Exact Worst-case Performance of First-order Methods for Composite Convex Optimization

A.B. Taylor · J.M. Hendrickx · F. Glineur

Date of current version: February 6, 2017

Abstract We provide a framework for computing the exact worst-case performance of any algorithm belonging to a broad class of oracle-based first-order methods for composite convex optimization, including those performing explicit, projected, proximal, conditional and inexact (sub)gradient steps. We simultaneously obtain tight worst-case guarantees and explicit instances of optimization problems on which the algorithm reaches this worst-case. We achieve this by reducing the computation of the worst-case to solving a convex semidefinite program, generalizing previous works on performance estimation by Drori and Teboulle [13] and the authors [43].

We use these developments to obtain a tighter analysis of the proximal point algorithm and of several variants of fast proximal gradient, conditional gradient, subgradient and alternating projection methods. In particular, we present a new analytical worst-case guarantee for the proximal point algorithm that is twice better than previously known, and improve the standard worst-case guarantee for the conditional gradient method by more than a factor of two.

We also show how the optimized gradient method proposed by Kim and Fessler in [22] can be extended by incorporating a projection or a proximal operator, which leads to an algorithm that converges in the worst-case twice as fast as the standard accelerated proximal gradient method [2].

1 Introduction

Consider the composite convex minimization problem

$$\min_{x\in\mathbb{E}} \left\{ F(x) \equiv \sum_{k=1}^{n} F^{(k)}(x) \right\}, \tag{CM}$$

where $\mathbb{E}$ is a finite-dimensional real vector space and each functional component $F^{(k)} : \mathbb{E} \to \mathbb{R} \cup \{\infty\}$ is a convex function belonging to some class $\mathcal{F}_k(\mathbb{E})$ — e.g., smooth or non-smooth, strongly convex or not, indicator functions, etc. — for which some operations are assumed to be available in closed-form (e.g., computing a gradient, projecting on the domain, computing a proximal step, etc.).

We are interested in the composite optimization problem (CM) because it naturally allows representing and exploiting a lot of the structure in many problems, which can play a major role in our ability to efficiently solve them (see [32] among others). In addition, the class of composite convex optimization problems arises

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office, and of the Concerted Research Action (ARC) programme supported by the Federation Wallonia-Brussels (contract ARC 14/19-060). The scientific responsibility rests with its authors. A.B.T. is a F.R.I.A. fellow.

A.B. Taylor, J.M. Hendrickx, F. Glineur
Université catholique de Louvain, ICTEAM Institute/CORE, B-1348 Louvain-la-Neuve, Belgium
E-mail: Adrien.Taylor@uclouvain.be; Julien.Hendrickx@uclouvain.be; Francois.Glineur@uclouvain.be

very commonly in practice, as it contains for example constrained, ℓ_1- and ℓ_2-regularized convex optimization problems.

We focus on black-box oracle-based algorithms that use first-order information to approximately solve (CM), and in particular on obtaining exact and global worst-case guarantees on their performances. That is, for a given algorithm, we simultaneously seek to obtain worst-case guarantees — for example on objective function accuracy — and an instance of (CM) on which the algorithm behaves as such. In this work, we investigate fixed-step linear first-order methods, which include among others fixed-step projected, proximal, conditional and inexact (sub)gradient methods.

This work builds on the recent idea of performance estimation, first developed by Drori and Teboulle in [13] and followed-up by Kim and Fessler [22] and the authors [43]. The approach was initially tailored for obtaining upper bounds on the worst-case behavior of fixed-step gradient methods for unconstrained minimization of a single smooth convex objective function. Motivated by subsequent results (see among others [21,22]) we extend the framework of performance estimation to the composite case involving a much broader class of algorithms and function classes (see Section 1.4 for more details about previous works).

