2019/22

DP

Yurii Nesterov and Mihai I. Florea

Gradient methods with memory

CORE
CENTER FOR OPERATIONS RESEARCH AND ECONOMETRICS

CORE

Voie du Roman Pays 34, L1.03.01

B-1348 Louvain-la-Neuve

Tel (32 10) 47 43 04

Email: immaq-library@uclouvain.be

https://uclouvain.be/en/research-institutes/lidam/core/discussion-papers.html

CORE DISCUSSION PAPER

2019/22

Gradient Methods with Memory

Yurii Nesterov * and Mihai I. Florea†

November 26, 2019

Abstract

In this paper, we consider gradient methods for minimizing smooth convex functions, which employ the information obtained at the previous iterations in order to accelerate the convergence towards the optimal solution. This information is used in the form of piece-wise linear model of the objective function, which provides us with much better prediction abilities as compared with the standard linear model. To the best of our knowledge, this approach was never really applied in Convex Minimization to differentiable functions in view of the high complexity of the corresponding auxiliary problems. However, we show that all necessary computations can be done very efficiently. Consequently, we get new optimization methods, which are better than the usual Gradient Methods both in the number of calls of oracle and in the computational time. Our theoretical conclusions are confirmed by preliminary computational experiments.

Keywords: Convex Optimization, gradient methods, relative smoothness, rate of convergence, piece-wise linear model.

*Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL). E-mail: Yurii.Neterov@uclouvain.be. Research results presented in this paper were obtained in the framework of ERC Advanced Grant 788368.

†INMA, Catholic University of Louvain. Email: Mihai.Florea@uclouvain.be.

1 Introduction

Motivation. The first-order gradient methods for minimizing smooth convex functions generate the sequence of test point based on the information obtained from the oracle, the function values and the gradients. Most of the methods, either use the information from the last test point, or accumulate it in the form of an aggregated linear function (see, for example, Chapter 2 in [4]). This approach is very different from the technique used in Nonsmooth Optimization, where the piece-wise linear model of the objective is a standard and powerful tool. It is enough to mention Bundle Method, Level Method, cutting plane schemes, etc. The reason for this situation is quite clear. The presence of piece-wise linear models in the auxiliary problems, which we need to solve at each iteration of the method, usually significantly increases the complexity of corresponding computations. This is acceptable in Nonsmooth Optimization, which has a reputation of a difficult field. In the contrast, Smooth Optimization admits very simple and elegant schemes, with very small computational cost of each iteration, preventing us from introducing there such a heavy machinery.

The main goal of this paper is the demonstration that the above situation is not as clear as it looks like. We will show that the Gradient Method, equipped with a piece-wise linear model of the objective function, has much more chances to accelerate on the particular optimization problems. At the same time, it appears that the corresponding auxiliary optimization problems can be easily solved by an appropriate version of Frank-Wolfe algorithm. All our claims are supported by the complexity analysis. In the end, we present preliminary computational results, which show that very often the new schemes have much better computational time.

Contents. In Section 2, we analyze the Gradient Method with Memory as applied to the composite form of smooth convex optimization problem [3]. In order to measure the level of smoothness of our objective function, we introduce the *relative smoothness* condition [1, 5], based on arbitrary strictly convex distance function. The main novelty here is the piece-wise linear model of the objective function, formed around the current test point. We analyze the corresponding auxiliary optimization problem and propose a condition on its approximate solution which does not destroy the rate of convergence of the algorithm. In Section 3, we analyze complexity of solving the auxiliary optimization problem (more precisely, its anti-dual problem) using Frank-Wolfe algorithm. In this section, we restrict ourselves by the strongly convex distance functions. We show that our auxiliary problem can be easily solved by Frank-Wolfe method. Its complexity is proportional to the maximal squared norm of the gradient in the current model of the objective divided by the desired accuracy.

In Section 4, we specify our complexity results for the Euclidean setup, when all distances are measured by a Euclidean norm. We show that, for some strategies of updating the piece-wise linear model, the complexity of the auxiliary computations is very low.

Finally, in Section 5 we present preliminary computational results. We compare the usual Gradient Method with two gradient methods with memory, which use different strategies for updating the piece-wise model of the objective function. Our conclusion is that the new schemes are always better, both in terms of the calls of oracle and in the total computational time.

Notation and Generalities. In what follows, denote by $\mathbb{E}$ a finite-dimensional real vector space and by $\mathbb{E}^*$ its dual space, the space of linear functions on $\mathbb{E}$. The value of function $s \in \mathbb{E}^*$ at point $x \in \mathbb{E}$ is denoted by $\langle s, x \rangle$. Let us fix some arbitrary (possibly non-Euclidean) norm $\|\cdot\|$ on the space $\mathbb{E}$ and define the dual norm $\|\cdot\|_*$ on $\mathbb{E}^*$ in the standard way:

$$\|s\|_* \stackrel{\text{def}}{=} \sup_{h \in \mathbb{E}} \{\langle s, h \rangle : \|h\| \leq 1\}.$$

Let us choose a simple strictly convex closed prox-function $d(\cdot)$, which is differentiable at the interior of its domain. This function must be strongly convex:

$$d(y) > d(x) + \langle \nabla d(x), y - x \rangle, \quad x \in \operatorname{int}(\operatorname{dom} d),\ y \in \operatorname{dom} d. \tag{1.1}$$

Using this function, we can define the *Bregmann distance* between two points x and y:

$$\beta_d(x, y) = d(y) - d(x) - \langle \nabla d(x), y - x \rangle, \quad x \in \operatorname{int}(\operatorname{dom} d),\ y \in \operatorname{dom} d. \tag{1.2}$$

Clearly, $\beta_d(x, y) \stackrel{(1.1)}{>} 0$ for $x \neq y$ and $\beta_d(x, x) = 0$.

