Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Efficient Generalized Conditional Gradient with Gradient Sliding for Composite Optimization

Yiu-ming Cheung[1,2] **and Jian Lou**[1]
[1]Department of Computer Science, Hong Kong Baptist University, Hong Kong SAR, China
[2]United International College, Beijing Normal University — Hong Kong Baptist University, Zhuhai, China
ymc@comp.hkbu.edu.hk, jianlou@comp.hkbu.edu.hk

Abstract

Generalized conditional gradient method has regained increasing research interest as an alternative to another popular proximal gradient method for sparse optimization problems. For particular tasks, its low computation cost of linear subproblem evaluation on each iteration leads to superior practical performance. However, the inferior iteration complexity incurs excess number of gradient evaluations, which can counteract the efficiency gained by solving low cost linear subproblem. In this paper, we therefore propose a novel algorithm that requires optimal graduate evaluations as proximal gradient. We also present a refined variant for a type of gauge regularized problem where approximation techniques are allowed to further accelerate linear subproblem computation. Experiments of CUR-like matrix factorization problem with group lasso penalty on four real-world datasets demonstrate the efficiency of the proposed method.

1 Introduction

This paper studies unconstrained composite optimization problems of the form

$$\min_{x \in \mathcal{X}} F(x) = l(x) + r(x), \tag{1}$$

where $\mathcal{X}$ is a general vector space equipped with inner product (e.g. $\mathbb{R}^d$ equipped with ℓ_2 norm). $l(x)$ is a smooth convex loss function, i.e. it is continuously differentiable with L-Lipschitz continuous gradient:

$$||\nabla l(x) - \nabla l(y)|| \leq \frac{L}{2}||x - y||, \; \forall x, y \in \mathcal{X}. \tag{2}$$

r(x) stands for the regularizer which is a nonsmooth closed proper convex function. Problem (1) is of vital importance in machine learning because many sparse estimation problems fit into this model. For example, in convex relaxed low rank matrix completion problem [Candès and Recht, 2009], $l(x)$ is the Frobenius norm between the observed matrix and the low rank estimation matrix, and $r(x)$ is the trace norm of the low rank estimation matrix. In regularized empirical risk minimization problem, $l(x)$ can be logistic loss or least square loss, and $r(x)$ can vary from simple ones like ℓ_1 norm [Tibshirani, 1996] to very complex form such as graph-guided lasso [Kim and Xing, 2009] and group lasso [Yuan and Lin, 2006] for inducing structured sparsity.

Many different lines of methods exist for solving such sparse optimization problems, among them is the popular proximal gradient (PG) based approach ([Beck and Teboulle, 2009]; [Nesterov, 2013]). This kind of methods can achieve the optimal rate of convergence under certain problem settings, hence it enjoys low iteration complexity. The per-iteration cost mainly comes from gradient evaluation and a proximal map (PM) related to the type of the regularizer. On the one hand, due to the optimal iteration complexity, the number of gradient evaluations is optimal for PG method. On the other hand, the proximal map itself is a quadratic optimization problem composed by $r(x)$ and a quadratic term. For particular regularizers, it admits efficient evaluation. For example, the proximal map related to the lasso regularizer is simply the soft-thresholding. However, in some cases, evaluating the proximal map can be quite computational demanding. For instance, to solve the proximal map related the trace norm regularizer, it requires a full singular value decomposition (SVD) in each iteration [Candès and Recht, 2009]. As a result, the high per-iteration cost raised by the proximal map becomes the bottleneck of the PG method for those problems.

To address the high per-iteration cost raised by the PM, generalized conditional gradient method (GCG) ([Jaggi, 2013]; [Yu *et al.*, 2014]; [Jaggi, 2011]; [Clarkson, 2010]), has been receiving increasing research interest. It only requires to evaluate a linear operator (LO) in each iteration, which is intuitively much easier to solve than the quadratic subproblem of PM. In fact, this intuition is elaborated by many structured sparse regularizer, e.g. LO requiring spectral norm versus PM requiring full SVD for trace norm regularized problem, in which the former is much more computational efficient [Yu *et al.*, 2014]. As a result, although the iteration complexity of GCG based methods are inferior to their PG counterpart [Dudik *et al.*, 2012], some studies have found that the low per-iteration cost can sometimes compensate for the extra iterations leading to less overall execution time than PG. Nevertheless, the inferior iteration complexity leads to extra gradient evaluations. Although the per-iteration cost for LO can be small enough to afford excess iterations, the increased demand for gradient evaluation can raise an inevitable trade-off, especially for large scale problems. In addition, the convergence results become weaker when approximation techniques are introduced to solve LO. As a result, these approximation techniques also bring about an increased count of gradient evaluations, which can counteract the efficiency gained from the accelerated LO evaluations.

