Master's Thesis

# Efficient Block Selections in Block Coordinate Gradient Descent Methods for Linearly Constrained Optimization Problem

Guidance

Professor Nobuo YAMASHITA

# Shinya KAMIURA

Department of Applied Mathematics and Physics

Graduate School of Informatics

Kyoto University



February 2016

#### Abstract

A block coordinate gradient descent method (BCGD) is frequently used for the large-scale optimization problem. It updates a block (set of variables) in each iteration with other variables fixed.

In this paper we consider BCGD for optimization problem with linearly equality constraints. When the block is not selected properly, BGCD does not necessarily converge globally. To guarantee the global convergence, we may select a block that satisfies the Gauss-Southwellq Rule. However, we cannot adopt the rule for the problems with more than three equality constraints, because the computational cost for finding a block satisfying the rule is  $O(n^2)$  in each iteration.

In this paper, we propose novel methods of selecting blocks which incorporates both low computational cost and the global convergence. The proposed method first selects a set of the blocks before starting the BCGD. Therefore, we do not need to construct a block in each iteration. Then BCGD selects a block from the set in a cyclic, greed or random manner. We show the global convergence of the proposed method. Moreover, numerical experiments are conducted to investigate its validity. These numerical results support the global convergence of the proposed method with a little computational cost for selecting blocks.

## Contents

1	Introduction	1		
2	BCGD with new block selection rules			
	2.1 Procedure 1: New block selection rules	2		
	2.2 Procedure 2: The BCGD method with the set of blocks	6		
3	3 Gloabal Convergence			
4	Concrete block selection algorithms	9		
	4.1 Block construction algorithm for Rule A	10		
	4.2 Block construction algorithm for Rule B	12		
5	5 Numerical experience			
6	Conclusion	15		

## **1** Introduction

The recent developments of the information technology produce so-called big data, and we face applications with such data, which appear in machine learning and statics [6, 2]. Then some of these applications are formulated as large-scale optimization problems.

In this paper, we consider the following linearly equality constraint minimization problem.

$$\begin{array}{l} \min \quad f(x) \\ \text{s.t.} \quad Ax = b, \end{array} \tag{1}$$

where  $f : \mathbb{R}^n \to \mathbb{R}$  is smooth,  $A \in \mathbb{R}^{m \times n}$  and  $b \in \mathbb{R}^m$ . We assume that *n* is very large.

The standard solution methods for problem (1) are Newton type methods. However the methods are not applicable for the large-scale problems since they have to solve n + m linear equations at each iteration. Moreover, if the number of variables is more than hundreds of thousands, we cannot apply the gradient methods, such as the projected gradient method. There are a few solution methods that can handle such large scale problems. One of them is the block coordinate gradient descent method (BCGD).

The BCGD updates a block (set of variables) in each iteration with other variables. Roughly speaking, its kth iteration is written as

Choose 
$$\mathcal{J}^{k} \subseteq \{1, \cdots, n\}$$
.  
Set  $x_{\mathcal{J}^{k}}^{k+1} \approx \underset{x_{\mathcal{J}^{k}}}{\operatorname{arg min}} f(x)$ ,  
 $x_{\mathcal{J}^{k}}^{k+1} = x_{\mathcal{J}^{k}}^{k}$ ,

where  $\mathcal{J}^k$  is a subset of indices set of variables,  $\overline{\mathcal{J}}$  is a complement set of  $\mathcal{J}$  for  $\{1, \ldots, n\}$ . Moreover  $x_{\mathcal{J}}$  is a vector consisting of  $x_i$   $i \in \mathcal{J}$ . Throughout the paper, we call  $x_{\mathcal{J}^k}$  or  $\mathcal{J}^k$  a block at the *k*th iteration.

When we solve the unconstrained minimization problem by the BCGD, we do not care about a selection of blocks for global convergence. Even if  $\mathcal{J}^k$  is singlton, that is  $\mathcal{J}^k = \{i\}$ , and *i* is chosen in a cyclic manner, the BCGD converges globally. If the problem has constraints, we have to choose block carefully. Otherwise, the BGCD may not converge to a solution of the problem.

The Gauss-Southwell-q rule [8] is a block selection rule that guarantees the global convergence of the BCGD for problem (1). The rule requires that the next iterate  $x^{k+1}$  reduces a value of a certain model function of f as much as that by a full gradient method. To get a block satisfying the rule, we have to solve some complicated problem at each iteration. When m = 1 and m = 2, the computational costs to get the block are O(n) and  $O(n \ln n)$ , respectively [8]. Unfortunately, it costs become  $O(n^2)$  for  $m \ge 3$ . Note that the cost  $O(n^2)$  is almost same as that of single iteration of the full gradient method. Then we cannot adopt the rule for the BCGD, since its computation is required at in each iteration.

In this paper, we propose a new block selection technique that incorporate both low computational cost and the global convergence of the BCGD for problem (1) with  $m \ge 3$ . Moreover, the technique can choose a block with any size. The proposed technique method first selects a set of candidates of blocks before starting the BCGD. Then the BCGD selects a block from the set in a cyclic, greed or random manner. Therefore, we do not need to construct a block at each iteration as the Gauss-Southwell-q rule does. We give concrete algorithms generating the set within  $O(m^3n)$ . Note that the algorithm is conducted only once before the BCGD starts. Thus it is applicable for problem (1) with  $m \ge 3$ . Moreover we show the global convergence of the BCGD with the proposed block selection technique.

The paper is organized as follow. In section 2, we propose new block selection techniques. Then we show the global convergence of the BCGD with the selection in section 3. Section 4 presents concrete algorithms that get the set of blocks. In section 5, we give some numerical results for the proposed method. In section 6, we make some concluding remarks and discuss future research topics.

