{100\times 1}\,(k = 1,\dots,1000)$ randomly. We choose the $x_i^* \in (\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}})\,(i = 1,\dots,n)$ and $a_i^k \in (\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}})\,(i = 1,\dots,n)\,(k = 1,\dots,1000)$ from the uniform distribution. Using x^* and a^k, we decide b^k with some noises as follows.

$$b^k = \begin{cases} (a^k)'x^* & (k \bmod 100) \neq 0 \\ 2(a^k)'x^* & (k \bmod 100) = 0 \end{cases} \quad (k = 1,\dots,1000).$$

Then, the least square problem is described as follows.

$$\min \quad \sum_{t=1}^{T} ((\tilde{a}^t)'x - \tilde{b}^t)^2\,, \tag{19}$$

where $\tilde{a}^t = a^{h(t)}$ and $\tilde{b}^t = b^{h(t)}$ with $h(t) = (t \bmod 1000) + 1$. We can treat the large number T of functions from 1000 date.

We solved the problem (19) by the proposed method and the proximal gradient method. We applied the following rule of choosing the block I_t at each iteration. First, we choose the blocksize of I_t from 10, 30, 50. The block I_t is randomly chosen so that the blocksize is fulfilled. We also set the initial stepsize $\eta_1 = 10$ and the initial point $x^1 = \mathbf{0}$ for all of the methods.

Since it is time consuming to calculate the regret $R(T)$ at each iteration, we use the following function to evaluate the sequence $\{x^1, \ldots, x^T\}$ generated by the algorithms.

$$\hat{R}(T) = \sum_{t=1}^{T} ((a^{h(t)})' x^t - b^{h(t)})^2.$$

Figure 1 illustrates the result. It shows that the proposed method converges without relation to blocksize and the method requires fewer iterations to converge when the blocksize is large. Therefore, if the calculation time is proportional to the blocksize, the proposed method is useful to get an approximate solution in real time.

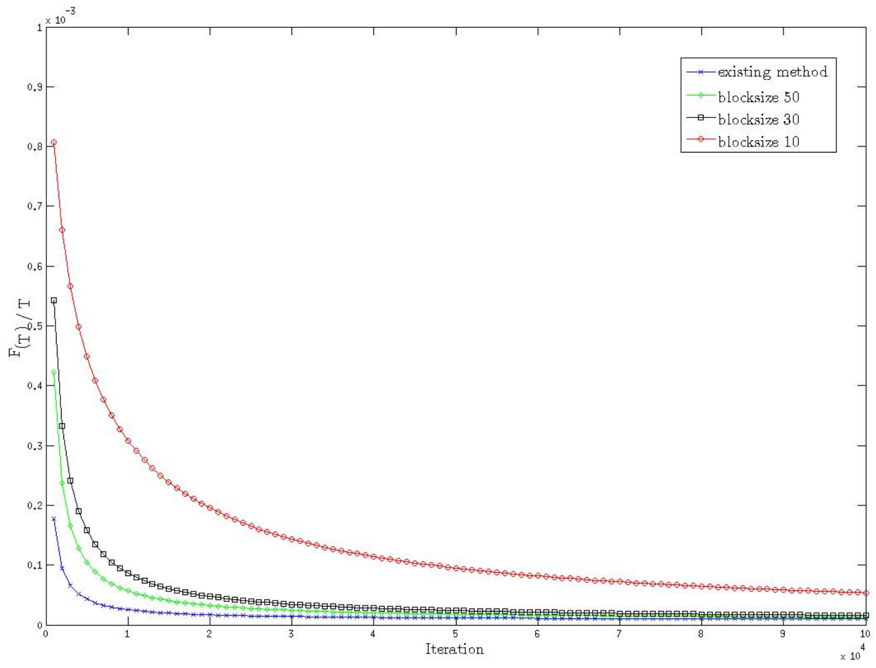

Figure 1 Comparison the proposed method with the existing method on the problem (19).

4.2 Experiment with Real Data

We consider the online learning with the Multi-Domain Sentiment Dataset [2]. This dataset consists of product reviews taken from Amazon.com for many product types. In this experiment, we used the samples in the books domain in the Multi-Domain Sentiment Dataset. The books domain has $K = 4465$ samples. For $k = 1, \ldots, K$, each sample has the input $a^k \in \mathbb{R}^n$ and the output $b^k \in \{-1, 1\}$. The input a^k consists of the frequency of the features (words or phrases) that the kth review contains. The output b^k means whether the kth review is positive (with rating ☆ 4 or ☆ 5) or negative (with rating ☆ 2 or ☆ 1). The dimension n of a corresponds to all the features in the books domain and it is 332440. The input a^k has at most 5412 (at least 11) non-zero elements, and it has 228 non-zero elements on the average.