In this paper, we therefore propose a novel algorithm called Generalized Conditional Gradient with Gradient Sliding (GCG-GS) and its refined variant for gauge regularized problems. We first extend a recent optimization scheme called gradient sliding to general unconstrained composite convex optimization problems. Instead of evaluating gradient on each iteration, we skip it from time to time, which can be viewed as many LO evaluations sharing the same gradient value. For gauge regularized problem ([Jaggi, 2013]; [Yu *et al.*, 2014]), where efficient approximation techniques exist for handling $r(x)$, we present an improved variant of the general GCG-GS algorithm to incorporate these techniques to further accelerate the algorithm. As a result, our algorithm has optimal count of gradient evaluations as their PG counterpart, and more importantly, it allows efficient approximation techniques to be used without increasing the optimal count of gradient evaluation. Experiments of CUR-like matrix factorization problem with group lasso penalty on four real-world datasets have demonstrated the efficiency of the proposed method.

2 Background

2.1 Generalized Conditional Gradient

Generalized conditional gradient method (GCG) is suitable for Problem (1). It is a generalization of the classic conditional gradient method (CG) [Frank and Wolfe, 1956], which solves constraint problem on a subset of $\mathcal{X}$, e.g. convex compact set [Jaggi, 2013]. Also, to keep the nonsmooth part intact, GCG defines the following alternative duality gap at iteration k ([Yu *et al.*, 2014]; [Bredies *et al.*, 2005]),

$$\begin{aligned} G(x_k) =& l(x_k) + r(x_k) \\ &- \inf_{x\in\mathcal{X}}\{(l(x_k) + \langle x - x_k, \nabla l(x_k)\rangle + r(x))\}, \\ =& r(x_k) - \inf_{x\in\mathcal{X}}\{r(x) + \langle x - x_k, \nabla l(x_k)\rangle\} \\ =& \sup_{x\in\mathcal{X}}\{r(x_k) - r(x) - \langle x - x_k, \nabla l(x_k)\rangle\}, \end{aligned} \tag{3}$$

instead of taking the sub-differential of nonsmooth $r(x)$ as some CG methods proposed [Jaggi, 2013]. The duality gap is essential for constructing the linear operator because it is related to the optimal necessary condition and is always an upper bound approximation to the primal gap $F(x) - F(x^*)$, as described in the following lemma [Yu *et al.*, 2014]:

Lemma 1. *For Problem 1 with any $x \in \mathcal{X}$, the duality gap $G(x) \geq 0$ and $G(x) = 0$ iff x satisfies the necessary optimal condition*[1]*. Also, the duality gap is always an upper bound of the primal gap that $G(x) \geq F(x) - F(x_*)$, where x^* denotes the global optimal point.*

[1]We say that x satisfies the necessary optimal condition if $0 \in \nabla l(x) + \partial r(x)$.

Hence, GCG solves Problem (1) by minimizing the duality gap on each iteration, which amounts to evaluating a linear operator (LO), i.e.

$$\begin{aligned} &d_k = \arg\max_{x\in\mathcal{X}} r(x_k) - r(x) - \langle x - x_k, \nabla l(x_k)\rangle, \\ \Leftrightarrow &d_k = \arg\min_{x\in\mathcal{X}} \langle x, \nabla l(x_k)\rangle + r(x). \end{aligned} \tag{4}$$

Note that additional assumptions should be made to $F(x)$, otherwise the above linear subproblem may diverge. To avoid introducing additional complexity, we assume that the solution sequences d_k and x_k (also a_t and u_t to be introduced later) are finite whose maximum distance between each other is upper bounded by a positive constant D_s. This assumption is the same as Assumption 3 in [Yu *et al.*, 2014], where more sophisticated equivalent assumptions are also discussed (see Proposition 3 in [Yu *et al.*, 2014]).

Then, the next step x_{k+1} can be obtained by

$$x_{k+1} = (1 - \alpha_k)x_k + \alpha_k d_k, \tag{5}$$

where the step size α_k can be set to deterministic sequence for example of order $O(\frac{1}{k})$ or choosing by optimizing the following problem,

$$\alpha_k = \arg\min_{\alpha\in[0,1]} F((1 - \alpha)x_k + \alpha d_k). \tag{6}$$

GCG method has $O(\frac{1}{K})$ convergence rate for Problem (1), where K is the total number of iterations. Or equivalently, it needs $O(\frac{1}{\epsilon})$ iterations to find an ϵ accurate solution.[2] Apparently, this incurs additional count of gradient evaluations than their PG counterpart based on this one gradient evaluation per-iteration scheme.

2.2 Approximation Techniques for Accelerating Linear Operator Evaluation

The linear operator in Eq.(4) admits low per-iteration cost that allows them to afford excess count of iterations, which is the major motivation for adopting GCG method rather than PG method for some tasks. This evaluation can be further accelerated by various approximation techniques. Specifically, we consider the case when $r(x)$ is the so-called generalized gauge function [Yu *et al.*, 2014] defined as

$$r(x) = h(\kappa(x)), \tag{7}$$

where $h(\cdot)$ is a convex increasing function and $\kappa(x)$ is a gauge function (convex, positively homogeneous, also see [Friedlander *et al.*, 2014] for detail) satisfying

$$\kappa(a) = \inf\{\rho : a \in \rho\mathcal{C}\}. \tag{8}$$

The convex compact subset $\mathcal{C}$ is the unit ball of the gauge $\kappa(x)$, namely

$$\mathcal{C} = \{a \in \mathcal{X} : \kappa(a) \leq 1\}. \tag{9}$$

This formal definition of $r(x)$ is a little complicated. Indeed, one can simply take it as a generalized norm function.