Throughout the paper we use the following notations. Let X be a feasible set of problem (1), that is,  $X = \{x \in \mathbb{R}^n \mid Ax = b\}$ . Let  $A_i$  be the *i*th column vector of A. For given a set  $\mathcal{J} \subseteq \{1, 2, ..., n\}$ ,  $A_{\mathcal{J}}$  denotes an  $m \times |\mathcal{J}|$  matrix whose columns are  $A_i \ i \in \mathcal{J}$ , and  $b_{\mathcal{J}}$  is a  $|\mathcal{J}|$  dimensional vector whose elements are  $b_i \in \mathcal{J}$ . Moreover  $\nabla_{\mathcal{J}} f(x)$  denotes a  $|\mathcal{J}|$  dimensional vector whose elements are  $\frac{\partial f(x)}{\partial x_i} \in \mathcal{J}$ . For a given matrix M, M > 0 means that M is positive definite. Let X be the feasible set of problem (1), that is  $X = \{x \in \mathbb{R}^n \mid Ax = b\}$ .

## 2 BCGD with new block selection rules

In this section, we propose new block selection rules that guarantee the global convergence of BCGD for problem (1).

We first describe a prototype of the BCGD with the new block selection rules, which first constructs a set of candidates of blocks in order to satisfy the new rules, then applies the BCGD that choses a block from the set at each iteration.

Algorithm 1	Prototype of BCGD with new block selection rules
Procedure 1:	Construct a set of candidates of blocks.

$$\mathcal{B} = \{\mathcal{J}_1, \mathcal{J}_2, \cdots, \mathcal{J}_t\}.$$

Procedure 2: Execute a BCGD whose *k*th iteration choses a block  $\mathcal{J}^k \in \mathcal{B}$ .

Note that the BCGD in Procedure 2 does not have to construct a block at each iteration, and it only choose a block from  $\mathcal{B}$  that has already constructed in Procedure 1. Thus each iteration of the BCGD can be executed very quickly. However, as written in Introduction, if the set  $\mathcal{B}$  is constructed properly, the BCGD may not converges to a solution of problem (1).

In the next subsection, we propose two rules for  $\mathcal{B}$  to be satisfied for the global convergence of the BCGD. Then we present a concrete BCGD with the rules.

#### 2.1 Procedure 1: New block selection rules

We first discuss the Karush-Kuhn-Tucker conditions of (1), which is a basis of the new block selection rules.

The KKT conditions for problem (1) are written as I

$$\nabla f(x^*) + A^\top \lambda = 0, \quad Ax^* = b, \tag{2}$$

where  $\lambda \in \mathbb{R}^m$  is Lagrange multipliers for Ax = b. Note that if  $x^*$  is a local optimal solution of (1), then there exist constants  $\lambda$  that satisfies the KKT conditions (2). Moreover, its converse holds if f is convex.

Decomposing the KKT conditions (2) with respect to  $\mathcal{J}$  and  $\overline{\mathcal{J}}$ , (2) is rewritten as

$$\begin{pmatrix} \nabla_{\mathcal{J}} f(x^*) \\ \nabla_{\bar{\mathcal{J}}} f(x^*) \end{pmatrix} + \begin{pmatrix} A_{\mathcal{J}}^\top \\ A_{\bar{\mathcal{J}}}^\top \end{pmatrix} \lambda = 0, \quad \begin{pmatrix} A_{\mathcal{J}} & A_{\bar{\mathcal{J}}} \end{pmatrix} \begin{pmatrix} x^*_{\mathcal{J}} \\ x^*_{\bar{\mathcal{J}}} \end{pmatrix} = b.$$
(3)

When we fix  $x_{\bar{\mathcal{J}}}$  to  $x_{\bar{\mathcal{T}}}^*$ , problem (1) is reduced to

$$\min_{x_{\mathcal{J}}} \quad f(x_{\mathcal{J}}, x_{\bar{\mathcal{J}}}^*)$$
s. t. 
$$A_{\mathcal{J}} x_{\mathcal{J}} = b - A_{\bar{\mathcal{J}}} x_{\bar{\mathcal{J}}}^*.$$
(4)

The KKT conditions for the problem is written as

$$\nabla_{\mathcal{J}} f(x^*) + A_{\mathcal{J}}^{\top} \lambda^1 = 0, \quad A_{\mathcal{J}} x_{\mathcal{J}}^* = b - A_{\tilde{\mathcal{J}}} x_{\tilde{\mathcal{J}}}^*.$$
(5)

We call the conditions the KKT conditions on the block  $\mathcal{J}$ .

On the other hand, when we fix  $x_{\mathcal{J}}$  to  $x_{\mathcal{J}}^*$ , problem (1) is reduced to

$$\begin{split} \min_{x_{\bar{\mathcal{J}}}} & f(x_{\mathcal{J}}^*, x_{\bar{\mathcal{J}}}) \\ \text{s. t.} & A_{\bar{\mathcal{J}}} x_{\bar{\mathcal{J}}} = b - A_{\mathcal{J}} x_{\mathcal{J}}^*. \end{split}$$

Then its KKT conditions are

$$\nabla_{\mathcal{J}}f(x^*) + A_{\mathcal{J}}^{\top}\lambda^2 = 0, \quad A_{\mathcal{J}}x_{\mathcal{J}}^* = b - A_{\tilde{\mathcal{J}}}x_{\tilde{\mathcal{J}}}^*. \tag{6}$$

We note that the conditions (5) and (6) are different from (2) for problem (1) since  $\lambda^1$  may not be  $\lambda^2$ . This is seen in the following example.

**Example 2.1.** Consider the following problem.

$$\min x_1^2 + x_2^2,$$
  
s.t.  $x_1 + x_2 = 1.$  (7)

The KKT conditions for (7) are written as.

$$2x_1^* + \lambda = 0, \ 2x_2^* + \lambda = 0, \ x_1^* + x_2^* = 1.$$
(8)

From the conditions we have  $(x_1^*, x_2^*, \lambda) = (\frac{1}{2}, \frac{1}{2}, -1)$ . Now, we let  $\mathcal{J} = \{1\}$  and  $(\bar{x}_1, \bar{x}_2) = (1, 0)$ . Then the problem with  $x_2$  fixed is written as

$$\min x_1^2$$
  
s.t.  $x_1 = 1$ .