We will use Bregmann distances for measuring the level of *relative smoothness* of convex functions (see [5]). Namely, for a differentiable closed convex function f with $\operatorname{dom} f \subseteq \operatorname{dom} d$ we define two constants, $L_d(f) \geq \mu_d(f) \geq 0$, such that

$$\begin{array}{rcl} f(y) - f(x) - \langle \nabla f(x), y - x \rangle & \geq & \mu_d(f) \beta_d(x, y), \\ f(y) - f(x) - \langle \nabla f(x), y - x \rangle & \leq & L_d(f) \beta_d(x, y), \end{array} \qquad \forall x, y \in \operatorname{dom} f, \tag{1.3}$$

(see [1] and [5] for definitions, motivations, and examples). Using these constants, we can define the *relative condition number*

$$\kappa_d(f) \quad = \quad \tfrac{\mu_d(f)}{L_d(f)}.$$

2 Gradient Method with Memory

In this paper, we are solving the following composite minimization problem:

$$\min_{x \in \operatorname{dom} \psi} \Big\{ F(x) \equiv f(x) + \psi(x) \Big\}, \tag{2.1}$$

where function f satisfies the relative smoothness condition (1.3), possibly with $\mu_d(f) = 0$. The function $\psi : \mathbb{E} \to \mathbb{R} \cup \{+\infty\}$ is a proper closed convex function, which is *simple*, but possibly non-differentiable or even discontinuous. We assume that $\operatorname{dom} f \supseteq \operatorname{dom} \psi$. We also assume that a solution $x^* \in \operatorname{dom} \psi$ of problem (2.1) does exist, denoting $F^* = F(x^*)$.

The simplest method for solving the problem (2.1) is the usual Gradient Method:

$$\begin{array}{l} \text{Choose } x_0 \in \operatorname{int}(\operatorname{dom} \psi). \text{ For } k \geq 0, \text{ iterate:} \\ x_{k+1} = \arg \min_{y \in \operatorname{dom} \psi} \left\{ f(x_k) + \langle \nabla f(x_k), y - x_k \rangle + \psi(y) + L \beta_d(x_k, y) \right\}. \end{array} \tag{2.2}$$

The constant L in this method has to be big enough in order to ensure

$$f(x_{k+1}) \quad \leq \quad f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + L\beta_d(x_k, x_{k+1}).$$

In view of (1.3), this is definitely true for $L \geq L_d(f)$. However, we are interested in choosing L as small as possible since this would significantly increase the rate of convergence of the scheme.

Method (2.2) is based on the simplest *linear model* of function $f(\cdot)$ around the point x_k. In our paper, we suggest to replace it by a piece-wise linear model, defined by the information collected at other test points.

Namely, for each $k \geq 0$ define a *discrete set* $\mathcal{Z}_k$ of m_k feasible points ($m_k \geq 1$):

$$\mathcal{Z}_k = \{z_i \in \operatorname{dom} \psi,\ i = 1, \ldots, m_k\}.$$

Then we can use a more sophisticated model of the smooth part of the objective function,

$$f(y) \geq \ell_k(y) \stackrel{\text{def}}{=} \max_{z_i \in \mathcal{Z}_k} \{f(z_i) + \langle \nabla f(z_i), y - z_i \rangle\}, \quad y \in \operatorname{dom} \psi. \tag{2.3}$$

This model is always better than the initial linear model provided that

$$x_k \in \mathcal{Z}_k. \tag{2.4}$$

In what follows, we always assume that this condition is satisfied.

Thus, we come to the following natural generalization of the method (2.2), which we call the *Gradient Method with Memory* (GMM):

Choose $x_0 \in \operatorname{int}(\operatorname{dom} \psi)$.

For $k \geq 0$**, iterate:**

$$x_{k+1} = \arg \min_{y \in \operatorname{dom} \psi} \{\ell_k(y) + \psi(y) + L\beta_d(x_k, y)\}. \tag{2.5}$$

Remark 1 *Note that for any $x \in \operatorname{dom} f$ we have*

$$f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + L\beta_d(x_k, x) \stackrel{(2.4)}{\leq} \ell_k(x) + L\beta_d(x_k, x)$$

$$\stackrel{(2.3)}{\leq} f(x) + L\beta_d(x_k, x) \stackrel{(1.3)}{\leq} f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + (L + L_d(f))\beta_d(x_k, x).$$

Therefore, we can count on a better convergence of method (2.5) only if we will be able to choose the parameter L significantly smaller than $L_d(f)$.

At each iteration of method (2.5), we need to solve a nontrivial auxiliary minimization problem. Therefore, the practical efficiency of this method crucially depends on the complexity of this computation. In what follows, we suggest to solve this problem approximately using a special method for its dual problem.

Let us start from presenting the corresponding technique. For the sake of notation, we omit the index of the iteration. Thus, our auxiliary problem is as follows:

$$\min_{y \in \mathrm{dom}\, \psi} \max_{\lambda \in \Delta_m} \left\{ \sum_{i=1}^{m} \lambda^{(i)} [f_i + \langle g_i, y - z_i \rangle] + \psi(y) + L \beta_d(\bar{x}, y) \right\},$$

where $f_i = f(z_i)$, $g_i = \nabla f(z_i)$, $i = 1, \ldots, m$, and Δ_m is the standard simplex in $\mathbb{R}^m$. Introducing now the vector $f_* \in \mathbb{R}^m$ with coordinates

$$f_*^{(i)} = \langle g_i, z_i \rangle - f_i, \quad i = 1, \ldots, m, \tag{2.6}$$

we get the following representation of our problem:

$$\min_{y \in \mathrm{dom}\, \psi} \max_{\lambda \in \Delta_m} \left\{ \langle \lambda, G^T y - f_* \rangle + \psi(y) + L \beta_d(\bar{x}, y) \right\}, \tag{2.7}$$

where $G = (g_1, \ldots, g_m) \in \mathbb{E}^* \times \mathbb{R}^m$. Note that the pay-off function in this saddle point problem can be written as follows:

$$\begin{aligned} & \langle \lambda, G^T y - f_* \rangle + \psi(y) + L \beta_d(\bar{x}, y) \\ = \; & \langle \lambda, G^T y - f_* \rangle + \psi(y) + L [d(y) - d(\bar{x}) - \langle \nabla d(\bar{x}), y - \bar{x} \rangle] \\ = \; & L d(y) - \langle L \nabla d(\bar{x}) - G\lambda, y \rangle + \psi(y) - \langle \lambda, f_* \rangle + L [\langle \nabla d(\bar{x}), \bar{x} \rangle - d(\bar{x})]. \end{aligned}$$

Hence, we need to introduce the following dual function:

$$\Phi_L(s) = \max_{y \in \mathrm{dom}\, \psi} \left\{ \langle s, y \rangle - L d(y) - \psi(y) \right\}, \quad s \in \mathbb{E}^*. \tag{2.8}$$

Our main joint assumption on functions $d(\cdot)$ and $\psi(\cdot)$ is as follows.

Assumption 1 *For any $L > 0$, function $\Phi_L(s)$ is defined at any $s \in \mathbb{E}^*$.*

This can be ensured, for example, by strong convexity of function $d(\cdot)$, or by the boundedness of $\mathrm{dom}\, \psi$, or in many other ways.