The key idea to accelerate LO evaluation is to circumvent direct computation of $\kappa(x)$ by approximation techniques.

[2]The ϵ accurate solution refers to a solution x having the primal gap no larger than ϵ.

Following [Yu *et al.*, 2014], we convert Eq.(4) to a constrained form, it gives

$$d_k = \arg \min_{x:r(x)\leq\zeta} \langle x, \nabla l(x_k)\rangle. \tag{10}$$

To avoid estimating ζ, it derives d_k by direction and scalar separately. For estimating the direction a_k, it follows

$$a_k = \arg \min_{a:\kappa(a)\leq 1} \langle a, \nabla l(x_k)\rangle = \arg\min_{a\in\mathcal{C}} \langle a, \nabla l(x_k)\rangle. \tag{11}$$

We leave the derivation of scalar later when used.

The approximation for accelerating the evaluation of LO lies in two parts. The first is to allow d_k or a_k to be calculated approximately, i.e. a_k is relaxed to satisfy:

$$\langle a_k, \nabla l(x_k)\rangle \leq \epsilon_k + \min_{a\in\mathcal{C}} \langle a, \nabla l(x_k)\rangle. \tag{12}$$

This approximation can further reduce the per-iteration cost of LO evaluation, and thus the reduction of overall running time is observed practically [Jaggi, 2013]. However, with such approximation, the iteration complexity guarantee is weakened by a factor associated with the degree of approximation allowed [Jaggi, 2011]. As a result, the count of gradient evaluations is increased which entails a trade-off between the time increase of gradient evaluation and the decrease of LO evaluation, especially for large scale problems whose gradient are expensive to compute.

The other approximation is related to the constraint set. When $\mathcal{C}$ is a convex hull of atomic domain $\mathcal{A}$, Eq.(11) can be equivalently solved on $\mathcal{A}$:

$$\min_{a\in\mathcal{C}} \langle a, \nabla l(x_k)\rangle = \min_{a\in\mathcal{A}} \langle a, \nabla l(x_k)\rangle = -\kappa^o(-\nabla l(x_k)), \tag{13}$$

where $\kappa^o(-\nabla l(x_k)) = \max_{a\in\mathcal{A}} \langle a, -\nabla l(x_k)\rangle$ is called the polar operator [Zhang *et al.*, 2013]. Hence, the evaluation of LO is converted to the evaluation of the polar operator, which is more efficient to deal with than $\kappa(x)$ itself and is the actual form adopted practically. A simple example of this polar operator is, when $\kappa(x)$ is a norm, we can immediately find the associated $\kappa^o(g)$ is its dual norm.

2.3 Conditional Gradient Sliding Algorithm (CGS)

Paper [Lan and Zhou, 2014] has proposed a gradient sliding technique for constraint smooth objective function to reduce gradient evaluations of CG algorithm. Under CG framework, although the requirement for solving linear operator is still $O(\frac{1}{\epsilon})$, gradient evaluations can be surprisingly reduced to match the iteration complexity of PG counterpart under the same problem settings.

The CGS separates into outer and inner iteration. On each outer iteration, two additional sequences are maintained. It can be seen as a variant to Nesterov's optimal gradient method [Nesterov, 2013], with the modification of the step (16) calling a CG subroutine rather than a gradient descent or proximal mapping procedure for PG, as shown in the following:

$$z_{k+1} = (1-\gamma_k)y_k + \gamma_k x_k; \tag{14}$$

$$g_k = \nabla F(z_{k+1}); \tag{15}$$

$$x_{k+1} = CG(g_k, x_k, \beta_k, \eta_k); \tag{16}$$

$$y_{k+1} = (1-\gamma_k)y_k + \gamma_k x_{k+1}. \tag{17}$$

The inner loop, namely the CG subroutine, applies the classic CG algorithm to optimize the following subproblem:

$$\phi(v) = \langle v, g_k\rangle + \frac{\beta_k}{2}||v - x_k||^2. \tag{18}$$

According to CG method, on each inner iteration, it optimizes the duality gap:

$$v_t = \arg\max_{v\in\mathcal{D}} G(u_t, v) = \arg\max_{v\in\mathcal{D}} \langle u_t - v, \nabla\phi(u_t)\rangle, \tag{19}$$

where u_t is the solution sequence of the inner loop subroutine[3], and $\mathcal{D}$ is the constraint set of the problem [Lan and Zhou, 2014] considers. Here, we have slightly extend the notation of the duality gap to incorporate the additional variable v. The subprocess returns the latest u_t once the duality gap is less than η_k. That is, it returns u_{t_k} when

$$G(u_{t_k}, v_{t_k}) \leq \eta_k. \tag{20}$$

As a result, it can be viewed as if many LO evaluations can share the same gradient to maintain the same convergence rate, instead of updating the gradient for each LO evaluation.