Our performance estimation framework relies on formulating the worst-case computation problem as a tractable semidefinite program (SDP), which can be tackled with standard solvers [24,26,41]. It enjoys the following attractive features:

- any primal feasible solution to this SDP leads to a lower bound on the worst-case performance of the method under consideration, by exhibiting a particular instance of (CM),
- any dual feasible solution to this SDP corresponds to an upper bound on the worst-case performance of the method under consideration, that can be converted into an explicit proof based on a combination of valid inequalities.

1.1 Notations

In this paper, we work in a finite-dimensional real vector space $\mathbb{E}$ and the corresponding dual space $\mathbb{E}^*$ consisting of all linear functions on $\mathbb{E}$, and denote their dimension by $d = \dim \mathbb{E} = \dim \mathbb{E}^*$. We consider a dual pairing[1] between those spaces, denoted by $\langle .,. \rangle : \mathbb{E}^* \times \mathbb{E} \to \mathbb{R}$. We also consider a self-adjoint positive definite[2] linear operator $B : \mathbb{E} \to \mathbb{E}^*$ for $\langle .,. \rangle$, which allows defining the following primal and dual norms:

$$\|x\|_{\mathbb{E}}^2 = \langle Bx, x\rangle,\ \forall x \in \mathbb{E}, \quad \|s\|_{\mathbb{E}^*}^2 = \left\langle s, B^{-1}s \right\rangle,\ \forall s \in \mathbb{E}^*.$$

We denote $\langle x,y\rangle_{\mathbb{E}} = \langle Bx, y\rangle$ for $x,y \in \mathbb{E}$ and $\langle x,y\rangle_{\mathbb{E}^*} = \left\langle x, B^{-1}y\right\rangle$ for $x,y \in \mathbb{E}^*$. The usual case is simply $\mathbb{E} = \mathbb{E}^* = \mathbb{R}^d$ with $\langle x,y\rangle = x^\top y$ the standard Euclidean inner product and B the identity operator, for which we also have $\|x\|_{\mathbb{E}}^2 = \|x\|_{\mathbb{E}^*}^2 = \langle x,x\rangle$.

In addition, we use the notation $\mathcal{F}_{0,\infty}$ for the set of closed, proper and convex functions. For a convex function $f : \mathbb{E} \to \mathbb{R} \cup \{\infty\}$, we denote by $f^* : \mathbb{E}^* \to \mathbb{R} \cup \{\infty\}$ its Legendre-Fenchel conjugate

$$f^*(y) = \sup_{x \in \mathbb{E}} \langle y, x\rangle - f(x),$$

by $\partial f(x)$ the subdifferential of f at x (set of all subgradients of f at x), and by $\tilde{\nabla} f(x)$ a particular subgradient of f at x. Similarly, the gradient of a differentiable function f at x is denoted by $\nabla f(x)$.

For notational convenience we denote by $K = \{1, \ldots, n\}$ the set of indices corresponding to the different components $F^{(k)}$ in the objective function of (CM). We also denote by $\mathcal{F}_K(\mathbb{E})$ the set of functions of the form (CM) with components $F^{(k)} \in \mathcal{F}_k(\mathbb{E})\ \forall k \in K$ — that is, $F \in \mathcal{F}_K(\mathbb{E})$.

Finally, we use the standard notation e_i for the unit vector having a single 1 as its i^{th} component.

[1] The dual pairing is a real bilinear map $\langle .,. \rangle : \mathbb{E}^* \times \mathbb{E} \to \mathbb{R}$ satisfying (i) $\forall x \in \mathbb{E} \backslash \{0\}, \exists s \in \mathbb{E}^*$ such that $\langle s, x\rangle \neq 0$, and (ii) $\forall s \in \mathbb{E}^* \backslash \{0\}, \exists x \in \mathbb{E}$ such that $\langle s, x\rangle \neq 0$.

[2] That is, a linear operator B satisfying (i) $\langle Bx, y\rangle = \langle By, x\rangle\ \forall x,y \in \mathbb{E}$ (self-adjoint), and (ii) $\langle Bx, x\rangle > 0\ \forall x \in \mathbb{E}\backslash\{0\}$ (positive definite). A direct consequence of B satisfying those assumptions is the existence of the linear operator B^{-1}.

