The Cyclic Block Conditional Gradient Method for Convex Optimization Problems

Amir Beck* Edouard Pauwels* Shoham Sabach*

February 12, 2015

Abstract

In this paper we study the convex problem of optimizing the sum of a smooth function and a compactly supported non-smooth term with a specific separable form. We analyze the block version of the generalized conditional gradient method when the blocks are chosen in a cyclic order. A global sublinear rate of convergence is established for two different stepsize strategies commonly used in this class of methods. Numerical comparisons of the proposed method to both the classical conditional gradient algorithm and its random block version demonstrate the effectiveness of the cyclic block update rule.

Keywords: Conditional gradient, cyclic block decomposition, iteration complexity, linear oracle, nonsmooth convex minimization, support vector machine.

1 Introduction

With the growth of size of problems commonly encountered in many applied fields, there is a strong demand for numerical methods featuring low computational cost iterative schemes. By low computational cost, we mean algorithms which require at most matrix by vector multiplication (inversion of matrices are, for example, too expensive). In this context, it is necessary to propose and analyze numerical schemes that

- are based on computationally efficient steps;
- exploit problem structure and data information;
- enjoy global convergence properties and iteration complexity estimates.

We consider structured convex problems consisting of minimizing the sum of two terms: a smooth term, which is a composition of a smooth function and a linear mapping and a nonsmooth separable term. We focus on programs for which the geometry of the non-smooth part exhibits such a degree of complexity that proximal-based methods [5, 6, 24] do not constitute a viable alternative. Indeed, the efficiency of these methods considerably deteriorates in situations that do not fit in a “favorable geometric settings”, see [11, 16] for a more detailed description of this concept. Algorithms based on linear oracles, such as the conditional gradient method (also known as the Frank-Wolfe algorithm) [14, 21, 12, 13, 17] and their extensions to structured composite problems [8, 9, 1], or block separable problems [19], are based on the principle of iteratively solving linearized subproblems. In settings where computing the proximal operator is too expensive, this approach has proven to be competitive. Successful examples of applications which benefit from this approach include trace-norm constrained

*Faculty of Industrial Engineering and Management, Technion, Haifa, Israel (becka, epauwels, ssbach}@ie.technion.ac.il).
or penalized problems [18, 11] and structured multiclass classification with extremely large number of classes [19].

Continuously increasing problem dimensions have motivated the principle of taking advantage of available block structure of the problem at hand. This led to the development of variable decomposition methods which break down the original large-scale problem into several much smaller subproblems that could be solved efficiently.

In recent works, [23] and [26] analyzed the average case iteration complexity of block versions of gradient, projected gradient and forward-backward methods where at each iteration, the block to be updated is selected at random. We refer to such a block selection rule as the random update rule. In the context of linear oracles, [19] applied this approach to the conditional gradient method focusing on implementing it for the structural Support Vector Machine (SVM) training problem (see more details in Section 5.3). Analysis of algorithms involving random update rules typically provides average case complexity results. A different kind of works consider updating the blocks in a cyclic order. We refer to such a deterministic update rule as the cyclic update rule. In this context, [22] provides an asymptotic analysis of exact coordinate minimization for composite strongly convex problems. In [6], a global rate of convergence result was established for the cyclic block coordinate gradient projection method for convex problems over feasible sets with a separable structure. For this line of works, the estimates given for the the cyclic update rule deterministically hold for the sequence of function values. As we already mentioned, this is not the case for the random update rule for which only average case estimates are available.

Another relevant feature of block decomposition methods is the fact that they usually allow to take a substantially larger step at each iteration compared to their classical non-block counterparts. This fact potentially gives a numerical advantage to block decomposition algorithms compared to classical variants resulting in a speedup in convergence, see [6].

In this work, we propose a cyclic block version of the generalized conditional gradient method [8, 9, 1] which for ease of reference is called Cyclic Block Conditional Gradient (CBCG). We provide a sublinear global convergence rate estimates for a the predefined stepsize strategy [13, 17] and an adaptive stepsize strategy [21]. We also establish rate estimates for an optimality measure in the spirit of [17, 1]. We numerically compare the proposed method to its random update rule counterpart, that is, the Random Block Conditional Gradient (RBCG), which was proposed and analyzed in [19]. Extensive simulations on a large number of synthetic examples suggest that CBCG compares favorably to both RBCG and the classical conditional gradient algorithm (CG). Finally, we also compare CBCG and RBCG on the problem of training the structural SVM [28, 29] for the optical character recognition task (OCR) originally proposed in [28].

The next section is dedicated to the presentation of the model and main assumptions (Section 2.1) and the description of the CBCG algorithm (Section 2.2). Section 3 presents few auxiliary results for an optimality measure which is commonly encountered when using methods which are based on linear oracles. In Section 4 we present our theoretical findings about the rate of convergence results of the CBCG method. We split this section into two subsections which deal with the two different stepsize strategies that we analyze in this paper (Section 4.1 for the predefined stepsize and Section 4.2 for the adaptive stepsize). We conclude Section 4 with a discussion on a backtracking version of the CBCG method (see Section 4.3). Numerical experiments on synthetic data and on the structural SVM training problem are presented in Section 5.

Conventions. Throughout the paper the underlying vector space is the $n$-dimensional Euclidean space $\mathbb{R}^n$ with the $l_2$-norm which is denoted by $\|\cdot\|$. This notation is also used for the matrix norm, which is assumed to be the spectral norm. We will consider the partition of an arbitrary vector $x \in \mathbb{R}^n$ into $N$ blocks where each block consists of a subset of the $n$ coordinates. The size of each block (the number of coordinates) is given by the integer $n_i$ for $i = 1, 2, \ldots, N$, such that $\sum_{i=1}^{N} n_i = n$. 

The $i$th block of a vector $x \in \mathbb{R}^n$ is denoted by $x_i$. We assume that $x \in \mathbb{R}^n$ can be written as follows

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}.$$ 

For any $i = 1, 2, \ldots, N$ we define the matrix $U_i \in \mathbb{R}^{n_i \times n_i}$ as the sub-matrix of the $n \times n$ identity matrix consisting of the columns corresponding to the $i$th block. Thus, in particular,

$$(U_1, U_2, \ldots, U_N) = I_n.$$ 

It is clear that using these notations, we have for any $x \in \mathbb{R}^n$ that

$$x_i = U_i^T x \quad \text{and} \quad x = \sum_{i=1}^N U_i x_i.$$ 

Finally, for any subset $S$ of $\mathbb{R}^n$, $\delta_S$ denotes the indicator function of $S$ which takes the value 0 on $S$ and $+\infty$ otherwise.

## 2 The Optimization Model and Algorithm

### 2.1 Problem Formulation and Assumptions

We consider the following optimization model

$$\min_{x \in \mathbb{R}^n} \left\{ H(x) \equiv F(Ax) + \sum_{i=1}^N g_i(x_i) \right\}, \quad (2.1)$$

where $A \in \mathbb{R}^{m \times n}$. We make the following standing assumption on model $(2.1)$.

**Assumption 1.**

(i) $g_i : \mathbb{R}^{n_i} \to (-\infty, \infty]$, $i = 1, 2, \ldots, N$, is a proper, lower semicontinuous and convex function which satisfies

- $X_i \equiv \text{dom} \, g_i \subseteq \mathbb{R}^{n_i}$ is a compact set with diameter $D_i$, that is,
  $$\sup_{x_i, y_i \in X_i} \|x_i - y_i\| = D_i.$$

- $g_i$ is globally Lipschitz on $X_i$ with constant $l_i$, that is,
  $$|g_i(x_i) - g_i(y_i)| \leq l_i \|x_i - y_i\|, \quad \forall \, x_i, y_i \in X_i. \tag{2.2}$$

(ii) $F : \mathbb{R}^m \to (-\infty, \infty]$ is convex and continuously differentiable over $A (X_1 \times X_2 \times \cdots \times X_N) \subseteq \mathbb{R}^m$ and has Lipschitz continuous gradient with constant $L_F$, that is,

$$\|\nabla F(x) - \nabla F(y)\| \leq L_F \|x - y\|, \quad \forall \, x, y \in A (X_1 \times X_2 \times \cdots \times X_N).$$

Since $g_i$ is assumed to be convex, it immediately follows that the domain $X_i$ is a convex subset of $\mathbb{R}^{n_i}$ for $i = 1, 2, \ldots, N$. We set $g(x) \equiv \sum_{i=1}^N g_i(x_i)$ and $f(x) \equiv F(Ax)$. The domain of $g$ is denoted by $\text{dom} \, g \equiv X$ and its diameter is denoted by $D$. Using these simplified notations, problem $(2.1)$ actually consists of minimizing $f + g$. Furthermore, we have $X \equiv X_1 \times X_2 \times \cdots \times X_N$ and therefore, it holds that

$$D^2 = \sum_{i=1}^N D_i^2. \tag{2.3}$$
Remark 2.1. By setting \( g(\cdot) = \delta_X(\cdot) \), we recover the constrained optimization model that motivated the development of the traditional conditional gradient method (see [17] and references therein). This is also the case when we add a linear term to the indicator.