3 The Proposed Algorithm

In this section, we propose our novel algorithm for Problem (1), called Generalized Conditional Gradient with Gradient Sliding (GCG-GS). We first present the GCG-GS for general regularizer. Our algorithm is related to [Lan and Zhou, 2014], but it can suit to more general composite optimization problems with the unconstrained domain. Although for some problems it can be equivalently transformed between regularization and constraint form by Lagrangian duality, the regularization form we consider here allows additional heuristic local optimization, which is hardly known for constraint form. More importantly, we will propose a refined GCG-GS algorithm, which admits various approximation techniques for further accelerating the LO evaluation. When the approximation techniques are involved, neither [Lan and Zhou, 2014] nor our general GCG-GS are applicable. This issue is mainly because the stopping criteria of Eq.(20) is either no longer computable, or computationally expensive to obtain. Our refined GCG-GS will handle this issue.

3.1 General GCG-GS

Algorithm Description:

To involve the gradient sliding scheme, our algorithm also separates into outer and inner loops. In the outer loop, we evaluate the gradient of the smooth part $g_k = \nabla l(z_{k+1})$. Then the subroutine is called. As for the subroutine, we consider $\Phi(v)$ composed of $\phi(v) = \langle v, g\rangle + \frac{\beta}{2}||v-u||^2$ and nonsmooth $r(v)$, i.e. $\Phi(v) = \phi(v) + r(v)$. We use a different definition of $G(u_t, v)$ according to Eq.(3), which is to be optimized as follows:

$$G(u_t, v_t) = \max_{v\in\mathcal{X}} \{r(u_t) - r(v) - \langle v - u_t, \nabla\phi(u_t)\rangle\}. \tag{21}$$

Note that $G(u_t, v_t)$ is the duality gap of $\Phi(v)$ at u_t. Thus, the subroutine actually solves the minimization problem of $\Phi(v)$

[3]We use k to represent sequences related to outer loop and t for inner loop in this paper.

by GCG algorithm until certain duality gap is obtained. In addition, α_t can be optimally chosen by solving

$$\alpha_t = \arg \min_{\alpha \in [0,1]} \phi((1-\alpha)u_t + \alpha v_t) + r((1-\alpha)u_t + \alpha v_t). \tag{22}$$

Finally, the next variable is obtained by

$$u_{t+1} = (1-\alpha_t)u_t + \alpha_t v_t. \tag{23}$$

For clarity, we summarize the general GCG-GS in Algorithm (1) and Algorithm (2).

Algorithm 1 General-GCG-GS

```
Require: $x_0, K, \gamma_k, \beta_k, \eta_k$
 1: for $k = 0, 1, ..., K-1$ do
 2:     $z_{k+1} = (1-\gamma_k)y_k + \gamma_k x_k$;
 3:     $g_k = \nabla l(z_{k+1})$;
 4:     $x_{k+1} = GCG(g_k, x_k, \beta_k, \eta_k)$;
 5:     $y_{k+1} = (1-\gamma_k)y_k + \gamma_k x_{k+1}$;
 6: end for
Ensure: $y_K$
```

Algorithm 2 GCG: General-GCG-GS subroutine

```
Require: $g, u, \beta, \eta$
 1: let $\phi(x) = \langle g, x\rangle + \frac{\beta}{2}||x-u||^2$, $u_0 = u$;
 2: for $t = 0, 1, ...,$ do
 3:     $G(u_t, v) = r(u_t) - r(v) - \langle v - u_t, \nabla\phi(u_t)\rangle$;
 4:     $v_t = \arg\max_{v\in\mathcal{X}} G(u_t, v)$;
 5:     if $G(u_t, v_t) \leq \eta$ then
 6:         break;
 7:     end if
 8:     $\alpha_t = \arg\min_{\alpha\in[0,1]} \phi((1-\alpha)u_t + \alpha v_t) + r((1-\alpha)u_t + \alpha v_t)$;
 9:     $u_{t+1} = (1-\alpha_t)u_t + \alpha_t v_t$;
10: end for
11: Return: $u^+ = u_t$;
```

Convergence Analysis:
For the general GCG-GS algorithm, we have the following convergence guarantee. We first introduce the following notation:

$$\Gamma_0 = 1;\ \Gamma_k = \Pi_{i=1}^{k}(1-\gamma_i), k = 1, 2, ... \tag{24}$$

Theorem 1. *Under finite solution sequence assumptions, apply GCG-GS to Problem (1),*
a) for any x, the output y_K satisfies,

$$\begin{aligned} F(y_K) - F(x) &\leq \Gamma_{K-1}(1-\gamma_0)(F(y_0) - F(x)) \\ &+ \sum_{k=0}^{K-1} \frac{\Gamma_{K-1}\gamma_k\beta_k}{2\Gamma_k}(||x_k - x||^2 - ||x_{k+1} - x||^2) \\ &+ \sum_{k=0}^{K-1} \frac{\Gamma_{K-1}\gamma_k\eta_k}{\Gamma_k}; \end{aligned} \tag{25}$$

b) consider the inner loop, namely the LO evaluations, for a particular stage k, denote $\beta = \beta_k$, the duality gap satisfies,

$$\min_{i=0}^{t} G(u_i, v_i) \leq \frac{6\beta D_s}{t+2}, \tag{26}$$

where D_s is the upper bound of the solution sequence.