1.2 Performance estimation problems

In [43], we introduced a formal definition for the performance estimation problem in the case of a black-box first-order method for unconstrained minimization of a single convex function F. We now generalize the performance estimation framework for handling multiple components in the objective function.

First, we formalize black-box methods using the concept of *black-box oracles*. That means that methods are only allowed to access the different components of the objective function by calling some routines, or oracles, returning some information about them at a given point. In particular, we focus on the standard first-order oracle for $F^{(k)}$: $\mathcal{O}_{F^{(k)}}(x) = \left(F^{(k)}(x), \tilde{\nabla} F^{(k)}(x)\right)$ in the sequel, where $\tilde{\nabla} F^{(k)}(x) \in \partial F^{(k)}(x)$ is a subgradient of $F^{(k)}$ at x. The general formalism of the approach is nevertheless also valid for other standard oracles, as for example zeroth-order or second-order ones — that is, $\mathcal{O}_{F^{(k)}}(x) = \left(F^{(k)}(x)\right)$ or $\mathcal{O}_{F^{(k)}}(x) = \left(F^{(k)}(x), \nabla F^{(k)}(x), \nabla^2 F^{(k)}(x)\right)$. However, as we will see, our ability to solve the corresponding performance estimation problems in an exact way is currently limited to first-order oracles.

Second, we consider a sequence of $N+1$ iterates $\{x_i\}_{0\leq i\leq N} \subset \mathbb{E}$, corresponding to a method that performs N steps from an initial iterate x_0. For each of those iterates we consider the set oracle calls for each functional component[3] $\mathcal{O}_{F^{(k)}}$: $\{\mathcal{O}_{F^{(k)}}(x_i)\}_{0\leq i\leq N}$.

Third, we consider a method $\mathcal{M}$ whose iterates can be computed by combining past and current oracle information about F. This means that, after the method has performed $i-1$ steps, the next iterate x_i should be computable as a solution to an equation of the form:

$$\textsc{Equation}(x_0, \{\mathcal{O}_{F^{(k)}}(x_0)\}_{k\in K}, x_1, \{\mathcal{O}_{F^{(k)}}(x_1)\}_{k\in K}, \ldots, x_i, \{\mathcal{O}_{F^{(k)}}(x_i)\}_{k\in K}). \tag{EQ$_\text{i}$}$$

Note that the only unknown in this equation is x_i, and that it thus provides an implicit definition for the next step. We will see later that this assumption on $\mathcal{M}$ includes a large number of existing methods for composite optimization.

Finally, we consider a real-valued performance criterion $\mathcal{P}$ for evaluating the efficiency of the method. In the sequel, we assume without loss of generality that the lower the value of $\mathcal{P}$, the better the corresponding method. Examples of such performance criteria include objective function accuracy $F(x_N) - F(x_*)$ (where x_* is any optimal solution of (CM)), and distance to an optimal solution $\|x_N - x_*\|^2_{\mathbb{E}}$.

In our framework, this performance criterion is generally allowed to depend on information returned by the oracles $\mathcal{O}_{F^{(k)}}$ at all the iterates $\{x_i\}_{0\leq i\leq N}$, but also at an extra point $x_* \in \mathbb{E}$ assumed to be an optimal solution to problem (CM). Also, we allow $\mathcal{P}$ to depend on the iterates themselves. For notational convenience we introduce an index set for all iterates (including optimal solution) $I = \{0, 1, \ldots, N, *\}$.