Under Assumption 1, problem (2.1) is guaranteed to attain its optimal value, therefore the optimal set, which is denoted by \( X^* \), is nonempty and the corresponding optimal value is denoted by \( H^* \in \mathbb{R} \).

For each block \( i \in \{1, 2, \ldots, N\} \), we employ the following notation:

- \( A_i \equiv AU_i \), so that \( A = (A_1, A_2, \ldots, A_N) \);
- \( \nabla_i f(x) \equiv U_i^T \nabla f(x) \) denotes the partial gradient of \( f \).

Using the previous notations, we have that \( \nabla_i f(x) = A_i^T \nabla F(Ax) \). We will use a refined notion of Lipschitz continuity that fits our block separable composite setting. This is expressed by the following standing assumption.

Assumption 2. For each \( i = 1, 2, \ldots, N \), there exists a constant \( \beta_i > 0 \), such that for any \( x \in X \) and any \( h_i \in \mathbb{R}^{n_i} \) satisfying \( x + U_i h_i \in X \), it holds that

\[
\|\nabla F(Ax + A_i h_i) - \nabla F(Ax)\| \leq \beta_i \|A_i h_i\|.
\]

Assumption 2 can be seen as a consequence of Assumption 1. Indeed, it is always possible to set \( \beta_i = L_F, i = 1, 2, \ldots, N \). However, adopting this more refined convention provides additional algorithmic flexibility which allows to take advantage of conditioning disparities between different blocks. See for example Section 5.1, where it is shown how this approach allows the usage of exact line search for quadratic problems. We also note that when \( A = I \), then \( \beta_i \) can be chosen to be the \( i \)th block Lipschitz constant of the gradient of \( F \) (see e.g., [6, 2]), which is always a quantity smaller than \( L_F \). We define the following quantity

\[
\beta_{\text{min}} \equiv \min \{\beta_1, \beta_2, \ldots, \beta_N\} > 0.
\]

The following important result will play a central role in the forthcoming analysis. The proof is almost identical to the well known proof of the descent lemma (see [7]), and is thus given in the appendix.

Lemma 2.2 (Composite block descent lemma). Let \( i \in \{1, 2, \ldots, N\} \), then for any \( x \in X \) and \( h_i \in \mathbb{R}^{n_i} \) such that \( x + U_i h_i \in X \), we have

\[
f(x + U_i h_i) \leq f(x) + \langle \nabla_i f(x), h_i \rangle + \frac{\beta_i}{2} \|A_i h_i\|^2.
\]

Proof. See Appendix A

2.2 The Cyclic Block Conditional Gradient Method

The generalized conditional gradient method [8, 9, 1] can be applied to problem (2.1) when the corresponding linear oracle is available. That is, we assume that for any \( x \in X \), the solution to the following problem can be easily computed:

\[
\min_{v \in X} \{\langle \nabla f(x), v \rangle + g(v)\}.
\]

We exploit here the separability of the function \( g \) (see Section 2.1) and propose a block decomposition extension of the generalized conditional gradient method which we call the Cyclic Block
Conditional Gradient method (CBCG). Before stating the algorithm, we will need the following additional notation. Let \( \{x^k\}_{k \in \mathbb{N}} \) be a given sequence, then for any \( i = 1, 2, \ldots, N \), we define

\[
x^{k,i} = \begin{pmatrix} x_{k+1}^1 \\ \vdots \\ x_{k+1}^i \\ x_k^i \\ \vdots \\ x_N^k \end{pmatrix}.
\] (2.6)

That is, the first \( i \) blocks in \( x^{k,i} \) are those of \( x^{k+1} \) and the remaining \( N - i \) blocks are those of \( x^k \). It is clear that using this notation we have \( x^{k,0} = x^k \) and \( x^{k+1} = x^{k,N} \). The algorithm is given now.

**CBCG: Cyclic Block Conditional Gradient**

**Initialization.** \( x^0 \in X \) and \( \alpha_k^i \in [0, 1] \) for all \( k \in \mathbb{N} \) and \( i = 1, 2, \ldots, N \).

**General Step.** For \( k = 1, 2, \ldots, \)

\[\begin{align*}
(1) \text{ For any } i = 1, 2, \ldots, N, \text{ compute } & \quad p_k^i \in \arg\min_{p_i \in X_i} \left\{ \langle \nabla_i f(x^{k,i-1}), p_i \rangle + g_i(p_i) \right\}, \quad (2.7) \\
& \text{ and then } \quad x^{k,i} = x^{k,i-1} + \alpha_k^i U_i(p_k^i - x^{k,i-1}_i). \quad (2.8)
\end{align*}\]

\( (2) \) Update \( x^{k+1} = x^{k,N} \).

We will first analyze, in Section 4.1, the convergence rate of the CBCG method using a predefined stepsize [13]. Here, the predefined stepsize that we use is given, for any \( i = 1, 2, \ldots, N \), by

\[
\alpha_k^i = \alpha_k = \frac{2}{k + 2}.
\]

In Section 4.2, we will consider an adaptive stepsize rule [21], which is determined by the minimization of the quadratic upper bound of \( H \) related to (2.5). The expression of this stepsize will be made precise below (see (4.12)).

## 3 The Optimality Measure

In this section, we describe a few properties of an optimality measure together with its block counterparts. This measure is typical when discussing methods which are based on linear oracles and usually plays a crucial role in the convergence analysis, see [4] for an overview and [19, 1] for a link with Fenchel duality. For any \( x \in X \), we define the following quantity

\[ p(x) \in \arg\min_{p \in X} \left\{ \langle \nabla f(x), p \rangle + g(p) \right\}, \quad (3.1) \]

as well as the optimality measure

\[ S(x) \equiv \max_{p \in X} \left\{ \langle \nabla f(x), x - p \rangle + g(x) - g(p) \right\} = \langle \nabla f(x), x - p(x) \rangle + g(x) - g(p(x)), \quad (3.2) \]

where the last equality follows from (3.1). The function \( S \) is an optimality measure in the sense that it is non-negative on \( X \) and it is zero only on \( X^* \). Furthermore, for any \( x \in X \), the quantity \( S(x) \) is an upper bound on \( H(x) - H^* \) as stated in the following lemma which short proof is given for the sake of completeness.
Lemma 3.1. $S(x) \geq H(x) - H^*$.  

Proof. For any $x^* \in X^*$ we have

$$S(x) = \langle \nabla f(x), x - p(x) \rangle + g(x) - g(p(x))$$

$$= \langle \nabla f(x), x \rangle + g(x) - [\langle \nabla f(x), p(x) \rangle + g(p(x))]$$

$$\geq \langle \nabla f(x), x \rangle + g(x) - [\langle \nabla f(x), x^* \rangle + g(x^*)]$$

$$= \langle \nabla f(x), x - x^* \rangle + g(x) - g(x^*),$$

where the inequality follows from (3.1). Using the convexity of $f$, we obtain that

$$S(x) \geq \langle \nabla f(x), x - x^* \rangle + g(x) - g(x^*) \geq f(x) - f(x^*) + g(x) - g(x^*) = H(x) - H^*.$$

This proves the desired result. \hfill \Box

We refine the notations introduced in (3.1) and (3.2) in order to fit our block structured setting. For any $x \in X$ and any $i = 1, 2, \ldots, N$, we set

$$p_i(x) \in \text{argmin}_{p_i \in X_i} \{ \langle \nabla f_i(x), p_i \rangle + g_i(p_i) \},$$

and define the block optimality measure

$$S_i(x) \equiv \max_{p_i \in X_i} \{ \langle \nabla f_i(x), x_i - p_i \rangle + g_i(x_i) - g_i(p_i) \}. $$

It is clear that in this case it is also true that

$$S_i(x) = \langle \nabla f_i(x), x_i - p_i(x) \rangle + g_i(x_i) - g_i(p_i(x)).$$

Using the separability of both $g$ and $X$, we have for any $x \in X$ that

$$S(x) = \sum_{i=1}^N S_i(x).$$

There might be multiple optimal solutions for problem (3.1) and also for (3.3). Our only assumption is that the choices of $p_1(x), \ldots, p_N(x)$ and $p(x)$ are made under the restriction that

$$p(x) = \begin{pmatrix} p_1(x) \\ p_2(x) \\ \vdots \\ p_N(x) \end{pmatrix}.$$

The following Lipschitz-type property of the block optimality measure $S_i$, $i = 1, 2, \ldots, N$, will be crucial in the forthcoming analysis.