Moreover, the problem with  $x_2$  fixed is

min 
$$x_2^2$$
  
s.t.  $x_2 = 0$ 

Then the KKT conditions of both problems are written as

$$2\hat{x}_1 + \lambda^1 = 0, \quad \hat{x}_1 = 1$$

and

$$2\hat{x}_2 + \lambda^2 = 0, \quad \hat{x}_2 = 0$$

respectively. Note that  $(\hat{x}_1, \hat{x}_2) = (1, 0) = (\bar{x}_1, \bar{x}_2)$ , but it is not a solution of (7). This is because  $\lambda^1 = -2$ ,  $\lambda^2 = 0$  and they are different. If  $\lambda^1 = \lambda^2$ , then we may get a solution of (7).

The above discussion indicates that Lagrange multipliers in the KKT conditions of problem (4) for each block  $\mathcal{J} \in \mathcal{B}$  must be same. Now consider two blocks  $\mathcal{J}_1, \mathcal{J}_2 \in \mathcal{B}$ . Then the first equation in the KKT conditions (5) for each blocks is written as

$$\nabla_{\mathcal{J}_1} f(x^*) + A_{\mathcal{J}_1}^\top \lambda^1 = 0, \quad \nabla_{\mathcal{J}_2} f(x^*) + A_{\mathcal{J}_2}^\top \lambda^2 = 0,$$

where  $\lambda^1, \lambda^2 \in \mathbb{R}^m$ . We want  $\lambda^1$  and  $\lambda^2$  to be same. Then we suppose that  $A_{\mathcal{J}_1 \cap \mathcal{J}_2}$  is full row rank, that is,  $A_{\mathcal{T} \cup \mathcal{T}}^{\mathsf{T}}$  is full column rank. From the above equations, we have

$$A_{\mathcal{J}_{1}\cap\mathcal{J}_{2}}^{\top}\lambda^{1} - A_{\mathcal{J}_{1}\cap\mathcal{J}_{2}}^{\top}\lambda^{2} = -\nabla_{\mathcal{J}_{1}\cap\mathcal{J}_{2}}f(x^{*}) + \nabla_{\mathcal{J}_{1}\cap\mathcal{J}_{2}}f(x^{*}) = 0$$

Therefore we have  $\lambda^1 = \lambda^2$  when  $A_{\mathcal{J}_1 \cup \mathcal{J}_2}^{\top}$  is full column rank. Based on the above discussion, we now propose two rules for the BCGD to converge globally. Here let the number of blocks in  $\mathcal{B}$  is t, that is,  $\mathcal{B} = \{\mathcal{J}_1, \mathcal{J}_2, \cdots, \mathcal{J}_t\}$ . Moreover let  $\mathcal{N} = \{\mathcal{J}_1, \mathcal{J}_2, \cdots, \mathcal{J}_t\}$ .  $\{1, 2, \cdots, n\}$  and  $\mathcal{T} = \{1, 2, \dots, t\}$ 

The first rules are following.

#### **Rule** A

- (i)  $\bigcup_{\mathcal{T}\in\mathcal{B}} \mathcal{J} = \mathcal{N}$ .
- (ii) A graph  $(\mathcal{T}, E)$  is connected, where  $E = \{(p, q) \in \mathcal{T} \times \mathcal{T} \mid A_{\mathcal{J}_p \cap \mathcal{J}_q} \text{ is full row rank}\}.$
- (iii) For each  $\mathcal{J} \in \mathcal{B} A_{\mathcal{J}}^{\top}$  is full column rank.

**Rule A** (i) means that  $\mathcal{B}$  must cover all variable. If the BCGD chooses all blocks in  $\mathcal{B}$ , then it updates all the variables. **Rule A** (ii) means that for any block  $\mathcal{J}_p \in \mathcal{J}$ , there exists another block  $\mathcal{J}_q$  such that the Lagrange multipliers in the KKT conditions on  $\mathcal{J}_p$  is same as those on  $\mathcal{J}_q$ 

The second rule is given as follows.

### Rule B

- (i)  $\bigcup_{\mathcal{J} \in \mathcal{B}} \mathcal{J} = \mathcal{N}$ .
- (ii)  $A_{(\mathcal{J}_1 \cup \mathcal{J}_2 \cup \cdots \cup \mathcal{J}_{p-1}) \cap \mathcal{J}_p}$  is full row rank for all  $p \in \{2, 3, \cdots, t\}$ .
- (iii) For each  $\mathcal{J} \in \mathcal{B} A_{\mathcal{J}}^{\top}$  is full column rank.

Rule B (ii) also makes that Lagrange multipliers of all block to be same each other.

We now present a property on **Rule A** and **Rule B**, which is a key to show the global convergence of the BCGD with the rules. (2).

**Lemma 2.1.** Suppose a set  $\mathcal{B}$  of blocks satisfies **Rule A** or **Rule B**. Suppose also that  $(x^*, \lambda^p)$  satisfies the KKT conditions (5) on  $\mathcal{J}_p$  for any  $p \in \mathcal{T}$ . Then, for any  $p \in \mathcal{T}$ ,  $(x^*, \lambda^p)$  satisfies the KKT conditions (2) of problem (1).

Proof. From the assumption we have

$$\nabla_{\mathcal{J}_p} f(x^*) + A_{\mathcal{J}_p}^\top \lambda^p = 0 \quad \forall p \in \mathcal{T}.$$
<sup>(9)</sup>

First, we consider the case where **Rule A** holds. It then follows from **Rule A** (ii) that for any  $p \in \{1, 2, \dots, N\}$ , there exists *q* such that  $(p, q) \in E$ . Then we have from (9)

$$\nabla_{\mathcal{J}_p \cap \mathcal{J}_q} f(x) + A_{\mathcal{J}_p \cap \mathcal{J}_q}^{\top} \lambda^p = 0,$$
  
$$\nabla_{\mathcal{J}_p \cap \mathcal{J}_q} f(x) + A_{\mathcal{J}_p \cap \mathcal{J}_q}^{\top} \lambda^q = 0.$$

From the equations, we have

$$A_{\mathcal{J}_p \cap \mathcal{J}_q}^{\top}(\lambda^p - \lambda^q) = 0.$$

Since  $A_{\mathcal{J}_p \cap \mathcal{J}_q}$  is full row rank by  $(p, q) \in E$ , we have

$$\lambda^p = \lambda^q$$

Let  $\lambda^* = \lambda^p$ . Then  $\lambda^i = \lambda^*$  for any *i* such that  $(i, p) \in E$ . Since *E* is connected by **Rule A** (ii),  $\lambda^i = \lambda^*$  for  $i \in \mathcal{T}$ , and hence we have

$$\nabla_{\mathcal{J}_p} f(x) + A_{\mathcal{J}_p}^{\top} \lambda^* = 0 \quad \forall p \in \mathcal{T}.$$
 (10)

Since all variables are covered by the blocks in Rule A (i), we have

$$\nabla f(x^*) + A^\top \lambda^* = 0, \ Ax^* = b$$

which is the KKT condition of problem (1).