The worst-case performance of method $\mathcal{M}$ on (CM) is then the optimal value of the following optimization problem, with both functions $\left\{F^{(k)}\right\}_{k\in K}$ and iterates $\{x_i\}_{i\in I}$ as variables, which we call a performance estimation problem (PEP).

$$\begin{aligned} \sup_{\left\{F^{(k)}\right\}_{k\in K}, \{x_i\}_{i\in I}} \quad & \mathcal{P}(\{\mathcal{O}_{F^{(k)}}(x_i)\}_{i\in I, k\in K}, \{x_i\}_{i\in I}) \\ \text{subject to } \quad & F^{(k)} \in \mathcal{F}_k(\mathbb{E}) \quad \text{for all } k \in K, \\ & x_0 \text{ satisfies some initialization condition}, \\ & x_i \text{ is computed by } \mathcal{M} \text{ according to (EQ}_\text{i}\text{) for all } 1 \leq i \leq N, \\ & x_* \text{ is a minimizer of } F(x). \end{aligned} \tag{PEP}$$

That is, a solution to (PEP) corresponds to an instance of problem (CM) on which method $\mathcal{M}$ behaves as badly as possible with respect to the performance criterion $\mathcal{P}$. The initialization condition on x_0 is required as most methods exhibit unbounded worst-case performance without it. In the sequel we will mostly restrict ourselves to the classical approach which consists in bounding the initial distance to an optimal solution with a constant R, i.e., assume $\|x_0 - x_*\|_{\mathbb{E}} \leq R$.

[3] That is, we chose to associate a call to each oracle to every iterate. This is mostly for notational convenience and does not induce any loss of generality. Indeed, a method can always choose not to use the information returned by one of the oracles at some iterations.

Note that (PEP) is inherently an infinite-dimensional optimization problem, as functions $F^{(k)}$ appear as variables. However, a crucial observation is that, due to the black-box assumption on the objective components, this problem can be cast completely equivalently in a finite-dimensional fashion. Indeed, introducing the *outputs* of the oracle calls as variables, namely $O_i^{(k)} = \mathcal{O}_{F^{(k)}}(x_i)$ for all iterates $i \in I$ and oracles $k \in K$, we observe that steps of method $\mathcal{M}$ can be still be computed using only information contained in variables $O_i^{(k)}$, so that we can reformulate (PEP) as

$$\sup_{\left\{O_i^{(k)}\right\}_{i\in I,k\in K},\{x_i\}_{i\in I}} \mathcal{P}\left(\left\{O_i^{(k)}\right\}_{i\in I,k\in K},\{x_i\}_{i\in I}\right), \tag{PEP2}$$
$$\begin{aligned}\text{subject to } &\exists F^{(k)} \in \mathcal{F}_k(\mathbb{E}) \text{ satisfying } \mathcal{O}_{F^{(k)}}(x_i) = O_i^{(k)} \text{ for all } i \in I, k \in K,\\ &x_0 \text{ satisfies some initialization condition,}\\ &x_i \text{ is computed by } \mathcal{M} \text{ according to } (\mathrm{EQ_i}) \text{ for all } 1 \le i \le N,\\ &x_* \text{ is a minimizer of } F(x).\end{aligned}$$

Note the central role played by the interpolation conditions $\mathcal{O}_{F^{(k)}}(x_i) = O_i^{(k)}$ for all $i \in I$ and $k \in K$, which enforce the existence of functions $F^{(k)}$ compatible with the output of the oracles. In the next subsection we describe situations for which this formulation is tractable.

1.3 First-order methods and first-order convex interpolation

In the remainder of this work, we restrict ourselves to first-order oracles and methods. We now investigate the concept of (first-order) convex interpolability, in order to make existence constraints from (PEP2) tractable — more precise requirements are detailed in Section 2. From the assumptions, the existence constraint for function $F^{(k)}$

$$\exists F^{(k)} \in \mathcal{F}_k(\mathbb{E}) \text{ satisfying } \mathcal{O}_{F^{(k)}}(x_i) = O_i^{(k)} \text{ for all } i \in I,$$

found in (PEP2), may be expressed in terms of first-order information only. Considering oracles returning first-order information $\mathcal{O}_{F^{(k)}}(x) = (F^{(k)}(x), \tilde{\nabla} F^{(k)}(x))$, we denote their output at point x_i by $\mathcal{O}_{F^{(k)}}(x_i) = O_i^{(k)} = (f_i^{(k)}, g_i^{(k)})$. The above existence constraint can be rephrased into the following set of interpolation conditions

$$\exists F^{(k)} \in \mathcal{F}_k(\mathbb{E}) \text{ satisfying } F^{(k)}(x_i) = f_i^{(k)} \text{ and } g_i^{(k)} \in \partial F^{(k)}(x_i), \tag{INT}$$

which leads us to introduce the following general definition.