Since the objective function in the definition (2.8) is strictly concave, its solution

$$y_L^*(s) = \arg \max_{y \in \mathrm{dom}\, \psi} \left\{ \langle s, y \rangle - L d(y) - \psi(y) \right\}$$

is uniquely defined for any $s \in \mathbb{E}^*$. Moreover, function $\Phi_L(\cdot)$ is differentiable and

$$\nabla \Phi_L(s) = y_L^*(s), \quad s \in \mathbb{E}^*. \tag{2.9}$$

Now we can write down the problem anti-dual to (2.7)

$$\xi_L^* \stackrel{\mathrm{def}}{=} \min_{\lambda \in \Delta_m} \left\{ \xi_L(\lambda) \stackrel{\mathrm{def}}{=} \Phi_L(L \nabla d(\bar{x}) - G\lambda) + \langle \lambda, f_* \rangle + \alpha \right\}, \tag{2.10}$$

where $\alpha = L[d(\bar{x}) - \langle \nabla d(\bar{x}), \bar{x} \rangle]$. This is a convex optimization problem with differentiable objective function. Our second main assumption is as follows.

Assumption 2 *Function $\Phi_L(\cdot)$ in the problem (2.10) is easily computable.*

We will discuss the reasonable strategies for finding an approximate solution to problem (2.10) in Section 3. At this moment, it is enough to assume that we are able to compute a point $\bar{\lambda} = \bar{\lambda}(\bar{x}, \mathcal{Z}, L)$ such that

$$\langle \bar{\lambda} - \lambda, \nabla \xi_L(\bar{\lambda}) \rangle \quad \leq \quad \delta, \quad \forall \lambda \in \Delta_m, \tag{2.11}$$

where $\delta \geq 0$ is some tolerance parameter. Clearly, if $\delta = 0$, then $\bar{\lambda}$ is the optimal solution to the problem (2.10). Note that condition (2.11) ensures also a small functional gap:

$$\xi_L(\bar{\lambda}) - \xi_L^* \quad = \quad \max_{\lambda \in \Delta_m} [\xi_L(\bar{\lambda}) - \xi_L(\lambda)] \; \leq \; \max_{\lambda \in \Delta_m} \langle \bar{\lambda} - \lambda, \nabla \xi_L(\bar{\lambda}) \rangle \overset{(2.11)}{\leq} \delta. \tag{2.12}$$

Condition (2.11) immediately leads to the following result.

Lemma 1 *Let $\bar{\lambda} \in \Delta_m$ satisfies condition (2.11). Then for $\bar{s} = L \nabla d(\bar{x}) - G\bar{\lambda}$ we have*

$$\sum_{i=1}^{m} \bar{\lambda}^{(i)} [f_i + \langle g_i, y_L^*(\bar{s}) - z_i \rangle] \quad \geq \quad \max_{1 \leq i \leq m} [f_i + \langle g_i, y_L^*(\bar{s}) - z_i \rangle] - \delta. \tag{2.13}$$

Proof:
Indeed, $\nabla \xi_L(\bar{\lambda}) = f_* - G^T y_L^*(\bar{s})$. Thus, inequality (2.11) can be rewritten as follows:

$$\langle \bar{\lambda}, G^T y_L^*(\bar{s}) - f_* \rangle \quad \geq \quad \langle \lambda, G^T y_L^*(\bar{s}) - f_* \rangle - \delta, \quad \forall \lambda \in \Delta_m.$$

It remains to note that

$$(G^T y_L^*(\bar{s}) - f_*)^{(i)} = f_i + \langle g_i, y_L^*(\bar{s}) - z_i \rangle, \quad i = 1, \ldots, m. \qquad \square$$

Now we are able to analyze one iteration of the inexact version of method (2.5).

Input:	Point $\bar{x} \in \operatorname{int}(\operatorname{dom} \psi)$, set of test points $\mathcal{Z}$, containing $\bar{x}$, constant $L > 0$, and tolerance $\delta \geq 0$.	(2.14)
Iteration:	Using the input data, form the optimization problem (2.10) and compute its approximate solution $\bar{\lambda}$ satisfying condition (2.11).	
Output:	Points $\bar{s} = L \nabla d(\bar{x}) - G\bar{\lambda}$ and $x_+ = y_L^*(\bar{s})$.	

Theorem 1 *1. Let point x_+ be generated by one iteration (2.14) of Inexact Gradient Method with Memory (IGMM), and let $L \geq L_d(f)$. Then for any $y \in \operatorname{dom} \psi$ we have:*

$$\beta(x_+, y) \quad \leq \quad \beta(\bar{x}, y) + \tfrac{1}{L} [F(y) - F(x_+) + \delta]. \tag{2.15}$$

2. Let $\beta_d(z_i, y) \geq \beta_d(\bar{x}, y)$, $i = 1, \ldots, m$. Then

$$\beta(x_+, y) \quad \leq \quad \left(1 - \tfrac{1}{L} \mu_d(f)\right) \beta(\bar{x}, y) + \tfrac{1}{L} [F(y) - F(x_+) + \delta]. \tag{2.16}$$

Proof:
Note that

$$\begin{aligned}
&\beta(x_+, y) - \beta(\bar{x}, y) \\
\overset{(1.2)}{=}\ & d(y) - d(x_+) - \langle \nabla d(x_+), y - x_+ \rangle - d(y) + d(\bar{x}) + \langle \nabla d(\bar{x}), y - \bar{x} \rangle \\
\overset{(1.2)}{=}\ & \langle \nabla d(\bar{x}) - \nabla d(x_+), y - x_+ \rangle - \beta(\bar{x}, x_+).
\end{aligned}$$

The first-order optimality condition defining the point x_+ is as follows:

$$\langle \bar{s} - L\nabla d(x_+), y - x_+ \rangle - \psi(y) \leq -\psi(x_+), \quad \forall y \in \operatorname{dom} \psi.$$

Since $\bar{s} = L\nabla d(\bar{x}) - G\bar{\lambda}$, it can be written in the following form:

$$\psi(x_+) \leq \quad \psi(y) + \langle G\bar{\lambda} + L(\nabla d(x_+) - \nabla d(\bar{x})), y - x_+ \rangle, \quad \forall y \in \operatorname{dom} \psi.$$

Hence,

$$\beta(x_+, y) - \beta(\bar{x}, y) \quad \leq \quad \tfrac{1}{L} \left[\psi(y) - \psi(x_+) + \langle G\bar{\lambda}, y - x_+ \rangle \right] - \beta(\bar{x}, x_+).$$