Corollary 1. *With the sequences setting as $\beta_k = \frac{2L}{k+1}$, $\gamma_k = \frac{2}{k+2}$, $\eta_k = \frac{2LD_0}{K(k+1)}$ and denoting $D_0 = ||x_0 - x||^2$, we have:*
a)

$$F(y_K) - F(x) \leq \frac{6LD_0}{K(K+1)}. \tag{27}$$

For finding an ϵ solution, we get:

$$K = \sqrt{\frac{6LD_0}{\epsilon}}. \tag{28}$$

b) the total number of inner LO evaluations is

$$T_K = \frac{6D_s}{D_0}K^2 + K. \tag{29}$$

For finding an ϵ solution, we have:

$$T_K = \frac{36LD_s}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}. \tag{30}$$

Discussion:
There are various assignment of sequence, please see [Lan and Zhou, 2014]. For the particular sequence we adopt here, it is apparent that our number of gradient evaluations for finding an ϵ solution is $O(\sqrt{\frac{1}{\epsilon}})$, which is the same as those optimal complexity of PG methods for Problem (1). In addition, in terms of the total number of LO evaluations T_K, the proposed method maintains the same order of complexity as those plain GCG methods which is also optimal for GCG.

However, the above algorithm is only conceptual in some sense. Note that both the Subproblems (21) and (22) can be difficult to solve for some $r(x)$. Also, even they were solvable, we often prefer avoiding directly computing $r(x)$ by considering more efficient substitution such as polar operator in Section 2.2. Inspired by this, we will propose the refined GCG-GS algorithm in the next subsection to allow more efficient inner loop execution.

3.2 Refined GCG-GS for Gauge Regularized Problem

In this subsection, we follow the assumption as in Section 2.2, where the regularizer is a generalized gauge function defined by Eq.(7). Note that most practically used sparsity inducing and rank minimization regularizers can be seen as generalized gauge function.

Efficient Approximation Techniques:
Essentially, we apply the efficient approximation techniques introduced in Section 2.2 to minimize $\Phi(v)$. To efficiently minimize Eq.(21), we first convert it to constraint form:

$$v_t \in \arg \min_{v:h(\kappa(v))\leq\zeta} \langle v, \nabla\phi(u_t)\rangle. \tag{31}$$

Then we update v_t by solving the direction and scalar separately, namely $\alpha_t v_t \approx \theta_t a_t$ (θ_t denotes the approximate scalar). The direction is updated by

$$a_t \in \arg \min_{a:\kappa(a)\leq 1} \langle a, \nabla\phi(u_t)\rangle. \tag{32}$$

Incorporating the approximation Eq.(12) and solving it on atomic domain $\mathcal{A}$, we can obtain a_t more efficiently by

$$\langle a_t, \nabla\phi(u_t)\rangle \leq \epsilon_t + \min_{a\in\mathcal{A}}\langle a, \nabla\phi(u_t)\rangle = \epsilon_t - \kappa^o(-\nabla\phi(u_t)). \tag{33}$$

The scalar, denoted as θ_t, would be originally chosen as

$$\theta_t = \arg\min_\theta \phi((1-\alpha_t)u_t + \theta a_t) + h(\kappa((1-\alpha_t)u_t + \theta a_t)), \tag{34}$$

where α_t is a deterministic sequence to be specified in our next theorem. Again, to avoid direct evaluation of $\kappa(u_t)$, an upper substitution ρ_t is used as in [Yu *et al.*, 2014]. This is achieved by choosing $\rho_0 \geq \kappa(u_0)$ and the update scheme $\rho_{t+1} = (1-\alpha_t)\rho_t + \theta_t$. Then $\rho_t \geq \kappa(u_t)$ can be held iteratively, see [Yu *et al.*, 2014]. Thus, by alternatively using $h(\rho_t)$ provided that $h(\cdot)$ is increasing convex and $\kappa(\cdot)$ is convex, θ_t can be obtained by

$$\theta_t = \arg\min_\theta \phi((1-\alpha_t)u_t+\theta a_t)+(1-\alpha_t)h(\rho_t)+\alpha_t h(\frac{\theta}{\alpha_t}). \tag{35}$$