Definition 1 ($\mathcal{F}(\mathbb{E})$-interpolation) Let I be an index set and $\mathcal{F}(\mathbb{E})$ a class of convex functions, and consider the set of triples $S = \{(x_i, g_i, f_i)\}_{i\in I}$ where $x_i \in \mathbb{E}$, $g_i \in \mathbb{E}^*$ and $f_i \in \mathbb{R}$ for all $i \in I$. The set S is $\mathcal{F}(\mathbb{E})$-interpolable if and only if there exists a function $F \in \mathcal{F}(\mathbb{E})$ such that both $g_i \in \partial F(x_i)$ and $F(x_i) = f_i$ hold for all $i \in I$.

The notion of $\mathcal{F}(\mathbb{E})$-interpolation can be considered for any class of convex functions. It allows us to formulate our performance estimation problem in its final form

$$\sup_{\left\{(f_i^{(k)},g_i^{(k)})\right\}_{i\in I,k\in K},\{x_i\}_{i\in I}} \mathcal{P}\left(\left\{(f_i^{(k)},g_i^{(k)})\right\}_{i\in I,k\in K},\{x_i\}_{i\in I}\right), \tag{f-PEP}$$
$$\begin{aligned}\text{subject to } &\left\{(x_i, g_i^{(k)}, f_i^{(k)})\right\}_{i\in I} \text{ is } \mathcal{F}_k\text{-interpolable for all } k \in K,\\ &x_0 \text{ satisfies some initialization condition,}\\ &x_i \text{ is computed by } \mathcal{M} \text{ according to } (\mathrm{EQ_i}) \text{ for all } 1 \le i \le N,\\ &x_* \text{ is a minimizer of } F(x).\end{aligned}$$

We conclude that identifying explicit conditions for convex interpolability by a given class of functions will be the key to eliminate the infinite-dimensional functional variables from (PEP) and transform it into a tractable estimation problem.

First-order convex interpolation was originally developed in [43] for classes of (possibly) L-smooth and (possibly) μ-strongly convex functions. In Section 3, we extend these results to classes of functions involving simultaneously strong convexity, smoothness, gradient boundedness and domain boundedness (for different norms). Those extensions also allow to consider interpolation by indicator or support functions, which may among others be used for problems involving constraints.

Also, note that the notion of first-order interpolability can be adapted for non-convex functions as well. Replacing the concept of subdifferentiability by standard differentiability can be used to study the convergence of first-order algorithms in the cases where some components $F^{(k)}$ are not convex (see Section 3.4).

1.4 Prior work

The concept of performance estimation showed itself very promising in the pioneer work of Drori and Teboulle [13], and later in the work of Kim and Fessler [22]. In their work [13], Drori and Teboulle proposed a convex relaxation to obtain numerical upper bounds on the worst-case behaviour of fixed-step first-order algorithms minimizing a single smooth convex function over $\mathbb{R}^d$, which turned out to be tight in surprisingly many situations[4]. They also proposed a way to numerically optimize the step size parameters of a fixed-step algorithm by minimizing an upper bound on its worst-case. Their approach is based on semidefinite relaxations of (PEP), and was taken further by Kim and Fessler [22], who derived analytically the optimized gradient method previously identifed numerically by Drori and Teboulle.

The performance estimation approach on the same smooth unconstrained minimization is further studied in [43], where convex interpolation allows the derivation of an exact convex reformulation of the problem, leading to tight worst-case estimates. The obtained semidefinite formulation also forms the basis for this work.