Note that

$$\begin{aligned}
\langle G\bar{\lambda}, y - x_+ \rangle \quad &= \quad \langle \bar{\lambda}, G^T(y - x_+) \rangle \ = \ \sum_{i=1}^{m} \bar{\lambda}^{(i)} \langle g_i, y - x_+ \rangle \\
&= \quad \sum_{i=1}^{m} \bar{\lambda}^{(i)} [\langle g_i, z_i - x_+ \rangle + \langle g_i, y - z_i \rangle] \\
&\overset{(1.3)}{\leq} \quad \sum_{i=1}^{m} \bar{\lambda}^{(i)} [\langle g_i, z_i - x_+ \rangle + f(y) - f_i - \mu_d(f) \beta_d(z_i, y)] \\
&= \quad f(y) - \sum_{i=1}^{m} \bar{\lambda}^{(i)} [f_i + \langle g_i, x_+ - z_i \rangle] - \mu_d(f) \sum_{i=1}^{m} \bar{\lambda}^{(i)} \beta_d(z_i, y).
\end{aligned}$$

Under conditions of Item 1, we drop the last term in the above inequality and by Lemma 1 obtain the following:

$$\begin{aligned}
\beta(x_+, y) - \beta(\bar{x}, y) \quad &\leq \quad \tfrac{1}{L} \left[F(y) - \sum_{i=1}^{m} \bar{\lambda}^{(i)} [f_i + \langle g_i, x_+ - z_i \rangle] - \psi(x_+) - L\beta(\bar{x}, x_+) \right] \\
&\leq \quad \tfrac{1}{L} \left[F(y) - \max_{1 \leq i \leq m} [f_i + \langle g_i, x_+ - z_i \rangle] - \psi(x_+) - L\beta(\bar{x}, x_+) + \delta \right].
\end{aligned}$$

Under conditions of Item 2, by the same reason we get

$$\begin{aligned}
&\beta(x_+, y) - \beta(\bar{x}, y) \\
&\leq \ \tfrac{1}{L} \left[F(y) - \mu_d(f) \beta_d(\bar{x}, y) - \sum_{i=1}^{m} \bar{\lambda}^{(i)} [f_i + \langle g_i, x_+ - z_i \rangle] - \psi(x_+) - L\beta(\bar{x}, x_+) \right] \\
&\leq \ \tfrac{1}{L} \left[F(y) - \mu_d(f) \beta_d(\bar{x}, y) - \max_{1 \leq i \leq m} [f_i + \langle g_i, x_+ - z_i \rangle] - \psi(x_+) - L\beta(\bar{x}, x_+) + \delta \right].
\end{aligned}$$

In both cases, since $\bar{x} \in \mathcal{Z}$, we have

$$\begin{aligned} \max_{1 \le i \le m} [f_i + \langle g_i, x_+ - z_i \rangle] + L\beta(\bar{x}, x_+) &\geq f(\bar{x}) + \langle \nabla f(\bar{x}), x_+ - \bar{x} \rangle + L\beta(\bar{x}, x_+) \\ &\overset{(1.3)}{\geq} f(x_+). \end{aligned}$$

Thus, we obtain inequalities (2.15) and (2.16). □

Remark 2 *In the end of the proof, we have seen that the statement of the Theorem 1 remains valid if the condition $L \geq L_d(f)$ is replaced by the following:*

$$\max_{1 \le i \le m} [f_i + \langle g_i, x_+ - z_i \rangle] + L\beta(\bar{x}, x_+) \quad \geq \quad f(x_+). \tag{2.17}$$

Denote the output of the iteration (2.14) by $x_{\delta,L}(\bar{x}, \mathcal{Z})$. Then we can define the following Inexact Gradient Method with Memory.

Choose $x_0 \in \operatorname{int}(\operatorname{dom} \psi)$, $\delta \geq 0$ and $L > 0$.

For $k \geq 0$**, iterate:**

1). Choose the set $\mathcal{Z}_k$ containing x_k.

2). Compute $x_{k+1} = x_{\delta,L}(x_k, \mathcal{Z}_k)$. (2.18)

Let us describe the rate of convergence of this process.

Theorem 2 *Let sequence $\{x_k\}_{k \geq 1}$ be generated by IGMM (2.18) with $L \geq L_d(f)$. Then, for any $T \geq 1$ and $y \in \operatorname{dom} \psi$ we have*

$$\frac{1}{T} \sum_{k=1}^{T} F(x_k) \quad \leq \quad F(y) + \frac{L}{T} \beta_d(x_0, y) + \delta. \tag{2.19}$$

Proof:
Indeed, in view of inequality (2.15), we have

$$\beta_d(x_{k+1}, y) \quad \leq \quad \beta_d(x_k, y) + \frac{1}{L} \left[F(y) - F(x_{k+1}) + \delta \right], \quad k \geq 0.$$

Summing up these inequalities for $k = 0, \ldots, T-1$, we get inequality (2.19). □

In the above result, the only restriction for the sets $\mathcal{Z}_k$ is the inclusion (2.4). If we apply a more accurate strategy of choosing $\mathcal{Z}_k$, we can get for this scheme a finer estimate of its rate of convergence.

Theorem 3 *Let sequence $\{x_k\}_{k\geq 1}$ be generated by IGMM (2.18) with $L \geq L_d(f)$. Assume that besides the condition (2.4), the sets $\mathcal{Z}_k$ satisfy also the following condition:*

$$\mathcal{Z}_k \;\subseteq\; \{x_0, \ldots, x_k\}, \quad k \geq 0. \tag{2.20}$$

Suppose that $\mu_d(f) > 0$ and for all k, $1 \leq k \leq T$, we have

$$F(x_k) - F^* \geq \delta. \tag{2.21}$$

Then for $\Delta_T^* = \min\limits_{1\leq k\leq T} [F(x_k) - F^*]$ we get the following rate of convergence

$$\begin{aligned} \Delta_T^* &\leq \delta + \tfrac{(1-\gamma)^T \mu_d(f)}{1-(1-\gamma)^T}\, \beta_d(x_0, x^*) \\ &\leq \delta + \tfrac{\mu_d(f)}{e^{\gamma T}-1}\, \beta_d(x_0, x^*) \;\leq\; \delta + \tfrac{L}{T}\beta_d(x_0, x^*), \end{aligned} \tag{2.22}$$

where $\gamma = \frac{1}{L}\, \mu_d(f)$.