Lemma 3.2. Let $x, y \in X$ be two vectors which satisfy $x_i = y_i$ for some $i = 1, 2, \ldots, N$. Then the following inequality holds

$$|S_i(x) - S_i(y)| \leq L_F D_i \| A_i \| \cdot \| A(x - y) \|.$$

Proof. From (3.5) we have

$$S_i(x) = \langle \nabla f_i(x), x_i - p_i(x) \rangle + g_i(x_i) - g_i(p_i(x))$$

$$= \langle \nabla f_i(y), x_i - p_i(x) \rangle + g_i(x_i) - g_i(p_i(x)) + \langle \nabla f_i(x) - \nabla f_i(y), x_i - p_i(x) \rangle.$$
Now, using the fact that \( f(x) \equiv F(Ax) \) and Assumption 1(ii), we obtain
\[
\langle \nabla f_i(x) - \nabla f_i(y), x_i - p_i(x) \rangle = \langle A_i^T (\nabla F(Ax) - \nabla F(Ay)), x_i - p_i(x) \rangle \\
= \langle \nabla F(Ax) - \nabla F(Ay), A_i(x_i - p_i(x)) \rangle \\
\leq \| \nabla F(Ax) - \nabla F(Ay) \| \cdot \| A_i(x_i - p_i(x)) \| \\
\leq L_F \| A(x - y) \| \cdot \| A_i(x_i - p_i(x)) \| \\
\leq L_F D_i \| A_i \| \cdot \| A(x - y) \|, 
\]
where the first inequality follows from the Cauchy-Schwarz inequality and the last inequality follows from the fact that both \( x_i \) and \( p_i(x) \) belong to \( X_i \) (see Assumption 1(i)). Finally, by combining (3.7) with (3.8), and using the fact that \( x_i = y_i \), we obtain that
\[
S_i(x) \leq \langle \nabla f_i(y), x_i - p_i(x) \rangle + g_i(x_i) - g_i(p_i(x)) + L_F D_i \| A_i \| \cdot \| A(x - y) \| \\
= \langle \nabla f_i(y), y_i - p_i(x) \rangle + g_i(y_i) - g_i(p_i(x)) + L_F D_i \| A_i \| \cdot \| A(x - y) \| \\
\leq S_i(y) + L_F D_i \| A_i \| \cdot \| A(x - y) \|, 
\]
where the last inequality follows from the definition of \( S_i \) (see (3.4)). Changing the roles of \( x \) and \( y \), we also obtain that
\[
S_i(y) \leq S_i(x) + L_F D_i \| A_i \| \cdot \| A(x - y) \|, 
\]
which along with (3.9) yields the desired result.

\[ \square \]

## 4 Convergence Analysis of the CBCG Method

This section is devoted to the convergence analysis of the CBCG algorithm. We will first prove, in Section 4.1, a sublinear convergence rate for the variant with the predefined stepsize rule. A similar rate of convergence will be then established, in Section 4.2, for the variant with the adaptive stepsize rule. Finally, we describe a backtracking procedure in Section 4.3 which allows to use the CBCG method when the constants given in Assumption 2 are unknown. We begin with an extension of Lemma 2.2 that holds for any choice of stepsize.

**Lemma 4.1.** Let \( \{x_k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method. Then for any \( k \geq 0 \) and \( i \in \{1, 2, \ldots, N\} \), we have
\[
H(x^{k,i}) \leq H(x^{k,i-1}) - \alpha_i^k S_i(x^{k,i-1}) + \frac{(\alpha_i^k)^2 \beta_i}{2} \| A_i(p_i^k - x_i^{k,i}) \|^2. 
\]

**Proof.** First, by the definition of the main step of the CBCG method (see (2.8)), we have
\[
H(x^{k,i}) = f(x^{k,i}) + g(x^{k,i}) \\
= f(x^{k,i-1} + \alpha_i^k U_i(p_i^k - x_i^{k,i-1})) + g(x^{k,i-1} + \alpha_i^k U_i(p_i^k - x_i^{k,i-1})). 
\]

We can now use Lemma 2.2 to obtain
\[
H(x^{k,i}) \leq f(x^{k,i-1}) + \alpha_i^k \langle \nabla f(x^{k,i-1}), p_i^k - x_i^{k,i-1} \rangle \\
+ \frac{(\alpha_i^k)^2 \beta_i}{2} \| A_i(p_i^k - x_i^{k,i-1}) \|^2 + g(x^{k,i-1} + \alpha_i^k U_i(p_i^k - x_i^{k,i-1})). 
\]

The last term can be bounded from above as follows:
\[
g(x^{k,i-1} + \alpha_i^k U_i(p_i^k - x_i^{k,i-1})) = \sum_{j=1,j\neq i}^N g_j(x_j^{k,i-1} + \alpha_i^k x_i^{k,i-1} + \alpha_i^k p_i^k) \\
\leq \sum_{j=1,j\neq i}^N g_j(x_j^{k,i-1} + (1 - \alpha_i^k) x_i^{k,i-1} + \alpha_i^k p_i^k) \\
= g(x^{k,i-1}) + \alpha_i^k (g_i(p_i^k) - g_i(x_i^{k,i-1})). 
\]
where the inequality follows from the convexity of each $g_i$, $i = 1, 2, \ldots, N$. Now, combining (4.1) with (4.2) and using the definition of $S_i$ (see (3.5)), we obtain

\[
H(x^{k,i}) \leq f(x^{k,i-1}) + g(x^{k,i-1}) - \alpha_i^k(\nabla_i f(x^{k,i-1}), x^{k,i-1} - p_i^k) + g_i(x^{k,i-1} - g_i(p_i^k)) \\
+ \frac{(\alpha_i^k)^2 \beta_i}{2} \|A_i(p_i^k - x^{k,i-1})\|^2 \\
= H(x^{k,i-1}) - \alpha_i^k S_i(x^{k,i-1}) + \frac{(\alpha_i^k)^2 \beta_i}{2} \|A_i(p_i^k - x^{k,i-1})\|^2 \\
= H(x^{k,i-1}) - \alpha_i^k S_i(x^{k,i-1}) + \frac{(\alpha_i^k)^2 \beta_i}{2} \|A_i(p_i^k - x^k)\|^2,
\]

where the last inequality follows from the fact that $x^k_i = x^{k,i-1}$ (see (2.6)). This proves the desired result.

We also need an explicit expression of the distance between two consecutive iterates generated by the CBCG method. The following lemma also holds for any choice of stepsize.

**Lemma 4.2.** Let $\{x^k\}_{k \in \mathbb{N}}$ be the sequence generated by the CBCG method. Then for any $k \in \mathbb{N}$, we have

\[
\|x^{k+1} - x^k\|^2 = \sum_{j=1}^{N} (\alpha_j^k)^2 \|p_j^k - x^{k,j-1}\|^2.
\]

Furthermore, for any $i = 0, 1, \ldots, N$, we have

\[
\|x^{k,i} - x^k\|^2 \leq \|x^{k+1} - x^k\|^2.
\]

**Proof.** For any fixed $k \in \mathbb{N}$, we have from the definition of the iterates of the CBCG algorithm (see (2.8)) and (2.6) that

\[
\|x^{k+1} - x^k\|^2 = \sum_{j=1}^{N} \|x^{k+1}_j - x^k\|^2 = \sum_{j=1}^{N} (\alpha_j^k)^2 \|p_j^k - x^{k+1,j-1}\|^2,
\]

which proves the first statement. Furthermore, since for any $i = 0, 1, \ldots, N$, $x^{k,i}_j = x^k_j$ for all $j > i$ and $x^{k,i}_j = x^{k,N}_j$ for all $j \leq i$, we have

\[
\|x^{k,i} - x^k\|^2 = \sum_{j=1}^{N} \|x^{k,i}_j - x^k\|^2 = \sum_{j=1}^{i} \|x^{k,i}_j - x^k\|^2 = \sum_{j=1}^{i} \|x^{k,N}_j - x^k_j\|^2 \\
\leq \sum_{j=1}^{i} \|x^{k,N}_j - x^k_j\|^2 = \|x^{k+1} - x^k\|^2,
\]

which proves the second statement and the proof is complete.