Finally, an additional local heuristic optimization can be adopted to further improve the practical performance, which is another motivation for using the regularized form rather than constrained form. Denoting such re-optimization by **Improve**, we adopt the following conceptual requirement for it, which is **Relaxed** assumption according to [Yu *et al.*, 2014]:

$$\begin{cases} \phi(u_{t+1}) + h(\rho_{t+1}) \leq \phi(u_t) + \langle \tilde{u}_{t+1} - u_t, \nabla\phi(u_t)\rangle \\ \quad + \dfrac{\beta}{2}||\tilde{u}_{t+1} - u_t||^2 + (1-\alpha_t)h(\rho_t) + \alpha_t h(\dfrac{\theta_t}{\alpha_t})) \\ \rho_{t+1} \geq \kappa(u_{t+1}). \end{cases}; \tag{36}$$

Weighted Average as Return Value:

An important issue with the above approximation is that the duality gap Eq.(21) is either no longer computable, or even when we can compute it, it is unreasonable for us to directly evaluate it because we do all the above approximations to avoid computing $\kappa(v)$ directly. As a result, the stopping criteria in general GCG-GS algorithm (also CGS algorithm) cannot be used. Furthermore, the choice of return value becomes a problem because the previous bound on duality gap only guarantees the minimum one. Again, as we cannot directly compute the duality gap, it also becomes unknown that on which particular u_t the duality gap is small enough.

To solve the above stopping criteria problem, we propose a simple alternative by estimating a maximum iteration count m of the inner LO evaluation loop. As shown in our convergence analysis, different outer loops can share the same m.

As for the choice of return value, instead of returning a particular u_t, we propose using the weighted average of u_t as the returned value. We show such averaged $\bar{u}_m$ can guarantee $G(\bar{u}_m, x)$ to be smaller than the desired η as long as the proper approximated m is used. In detail, the return value to the outer loop is

$$\bar{u}_m = \sum_{t=0}^{m-1} \nu_t u_t,\ \nu_t = \frac{2}{m(m+1)}(t+1). \tag{37}$$

Intuitively, variables of later iterations gain more weights. This intuition is compatible to the analysis in [Jaggi, 2011], where the one achieving the smallest duality gap lies in the last third iterations. This average scheme is by observing the special construct of ϕ, namely, the corresponding $G(u_t, x)$ is convex in u_t for arbitrary yet fixed x. Note that it also allows an online update $\bar{u}_{t+1} = (1-\frac{2}{t+2})\bar{u}_t + \frac{2}{t+2}u_t$, which is exactly what has been shown in the algorithm. We note that [Lacoste-Julien *et al.*, 2013] has used the same weighted average as an update option in block coordinate conditional gradient method. To summarize together, Algorithm (3) and Algorithm (4) show the implementation details of the Refined-GCG-GS.

Algorithm 3 Refined-GCG-GS

```
Require: $x_0, m, \beta_k, \gamma_k$
1: for $k = 0, 1, ..., K-1$ do
2:     $z_{k+1} = (1-\gamma_k)y_k + \gamma_k x_k$;
3:     $g_k = \nabla l(z_{k+1})$;
4:     $x_{k+1} = $ Re-GCG$(g_k, x_k, \beta_k, m)$;
5:     $y_{k+1} = (1-\gamma_k)y_k + \gamma_k x_{k+1}$;
6: end for
Ensure: $y_K$
```

Algorithm 4 Re-GCG:Refined-GCG-GS subroutine

```
Require: Input from outer loop: $g, u, \beta, m$; Sequence $\alpha_t$
1: let $\phi(x) = \langle g, x\rangle + \frac{\beta}{2}||x-u||_2^2$, $u_0 = u$;
2: for $t = 0, 1, ..., m-1$ do
3:     choose $a_t$ satisfy $\langle a_t, \nabla\phi(u_t)\rangle \leq \epsilon_t - \kappa^o(-\nabla\phi(u_t))$;
4:     $\theta_t = \arg\min_\theta \phi((1-\alpha_t)u_t + \theta a_t) + (1-\alpha_t)h(\rho_t) + \alpha_t h(\frac{\theta}{\alpha_t}))$;
5:     $\tilde{u}_{t+1} = (1-\alpha_t)u_t + \theta_t a_t$;
6:     $\tilde{\rho}_{t+1} = (1-\alpha_t)\rho_t + \theta_t$;
7:     $(u_{t+1}, \rho_{t+1})$ = Improve$(\tilde{u}_{t+1}, \tilde{\rho}_{t+1}, \phi, r)$;
8:     $\bar{u}_{t+1} = (1-\frac{2}{t+2})\bar{u}_t + \frac{2}{t+2}u_{t+1}$;
9: end for
10: Return: $u^+ = \bar{u}_m$;
```

Convergence Analysis:[4]