Proof:
In view of assumption (2.21) and inequality (2.15), we have

$$\beta_d(x_{k+1}, x^*) \;\leq\; \beta_d(x_k, x^*), \quad 0 \leq k \leq T-1.$$

Therefore, in view of inequality (2.16), for $r_k \stackrel{\text{def}}{=} \beta_d(x_k, x^*)$ and all $k = 0, \ldots, T-1$ we have

$$r_{k+1} \;\leq\; (1-\gamma) r_k - \tfrac{1}{L}[F(x_{k+1}) - F^* - \delta] \;\leq\; (1-\gamma) r_k - \tfrac{1}{L}[\Delta_T^* - \delta].$$

Applying this inequality recursively, we get

$$\tfrac{1}{L}[\Delta_T^* - \delta] \tfrac{1-(1-\gamma)^T}{1-(1-\gamma)} \;\leq\; (1-\gamma)^T r_0.$$

This can be rewritten as

$$\Delta_T^* \;\leq\; \delta + \tfrac{L\gamma(1-\gamma)^T}{1-(1-\gamma)^T}\beta_d(x_0, x^*) \;=\; \delta + \tfrac{(1-\gamma)^T \mu_d(f)}{1-(1-\gamma)^T}\beta_d(x_0, x^*). \qquad \square$$

Note that the rate of convergence given by inequality (2.22) is continuous as $\mu_d(f) \to 0$.

As we have mentioned in Remark 1, it is important to adjust the value of the constant L during the minimization process. Therefore we present an adaptive version of the

method (2.18).

Choose $x_0 \in \mathrm{int}\,(\mathrm{dom}\,\psi)$, $\delta \geq 0$, and some $L_0 \in (0, L_d(f)]$.

For $k \geq 0$**, iterate:**

1). Choose the set $\mathcal{Z}_k$ containing x_k.

2). Find the smallest integer $i_k \geq 0$ such that

for the point $x_k^+ = x_{\delta, 2^{i_k} L_k}(x_k, \mathcal{Z}_k)$ we have

$$f(x_k^+) \overset{(2.3)}{\leq} \ell_k(x_k^+) + L_k \beta_d(x_k, x_k^+).$$

3). Set $x_{k+1} = x_k^+$ and $L_{k+1} = 2^{i_k - 1} L_k$. (2.23)

The rate of convergence of this algorithm can be established exactly in the same way as that one of method (2.18). The main fact is that during the minimization process we always have

$$L_k \leq 2L_d(f), \quad k \geq 0.$$

Therefore, for any $y \in \mathrm{dom}\, psi$ we will have

$$\beta_d(x_{k+1}, y) \;\leq\; \beta_d(x_k, y) + \tfrac{1}{2L_d(f)}[F(y) - F(x_{k+1}) + \delta]$$

with corresponding consequences for the rate of convergence. At the same time, the average number of calls of oracle at each iteration of this method is bounded by two (see [4] for justification details).

3 Getting approximate solution of anti-dual problem

Complexity of solving the auxiliary problem (2.10) crucially depends on the properties of the prox-function $d(\cdot)$. In the previous section, we assumed its strict convexity and solvability of problem (2.8) (see Assumption 1). It is time now to make a stronger assumption, which ensures these two properties.

Assumption 3 *Function $d(\cdot)$ is differentiable in the interior of its domain and strongly convex with convexity parameter one:*

$$d(y) \;\geq\; d(x) + \langle \nabla d(x), y - x \rangle + \tfrac{1}{2}\|y - x\|^2, \quad x \in \mathrm{int}\,(\mathrm{dom}\, d),\ y \in \mathrm{dom}\, d. \tag{3.1}$$

Clearly, for all $x, y \in \mathrm{int}\,(\mathrm{dom}\, d)$ we have

$$\beta_d(x, y) \;\geq\; \tfrac{1}{2}\|y - x\|^2. \tag{3.2}$$

The main consequence of Assumption 3 is Lipschitz continuity of the gradient of function $\Phi(\cdot)$. Since usually this fact is proved for a function $\psi(\cdot)$ being an indicator function of a closed convex set, we provide it with a simple proof.

Lemma 2 *Let function $d(\cdot)$ satisfy Assumption 3. Then the gradient $\nabla\Phi_L(s) = y_L^*(s)$, $s \in \mathbb{E}^*$, is Lipschitz continuous:*

$$\|\nabla\Phi_L(s_1) - \nabla\Phi_L(s_2)\| \;\leq\; \tfrac{1}{L}\|s_1 - s_2\|_*, \quad s_1, s_2 \in \mathbb{E}^*. \tag{3.3}$$

Proof:
Let us write down the first-order optimality conditions for optimization problems defining the points $y_1 \stackrel{\text{def}}{=} y_L^*(s_1)$ and $y_2 \stackrel{\text{def}}{=} y_L^*(s_2)$:

$$\langle s_1 - L[\nabla d(y_1) - \nabla d(\bar{x})], y - y_1\rangle - \psi(y) \;\leq -\psi(y_1), \quad \forall y \in \operatorname{dom}\psi,$$

$$\langle s_2 - L[\nabla d(y_2) - \nabla d(\bar{x})], y - y_2\rangle - \psi(y) \;\leq -\psi(y_2), \quad \forall y \in \operatorname{dom}\psi.$$

Taking in the first inequality $y = y_2$ and $y = y_1$ in the second one, and adding the results, we obtain

$$\langle s_1 - s_2, y_2 - y_1\rangle \;\leq\; L\langle \nabla d(y_1) - \nabla d(y_2), y_2 - y_1\rangle.$$

Thus,

$$\langle s_1 - s_2, y_1 - y_2\rangle \;\geq\; L\langle \nabla d(y_2) - \nabla d(y_1), y_2 - y_1\rangle \stackrel{(3.1)}{\geq} L\|y_1 - y_2\|^2.$$

Therefore, by Cauchy-Schwartz inequality, we get

$$\|y_L^*(s_1) - y_L^*(s_2)\| \;\leq\; \tfrac{1}{L}\|s_1 - s_2\|_*. \qquad \square$$

Thus, in this section, our main problem of interest is as follows:

$$\xi_L^* \;=\; \min_{\lambda \in \Delta_m} \left\{ \xi_L(\lambda) \stackrel{\text{def}}{=} \Phi_L(L\nabla d(\bar{x}) - G\lambda) + \langle \lambda, f_*\rangle + \alpha \right\}, \tag{3.4}$$

where $\alpha = L[d(\bar{x}) - \langle \nabla d(\bar{x}), \bar{x}\rangle]$. This is a convex optimization problem over a simplex, where objective function has Lipschitz-continuous gradient.