### 4.1 Analysis for the Predefined Stepsize Strategy

In this section, we analyze the CBCG algorithm with the stepsize defined as follows:

\[
\alpha_i^k = \alpha^k \equiv \frac{2}{k+2}.
\]  

(4.3)

The main step towards the proof of convergence rate in this case is recorded in the following lemma.
Lemma 4.3. Let \( \{x_k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the stepsize given in (4.3). Then for any \( k \geq 0 \), we have

\[
H(x^{k+1}) \leq H(x^k) - \alpha^k S(x^k) + \frac{C_1}{2}(\alpha^k)^2,
\]

where

\[
C_1 = \sum_{i=1}^{N} \beta_i \|A_i\|^2 D_i^2 + 2L_F D \|A\| \sum_{i=1}^{N} D_i \|A_i\|.
\]

Proof. For any \( k \geq 0 \) and each \( i \in \{1, 2, \ldots, N\} \), we have from Lemma 4.1 that

\[
H(x^{k,i}) \leq H(x^{k,i-1}) - \alpha^k S_i(x^{k,i-1}) + \left(\frac{\alpha^k}{2}\right)^2 \|A_i(p_i^{k} - x_i^{k})\|^2.
\]

Summing (4.5) for \( i = 1, 2, \ldots, N \) yields

\[
H(x^{k+1}) = H(x^{k,N})
\]

\[
\leq H(x^{k,0}) - \alpha^k \sum_{i=1}^{N} S_i(x^{k,i-1}) + \frac{(\alpha^k)^2}{2} \sum_{i=1}^{N} (\beta_i \|A_i\|^2 D_i^2)
\]

\[
= H(x^k) - \alpha^k \sum_{i=1}^{N} S_i(x^k) + \frac{(\alpha^k)^2}{2} \sum_{i=1}^{N} (\beta_i \|A_i\|^2 D_i^2) + \alpha^k \sum_{i=1}^{N} (S_i(x^k) - S_i(x^{k,i-1}))
\]

\[
= H(x^k) - \alpha^k S(x^k) + \frac{(\alpha^k)^2}{2} \sum_{i=1}^{N} (\beta_i \|A_i\|^2 D_i^2) + \alpha^k \sum_{i=1}^{N} (S_i(x^k) - S_i(x^{k,i-1})),
\]

where the last equality follows from (3.6). Using Lemma 3.2, and the fact that \( x_i^{k,i-1} = x_i^k \) (see (2.6)) gives, for any \( i = 1, 2, \ldots, N \), that

\[
S_i(x^k) - S_i(x^{k,i-1}) \leq L_F D_i \|A_i\| \cdot \|A(x^k - x^{k,i-1})\|.
\]

Combining (4.6) and (4.7) yields

\[
H(x^{k+1}) \leq H(x^k) - \alpha^k S(x^k) + \frac{(\alpha^k)^2}{2} \sum_{i=1}^{N} (\beta_i \|A_i\|^2 D_i^2) + \alpha^k L_F \sum_{i=1}^{N} D_i \|A_i\| \cdot \|A(x^k - x^{k,i-1})\|.
\]

From Lemma 4.2, for any \( i = 1, 2, \ldots, N \), we have

\[
\|A(x^k - x^{k,i-1})\|^2 \leq \|A\|^2 \cdot \|x^k - x^{k,i-1}\|^2 \leq \|A\|^2 \cdot \|x^k - x^{k+1}\|^2
\]

\[
= (\alpha^k)^2 \|A\|^2 \sum_{j=1}^{N} \|p_j^k - x_j^k\|^2 \leq (\alpha^k)^2 \|A\|^2 \sum_{j=1}^{N} D_j^2
\]

\[
= (\alpha^k)^2 \|A\|^2 D^2.
\]

Combining (4.8) and (4.9) yields

\[
H(x^{k+1}) \leq H(x^k) - \alpha^k S(x^k) + \frac{(\alpha^k)^2}{2} \left[ \sum_{i=1}^{N} \beta_i \|A_i\|^2 D_i^2 + 2L_F D \|A\| \sum_{i=1}^{N} D_i \|A_i\| \right],
\]

which proves the desired result.
We now recall the following technical lemma on sequences of non-negative numbers (cf. [1, Lemma 1]).

**Lemma 4.4.** Let \( \{a_k\}_{k \in \mathbb{N}} \) and \( \{b_k\}_{k \in \mathbb{N}} \) be two sequences satisfying, for any \( k \geq 0 \), that
\[
a_{k+1} \leq a_k - t_kb_k + \frac{C}{2}t_k^2,
\]
where \( 0 \leq a_k \leq b_k \), \( t_k = 2/(k+2) \) and \( C > 0 \), then
(a) \( a_k \leq \frac{2C}{k+1} \);
(b) For any \( n \geq 1 \), we have
\[
\min_{k=\lceil n/2 \rceil,\ldots,n} b_k \leq \frac{8C}{n}.
\]

We are now ready to establish the sublinear rate of convergence of the CBCG algorithm with predefined stepsize.

**Theorem 4.5** (Sublinear rate for CBCG with predefined stepsizes). Let \( \{x^k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the predefined stepsize strategy given in (4.3). Then for any \( k \geq 0 \), we have
\[
H(x^k) - H^* \leq \frac{2C_1}{k+1},
\]
and, for any \( n \geq 1 \), we have
\[
\min_{k=\lceil n/2 \rceil,\ldots,n} S(x^k) \leq \frac{8C_1}{n},
\]
where
\[
C_1 = \sum_{i=1}^N \beta_i \|A_i\|^2 D_i^2 + 2L_F D \|A\| \sum_{i=1}^N D_i \|A_i\|.
\]

**Proof.** From Lemma 4.3 we have
\[
H(x^{k+1}) - H^* \leq H(x^k) - H^* - \alpha_k S(x^k) + \frac{C_1}{2} (\alpha_k^2).
\]
In addition, by Lemma 3.1, we have that \( S(x^k) \geq H(x^k) - H^* \). We can therefore invoke Lemma 4.4 with \( a_k = H(x^k) - H^* \) and \( b_k = S(x^k) \), and the desired result is established.

### 4.2 Analysis for the Adaptive Stepsize Strategy

We now turn to the analysis of CBCG with adaptive stepsize. The main insight is to choose a stepsize that minimizes the upper bound given in Lemma 4.1. In this setting, the adaptive stepsize is defined as follows:

\[
\alpha_i^k = \arg\min_{\alpha \in [0,1]} \left\{ -\alpha S_i(x^{k,i-1}) + \alpha^2 \cdot \frac{\beta_i}{2} \|A_i(p_i^k - x_i^k)\|^2 \right\}
= \min \left\{ \frac{S_i(x^{k,i-1})}{\beta_i \|A_i(p_i^k - x_i^k)\|^2}, 1 \right\}.
\]

Note that by Lemma 4.1, this choice of stepsize leads to a decrease of objective value at each step. The analysis employed for the predefined stepsize does not seem to be easily adjusted to the adaptive stepsize strategy. Thus, a different analysis is developed in this section. We begin with the following technical result.
Lemma 4.6. Let \( \{x^k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any \( k \geq 0 \) and \( i \in \{1, 2, \ldots, N\} \) we have

\[
H(x^{k,i-1}) - H(x^{k,i}) \geq \frac{\alpha_i^k}{2} S_i(x^{k,i-1}) - \frac{\alpha_i^k \beta_i}{2} \|A_i(p_i^k - x_i^k)\|^2.
\]

Proof. We split the proof into two cases. First, if \( \alpha_i^k = 1 \), and by Lemma 4.1 we have

\[
H(x^{k,i-1}) - H(x^{k,i}) \geq \frac{\alpha_i^k}{2} S_i(x^{k,i-1}) - \frac{\alpha_i^k \beta_i}{2} \|A_i(p_i^k - x_i^k)\|^2.
\]

The result now follows by combining (4.15) and (4.16).

We can now prove the following important result.