For the online learning, we set the objective function as the Logistic loss function $f^t(x) = \log(1 + \exp(-b^t\langle a^t, x\rangle))$ with l_2 regularization. The problem to solve is as follows.

$$\min_x \quad \sum_{t=1}^{T} \left\{ \log(1 + \exp(-\tilde{b}^t \langle \tilde{a}^t), x\rangle)) + \frac{\lambda}{2}\|x\|^2 \right\}, \tag{20}$$

where $\tilde{a}^t = a^{h(t)}$ and $\tilde{b}^t = b^{h(t)}$ with $h(t) = (t \bmod K) + 1$, and $\lambda > 0$ is the weight parameter for the l_2 regularization term. Here, we set $\lambda = 1$. Similarly to the previous subsection, we give the stepsize $\eta_t = \frac{\eta_1}{\sqrt{t}}$ and the initial point $x^1 = \mathbf{0}$, and we define the function $\hat{R}(T)$ as follows.

$$\hat{R}(T) = \sum_{t=1}^{T} \left\{ \log(1 + \exp(-\tilde{b}^t \langle \tilde{a}^t, x^t\rangle)) + \frac{\lambda}{2}\|x^t\|^2 \right\}. \tag{21}$$

We set the block I_t for the proposed method as follows. First, note that the input vector a^k is very sparse, and the sparsity induces $\nabla_i f^t(x^t) = 0$ for almost i. Thus, we set a high probability to choose $I_t = \{i \mid \tilde{a}_i^t \neq 0\}$. However, this choosing rule does not match the convergence theory if the probability that choose the complement $\bar{I}_t$ is very low or 0. For this reason, we should also update $x_{\bar{I}_t}$ with high probability. However, it takes much time since $|\bar{I}_t|$ is very large. Therefore, here, we apply the following heuristics (Algorithm 3).

After x_{I_t} is updated for many times $(t_1, t_2, \ldots, t_m)$, we update $x_{\bar{I}_t}$ $(t = t_1, \ldots, t_m)$ simultaniously. From the special structure of the proximal gradient method, it takes only $O(n)$ to update $x_{\bar{I}_t}$ $(t = t_1, \ldots, t_m)$.

4.2.1 Sensitivity Experiments

Firstly, we investigate the effect of the initial stepsize η_1. We solve the problem (20) by the proposed method with each initial stepsize $\eta_1 \in \{0.1, 0.05, 0.01, 0.001\}$. We experiment for an hour, and the results are illustrated in Figure 2. It shows that the proposed method performs well when we choose the initial stepsize $\eta_1 = 0.01$.

4.2.2 Comparison with the existing proximal gradient method

We solve the problem (20) by the proposed method and the proximal gradient method. We choose the initial stepsize $\eta_1 = 0.01$ in accordance with the result of the previous experiment. We experiment for an hour, and the results are illustrated in Figure 3 and Figure 4.

We plot the value $\frac{F(T)}{T}$ in Figure 3. Although the value of the proposed method is slightly larger than that of the existing method, they are almost equivalent. This result shows that the choosing rule is successful and the proposed method updates the critical components accurately.

Figure 4 illustrates a behavior of $\frac{F(T)}{T}$ per calculation time. It indicates that the proposed method return the appropriate solution quickly. Although the existing method takes about 12 seconds per K iterations, the proposed method takes about 1.1 seconds per K iterations. This result means that the proposed method is suitable for the case required to get some appropriate solution in real time.