Next, we consider the case where **Rule B** holds. In a way similar to the case of **Rule A**, we can show Lagrange multipliers in the KKT conditions on block  $\mathcal{J}_p$  for  $p \ge 2$  is clearly equal to those in the KKT conditions on block  $(\mathcal{J}_1 \cup \mathcal{J}_2 \cup \cdots \cup \mathcal{J}_{p-1})$ . Moreover, from **Rule B** (i), we have

$$\nabla f(x^*) + A^\top \lambda^p = 0, \ Ax^* = b,$$

which shows the desired property.

In section 4, we give concrete algorithms that find  $\mathcal{B}$  satisfying one of the rules.

#### 2.2 Procedure 2: The BCGD method with the set of blocks

In this subsection, we introduce a BCGD with the set  $\mathcal{B}$  of candidates of blocks. The BCGD is same as that in [8] except for a method choosing a block at each iteration. Thus we briefly explain it. The BCGD has three steps: (a) choosing a block, (b) finding a search direction, and (c) determining a step length to reduce the objective function value.

First, we explain how to choose a block. As written before, the BCGD select one block from the set  $\mathcal{B}$  that is constructed in Procedure 1. As usual BCGD methods for unconstrained minimization problem, we have the following three rules for choosing  $\mathcal{J}$  from  $\mathcal{B}$ .

#### **Random Rule**

Choose a block  $\mathcal{J}$  from  $\mathcal{B}$  randomly.

#### Almost Cyclic Rule

There exists a positive integer *C* such that all blocks  $\mathcal{J} \in \mathcal{B}$  is chosen at least once within C successive iterations.

#### Gauss-Southwell Rule (Greedy Rule)

For given  $x^k$ , choose  $\mathcal{J}$  such that

$$\mathcal{J} \in \underset{\mathcal{J} \in \mathcal{B}}{\operatorname{arg\,min}} \min_{d \in \mathbb{R}^n} \left\{ \nabla f(x^k)^\top d + \frac{1}{2} \|d\|^2 \mid Ad = 0, \ d_j = 0 \ \forall j \notin \mathcal{J} \right\}.$$

We will show the proposed method with any rule converges globally. Note that the almost cyclic rule chooses all blocks almost equally. Thus it may frequently choose blocks that does not sufficiently decrease the objective function. On the other hand, the Gauss-Southwell rule choose the best block from the viewpoint of decreasing the objective function at each iteration. However, it require much time to fine the best one.

Next we define a search direction with respect the chosen block  $\mathcal{J}$  and the current iterate  $x^k$  as follows.

$$d_H(x^k;\mathcal{J}) = \arg\min_d \left\{ \nabla f(x^k)^\top d + \frac{1}{2} d^\top H d \mid Ad = 0, \ d_j = 0 \ \forall j \notin \mathcal{J} \right\},\tag{11}$$

where *H* is an  $n \times n$  positive definite matrix. We may choose an approximate matrix of the Hessian  $\nabla^2 f(x)$ . Note that since  $d_j = 0 \forall j \notin \mathcal{J}$ , we only need a  $|\mathcal{J}| \times |\mathcal{J}|$  submatrix of *H*.

To guarantee that  $d_H(x; \mathcal{J})$  is well-defined, we assume that  $H_{\mathcal{J}\mathcal{J}}$  is positive definite on Null $(A_{\mathcal{J}})$ , that is,  $C_{\mathcal{J}}^{\top}H_{\mathcal{J}\mathcal{J}}C_{\mathcal{J}} > 0$ , where  $C_{\mathcal{J}}$  is a matrix whose columns forms an orthonormal basis for Null $(A_{\mathcal{J}})$ . Then we have the following lemma.

**Lemma 2.2.** [8, Lemma 2.1] Suppose that  $C_{\mathcal{J}}^{\top}H_{\mathcal{J}\mathcal{J}}C_{\mathcal{J}} > 0$ . Then we have

$$\nabla f(x^k)^{\mathsf{T}} d_H(x^k;\mathcal{J}) \le -\lambda_{\min}(C_{\mathcal{J}}^{\mathsf{T}} H_{\mathcal{J}\mathcal{J}} C_{\mathcal{J}}) \|d_H(x;\mathcal{J})\|^2$$
(12)

Lemma 2.2 means that  $d_H(x; \mathcal{J})$  is a direction for which the decrease of the objective function.

Finally we explain how to determine the stepsize. There are various stepsize rules for smooth optimization[1, 3, 4, 5]. We adapted the following simple Armijo rule.

#### Armijo rule

Let  $\beta$ ,  $\sigma$  be parameters such that  $0 < \beta < 1$  and  $0 < \sigma < 1$ . For given a search direction  $d^k$ , let a step size  $\alpha^k$  be the largest element of  $\{\beta^j\}_{j=0,1,\dots}$  satisfying

$$f(x^k + \alpha^k d^k) \le f(x^k) + \sigma \alpha^k \nabla f(x^k)^{\mathsf{T}} d^k.$$
(13)

If f is a quadratic function, then we may use the minimization rule instead of Armijo rule. Now we present a concrete algorithm of Procedure 2.

Algorithm 2 Block coordinate gradient descent method with  $\mathcal{B}$ 

Step 0 : Choose an initial point  $x^0 \in X$ . Set k := 0.

Step 1 : Choose a block  $\mathcal{J}^k$  from  $\mathcal{B}$  by Random rule, Almost cyclic rule or Gauss-Southwell rule. Choose a positive definite matrix  $H^k$ .