Lemma 4.7. Let \( \{x^k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any \( k \geq 0 \) and \( i \in \{1, 2, \ldots, N\} \), we have

\[
H(x^{k,i-1}) - H(x^{k,i}) \geq \frac{S_i(x^{k,i-1})^2}{2 \max \{\beta_i \|A_i\|^2 D_i^2, K_i\}},
\]

where, for each \( i = 1, 2, \ldots, N \),

\[
M_i = \max_{x \in X} \|\nabla_i f(x)\|, \quad K_i = (M_i + l_i) D_i.
\]

Proof. We again split the proof into two cases. First, if \( \alpha_i^k = 1 \), then by Lemma 4.6 we have

\[
H(x^{k,i-1}) - H(x^{k,i}) \geq \frac{1}{2} S_i(x^{k,i-1}) - \frac{S_i(x^{k,i-1})^2}{2K_i}.
\]
where the last inequality follows from the fact that for any $x \in X$:

$$
S_i(x) = \langle \nabla_i f(x), x_i - p_i(x) \rangle + g_i(x_i) - g_i(p_i(x)) \\
\leq \|\nabla_i f(x)\| \cdot \|x_i - p_i(x)\| + l_i \|x_i - p_i(x)\| \\
\leq (M_i + l_i)D_i.
$$

(4.20)

In the second case, when $\alpha_i^k < 1$, we have that $\alpha_i^k = \frac{S_i(x_{i-1}^k)}{\|A_i(p_i^k - x_i^k)\|^2}$, and by Lemma 4.6 we can write

$$
H(x_{i-1}^k) - H(x_i^k) \geq \frac{S_i(x_{i-1}^k)^2}{2\beta_i \|A_i(p_i^k - x_i^k)\|^2} \geq \frac{S_i(x_{i-1}^k)^2}{2\beta_i \|A_i\|^2 D_i^2},
$$

(4.21)

The result now follows by combining (4.20) and (4.21).

Proof. By Lemma 4.6 we have

$$
H(x_{i-1}^k) - H(x_i^k) \geq \frac{(\alpha_i^k)^2 \beta_i}{2} \|A_i(p_i^k - x_i^k)\|^2.
$$

(4.22)

We will now prove an additional technical lemma that establishes a “sufficient decrease” property for the CBCG method with adaptive stepsize.

Lemma 4.8. Let $\{x_i\}_{i \in \mathbb{N}}$ be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any $k \geq 0$ and $i \in \{1, 2, \ldots, N\}$, we have

$$
H(x_k) - H(x_{k+1}) \geq \frac{\beta_{\text{min}}}{2N} \|A_i(x_k - x_{k,i-1})\|^2,
$$

where $\beta_{\text{min}}$ is given by (2.4).

Proof. By Lemma 4.6 we have

$$
H(x_{i-1}^k) - H(x_i^k) \geq \frac{(\alpha_i^k)^2 \beta_i}{2} \|A_i(p_i^k - x_i^k)\|^2.
$$

(4.22)

Now, for any $i \in \{1, 2, \ldots, N\}$, we can write

$$
\|A_i(x_k - x_{k,i-1})\|^2 = \left\| \sum_{j=1}^N A_j(x_j^k - x_{j,i-1}^k) \right\|^2 \\
\leq N \sum_{j=1}^N \|A_j(x_j^k - x_{j,i-1}^k)\|^2 \\
= N \sum_{j=1}^{i-1} \|A_j(x_j^k - x_{j,i-1}^k)\|^2 \\
= N \sum_{j=1}^{i-1} (\alpha_j^k)^2 \|A_j(x_j^k - p_j^k)\|^2 \\
\leq \frac{2N}{\beta_{\text{min}}} \sum_{j=1}^{i-1} (H(x_j^k - x_{j,i-1}^k)) \\
= \frac{2N}{\beta_{\text{min}}} (H(x_k) - H(x_{k,i-1}^k)) \\
\leq \frac{2N}{\beta_{\text{min}}} (H(x_k) - H(x_{k+1})),
$$

where the first inequality follows from the fact that for any $N$ vectors $u_1, u_2, \ldots, u_N$, the inequality $\left\| \sum_{j=1}^N u_j \right\|^2 \leq N \sum_{j=1}^N \|u_j\|^2$ holds; the second inequality follows from (4.22), and the last inequality follows from Lemma 4.6, which shows that the sequence of function values is non-increasing. □
Remark 4.9. The bound in Lemma 4.8 can be improved in some situations where additional information on the structure of $A$ is available. For example, if the column space of each $A_i$ span orthogonal spaces, that is $A_i^T A_j = 0$ for any $1 \leq i < j \leq N$, then the factor $N$ can be avoided.

The next lemma constitutes the crucial step towards the establishment of a sublinear convergence rate of the CBCG method with adaptive stepsize.

Lemma 4.10. Let $\{x^k\}_{k \in \mathbb{N}}$ be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any $k \geq 0$, we have

$$H(x^k) - H(x^{k+1}) \geq \frac{1}{NC_2} S(x^k)^2,$$

where

$$C_2 = 4 \left[ \max_{i=1,2,\ldots,N} \max \left\{ \beta_i \|A_i\|^2 D_i^2, K_i \right\} + \frac{NL_2^2 D^2 \max_{i=1,2,\ldots,N} \|A_i\|^2}{\beta_{\min}} \right], \quad (4.23)$$

$\beta_{\min}$ is defined in (2.4) and $K_i$ is defined in equation (4.17), for $i = 1, 2, \ldots, N$.

Proof. For any $i \in \{i = 1, 2, \ldots, N\}$ we have that

$$S_i(x^k)^2 = (S_i(x^{k,i-1}) + S_i(x^k) - S_i(x^{k,i-1}))^2$$

$$\leq 2S_i(x^{k,i-1})^2 + 2(S_i(x^k) - S_i(x^{k,i-1}))^2$$

$$\leq 2S_i(x^{k,i-1})^2 + 2L_F^2 D_i^2 \|A_i\|^2 \cdot \|A(x^k - x^{k,i-1})\|^2$$

$$\leq 2S_i(x^{k,i-1})^2 + \frac{4NL_2^2 D_i^2 \|A_i\|^2}{\beta_{\min}} (H(x^k) - H(x^{k+1})), \quad (4.24)$$

where the second inequality follows from Lemma 3.2 and the last inequality follows from Lemma 4.8. Invoking Lemma 4.7, we obtain from (4.24) that

$$S_i(x^k)^2 \leq 2S_i(x^{k,i-1})^2 + \frac{4NL_2^2 D_i^2 \|A_i\|^2}{\beta_{\min}} (H(x^k) - H(x^{k+1}))$$

$$\leq 4 \max \left\{ \beta_i \|A_i\|^2 D_i^2, K_i \right\} (H(x^{k,i-1}) - H(x^{k,i})) + \frac{4NL_2^2 D_i^2 \|A_i\|^2}{\beta_{\min}} (H(x^k) - H(x^{k+1})). \quad (4.25)$$

Summing (4.25) for $i = 1, 2, \ldots, N$ yields

$$\sum_{i=1}^N S_i(x^k)^2 \leq 4 \sum_{i=1}^N \max \left\{ \beta_i \|A_i\|^2 D_i^2, K_i \right\} (H(x^{k,i-1}) - H(x^{k,i}))$$

$$+ \frac{4NL_2^2 D_i^2 \|A_i\|^2}{\beta_{\min}} \sum_{i=1}^N (H(x^k) - H(x^{k+1}))$$

$$\leq 4 \max_{i=1,2,\ldots,N} \max \left\{ \beta_i \|A_i\|^2 D_i^2, K_i \right\} \sum_{i=1}^N (H(x^{k,i-1}) - H(x^{k,i}))$$

$$+ \frac{4NL_2^2 D^2 \max_{i=1,2,\ldots,N} \|A_i\|^2}{\beta_{\min}} (H(x^k) - H(x^{k+1}))$$

$$= C_2 (H(x^k) - H(x^{k+1})).$$

Finally, using (3.6) we have

$$S(x^k)^2 = \left[ \sum_{i=1}^N S_i(x^k) \right]^2 \leq N \sum_{i=1}^N S_i(x^k)^2 \leq NC_2 (H(x^k) - H(x^{k+1})),$$

which proves the desired result.

\qed
Combining Lemma 3.1 with Lemma 4.10, we obtain the following corollary.

**Corollary 4.11.** Let \( \{x^k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any \( k \geq 0 \) we have

\[
H(x^k) - H(x^{k+1}) \geq \frac{1}{NC_2} (H(x^k) - H^*)(H(x^k) - H^*)^2,
\]

where \( C_2 \) is given in (4.23).