Another recent and closely related approach for studying performances of first-order methods consists in viewing optimization algorithms as dynamical systems, and to use the related stability theory in order to numerically analyse them. This idea is proposed by Lessard et al. in [23], and is attractive because it requires solving a single semidefinite program to obtain a bound that is valid for all subsequent iterations. This technique is particularly efficient for problems involving strong convexity, for which tight linear convergence rates are often recovered. However, as they aim at finding global rates of convergence, they are naturally more conservative than the general performance estimation approach.

For more details on the general topic of convergence analysis of first-order methods, we refer to the seminal books of Yudin and Nemirovski [27], Polyak [36], Nesterov [29] and the more recent book of Bertsekas [4]. Concerning the development of accelerated methods, we specifically refer to the original work of Nesterov [28, 29], and to the later extensions to minimize smoothed convex functions [30] and composite functions [2, 32].

1.5 Paper organization and main contributions

This work is divided into three main parts. First, Section 2 is concerned with putting in place the performance estimation framework for large classes of first-order algorithms, objective functions, performance criteria and initialization conditions. The main idea of this section is to require every element of the performance estimation problem (PEP) to be *linearly Gram-representable* (defined in Section 2.2). This section contains multiple examples of standard settings for which the methodology applies — including among others those covering (sub)gradient methods (along with their projected and proximal counterparts) and conditional gradient methods.

Section 3 focuses on providing convex interpolation conditions for different classes of convex functions commonly arising in practice. Those classes include convex functions, possibly with strong convexity, smoothness, bounded domain and bounded (sub)gradient requirements. The subclasses of indicator and support functions are also explicitly handled. Those classes of functions can all be used directly in the performance

[4] An extension to provide upper bounds for the fixed-step projected gradient method is also provided in Drori's PhD thesis [11].

estimation framework of Section 2, since their corresponding interpolation conditions are linearly Gram-representable. This section ends with an extension of the convex interpolation results to cope with smooth non-convex functions in a linearly Gram-representable way.

In Section 4, we apply our approach to several concrete first-order algorithms. We obtain improvements on the analysis of several well-known methods, either analytically or numerically, including the proximal point algorithm and the conditional gradient method. We also use those results to provide an extension of the optimized gradient method proposed by Kim and Fessler [22] that incorporates a projection or a proximal operator to tackle constrained and composite problems.

2 Performance estimation framework for first-order algorithms

We start this section by formulating (f-PEP) in terms of a Gram matrix. This leads to a tractable convex formulation for (f-PEP) — once appropriate assumptions are made on the classes of objective function components, methods, performance criteria and initialization conditions. Those assumptions are motivated by practical applications, which we also provide in the following lines. The main point underlying those assumptions is to ensure that every element of the performance estimation problem can be formulated in a linear way in terms of both the entries of a Gram matrix and the function values at the iterates.

2.1 Gram representations

Let us consider $N+1$ iterates $x_0,\dots,x_N$ and an optimal solution x_*, and the set of corresponding oracle outputs $\left\{(f_i^{(k)},g_i^{(k)})\right\}_{i\in I,k\in K}$. The accumulated information after those $N+1$ oracle calls can be gathered into a $d\times(n+1)(N+2)$ matrix[5] P_N (using a slight abuse of notations) and a vector F_N of length $n(N+2)$:

$$P_N = [Bx_0\ \dots\ Bx_N \mid Bx_* \mid g_0^{(1)}\ \dots\ g_0^{(n)} \mid \dots \mid g_N^{(1)}\ \dots\ g_N^{(n)} \mid g_*^{(1)}\ \dots\ g_*^{(n)}], \tag{1}$$

$$F_N = [\ f_0^{(1)}\ \dots\ f_0^{(n)} \mid \dots \mid f_N^{(1)}\ \dots f_N^{(n)} \mid f_*^{(1)}\ \dots f_*^{(n)}]. \tag{2}$$