Step 2: Get  $d^k = d_{H^k}(x^k; \mathcal{J}^k)$  by solving (11).

Step 3 : Find a step size  $\alpha^k$  that satisfy the Armijo rule. Set  $x^{k+1} = x^k + \alpha^k d^k$ .

Step 4 : If  $x^{k+1}$  satisfies the KKT conditions, terminate. Otherwise, set k = k + 1 and go to Step 1.

## **3** Gloabal Convergence

In this section, we show the global convergence of Algorithm 2 with a set  $\mathbb{B}$  satisfying **Rule A** or **Rule B**. For this purpose, we need the following assumptions.

Assumption 1. (a) A level set  $\Omega = \{x \in X \mid f(x) \le f(x^0)\}$  is bounded.

(b) There exists a positive constant L such that

$$\|\nabla f(y) - \nabla f(z)\| \le L \|y - z\| \quad \forall y, z \in \Omega.$$
(14)

- (c) The set  $\mathcal{B}$  satisfies Rule A or Rule B.
- (d) There exist positive constant  $\lambda$  and  $\overline{\lambda}$  such that

$$\overline{\lambda}I \geq C_{\mathcal{J}^k}^{\top} H_{\mathcal{J}^k,\mathcal{J}^k}^k C_{\mathcal{J}^k} \geq \underline{\lambda}I \quad \forall k.$$

Assumption 1 (b) holds if f is twice continues differentiable. Assumption 1 (d) is necessary to get sufficiently decreasing search directions. It always holds when  $H^k = I$ .

The main theorem of the paper is described as follows.

**Theorem 3.1.** Suppose that Assumption 1 holds. Let  $\{x^k\}$  be a sequence generated by the Algorithm 2. Then any accumulation point  $x^*$  of  $\{x^k\}$  satisfies the KKT conditions (2).

We need several lemmas to prove this theorem. In the proofs of the lemmas we frequently use the following notations for simplicity:  $g = \nabla f(x^k)$ ,  $d = d^k$  and  $x = x^k$ .

The next lemma shows that we can find a step length  $\alpha^k$  that satisfies the Armijo rule.

**Lemma 3.1.** Suppose Assumption 1 holds. Then  $\underline{\lambda} ||d||^2 \leq -\nabla f(x)^\top d$ . Moreover, for any  $\sigma$  and  $\alpha$  such that  $\sigma \in (0, 1)$  and  $0 \leq \alpha \leq \frac{2\underline{\lambda}(1-\sigma)}{L}$ , we have

$$f(x^{k} + \alpha d^{k}) - f(x^{k}) \le \sigma \alpha \nabla f(x^{k})^{\mathsf{T}} d^{k}.$$
(15)

*Proof.* First note that  $\underline{\lambda} ||d||^2 \le -\nabla f(x)^\top d$  from Lemma 2.2. We have for any  $\alpha \ge 0$ 

$$f(x + \alpha d) - f(x) = \alpha \nabla f(x^k)^{\mathsf{T}} d + \int_0^1 (\nabla f(x + t\alpha d) - \nabla f(x))^{\mathsf{T}} (\alpha d) dt$$
  
$$\leq \alpha g^{\mathsf{T}} d + \alpha \int_0^1 \|\nabla f(x + t\alpha d) - \nabla f(x))^{\mathsf{T}}\| \|d\| dt$$
  
$$\leq \alpha g^{\mathsf{T}} d + \alpha^2 \frac{L}{2} \|d\|^2, \tag{16}$$

where the second inequality follows from Assumption 1 (b). Moreover, since  $\alpha \leq \frac{2\underline{\lambda}(1-\sigma)}{L}$ , we have

$$\alpha^{2} \frac{L}{2} \|d\|^{2} \leq \alpha (1 - \sigma) \underline{\lambda} \|d\|^{2}$$
$$\leq -\alpha (1 - \sigma) g^{\top} d.$$

Thus we obtain (15) by (16).

This lemma says that Algorithm 2 is well-defined. The next lemma shows that  $||d^k|| \rightarrow 0$  as  $k \rightarrow \infty$ .

**Lemma 3.2.** Suppose that Assumption 1 holds. Then  $\{f(x^k)\}$  converges. Moreover  $\{\nabla f(x^k)^\top d^k\} \rightarrow 0$  and  $\{d^k\} \rightarrow 0$ .

*Proof.* First note that  $\{f(x^k)\}$  is non-increasing due to Lemma 3.1. It then follows from Assumption 1 (i) that  $\{x^k\}$  is bounded. Let  $\{x^k\}_{\mathcal{K}}$  be a subsequence of  $\{x^k\}$  such that  $\{x^k\}_{\mathcal{K}} \to \overline{x}$ . Since f is smooth, we have  $f(\overline{x}) = \lim_{\substack{k \to \infty \\ k \in \mathcal{K}}} f(x^k)$ . Moreover, since  $\{f(x^k)\}$  is non-increasing, the whole sequence  $\{f(x^k)\}$  converges to  $f(\overline{x})$ .

From the Armijo rule and Lemma 3.1, we have  $\frac{\alpha^k}{\beta} > \min\{1, \frac{2\underline{\lambda}(1-\sigma)}{L}\}$ . Since  $\{f(x^k)\}$  congerges, it then follows from Lemma 3.1 that  $\{\nabla^{\top} f(x^k)d^k\} \to 0$ . Moreover, we have  $\{d^k\} \to 0$  from Lemma 3.1.

Next we show that  $d_{H^k}(x^k; \mathcal{J}) \to 0$  for all  $\mathcal{J} \in \mathcal{B}$ .

**Lemma 3.3.** Suppose that Assumption 1 holds. Then for any  $\mathcal{J} \in \mathcal{B}$ , there exist a subsequence  $\{x^k\}_{\mathcal{K}}$  of  $\{x^k\}$  such that

$$\lim_{k \in \mathcal{K}, k \to \infty} d_{H^k}(x^k; \mathcal{J}) \to 0.$$
(17)

*Proof.* We only consider the almost cyclic rule. For  $\mathcal{J} \in \mathcal{B}$ , there exists an infinite subsequence  $\mathcal{K}$  of  $\{1, 2, ...\}$  such that  $\mathcal{J}^k = \mathcal{J}$ . From Lemma 3.2  $||d^k|| \to 0$  as  $k \to \infty$ . Therefore we have  $\lim_{k \in \mathcal{K}, k \to \infty} d_{H^k}(x^k; \mathcal{J}) \to 0$ .