In order to get rate of convergence result in this case we will need the following technical lemma (cf. [6, Lemma 3.5]).

**Lemma 4.12.** Let \( \{a_k\}_{k \in \mathbb{N}} \) be a sequence of non-negative real numbers satisfying, for any \( k \in \mathbb{N} \), the following property

\[
a_k - a_{k+1} \geq \gamma a_k^2,
\]

and \( a_0 \leq 1/(\gamma m) \) for some positive \( \gamma \) and \( m \). Then for any \( k \in \mathbb{N} \), we have that

\[
a_k \leq \frac{1}{\gamma} \cdot \frac{1}{m + k}.
\]

Combining Lemma 4.12 along with Corollary 4.11 and Lemma 3.1, we can establish the sublinear rate of convergence of the CBCG method with adaptive stepsize.

**Theorem 4.13** (Sublinear rate for CBCG with adaptive stepsize). Let \( \{x^k\}_{k \in \mathbb{N}} \) be the sequence generated by the CBCG method with the adaptive stepsize strategy given in (4.12). Then for any \( k \geq 0 \) we have

\[
H(x^k) - H^* \leq \frac{NC_2}{k+4},
\]

and

\[
\min_{n=0,1,...,k} S(x^n) \leq \frac{2NC_2}{k+4},
\]

where

\[
C_2 = 4 \left[ \max_{i=1,2,...,N} \max \left\{ \beta_i \|A_i\|^2 D_i^2, K_i \right\} + \frac{NL_{f}^2 D_{f}^2 \max_{i=1,2,...,N} \|A_i\|^2}{\beta_{\min}} \right],
\]

\( \beta_{\min} \) is defined in (2.4) and \( K_i \) is defined in equation (4.17), for \( i = 1, 2, \ldots, N \).

**Proof.** Denote \( a_k = H(x^k) - H^* \). Thanks to Corollary 4.11 we get that (4.26) holds true with \( \gamma = 1/(NC) \). In addition, from Lemma 3.1 and (4.20), we have

\[
a_0 = H(x^0) - H^* \leq S(x^0) = \sum_{i=1}^{N} S_i(x^0) = \sum_{i=1}^{N} K_i \leq N \max_{i=1,2,...,N} K_i \leq \frac{NC_2}{4}.
\]

Hence, \( a_0 \leq 1/(4\gamma) \). Picking \( m = 4 \), we conclude from Lemma 4.12 that (4.27) holds. To find the bound on the optimality measure \( S(\cdot) \), from Lemma 4.10, we have for any \( n \geq 0 \)

\[
H(x^n) - H(x^{n+1}) \geq \gamma S(x^n)^2.
\]

For any \( k_0 \geq 0 \), summing the latter inequality for \( n = k_0, k_0 + 1, \ldots, 2k_0 \), we obtain that

\[
\gamma \sum_{n=k_0}^{2k_0} S(x^n)^2 \leq H(x^{k_0}) - H(x^{2k_0+1}) \leq H(x^{k_0}) - H(x^*) \leq \frac{1}{\gamma} \cdot \frac{1}{k_0 + 4},
\]
where the last inequality follows from (4.27). Hence,
\[
\min_{n=k_0,k_0+1,...,2k_0} S(x^n)^2 \leq \frac{1}{\gamma^2} \cdot \frac{1}{(k_0 + 1)(k_0 + 4)} \leq \frac{1}{\gamma^2} \cdot \frac{1}{(k_0 + 2)^2}.
\] (4.28)
where we have used the fact that \((k_0 + 1)(k_0 + 4) \geq (k_0 + 2)^2\). Similarly, for any \(k_0 \geq 0\),
\[
\gamma \sum_{n=k_0}^{2k_0+1} S(x^n)^2 \leq H(x^{k_0}) - H(x^{2k_0+2}) \leq H(x^{k_0}) - H(x^*) \leq \frac{1}{\gamma} \cdot \frac{1}{k_0 + 4}.
\]
Hence
\[
\min_{n=k_0,k_0+1,...,2k_0+1} S(x^n)^2 \leq \frac{1}{\gamma^2} \cdot \frac{1}{(k_0 + 2)(k_0 + 4)} \leq \frac{1}{\gamma^2} \cdot \frac{1}{(k_0 + 5/2)^2},
\] (4.29)
where we have used the fact that \((k_0 + 2)(k_0 + 4) \geq (k_0 + 5/2)^2\). Notice that \(k_0 + 5/2 = (2k_0 + 1)/2 + 2\). Therefore, by combining (4.28) when \(k\) is even and (4.29) when \(k\) is odd, we conclude that
\[
\min_{n=0,1,...,k} S(x^n) \leq \frac{1}{\gamma} \cdot \frac{1}{k/2 + 2} \leq \frac{2NC_2}{k + 4},
\]
which concludes the proof. \(\square\)

### 4.3 Backtracking Version of CBCG

Computing the adaptive stepsize requires to know the constants \(\beta_i\). In practice, these constants may not be known in advance or their known upper-approximations may be too loose. A common approach to overcome this problem is to use a backtracking scheme in order to estimate the unknown constants that are required to ensure convergence of the algorithm. This strategy is also effective in the context of CG-type algorithms.

The crucial property for the convergence analysis with adaptive stepsize is the block structured descent condition given in Lemma 4.6. When the constants in Lemma 4.6 are not known, a workaround for this problem is to use a backtracking scheme in order to estimate the unknown constants that are required to ensure convergence of the algorithm. This strategy is also effective in the context of CG-type algorithms.

The crucial property for the convergence analysis with adaptive stepsize is the block structured descent condition given in Lemma 4.6. When the constants in Lemma 4.6 are not known, a workaround for this problem is to use a backtracking scheme in order to estimate the unknown constants that are required to ensure convergence of the algorithm. This strategy is also effective in the context of CG-type algorithms.

**CBCG-B: Cyclic Block Conditional Gradient with Backtracking**

**Initialization.** \(x^0 \in X, \beta_{\text{init}} > 0, \kappa > 1, \xi_i^{-1} = 1, i = 1, 2, \ldots, N.\)

**General Step.** For \(k = 1, 2, \ldots,\)

(i) For any \(i = 1, 2, \ldots, N\), compute
\[
p_i^k \in \arg\min_{p_i \in X_i} \left\{ (\nabla_i f(x^k,i-1), p_i) + g_i(p_i) \right\},
\] (4.30)

Find \(\xi \in \mathbb{N}\), the smallest integer \(\xi \geq \xi_i^{k-1}\) such that
\[
H(x^k,i-1) - H(x^k,i-1 + \alpha U_i(p_i^k - x^k,i-1)) \geq \frac{\alpha}{2} S_i(x^k,i-1)
\] (4.31)

where \(\alpha = \min \left\{ \frac{S_i(x^k,i-1)}{\kappa \beta_{\text{init}} ||A_i(p_i^k - x^k,i-1)||^2}, 1 \right\}\) and set
\[
\xi_i^k = \xi, \quad \beta_i^k = \kappa \beta_{\text{init}}, \quad \alpha_i^k = \min \left\{ \frac{S_i(x^k,i-1)}{\beta_i^k ||A_i(p_i^k - x^k,i-1)||^2}, 1 \right\},
\] (4.32)

\[
x^k,i = x^k,i-1 + \alpha_i^k U_i(p_i^k - x^k,i-1).
\] (4.33)

(ii) Update: \(x^{k+1} = x^{k,N}\).
Under Assumption 2, we have from Lemma 4.6, that (4.31) holds provided that $\beta_k^i \geq \beta_i$. We therefore have for any $k \geq 0$ and any $i = 1, 2, \ldots, N$ that

$$\beta_{init} \leq \beta_k^i \leq \bar{\beta}_i \equiv \max \{ \kappa \beta_i, \beta_{init} \}.$$  \hspace{1cm} (4.34)

The main insight is given by the following lemma which proof follows the same arguments as that of Lemma 4.6 using (4.31).

**Lemma 4.14.** Let $\{x^k\}_{k \in \mathbb{N}}$ be the sequence generated by the CBCG-B method. Then for any $k \geq 0$ and $i \in \{1, 2, \ldots, N\}$, we have

$$H(x^{k,i-1}) - H(x^{k,i}) \geq \frac{\alpha_i^k}{2} S_i(x^{k,i-1}) \geq \frac{(\alpha_i^k)^2 \beta_i ^k}{2} \|A_i(p^k_i - x^k_i)\|^2,$$

where

$$\alpha_i^k = \min \left\{ \frac{S_i(x^{k,i-1})}{\beta_i^k \|A_i(p^k_i - x^k_i)\|^2}, 1 \right\}.$$

Using the bounds in (4.34), we obtain the following rate of convergence result for the CBCG algorithm with a backtracking scheme.