We also denote by $B^{-1}P_N$ the matrix

$$B^{-1}P_N = [x_0\ \dots\ x_N \mid x_* \mid B^{-1}g_0^{(1)}\ \dots\ B^{-1}g_*^{(n)}].$$

In order to formulate (PEP) in a tractable way for first-order methods, we use a Gram matrix. That is, we define a symmetric $(n+1)(N+2)\times(n+1)(N+2)$ Gram matrix $G_N\in\mathbb{S}^{(n+1)(N+2)}$, using the following construction :

$$G_N = \begin{pmatrix}
\langle x_0,x_0\rangle_{\mathbb{E}} & \dots & \langle x_0,x_N\rangle_{\mathbb{E}} & \langle x_0,x_*\rangle_{\mathbb{E}} & \langle g_0^{(1)},x_0\rangle & \dots & \langle g_*^{(n)},x_0\rangle \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\langle x_N,x_0\rangle_{\mathbb{E}} & \dots & \langle x_N,x_N\rangle_{\mathbb{E}} & \langle x_N,x_*\rangle_{\mathbb{E}} & \langle g_0^{(1)},x_N\rangle & \dots & \langle g_*^{(n)},x_N\rangle \\
\langle x_*,x_0\rangle_{\mathbb{E}} & \dots & \langle x_*,x_N\rangle_{\mathbb{E}} & \langle x_*,x_*\rangle_{\mathbb{E}} & \langle g_0^{(1)},x_*\rangle & \dots & \langle g_*^{(n)},x_*\rangle \\
\langle g_0^{(1)},x_0\rangle & \dots & \langle g_0^{(1)},x_N\rangle & \langle g_0^{(1)},x_*\rangle & \langle g_0^{(1)},g_0^{(1)}\rangle_{\mathbb{E}^*} & \dots & \langle g_0^{(1)},g_*^{(n)}\rangle_{\mathbb{E}^*} \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\langle g_*^{(n)},x_0\rangle & \dots & \langle g_*^{(n)},x_N\rangle & \langle g_*^{(n)},x_*\rangle & \langle g_*^{(n)},g_0^{(1)}\rangle_{\mathbb{E}^*} & \dots & \langle g_*^{(n)},g_*^{(n)}\rangle_{\mathbb{E}^*}
\end{pmatrix} \succeq 0.$$

This can be written more compactly as $[G_N]_{ij} = \langle P_N e_i, B^{-1}P_N e_j\rangle = \langle P_N e_i, P_N e_j\rangle_{\mathbb{E}^*}$, where $P_N e_k$ corresponds to the kth column of P_N. Also, note that the size of this matrix does not depend on the dimension d of the spaces we are working with.

[5] We recall that $B:\mathbb{E}\to\mathbb{E}^*$ is a positive definite operator which is chosen as the identity operator in standard situations (see Section 1.1).

Remark 1 Note that Gram matrix G_N is positive semidefinite for any matrix P_N (of the form (1)). The number of linearly independent columns of P_N is equal to the rank of G_N. Hence this rank is upper bounded by the dimension d of the ambient space of the iterates. It is possible to recover a matrix P_N of the form[6] (1) from any Gram matrix $G_N \succeq 0$ satisfying Rank $G_N \leq d$.

Our goal for the next subsections is to show that in a lot of situations, the performance estimation problem (f-PEP) can be expressed exactly as a semidefinite program in the F_N and G_N variables:

$$\begin{aligned} \sup_{F_N\in\mathbb{R}^{n(N+2)},G_N\in\mathbb{S}^{(n+1)(N+2)}} \quad & c^\top F_N + \operatorname{Tr}(CG_N) \\ \text{subject to} \quad & a_i + b_i^\top F_N + \operatorname{Tr}(D_iG_N) \leq 0 \quad \forall i \in S, \\ & G_N \succeq 0, \end{aligned} \tag{SDP-PEP}$$

with S some index set related to the constraints, and elements a_i, b_i, c, D_i and C of appropriate dimensions for writing the constraints and objective function linearly in terms of the Gram matrix G_N and of the objective function values F_N.