For Gauss-Southwell rule and the random rule we can show the lemma in a similar way.

Next we show that the KKT conditions on each  $\mathcal{J} \in \mathcal{B}$  hold.

**Lemma 3.4.** Let  $x^*$  be an accumulation point of  $x^k$ . Then  $x^*$  satisfies the KKT conditions on  $\mathcal{J}$  for each  $\mathcal{J} \in \mathcal{B}$ .

*Proof.* We only consider the almost cyclic rule. Let  $\mathcal{J} \in \mathcal{B}$ . Then there exists an infinite subsequence  $\mathcal{K}$  of  $\{1, 2, ...\}$  such that  $\mathcal{J}^k = \mathcal{J}$  and  $\{x^k\}_{\mathcal{K}} \to x^*$ .

Note that for any  $k \in \mathcal{K}$  the direction  $d^k = d_{H^k}(x^k; \mathcal{J})$  is the unique solution of

min 
$$\nabla_{\mathcal{J}} f(x^k)^{\mathsf{T}} d_{\mathcal{J}} + \frac{1}{2} d_{\mathcal{J}}^{\mathsf{T}} H_{\mathcal{J}\mathcal{J}}^k d_{\mathcal{J}}$$
  
s.t.  $A_{\mathcal{I}} d_{\mathcal{I}} = 0.$ 

Then there exists  $\lambda^k$  such that the following KKT condition holds.

$$\nabla_{\mathcal{J}} f(x^k) + H^k_{\mathcal{I}\mathcal{J}} d^k_{\mathcal{J}} + A^{\top}_{\mathcal{J}} \lambda^k = 0, \ A_{\mathcal{J}} d^k_{\mathcal{J}} = 0.$$

Since  $A_{\mathcal{J}}^{\top}$  is full column rank,  $\{\lambda^k\}$  is bounded. Without loss of generality, we assume that  $\lim_{k \in \mathcal{K}, k \to \infty} \lambda^k = \lambda^*$ . Then we have

$$\lim_{\substack{k \in \mathcal{K}, k \to \infty}} x^k = x^*,$$
$$\lim_{\substack{k \in \mathcal{K}, k \to \infty}} d^k_{\mathcal{J}} = 0,$$

where the second inequality follows from Lemma 3.3. Therefore  $x^*$  satisfies the KKT condition on the block  $\mathcal{J}$ .

Now we stand at a position to show the main theorem. We only give a proof for the almost cyclic rule.

*Proof.* For any block  $\mathcal{J} \in \mathcal{B}$  the KKT conditions on  $\mathcal{J}$  holds due to Lemma 3.4. Therefore  $x^*$  is a stationary point of problem 1 by Lemma 2.1.

### **4** Concrete block selection algorithms

In this section we propose concrete algorithms that satisfy **Rule A** or **Rule B**. In the remainder of the paper, we define *s* a parameter that controls a block size, which is m + 1 + s. More precisely, the size of blocks in the proposed algorithms becomes m + 1 + s.

#### 4.1 Block construction algorithm for Rule A

#### Block construction algorithm 1 for Rule A

In this algorithm, we make a block whose size is m + 1 + s, by sliding *m* elements in the index set. Although we can get a set of block in this way, this set does not necessarily satisfy the Rule A (ii). Therefore, we execute Step 5 and Step 6, and the set becomes to satisfy Rule A. Through this algorithm, we can get the set of blocks as Figure 1.



Figure 1: Example of Algorithm 3

Algorithm 3 Block construction algorithm 1 for Rule A

**Input:** *n*, *m*, *s*, *A* 

Output: blocklist

Step 0: blocklist:=[empty list], p := 0, start := 0, end := m + s

Step 1: block:=[list of integers from *start* to *end*], append block to blocklist.

Step 2: start := end - (m - 1), end := start + m + s, block:=[list of integers from start to end], append block to blocklist.

Step 3 : if end is n - 1 or more, then go to step 4, else then go to step 3.

Step 4 : if the last element of blocklist have *n* or more of the elements then we delete it. if the last block of blocklist is a subset to penultimate block then we delete it.

Step 5: let the intersection of blocklist[p] and blocklist[p + 1] be  $\mathcal{J}_p$ . If  $A_{\mathcal{J}_p}$  is full row rank then go to step 6. Else then we append an element of blocklist[p] which does not include in blocklist[p + 1] to blocklist[p + 1]. And go to step 5.

Step 6: If *p* is *n* or more, return blocklist and exit the algorithm , else go to step 5.

**Example 4.1.** n = 7, m = 2, s = 0  $\mathcal{B} = \{\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3, \mathcal{J}_4, \mathcal{J}_5\}$   $\mathcal{J}_1 = \{0, 1, 2\}, \mathcal{J}_2 = \{1, 2, 3\}, \mathcal{J}_3 = \{2, 3, 4\}, \mathcal{J}_4 = \{3, 4, 5\}, \mathcal{J}_5 = \{4, 5, 6\}$  **Example 4.2.** n = 10, m = 3, s = 2  $\mathcal{B} = \{\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3\}$  $\mathcal{J}_1 = \{0, 1, 2, 3, 4, 5\}, \mathcal{J}_2 = \{3, 4, 5, 6, 7, 8\}, \mathcal{J}_3 = \{6, 7, 8, 9\}$ 

#### Block construction algorithm 2 for Rule A

In this algorithm, we first extract m elements from an index set, where a matrix composed

of *m* column vectors from matrix *A* is full row rank. Then, we combine the *m* elements and components of the index set except the *m* elements, and we can compose a block as in Figure 2. If they are contributed decrease the value of objective function, the sequence converges rapidly, and otherwise the sequence converges slowly.



Figure 2: Example of Algorithm 4

In this algorithm, all blocks include the first m element. So if they are contributed to reduction of the objective function, the sequence converges rapidly. However otherwise the sequence converge slowly.