**Theorem 4.15** (Sublinear rate for CBCG-B). Let $\{x^k\}_{k \in \mathbb{N}}$ be the sequence generated by the CBCG-B method. Then for any $k \geq 0$ we have

$$H(x^k) - H^* \leq \frac{NC_3}{k + 4},$$

and

$$\min_{n=0, 1, \ldots, k} S(x^n) \leq 2NC_3 \frac{k}{k + 4},$$

where

$$C_3 = 4 \left[ \max_{i=1, 2, \ldots, N} \max \{ \tilde{\beta}_i \|A_i\|^2 D_i^2, K_i \} + \frac{NL_F^2 D^2 \max_{i=1, 2, \ldots, N} \|A_i\|^2}{\beta_{init}} \right].$$

$\tilde{\beta}_i$ is given in (4.34) and $K_i$ is defined in equation (4.17), for $i = 1, 2, \ldots, N$.

**Proof.** The arguments of this proof are similar to those in the proof of Theorem 4.13 where we replace Lemma 4.6 with Lemma 4.14. \hfill \Box

5 Numerical Experiments

The experiments presented in this section correspond to situations where the non-smooth function $g_i$ is taken to be the indicator of the set $X_i \subset \mathbb{R}^n_i$, with the eventual addition of a linear term, for $i = 1, 2, \ldots, N$. We therefore recover the smooth constrained optimization model of the traditional conditional gradient method (see also Remark 2.1). Furthermore, for the cyclic selection rule, in all the experiments, we use the “random permutation” approach which consists in randomly changing the order of the blocks at each iteration. The convergence analyses described so far do not depend on the order of the blocks and are still valid when it changes at each iteration. Therefore, all the results of Section 4 hold deterministically for this “random permutation” rule which we do not differentiate from the cyclic update rule, and hence the corresponding algorithms bear the same name in this section. We begin with a modeling remark in Section 5.1. Numerical results are given in Section 5.2 for synthetic examples and in Section 5.3 for the structural SVM training.
5.1 Exact Line Search for Quadratic Problems

We recall another well-known stepsize strategy – exact line search strategy (see, for example, [12]). This stepsize is defined as follows:

\[ \alpha_k^i \in \arg\min_{\alpha \in [0,1]} H(x^{k,i-1} + \alpha U_i (p_k - x^{k,i-1})) . \]  

(5.1)

The minimization step (5.1) can incur unnecessary additional computations, unless it can be carried out efficiently. This is the case for quadratic functions for which it reduces to the minimization of a one-dimensional quadratic over a segment. It is therefore tempting to use CBCG with this stepsize rule, but our convergence analysis does not cover it explicitly. However, in the quadratic case, exact line search can be recast as an adaptive stepsize strategy. Indeed, suppose for example the problem has the form

\[
\min \left\{ \frac{1}{2} \left\| \sum_{i=1}^{N} A_i x_i \right\|^2 + \sum_{i=1}^{n} (b_i, x_i) : x_i \in X_i \right\},
\]

where \( A_i \in \mathbb{R}^{n \times n_i} \) and \( b_i \in \mathbb{R}^{n_i} \). This problem fits model (2.1) with \( F(\cdot) = \frac{1}{2} \| \cdot \|^2, A = (A_1, A_2, \ldots, A_N) \) and \( g_i(\cdot) = (b_i, \cdot) + \delta_{X_i}(\cdot) \). Therefore, we can choose \( \beta_i = 1, i = 1, 2, \ldots, N \) and the exact line search strategy is equivalent to our adaptive stepsize strategy given in (4.12) since the upper bound of Lemma 4.1 holds with equality. Therefore, in the quadratic case, convergence for the exact line search strategy follows from Theorem 4.13.

5.2 Synthetic Problems

We generate artificial convex quadratic problems with box constraints in \( \mathbb{R}^{100} \). We consider problems of the following form

\[
\min_{\|x\|_\infty \leq 1} \frac{1}{2} (x - y)^T Q (x - y),
\]

(5.2)

where \( \|\cdot\|_\infty \) denotes the \( \ell_\infty \) norm, \( y \in \mathbb{R}^{100} \) and \( Q \in \mathbb{R}^{100 \times 100} \) are given. The problem has a natural coordinate-wise structure which allows to apply the CBCG algorithm where each block consists of a single coordinate. Indeed, by setting \( f(\cdot) = \frac{1}{2} (\cdot - y)^T Q (\cdot - y) \) and \( g_i(\cdot) = \delta_{[-1,1]}(\cdot), i = 1, 2, \ldots, 100 \), problem (5.2) fits our model (2.1). We generate random instances of problem (5.2) as follows.

- Set \( d = 100 \) and \( n = 200 \).
- Generate \( X \in \mathbb{R}^{n \times d} \) with standard normal entries.
- Set \( Q = \frac{1}{n} X^T X \).
- Generate \( y \) with standard normal entries.

We compare the standard Conditional Gradient (CG) algorithm [17], the Random Block Conditional Gradient (RBCG) algorithm [19] and the Cyclic Block Conditional Gradient (CBCG), which is the method of interest in this paper. For each algorithm, we compare three different stepsize strategies:

- **Predefined**: the stepsize which is given in (4.3). Note that for RBCG, there is a slight modification in the definition of the predefined stepsize, see [19].
- **Adaptive**: set \( A = I \) and compute the stepsize using the backtracking scheme.
- **Exact line search**: the stepsize is chosen to minimize the objective as in (5.1).

For problems of the form (5.2), the second strategy is questionable since a closed form expression is available for exact line search. However, it illustrates differences in algorithmic behaviors related to the choice of \( A \) in model (2.1).
Figure 1: Comparison of Conditional Gradient (CG), its random block version (RBDG) and its cyclic block version (CBCG) with the three different stepsize strategies based on 1000 randomly generated instances of problem (5.2). The central line is the median over the 1000 runs and the ribbons show 98%, 90%, 80%, 60% and 40% quantiles. For all methods, $k$ represents the number of effective passes through all the coordinates.

**Remark 5.1.** For CG and RBCG, sublinear convergence rate holds for the three stepsize strategies. For CBCG, in the case of the predefined and of the adaptive stepsizes, convergence is ensured by Theorems 4.5 and 4.15, respectively. Furthermore, since we consider quadratic objective functions, exact line search stepsize strategy can be seen as a case of our adaptive strategy and convergence follows from Theorem 4.13 (see also Section 5.1).

We generate 1000 random instances of problem (5.2), $f_w$ denoting the objective function of the $w$th randomly generated problem, $w = 1, 2, \ldots, 1000$. For each problem, we run the three different algorithms with the three different stepsize strategies (the initialization is at the origin). The results for the first 10 iterations are presented in Figure 1. For each algorithm, increasing $k$ by 1 means that $N = 100$ blocks have been queried, randomly for RBCG, sequentially for CBCG and all at once for CG. Since each objective function is generated randomly, it does not make sense to directly compare performance across different problems on the same graph. To overcome this, for each $w = 1, 2, \ldots, 1000$, we “center” and “scale” the function values. That is, for each objective function $f_w$, $w = 1, 2, \ldots, 1000$, we estimate the optimal value $f^*_w$ of (5.2) by running CBCG with exact line search for 200 more iterations. For such a number of iterations, we observed on preliminary experiments that the algorithm reached machine precision on random instances of problem (5.2). The quantity plotted in Figure 1 is given by the following affine transformation,

$$
\frac{f_w(x^k) - f_w^*}{f_w(0) - f_w^*},
$$

so that in Figure 1, the first value is always 1 and the represented quantities are positive and asymptotically tend to 0. The main comments regarding this experiment are the following:

- For each stepsize strategy, CBCG has an advantage.
- The predefined stepsize strategy leads to much slower convergence.
- Adaptive and exact line search strategies yield improved convergence speed for both CBCG and RBCG, which is not the case with the CG.
- Exact line search strategy has a slight advantage over adaptive strategy.
The last point deserves further comments. Indeed, in model (2.1), there are different possible choices of matrix $A$ and function $F$ that lead to equivalent problems. Despite being equivalent problems, different model choices may lead to variations in algorithmic performances when numerically solving a problem. This is what we observe here. Indeed, as pointed out in Section 5.1, the exact line search strategy corresponds to choosing an $F$ that is perfectly conditioned. This means that variations of $F$ around its minimum are isotropic. On the other hand, choosing $A = I$ leads to a choice of $F$ that is less well behaved. These numerical experiments suggest that choosing $A$ that corresponds to a better conditioned $F$ leads to better numerical performances.