The most natural algorithm for solving the problem (2.10) is *Frank-Wolfe algorithm* [2] (or *Conditional Gradients Method*). For our problem, it looks as follows.

Set $\lambda_0 = \frac{1}{m}\bar{e}_m$.

For $k \geq 0$ **iterate:**

1. Compute the gradient $\nabla\xi_L(\lambda_k)$.
2. Compute $i_k = \arg\min\limits_{1 \leq i \leq m} \nabla_i \xi_L(\lambda_k)$.
3. Set $\lambda_{k+1} = \frac{k}{k+2}\lambda_k + \frac{2}{k+2}e_{i_k}$.

(3.5)

In this scheme, $\bar{e}_m \in \mathbb{R}^m$ is the vector of all ones, and e_i is ith coordinate vector in $\mathbb{R}^m$.

In order to estimate the rate of convergence of this method, we introduce the following accuracy measure:

$$\delta_L(\bar{\lambda}) \;=\; \max_{\lambda \in \Delta_m} \langle \nabla \xi_L(\bar{\lambda}), \bar{\lambda} - \lambda \rangle.$$

For the sequence $\{\lambda_k\}_{k \geq 0}$ generated by the method (3.5), denote

$$\delta_L^*(T) \;=\; \min_{0 \leq k \leq T} \delta_L(\lambda_k), \quad T \geq 0.$$

For estimating the rate of convergence of method (3.5), we need to choose an appropriate norm in $\mathbb{R}^m$. Since the feasible set of the problem (2.10) is the standard simplex, it is reasonable to use ℓ_1-norm:

$$\|\lambda\|_1 \;=\; \sum_{i=1}^m |\lambda^{(i)}|, \quad \lambda \in \mathbb{R}^m.$$

Then, for measuring gradients of function $\xi_L(\cdot)$, we can use ℓ_∞-norm:

$$\|\lambda\|_\infty \;=\; \max_{1 \leq i \leq m} |\lambda^{(i)}|, \quad \lambda \in \mathbb{R}^m.$$

In this case, the Lipschitz constant for the gradients of function $\xi_L(\cdot)$ can be estimated as follows:

$$\begin{aligned}
& \|\nabla \xi_L(\lambda_1) - \nabla \xi_L(\lambda_2)\|_\infty \\
= \;& \max_{1 \leq i \leq m} |\langle g_i, \nabla \Phi(L \nabla d(\bar{x}) - G\lambda_2) - \nabla \Phi(L \nabla d(\bar{x}) - G\lambda_1) \rangle| \\
\leq \;& \max_{1 \leq i \leq m} \|g_i\|_* \cdot \|\nabla \Phi(L \nabla d(\bar{x}) - G\lambda_2) - \nabla \Phi(L \nabla d(\bar{x}) - G\lambda_1)\| \\
\overset{(3.3)}{\leq} \;& \max_{1 \leq i \leq m} \|g_i\|_* \cdot \tfrac{1}{L} \|G(\lambda_1 - \lambda_2)\|_* \;\leq\; \tfrac{1}{L} \max_{1 \leq i \leq m} \|g_i\|_*^2 \cdot \|\lambda_1 - \lambda_2\|_1.
\end{aligned}$$

Thus, the gradients of function $\xi_L(\cdot)$ are Lipschitz continuous with the constant

$$L(\xi_L) \;=\; \tfrac{1}{L} \max_{1 \leq i \leq m} \|g_i\|_*^2. \tag{3.6}$$

Since the diameter of the standard simplex in $\mathbb{R}^m$ in ℓ_1-norm is two, in accordance to the estimate (3.13) in [6], we can guarantee the following rate of convergence:

$$\delta_L^*(T) \;\leq\; \tfrac{18}{L \cdot T} \max_{1 \leq i \leq m} \|g_i\|_*^2, \quad t \geq 1. \tag{3.7}$$

(Here we replace the constant $\frac{136}{11 \ln 2}$ from [6] by a bigger value 18.) In accordance to the condition (2.11), this means that we need

$$N_L(\delta) \;=\; \tfrac{18}{L \cdot \delta} \max_{1 \leq i \leq m} \|g_i\|_*^2 \tag{3.8}$$

iterations of the method (3.5) in order to generate an appropriate dual solution $\bar{\lambda}$.

4 Unconstrained minimization in Euclidean setup

In this section we consider the simplest unconstrained minimization problem

$$f^* \;=\; \min_{x\in\mathbb{E}} f(x), \tag{4.1}$$

where $f(\cdot)$ is a smooth convex function. For measuring distances in $\mathbb{E}$, we introduce a Euclidean norm

$$\|x\| \;=\; \langle Bx, x\rangle^{1/2}, \quad x \in \mathbb{E},$$

where $B = B^* \succ 0$ is a linear operator from $\mathbb{E}$ to $\mathbb{E}^*$. Then the dual norm is defined as follows:

$$\|g\|_* \;=\; \langle g, B^{-1}g\rangle^{1/2}, \quad g \in \mathbb{E}^*.$$

Let us choose now the distance function $d(x) = \frac{1}{2}\|x\|^2$. Then the Bregmann distance is given by

$$\beta_d(x,y) \;=\; \tfrac{1}{2}\|x-y\|^2, \quad x, y \in \mathbb{E}.$$

In this case, the relative smoothness condition (1.3) is equivalent to strong convexity and Lipschitz continuity of the gradient:

$$\begin{array}{rcl} f(y) - f(x) - \langle \nabla f(x), y-x\rangle & \geq & \tfrac{1}{2}\mu_d(f)\|x-y\|^2, \\ f(y) - f(x) - \langle \nabla f(x), y-x\rangle & \leq & \tfrac{1}{2}L_d(f)\|x-y\|^2. \end{array} \qquad \forall x, y \in \operatorname{dom} f, \tag{4.2}$$

Let us write down now the specific form of the objective function $\xi_L(\cdot)$ in problem (2.10). Note that in our case

$$\Phi_L(s) \;=\; \max_{y\in\mathbb{E}}\{\langle s, y\rangle - \tfrac{1}{2}\|y\|^2\} \;=\; \tfrac{1}{2L}\|s\|_*^2, \quad s \in \mathbb{E}^*.$$