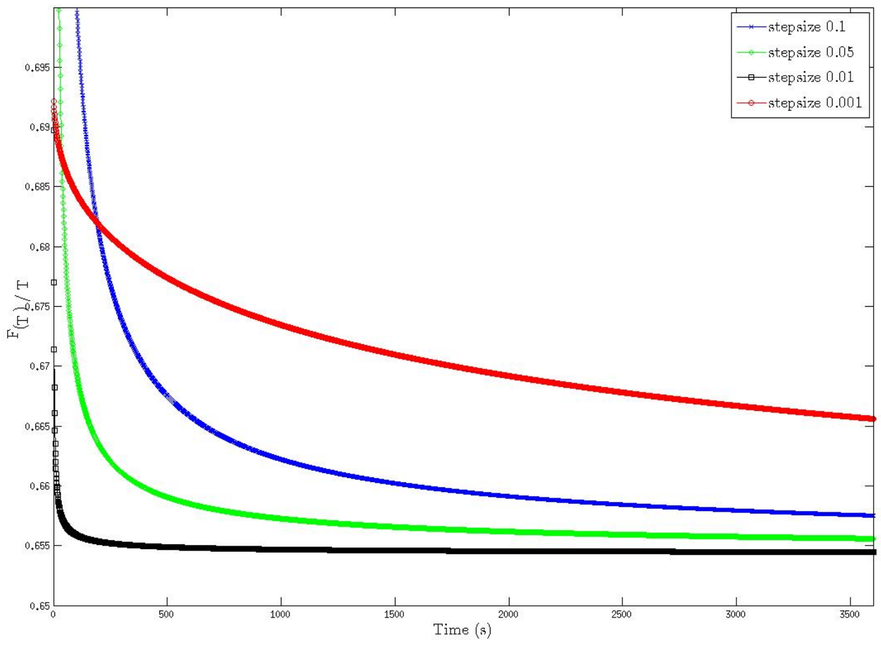

Figure 2 Sensitivity of the proposed method to initial stepsize on solving the problem (20).

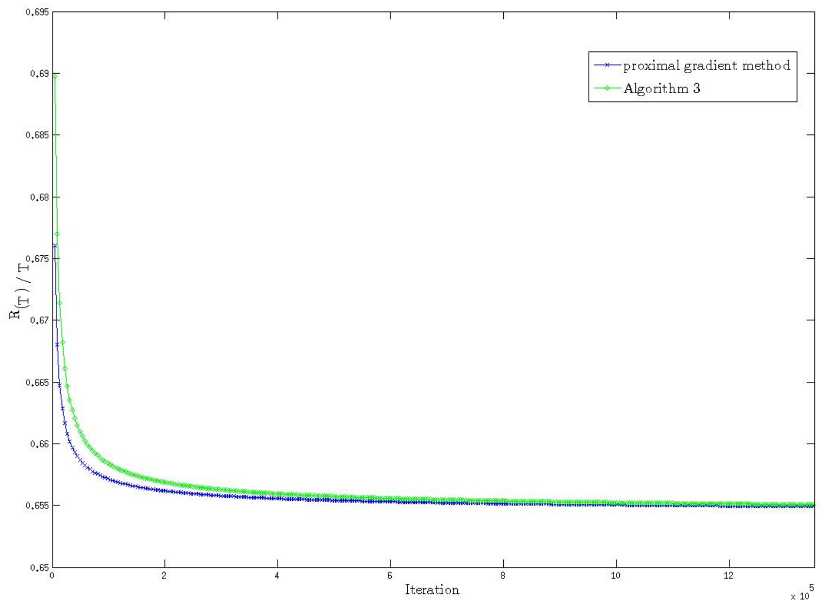

Figure 3 Comparison of the proposed method with the proximal gradient method by $\frac{F(T)}{T}$ and iterations

Algorithm 3 Heuristics for the l_2 regularized online learning

```
Input: Set the initial vector $x^1 = \mathbf{0}$, the initial stepsize $\eta_1 > 0$, and the initial vector $z^1 = \mathbf{0}$ and the
    initial value $\Gamma_1 = 0$ to accumulate stepsizes.
  for $t = 1, 2, \ldots$ do
    1. Choose a random number $a^t \in \{1, \ldots, K\}$
    if $a^t \neq K$ then
      2. Choose $I_t = \{i \,|\, x_i^{h(t)} \neq 0\}$.
      3. Update $w^{t+1}$ and $z^{t+1}$ as follows.
         $$w_i^{t+1} = \begin{cases} \frac{1}{1+\lambda\eta_t} w^t - \frac{\eta_t}{1+\lambda\eta_t} \nabla_i f^t(w^t) & (i \in I_t) \\ w_i^t & (i \notin I_t)\,, \end{cases}$$
         $$z_i^{t+1} = \begin{cases} z_i^t + \eta_t & (i \in I_t) \\ z_i^t & (i \notin I_t) \end{cases}$$
         $$\Gamma_{t+1} = \Gamma_t + \eta_t$$
      4. Update a stepsize $\eta_{t+1} = \frac{\eta_1}{\sqrt{t}}$.
    else
      2. Update $w^{t+1}$ as follows.
         $$w_i^{t+1} = \frac{1}{1 + \lambda(\Gamma_t - z_i^t)} w_i^t \quad (i = 1, \ldots, n)$$
      3. Set $z^{t+1} = \mathbf{0}$ and $\Gamma_{t+1} = 0$.
      4. Update a stepsize $\eta_{t+1} = \frac{\eta_1}{\sqrt{t}}$.
    end if
  end for
```

5 Conclusion

In this paper, we proposed the randomized block-coordinate descent method based on Nesterov's random coordinate descent method for online convex optimization problem. We also proved that the bound of $\frac{E[R(T)]}{T}$ is $O(\frac{1}{\sqrt{T}})$, and this result includes the result of the existing methods. Moreover, we experiment on the problem which have the large number of decision variables n. By choosing an appropriate block, the proposed method quickly returns an approximate solution. This result illustrates that the proposed method is useful for huge data in real time.