Algorithm 4 Block construction algor	ithm 2 for Rule A
--------------------------------------	-------------------

**Input:** *n*, *m*, *s*, *A* 

**Output:** blocklist

Step 0: blocklist:=[empty list], p := 0, allvar:= [list of integers from 0 to n - 1]

Step 1 : S:=[list of m elements extracted from allvar at random]

- Step 2: If  $A_S$  is full row rank, delete the element of allvar which is include in S, and go to step 3. Else go to step 1.
- Step 3: block:=[list which is composed of both (s + 1) elements extracted from allvar at random and elements in S]

, append block to blocklist. delete the (1 + s) elements in allvar and go to step 4

- Step 4: If number of allvar's element is (1 + s) or more, go to step 3. else if number of allvar's element is between 1 to s, s:=(number of allvar's element) and go to step 3. Else if number of allvar's element is equal to 0, go to Step 5.
- Step 5: let the intersection of blocklist[*p*] and blocklist[*p*+1] be  $\mathcal{J}_p$ . If  $A_{\mathcal{J}_p}$  is full row rank then go to step 6. Else then we append an element of blocklist[*p*] which does not include in blocklist[*p*+1] to blocklist[*p*+1]. And go to step 5.

Step 6: If p is n or more, return blocklist and exit the algorithm, else go to step 5

**Example 4.3.** n = 10, m = 3, s = 2 $\mathcal{B} = \{\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3\}$  $\mathcal{J}_1 = \{0, 1, 2, 3, 4, 5\}, \mathcal{J}_2 = \{2, 4, 5, 6, 7, 8\}, \mathcal{J}_3 = \{2, 4, 5, 9\}$  $\{2, 4, 5\}$  is common to all blocks.

#### 4.2 Block construction algorithm for Rule B

#### Block construction algorithm for Rule B

In this algorithm, we use M which is index set as a set of pre-treatment and  $\overline{M}$  which is index set as a set of post-processing.

Extracting the elements from M is corresponding to extracting the elements from union set of blocks which were created already. We choose m elements from the union set, where column vector of matrix A corresponding to the m elements is full row rank. And then we get a block by combining the m element and some elements of M. Through this algorithm, we can get the set of blocks as Figure 3.



Figure 3: Example of Algorithm 5

#### Algorithm 5 Block construction algorithm for Rule B

**Input:** *n*, *m*, *s*, *A* 

#### Output: blocklist

Step 0: M:=[list of integers from 0 to n - 1],  $\overline{M}$ :=[empty list], blocklist:= [empty list]

- Step 1 : p := 1 choose *m* elements of M, and add the*m* elements to  $\overline{M}$  and delete the *m* elements from M.
- Step 2 : If  $A_{\overline{M}}$  is full row rank, go to step 3. Else, go to step 0.
- Step 3 : tmp:= [list of m elements extracted from  $\overline{M}$  at random]
- Step 4 : If  $A_{tmp}$  is full row rank, go to step 5. Else, tmp:=[empty list], go to step 3.
- Step 5: block:= [list which is composed of both an element extracted from Ms at random and elements in tmp], append block to a blocklist. delete elements included in tmp from M, add elements included in tmp to M, go to step 6
- Step 6: If the number of M's element is 0, return blocklist and exit the algorithm. Else tmp:=[empty list], go to step 3

#### **Example 4.4.** n = 10, m = 3, s = 2M = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, $\overline{M} =$ {}

*Iteration 0*  $\{9, 7, 6\}$  from M where  $A_{\{9,7,6\}}$  is full row rank.

Updating M and  $\bar{M}$ , M = {0, 1, 2, 3, 4, 5, 8},  $\bar{M}$  = {9, 7, 6}

Iteration 1 {9,7,6} from  $\overline{\mathbb{N}}$  where  $A_{\{9,7,6\}}$  is full row rank {2,5,3} from  $\mathbb{N}$  $\mathcal{J}_1 = \{2,3,5,6,7,9\}$ Updating  $\mathbb{N}$  and  $\overline{\mathbb{N}}$ ,  $\mathbb{M} = \{0, 1, 4, 8\}, \overline{\mathbb{M}} = \{2, 3, 5, 6, 7, 9\}$ 

Iteration 2 {9, 2, 3} from  $\bar{\mathbb{N}}$  where  $A_{\{9,2,3\}}$  is full row rank {1, 8, 0} from  $\bar{\mathbb{N}}$  $\mathcal{J}_2 = \{0, 1, 2, 3, 8, 9\}$ Updating  $\mathbb{N}$  and  $\bar{\mathbb{N}}$ ,  $\mathbb{M} = \{4\}, \bar{\mathbb{N}} = \{0, 1, 2, 3, 5, 6, 7, 8, 9\}$ 

*Iteration 3* {0, 5, 9} *from*  $\bar{\mathbb{M}}$  *where*  $A_{\{0,5,9\}}$  *is full row rank* {4} *from*  $\bar{\mathbb{M}}$  $\mathcal{J}_3 = \{0, 5, 9, 4\}$ 

## **5** Numerical experience

In order to understand the behavior of the proposed algorithm, we have implemented the proposed method, and we report these numerical results.

## Problem for the numerical experiment

We solve the following problem in this section.

$$\begin{array}{ll} \min & x^\top Q x + q^\top x \\ \text{s.t.} & A x = b, \end{array}$$

where  $Q \in \mathbb{R}^{n \times n}$ ,  $q \in \mathbb{R}^n$ ,  $A \in \mathbb{R}^{m \times n}$  and  $b \in \mathbb{R}^m$ .