Therefore,

$$\xi_L(\lambda) = \tfrac{1}{2L}\|LB\bar{x} - G\lambda\|_*^2 + \langle \lambda, f_*\rangle + \alpha,$$

where $\alpha = -\frac{1}{2}L\|\bar{x}\|^2$. The gradient of function $\xi_L(\cdot)$ can be computed as follows:

$$\begin{array}{rcl} \nabla\xi_L(\lambda) & = & \tfrac{1}{L}G^*B^{-1}(G\lambda - LB\bar{x}) + f_*, \quad \lambda \in \mathbb{R}^m \\ & = & \tfrac{1}{L}Q\lambda - \bar{f}, \end{array} \tag{4.3}$$

where $Q = G^*B^{-1}G$ and $\bar{f} = G^*\bar{x} - f_*$. Note that

$$\bar{f}^{(i)} \;\overset{(2.6)}{=}\; f_i + \langle g_i, \bar{x} - z_i\rangle, \quad i = 1, \ldots, m.$$

Thus, in the Euclidean setup, our auxiliary problem (2.10) can be written as follows:

$$\min_{\lambda\in\Delta_m}\Big[\; \xi_L(\lambda) \;=\; \tfrac{1}{2L}\langle \lambda, Q\lambda\rangle - \langle \lambda, \bar{f}\rangle \;\Big]. \tag{4.4}$$

The stoping criterion (2.11) for this problem is as follows:

$$\langle \bar{\lambda}, \nabla\xi_L(\bar{\lambda})\rangle \;\overset{(4.3)}{=}\; \langle \bar{\lambda}, \tfrac{1}{L}Q\bar{\lambda} - \bar{f}\rangle \;\overset{(2.11)}{\leq}\; \delta + \min_{1\leq i\leq m} \nabla_i\xi_L(\bar{\lambda})$$

$$= \quad \delta + \min_{1\leq i\leq m} (\tfrac{1}{L}Q\bar{\lambda} - \bar{f})^{(i)}.$$

Note that the main output of the minimization process for problem (4.4) is

$$x_+ \;=\; y_*(LB\bar{x} - G\bar{\lambda}) \overset{(2.9)}{=} \tfrac{1}{L}B^{-1}(LB\bar{x} - G\bar{\lambda}) \;=\; \bar{x} - \tfrac{1}{L}B^{-1}G\bar{\lambda}.$$

Then

$$\bar{f} - \tfrac{1}{L}Q\bar{\lambda} \;=\; \bar{f} - \tfrac{1}{L}G^*B^{-1}G\bar{\lambda} \;=\; \bar{f} + G^*(x_+ - \bar{x}).$$

Hence, in the Euclidean case, stopping criterion (2.11) can be written as follows:

$$\sum_{i=1}^{m} \bar{\lambda}^{(i)}[f_i + \langle g_i, x_+ - z_i\rangle] \;\leq\; \delta + \max_{1\leq i\leq m}[f_i + \langle g_i, x_+ - z_i\rangle]. \tag{4.5}$$

Now we can estimate the computational expenses of the method (3.5) as applied to the auxiliary problem (4.4).

1. Computation of matrix Q: $O(m^2n)$ arithmetic operations. For certain strategies for *updating* the sets $\mathcal{Z}_k$, it can be reduced up to $O(mn)$ operations.
2. Computation of the vector $\bar{f}$: $O(mn)$ operations.
3. Computation of the initial gradient $u_0 = \frac{1}{L}\lambda_0 - \bar{f}$: $O(m^2)$ operations. For certain updating strategies it can be $O(m)$.
4. Expenses at each iterations:
   - Computing the index i_k: $O(m)$ operations.
   - Updating the point λ_k: $O(m)$ operations.
   - Updating the gradient $u_k = \frac{1}{L}Q\lambda_k - \bar{f}$: $O(m)$ operations.

Thus, taking into account the upper bound (3.8)for the number of iterations in method (3.5), obtain the following bound for the arithmetic complexity of problem (4.4) with reasonable updating strategies for the sets $\mathcal{Z}_k$:

$$O\left(mn + \tfrac{m}{L\cdot\delta}\max_{1\leq i\leq m}\|g_i\|_*^2\right). \tag{4.6}$$

Taking into account that we can expect that in the problem (4.1) we have

$$\tfrac{1}{2L_d(f)}\|\nabla f(z_i\|_*^2 \overset{(4.2)}{\leq} f(z_i) - f^* \;\to\; 0$$

as $i \to \infty$, the bound (4.6) looks very promising.

5 Numerical experiments

In this section we present preliminary computational results for method (2.23) as applied to the following unconstrained minimization problem:

$$\min_{x\in\mathbb{R}^n}\Big[\; f(x) \;=\; \mu\ln\left(\sum_{j=1}^{M} e^{(\langle a_j,x\rangle - b_j)/\mu}\right)\Big]. \tag{5.1}$$

The data defining this function is randomly generated in the following way. First of all, we generate a collection of random vectors

$$\hat{a}_1, \ldots, \hat{a}_M$$

with entries uniformly distributed in the interval $[-1, 1]$. Using the same distribution, we generate values b_j, $j = 1, \ldots, m$. Using this data, we form preliminary function

$$\hat{f}(x) \quad = \quad \mu \ln \left(\sum_{j=1}^{M} e^{(\langle a_j, x \rangle - b_j)/\mu} \right)$$

and compute $g = \nabla \hat{f}(0)$. Then, we define

$$a_j = \hat{a}_j - g, \quad j = 1, \ldots, n.$$

Clearly, in this case we have $\nabla f(0) = 0$, so the unique solution of our test problem (5.1) is $x^* = 0$. The starting point x_0 is chosen in accordance to the uniform distribution on the Euclidean sphere of radius one.

Thus, the problem (5.1) has three parameters, the dimension n, number of linear functions $M \geq n$, and the smoothness coefficient $\mu > 0$. In our experiments, we always choose $M = 6n$. Let us present our computational results for different values of n and μ.

IN the definition of method (2.23) we have some freedom in the choice of the bundle $\mathcal{Z}_k$. Let us bound its maximal size by a parameter $m \geq 1$. Then $m = 1$ corresponds to the usual Gradient Method. In the first series of our experiments (shown in Table 1 and Table 2) we always choose

$$m = n.$$

We also have some freedom in the updating strategy for the sets $\mathcal{Z}_k$. Clearly, at the firs m steps we can simply add all new points in the bundle. However, at the next iterations we need to decide on the strategy of replacement of the the old information. In our experiments we implemented two strategies:

- Cyclic replacement (Cyclic).
- Replacement of linear function with maximal norm of the gradient (Max-Norm).

The second strategy is motivated by the formula (3.6) for he Lipschitz constant of the gradient of function $\xi_L(\cdot)$. For both strategies, at each iteration we need to update only one column of matrix Q_k (see (4.3)), which costs $O(mn)$ operations.