5.3 Structural SVM

The main motivation for the introduction of block version of CG method with random update rule in [19] is that it leads to a new efficient algorithm for training the structural SVM [28, 29]. We refer the reader to [19] and the references therein for background on this problem and its relations to the conditional gradient method. In brief, the structural SVM solves a multi-class classification task. It is dedicated to problems for which the output classes are embedded in a combinatorial structure such as trees, graphs or sequences. In this setting the number of classes can be enormous which results in optimization problems with an untractable number of linear constraints. For some of these problems efficient decoding algorithms can be used as oracles to compute sub-gradients of the structural SVM problem. They can also be used as oracles to solve the linearized sub-problem required to run the conditional gradient and block conditional gradient methods on the dual of the structural SVM.

In this section, we briefly recall the mathematical formulation of the dual structural SVM problem and provide a numerical comparison between random and cyclic update rules for block conditional gradient in this context. The purpose is not to be exhaustive here and we solely focus on the aspects of the problem related to optimization. Therefore, we compare the numerical performances of the two selection rules on this real world example based on an optimization criterion. In this section, $N$ denotes the number of training examples, $M$ denotes the number of output classes and $d$ is an integer such that $\mathbb{R}^d$ is a space of tractable size (such that elements of $\mathbb{R}^d$ can be stored in memory). The dual variable of the structural SVM is a matrix $\alpha \in \mathbb{R}^{N \times M}$ with non-negative entries (this could actually be refined with example dependent output classes, but we stick to this notation as a first approximation). The dual problem of the structural SVM can be written as follows

$$
\min_{\alpha \geq 0} \left\{ \frac{\lambda}{2} \|A\alpha\|^2 - \text{Tr}(b^T\alpha) : \alpha 1^M = 1^N \right\},
$$

(5.3)

where $A : \mathbb{R}^{N \times M} \to \mathbb{R}^d$ is a linear map, $b \in \mathbb{R}^{N \times M}$ is a matrix, $\text{Tr}$ denotes the trace operator and $1^s$ denotes the vector in $\mathbb{R}^s$ which all entries are 1. Problem (5.3) has an interesting product structure. Indeed, its feasible set can be viewed as a product of $N$ simplices of dimension $M$. In the context of structural SVM it is not necessary to store $\alpha$ explicitly, it is sufficient to store and update $A\alpha \in \mathbb{R}^d$ and $\text{Tr}(b^T\alpha)$. With this information, we can use the specific decoding oracles to solve simplex constrained linear subproblems and run the conditional gradient algorithm. This constitutes the main advantage of the method here, it allows to explore a potentially very large space by using only implicit conditional gradient steps (recall that $M$ is the size of a set of combinatorial nature, see [19] for a complete derivation and more details). We use the code provided by the authors of [19] which gives the possibility to train the structural SVM using RBCG or CBCG on the Optical Character Recognition (OCR) originally proposed in [28]. We consider random update rule and cyclic update rule with varying block order (or random permutation). We only consider exact line search strategy (5.1). Note that our convergence analysis can be applied in this setting (see also Section 5.1). Numerical results in terms of the optimality measure $S$ for various values of $\lambda$ are given in Figure 2. The global behavior is similar to what we could observe on synthetic examples in Section 5.2. In particular, the cyclic update rule has a slight advantage.
Figure 2: Comparison of RBCG and CBCG for the structural SVM training on the OCR dataset of [28]. The quantity represented is the optimality measure defined by (3.1) and the heading represents the value of $\lambda$ (from (5.3)). The central line is the median over 20 runs and the ribbons show 80% and 50% quantiles. For all methods, $k$ represents the number of effective passes through all the blocks (which correspond to datapoints here).

6 Discussion and future work

We have described CBCG algorithm and provided an explicit sublinear convergence rate estimate. It is important to notice that the convergence speed is $O(1/k)$, which is the same as for the traditional CG algorithm (up to multiplicative factor) [25]. This asymptotic rate cannot be improved in general [10]. However, contrary to the average case complexity estimate of RBCG [19], the exact expression of the rate is not a direct generalization of that of CG algorithm. Recall that for traditional CG method, the multiplicative constant is proportional to $L_F D^2$ [17]. The constants given in Theorems 4.5 and 4.13 feature additional multiplicative and additive terms. In particular the quantity $C_1/L_F D^2$, for $C_1$ given by (4.4) or the quantity $NC_2/L_F D^2$ for $C_2$ given by (4.23), show multiplicative dependence in the number of blocks $N$. This behavior is consistent with previous observations comparing random and cyclic update rules [6]. It is expected that the analysis of cyclic rules lead to worse constants since they represent worst case complexity analysis. An important theoretical question is whether this can be leveraged or not for cyclic rules. In other words, is it possible to prove an explicit convergence rate of the form $M/k$ for the cyclic rule such that $M/(L_F D^2)$ does not depend on the number of blocks $N$?

For practical applications however, this remark should be mitigated since we are comparing upper bounds. These bounds can be very loose and their comparison may not shed much light on the comparative behavior of different rules on practical problems. Indeed, our numerical experiments on synthetic and real-world examples reproducibly suggest an advantage of cyclic over random update rule. This is again something that has already been observed in other contexts [6]. A question related to the discussion of the previous paragraph is to give a theoretical justification to this observation or eventually provide iteration dependent block update rules that comply with it (see for example [27] for an illustration in the context of gradient descent).

Finally, a natural question is that of the extension of specificities of the conditional gradient method in our block decomposition setting. Potential directions include

- **Linear convergence.** CG is known to converge linearly when the optimum is in the relative interior of the feasible set [15, 20] or when the feasible set is strongly convex and the gradient of the smooth part of the objective is non-zero on the feasible set [21].
• **Dual interpretation of the block decomposition.** Generalized CG is known to implicitly generate subgradient sequences [1] related to the mirror descent algorithm [3] applied to a dual problem. Similarly, a dual interpretation of RBCG is in terms of stochastic subgradient [19].

• **Generalization of the results to exact line search stepsize strategies.** The analysis of such stepsize strategies is not problematic for CG or RBCG [17, 19]. However, we could not generalize it for CBCG, except in the quadratic case (see Section 5.1), and thus developing an analysis for the exact line search strategy is in our opinion an important task.

## A Proof of Lemma 2.2

We adapt the standard proof of the descent Lemma, see e.g., [7, Proposition A.24]. Under the assumptions of the lemma, using the fundamental theorem of calculus for line integrals on the segment \([x, x + U_i h]\), we have

\[
f(x + U_i h) = f(x) + \int_0^1 \langle \nabla f(x + tU_i h), U_i h \rangle \, dt
\]

\[
= f(x) + \langle \nabla f(x), U_i h \rangle + \int_0^1 \langle \nabla f(x + tU_i h) - \nabla f(x), U_i h \rangle \, dt. \tag{A.1}
\]

We can bound the integrand term for any \(t \in [0, 1]\) as follows

\[
\langle \nabla f(x + tU_i h) - \nabla f(x), U_i h \rangle = \langle A^T \nabla F(A(x + tU_i h)) - A^T \nabla F(Ax), U_i h \rangle
\]

\[
= \langle \nabla F(A(x + tU_i h)) - \nabla F(Ax), AU_i h \rangle
\]

\[
= \langle \nabla F(Ax + tA_i h) - \nabla F(Ax), A_i h \rangle
\]

\[
\leq \| \nabla F(Ax + tA_i h) - \nabla F(Ax) \| \cdot \| A_i h \|
\]

\[
\leq t \beta_i \| A_i h \|^2, \tag{A.2}
\]

where we have used Cauchy-Schwartz inequality and Assumption 2 to obtain the last two inequalities. Combining (A.1) and (A.2), we have

\[
f(x + U_i h) \leq f(x) + \langle \nabla f(x), U_i h \rangle + \beta_i \| A_i h \|^2 \int_0^1 t \, dt
\]

\[
= f(x) + \langle \nabla f(x), h \rangle + \frac{\beta_i \| A_i h \|^2}{2},
\]

which proves the desired result. \(\square\)

## References