We generate random number as follows.  $Q: Q = PP^{\top}/n$ , where *P* is  $n \times n$  random matrix q: n-dimensional random vector  $A: m \times n$  random matrix b: m-dimensional random vector Random number is generated between 0 and 1. Initial iterrate point  $x^0 = A^{\dagger}b$ , where  $A^{\dagger}$  is pseudo-inverse matrix of *A*. we solve the following partial problem of (5) for computing the direction (11) in each iteration.

$$\min \nabla_{\mathcal{J}^k} f(x)^\top d_{\mathcal{J}^k} + \frac{1}{2} d_{\mathcal{J}^k}^\top H_{\mathcal{J}^k} d_{\mathcal{J}^k} d_{\mathcal{J}^k},$$
  
s.t.  $A_{\mathcal{J}^k} d_{\mathcal{J}^k} = 0.$ 

And KKT condition of this program is

$$\nabla_{\mathcal{J}^k} f(x^k) + H_{\mathcal{J}^k \mathcal{J}^k} d_{\mathcal{J}^k} + A_{\mathcal{J}^k}^{\mathsf{T}} \lambda = 0,$$
$$A_{\mathcal{J}^k} d_{\mathcal{J}^k} = 0.$$

Therefore we have

$$\begin{split} \lambda &= -(A_{\mathcal{J}^k} H_{\mathcal{J}^k \mathcal{J}^k}^{-1} A_{\mathcal{J}^k}^{\top})^{-1} A_{\mathcal{J}^k} H_{\mathcal{J}^k \mathcal{J}^k}^{-1} \nabla_{\mathcal{J}^k} f(x), \\ d_{\mathcal{J}^k} &= -H_{\mathcal{J}^k \mathcal{J}^k}^{-1} (\nabla_{\mathcal{J}^k} f(x) + A_{\mathcal{J}^k}^{\top} \lambda). \end{split}$$

we use

$$\alpha^{k} = \frac{x^{\top} Q_{\mathcal{J}^{k}} d_{\mathcal{J}^{k}}^{\top} + q_{\mathcal{J}^{k}}^{\top} d_{\mathcal{J}^{k}}}{d_{\mathcal{J}^{k}}^{\top} Q_{\mathcal{J}^{k}} \mathcal{J}^{k} d_{\mathcal{J}^{k}}}$$

as step-size.

#### Numerical experiment

We use Algorithm 3 and Cyclic Rule for block selection to solve the problem. We terminate iteration, if most larger ones of infinite norm of the all direction vectors in a set of block is less than  $-10^2$  in a cycle.

In the case of n = 100, we observe number of iteration and relative error for each m(number of constraint) and size of block at the time of the iteration end. Relative error is defined as follows

$$\left|\frac{f_{opt} - f_{stop}}{f_{opt}}\right|,$$

where  $f_{opt}$  is optimal value of objective function f, and  $f_{stop}$  is objective function value at the time of the iteration end.

For each n,Q,q is same and Constraint is a subset of the larger for each m.

There is numerical result in Table 1. We In case n = 100, the sequence generated from Algorithm 2 converges to the optimum solution. So the block selection enables BCGD to converge global.

However, global convergence are observed in the result of numerical experiments, there are many number of iterations. The reason of this, but this time the choice of the block was using only matrix *A* constraints by considering only the global convergence is believed to be because it did not use the information of the objective function.

The reason for this is considered that we used only matrix A for block selection for global convergence and did not use the information of objective function.

For example, if the optimal solution did to the variables included in the selected block is the same sign, this time, since the matrix of the constraint condition is positive, becomes smaller magnitude of the direction vector, the number of iterations is increased as a result It was it is conceivable.

n = 100					
m	blocksize	number of iteration	relative error		
	10	111402	2.41E - 03		
0	15	115827	1.93E - 03		
	20	94813	1.16E - 03		
	10	1060470	4.57E - 02		
1	15	377397	1.88E - 03		
	20	954693	7.61E - 03		
	10	367644	4.82E - 02		
2	15	358421	2.61E - 02		
	20	821024	1.59E - 02		
	10	563852	7.41E - 02		
3	15	458053	8.70 <i>E</i> – 03		
	20	267603	2.88E - 03		
	10	356424	4.17E - 02		
4	15	159460	1.12E - 02		
	20	234522	2.75E - 03		

Table 1: Numerical result

## 6 Conclusion

In this paper, we proposed new block selection techniques in the BCGD method for optimization problems with linear equality constraints. The techniques incorporate both low computational cost and the global convergence. Moreover, we showed that the BCGD method with them converge to the KKT point globally.

In this paper we only considered the global convergence and ignored rapid convergence. From numerical results we see that the BGCD needed many iterations to get a solution. This is because the proposed selection rules, **Rule A** and **Rule B**, exploit only *A*, and it ignores the nature of the objective function. It is worth to improve the block selection rules by exploiting the objective function. Moreover, it is important to extend the proposed methods for optimization problems with both linearly equality constraint and box constraints.

## Acknowledgments

When I did this study, Professor Yamashita guided me very kindly and precisely. Without him, I cannot complete this study. I cannot thank you enough. I am grateful to Assistant Professor Ellen Hidemi Fukuda as well. Thank you. And I am grateful to classmates in my laboratory Mr. Tsuyuguchi, Mr. Togari, Mr.Hama and Ms. Morishita, who encourage each other and and laboratory member as well. I am glad that I can this study in the same laboratory with you all. I met with the teachers and everyone and did this study. It is a treasure in my life. Thank you.

## References

- [1] D. P. Bertsekas: Nonlinear programming, Athena Scientific, 1999.
- [2] K. W. Chang, C. J. Hsieh, C. J. Lin: Coordinate descent method for large-scale l2-loss linear support vector machines, *The Journal of Machine Learning Research*, vol.9, pp.1369–1398, 2008.
- [3] R. Fletcher: Practical Methods of Optimization, John Wiley & Sons, 1987.
- [4] R. Fletcher: *An overview of unconstrained optimization*, Springer Netherlands, pp.109–143 1994.
- [5] J. Nocedal, S. J. Wright: *Numerical optimization*, Springer Science & Business Media, 1999.
- [6] S. Sonnenburg, G. Rätsch, C. Schfer, B. Schölkopf: Large scale multiple kernel learning, *The Journal of Machine Learning Research*, vol.7, pp.1531–1565, 2006.
- [7] P. Tseng, S. Yun: A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training, *Computational Optimization and Applications*, vol.47(2), pp.179–206, 2010.
- [8] P. Tseng, S. Yun: A coordinate gradient descent method for nonsmooth separable minimization, *Mathematical Programming*, vol.117(1-2), pp.387–423, 2009.