Let us present the results of our numerical experiments. All methods were stopped when the residual in the function value was smaller than $\epsilon = 10^{-6}$. Parameter δ for the stopping criterion (2.11) was chosen as $\delta = \epsilon/2$.

In the tables below, the first line indicates the total number of iterations. Second line displays the total number of oracle calls. Third line shows the average number of Frank-Wolfe steps per iterations (for Gradient Method we just put two). Next line indicates the total computational time (in seconds). Finally, at the last line we can see the average

time spent on one iteration of the corresponding method (in milliseconds).

$n = 100$	GM	Cyclic	Max-Norm	$n = 250$	GM	Cyclic	Max.Norm
Iter	2683	801	664	Iter	2148	227	227
NFunc	5371	1606	1332	NFunc	4302	459	459
FW/Iter	2	55	66	FW/Iter	2	243	243
Time (s)	1.94	0.88	0.73	Time (s)	10.20	1.36	1.36
IT(ms)	0.72	0.11	0.11	IT(ms)	4.75	5.99	5.99

$n = 500$	GM	Cyclic	Max.Norm
Iter	2902	268	268
NFunc	5809	537	537
FW/Iter	2	428	428
Time (s)	52.16	5.5	5.5
IT(ms)	17.97	22.35	22.35

Table 1. Smoothness parameter $\mu = 0.05$.

$n = 100$	GM	Cyclic	Max.Norm	$n = 250$	GM	Cyclic	Max.Norm
Iter	43893	4171	6710	Iter	116479	45183	25492
NFunc	87795	8351	13427	NFunc	232967	90377	50990
FW/Iter	2	59	17	FW/Iter	2	8	13
Time (s)	31.73	4.50	6.95	Time (s)	540.64	311.95	176.02
IT(ms)	0.72	1.08	1.04	IT(ms)	4.64	6.90	6.90

$n = 500$	GM	Cyclic	Max.Norm
Iter	105610	38144	29916
NFunc	211229	76297	59840
FW/Iter	2	10	17
Time (s)	1894.19	1010.30	788.97
IT(ms)	17.94	26.49	26.37

Table 2. Smoothness parameter $\mu = 0.01$.

As we can see from these tables, in all our experiments the gradient methods with memory were better than the standard Gradient Method, both in the number of iterations, and, what is rather surprising, in the total computational time. Max-Norm version usually outperforms the Cyclic version. It is interesting that the auxiliary algorithm (3.5) works very well. The average time spent on one iteration of the methods with memory is never increased more than on 50% of the time of the cheapest Gradient Method. This is partially explained by the fact that in our test problems the data is fully dense, so each call of oracle is very expensive ($O(Mn)$ operations).

Let us look now how the small bundles can accelerate the Gradient Method. In the tables below, the first line with parameter Bundle = 1 corresponds to the Gradient method with line search. Next lines display the results for different sizes of the bundle. We show the number of iterations, average number of Frank-Wolfe steps per iteration, and the total

computational time in seconds. Table 3 displays the results for the IGMM (2.23) with cyclic replacement strategy for the bundle, $\epsilon = 10^{-4}$, and $\delta = \epsilon/2$. In Table 4, we show the results for the max-norm replacement strategy. The smoothness parameter for our objective function is chosen as $\mu = 0.05$.

	$n = 100$			$n = 200$			$n = 400$		
Bundle	Iter	FW	Sec	Iter	FW	Sec	Iter	FW	Sec
1	2683	2	1.94	1753	2	4.78	1676	2	19.86
2	2543	3	1.84	1433	7	3.91	1669	5	19.81
4	1755	13	1.31	829	39	2.31	1084	18	12.92
8	1363	19	1.03	633	60	1.78	758	37	9.08
16	1220	22	0.97	579	68	1.67	720	44	8.69
32	1202	24	1.00	593	71	1.76	719	47	8.78
64	1138	28	1.06	423	114	1.39	636	58	8.02
128	880	60	1.06	247	320	0.94	349	127	4.63
256	207	361	0.36	173	507	0.70	202	260	2.70

Table 3. Gradient Method with Cyclic Memory Replacement

	$n = 100$			$n = 200$			$n = 400$		
Bundle	Iter	FW	Sec	Iter	FW	Sec	Iter	FW	Sec
1	2683	2	1.94	1753	2	4.76	1676	2	19.86
2	1288	4	0.95	737	11	2.05	796	7	9.47
4	592	27	0.45	316	79	0.91	385	38	4.61
8	400	48	0.33	269	121	0.78	283	86	3.42
16	362	58	0.33	418	91	1.22	335	86	4.05
32	786	33	0.67	354	116	1.08	354	89	4.34
64	838	39	0.80	379	127	1.23	421	85	5.31
128	529	100	0.66	285	277	1.09	352	127	4.64
256	207	361	0.36	173	507	0.70	202	260	2.70

Table 4. Gradient Method with Max-Norm Memory Replacement

As we can see from these tables, Max-Norm replacement strategy was always better than the cyclic one. Even a small size of the bundle, drops very quickly the total number of iteration, and, what is more important, decreases proportionally the computational time. Number of the auxiliary Frank-Wolfe steps remain on an acceptable level and cannot increase significantly the computational time of each iteration as compared with the Gradient Method. Recall that our test function has an expensive oracle, requiring $O(Mn)$ operations for computing the function value and the gradient. For Max-Norm version of IGMM, the optimal size of the bundle is probably between 8 and 16. Another candidate is 256, but it needs much more Frank-Wolfe steps.

May be our preliminary conclusions are problem specific. However, we believe that in any case they demonstrate a high potential of our approach in increasing efficiency of the gradient methods, both in accelerated and, hopefully, non-accelerated variants.

References

[1] H. Bauschke, J. Bolte, and M. Teboulle. A descent Lemma beyond Lipschitz gradient continuity: First-order methods revisited and applications. *Mathematics of Operations Research*, **42**(2), 330-348 (2017).

[2] M. Frank, P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, **3**, 149-154 (1956).

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

[4] Yu. Nesterov. Lectures on Convex Optimization. *Springer* (2018)

[5] H. Lu, R. Freund, and Yu. Nesterov. Relatively Smooth Convex Optimization by First-Order Methods, and Applications. *SIOPT*, **28**(1), 333-354 (2018).

[6] Yu. Nesterov. Complexity bounds for primal-dual methods minimizing the model of objective function. *Mathematical Programming*, **171**, 311-330 (2018).