Theorem 2. *Let the sequence settings be $\beta_k = \frac{2L}{k+1}$, $\gamma_k = \frac{2}{k+2}$, $\alpha_t = \frac{2}{t+2}$ and the inner loop count $m = \left\lceil \frac{6K\left(D_s+\delta\kappa(x)\right)}{D_0}\right\rceil$, where D_s is an upper bound on the distance of the solution path, D_0 is the distance between x_0 and x, δ is the constant satisfies $\epsilon_t \leq \frac{\delta\beta_k\alpha_t}{2}$. Then the following results hold for the above algorithm for any x in $\mathcal{X}$.*
a) After K outer loops, the output y_K satisfies

$$F(y_K) - F(x) \leq \frac{6LD_0}{K(K+1)}; \tag{38}$$

[4]We provide the proof in an anonymous dropbox folder for interested reader. https://www.dropbox.com/sh/xhfmrxr0xss1mxl/AAC00-hytB5tncUssuKRMyxca?dl=0

for finding an ϵ solution, and the number of FO evaluation requires

$$K = \sqrt{\frac{6LD_0}{\epsilon}}. \tag{39}$$

b) The total number of LO evaluation T_K for finding an ϵ solution requires

$$T_K = \frac{36L(D_s + \delta\kappa(x))}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}. \tag{40}$$

Discussion:
The outer loop complexity in this subsection is exactly the same as the one in the previous subsection, despite all the approximation we make to efficiently evaluate LO. As a result, the count of gradient evaluation keeps unchanged. As a sharp comparison, the convergence rate of [Jaggi, 2011] is degenerated by a factor of 2 when d_k is evaluated approximately. Apparently, the approximations made to the LO lead to the increasing count of evaluations of gradient.

We point out that, by properly restarting GCG-GS, this algorithm can also obtain optimal count of gradient evaluation for strongly convex problem [Lan and Zhou, 2014]. In this paper, we only discuss the problem under convex assumption due to space limitation. In fact, the extension to strongly convex case is straightforward in some sense.

4 Experiment

In this section, we demonstrate the efficiency of the proposed algorithm by a CUR-like matrix factorization task ([Mahoney and Drineas, 2009]; [Mairal *et al.*, 2011]) regularized by group lasso penalty. This experiment was conducted by using MATLAB on a laptop computer of Intel Core i7 2.7GHz processor with 8 GB RAM.

Experimental Setup:
We consider the following CUR-like matrix factorization problem [Mairal *et al.*, 2011].

$$\min_X \frac{1}{2}||D - DXD||_F^2 + \lambda(\sum_i ||X_{i:}||_\infty + \sum_j ||X_{:j}||_\infty), \tag{41}$$

where D is the input data matrix, $X_{i:}$ and $X_{:j}$ denote the row vectors and column vectors, respectively, and $||\cdot||_\infty$ is the max norm. We set $\lambda = 5 \times 10^{-4}$ in our experiment. We utilized the following four real datasets as used in [Yu *et al.*, 2014]: SRBCT, Brain_Tumor_2, 9_Tumor and Leukemia [5], which are of sizes 83×2308, 50×10367, 60×5762, and 72×11225, respectively. For comparison, we utilized GCG_TUM algorithm in [Yu *et al.*, 2014] [6]. For our GCG-GS algorithm, we implemented the outer loop routine, which then called the same polar operator of GCG_TUM for inner loop subroutine. Hence, the improved performance is gained purely from the gradient sliding scheme. We set our inner loop estimation m to 3 for all four datasets. Other input sequences were assigned exactly as the theoretical part. Note that we did not compare with PG based methods because they have already been shown to be less efficient than GCG_TUM in [Yu *et al.*, 2014].

[5]Download from http://www.gems-system.org.

[6]Download from http://users.cecs.anu.edu.au/∼xzhang/GCG.

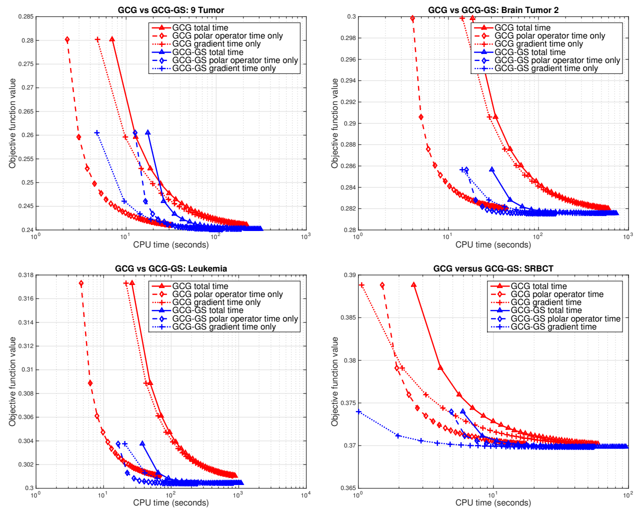

Figure 1: Objective function value versus total running time, polar operator time only, and gradient time only.

Experimental Result:
Figure 1 shows the experimental results, where three sets of plots are drawn: objective function value versus total running time, versus polar operator evaluation time only and versus gradient evaluation time only. We sampled every 30 iterations for GCG_TUM and every 10 outer loop iterations for GCG-GS. In general, our algorithm is much faster than GCG_TUM algorithm in terms of convergence speed, as illustrated by the curve of objective function value versus total running time. Also, our algorithm requires much less time on gradient evaluation to achieve certain decrease of objective function value on all four datasets. Finally, the time requirements for polar operator evaluation of our algorithm are similar to GCG_TUM on Brain Tumor 2 and SRBCT, superior than GCG_TUM on Leucamia and inferior than GCG_TUM on 9 Tumor.