Since the proposed method is very sensitive to the initial stepsize, it is important to analyze various stepsize rules for future research. Moreover, it is also worth to try analyizing the proposed method with the cyclic choosing rule.

Acknowledgements

First of all, I would like to express my sincere appreciation to Associate Professor Nobuo Yamashita

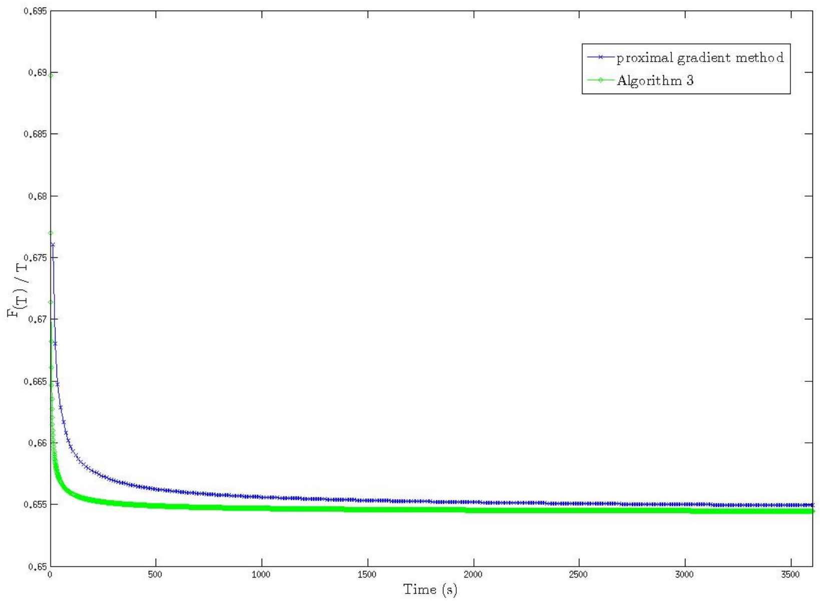

Figure 4 Comparison of the proposed method with the proximal gradient method by $\frac{F(T)}{T}$ and calculation time

for his kind guidance and suggestion. Although I sometimes troubled to him by my faults, he always supported me kindly. I am deeply indebted to Professor Masao Fukushima for his invaluable useful advice. His support always encouraged me to study. I am also grateful to Assistant Professor Shunsuke Hayashi for his numerous comments and encouragement. In addition, I also deeply thank all members of Fukushima Laboratory for their encouragements. Finally, I specially thank my family and my friends for their constant support.

References

[1] D. P. Bertsekas, Nonlinear Programming, 2nd edition, Athena Scientific, Belmont, 1999.

[2] J. Blitzer, M. Dredze and F. Pereira, Biographies, Bollywood, Boom-Boxes and Blenders: Domain Adaptation for Sentiment Classification, In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 440 - 447, 2007.

[3] G. Chen and M. Teboulle, Convergence Analysis of a Proximal-Like Minimization Algorithm Using Bregman Functions, *SIAM Journal on Optimization*, Vol. 3, No. 3, pp. 538 - 543, 1993.

[4] J. Duchi and Y. Singer, Efficient Online and Batch Learning Using Forward Backward Splitting, *Journal of Machine Learning Research*, Vol. 10, pp. 2899 - 2934, 2009.

[5] Y. Nesterov, Efficiency of Coordinate Descent Methods on Huge-Scale Optimization Problem, *SIAM Journal on Optimization*, Vol. 22, No. 2, pp. 341 - 362, 2012.

[6] P. Tseng, Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization,

SIAM Journal on Optimization, Vol. 109, No. 3, pp. 475 - 494, 2001.

[7] P. Tseng, Approximation Accuracy, Gradient Methods, and Error Bound for Structured Convex Optimization, *Mathematical Programming*, Vol. 125, No. 2, pp. 263 - 295, 2010.

[8] L. Xiao, Dual Averaging Methods for Regularized Stochastic Learning and Online Optimization, *Journal of Machine Learning Research*, Vol. 11, pp. 2543 - 2596, 2010.

[9] M. Zinkevich, Online Convex Programming and Generalized Infinitesimal Gradient Ascent, In *Proceedings 20th International Conference on Machine Learning*, pp. 928 - 936, 2003.