Choice of Inner Iteration Count m:
In this subsection, we study the effect of different estimates of m. We run the GCG-GS algorithm with 400 outer loops on the four datasets, while vary the maximum inner iteration from 2 to 7. Figure 2 plots the objective function value versus number of iterations. With the different inner iteration number, the algorithm actually converges at similar outer loop count. To be specific, all lines begin to converge around 50 numbers of outer iteration. Although the algorithm converges to different objective function value with different m, the difference is below 0.005. In addition, we observed that $m = 3$ always yields relatively superior performance. Hence, m is not hard to tune practically.

5 Conclusion

In this paper, we have proposed a new algorithm under GCG framework. Our algorithm has optimal count of gradient evaluations as those PG method, which is an order superior than plain GCG methods. Also, it admits the incorporation of ef-

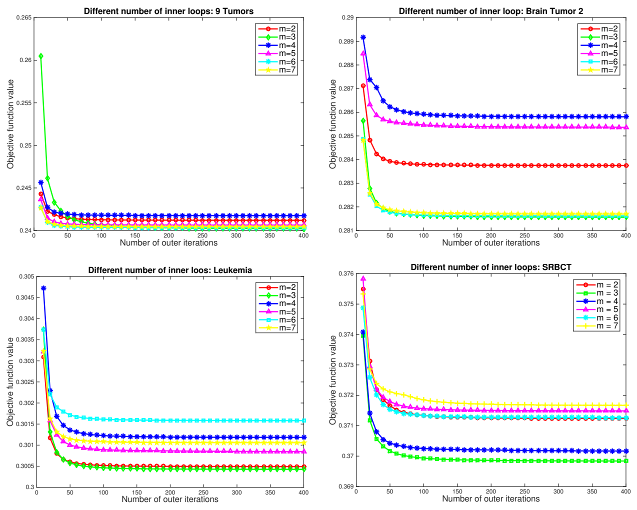

Figure 2: Objective function value versus number of outer iterations for different choice of inner loop count m.

ficient approximation techniques for accelerating the evaluation of linear operator that CGS lacks. Meanwhile, this count of gradient requirement remains unchanged. Experiment on a CUR-like matrix factorization task with group lasso penalty on four real datasets have demonstrated the efficiency of the proposed method.

Acknowledgments

This work was supported by the Faculty Research Grant of Hong Kong Baptist University under Projects: FRG2/14-15/075 and FRG1/14-15/041, and the National Science Foundation of China under Grant: 61272366.

References

[Beck and Teboulle, 2009] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[Bredies *et al.*, 2005] Kristian Bredies, Dirk Lorenz, and Peter Maass. *Equivalence of a generalized conditional gradient method and the method of surrogate functionals*. Zentrum für Technomathematik, 2005.

[Candès and Recht, 2009] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[Clarkson, 2010] Kenneth L Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, 6(4):63, 2010.

[Dudik *et al.*, 2012] Miro Dudik, Zaid Harchaoui, and Jérôme Malick. Lifted coordinate descent for learning with trace-norm regularization. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22, pages 327–336, 2012.

[Frank and Wolfe, 1956] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[Friedlander *et al.*, 2014] Michael P. Friedlander, Ives Macedo, and Ting Kei Pong. Gauge optimization and duality. *SIAM Journal on Optimization*, 24(4):1999–2022, 2014.

[Jaggi, 2011] Martin Jaggi. Convex optimization without projection steps. *arXiv preprint arXiv:1108.1170*, 2011.

[Jaggi, 2013] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning*, pages 427–435, 2013.

[Kim and Xing, 2009] Seyoung Kim and Eric P Xing. Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, 5(8):e1000587, 2009.

[Lacoste-Julien *et al.*, 2013] Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-Coordinate Frank-Wolfe optimization for structural SVMs. In *Proceedings of The 30th International Conference on Machine Learning*, pages 53–61, 2013.

[Lan and Zhou, 2014] Guanghui Lan and Yi Zhou. Conditional gradient sliding for convex optimization. *Technical Report*, 2014.

[Mahoney and Drineas, 2009] Michael W Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[Mairal *et al.*, 2011] Julien Mairal, Rodolphe Jenatton, Guillaume Obozinski, and Francis Bach. Convex and network flow optimization for structured sparsity. *The Journal of Machine Learning Research*, 12:2681–2720, 2011.

[Nesterov, 2013] Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

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

[Yu *et al.*, 2014] Yaoliang Yu, Xinhua Zhang, and Dale Schuurmans. Generalized conditional gradient for sparse estimation. *arXiv preprint arXiv:1410.4828*, 2014.

[Yuan and Lin, 2006] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[Zhang *et al.*, 2013] Xinhua Zhang, Yao-Liang Yu, and Dale Schuurmans. Polar operators for structured sparse estimation. In *Advances in Neural Information Processing Systems*, pages 82–90, 2013.