2.2 Tractable formulation of the performance estimation problem

In this section, we present our main result, stating that computing the exact worst-case performance of a method on a class of functions is tractable and can, in many cases, be formulated as (SDP-PEP). We start with the concept of Gram-representability for the different ingredients of the performance estimation problem.

Definition 2 A class of functions is Gram-representable (resp. linearly Gram-representable) if and only if its interpolation conditions (INT) can be formulated using a finite number of convex (resp. linear) constraints involving only the matrix G_N and the function values F_N.

The functional classes of smooth strongly convex functions, smooth convex functions with bounded (sub)gradients, and strongly convex functions with bounded domain are linearly Gram-representable. In addition, the particular subclasses of support and indicator convex functions share this same advantageous property. The details and proofs of these results are postponed to Section 3.

Definition 3 A performance measure is Gram-representable (resp. linearly Gram-representable) if and only if it can be expressed as a concave (resp. linear) function involving only the matrix G_N and the function values F_N.

The class of linearly Gram-representable performance criteria contains a large variety of choices, including most standard measures we are aware of. For example, it is easy to check that standard optimality criteria in function values $F(x_N) - F(x_*)$, in residual subgradient norm $\left\|\tilde{\nabla} F(x_N)\right\|_{\mathbb{E}^*}^2$, distance to optimality $\|x_N - x_*\|_{\mathbb{E}}^2$, and distance to feasibility $\|x_N - \Pi_Q(x_N)\|_{\mathbb{E}}^2$ can be handled.

On the other hand, multiple examples of non-linear Gram-representable performance criteria can also be handled with no difficulty. This includes performance measures involving the best values among all iterates, for example $\min_{0\leq i\leq N} F(x_i) - F(x_*)$, or the best residual gradient norm among the iterates $\min_{0\leq i\leq N} \|\nabla F(x_i)\|_{\mathbb{E}^*}^2$ (see also [43, Sect. 4.3]).

Definition 4 An initialization condition is Gram-representable (resp. linearly Gram-representable) if and only if it can be expressed using a finite number of convex (resp. linear) constraints involving only the matrix G_N and the function values F_N.

Standard examples of valid initial conditions include the classical bounds on the initial distance to optimality $\|x_0 - x_*\|_{\mathbb{E}}^2 \leq R^2$, on the initial function value $F(x_0) - F_* \leq R$, and on initial gradient value $\|\nabla F(x_0)\|_{\mathbb{E}^*}^2 \leq R^2$, for given values of $R \geq 0$.

[6] In the case $\mathbb{E} = \mathbb{E}^* = \mathbb{R}^d$ with the usual inner product $\langle x, y\rangle = x^\top y$ and B the identity operator, this can be done using the standard Cholesky factorization. In the general cases the exact same idea can be used, using the chosen inner product $\langle ., .\rangle_{\mathbb{E}^*}$ in the process.

Definition 5 A first-order method is Gram-representable (resp. linearly Gram-representable) if and only if the computation of its iterates, implicitly defined by an equation of type $(\mathrm{EQ_i})$, can be expressed using a finite number of convex (resp. linear) constraints involving only the matrix G_N and the function values F_N.

We refer to the next section for examples of linearly Gram-representable methods. Note that the necessary and sufficient condition for x_* to be optimal for F is always linearly Gram-representable. Indeed, it corresponds to requiring $\tilde{\nabla} F(x_*) = 0$, i.e.

$$\sum_{k\in K} \tilde{\nabla} F^{(k)}(x_*) = \sum_{k\in K} g_*^{(k)} = 0 \Leftrightarrow \left\| \sum_{k\in K} g_*^{(k)} \right\|_{\mathbb{E}^*}^2 = \left\langle \sum_{k\in K} g_*^{(k)}, \sum_{k\in K} g_*^{(k)} \right\rangle_{\mathbb{E}^*} = 0,$$

where the last condition is linear in the entries of G_N.

We can now state our main results concerning Gram-representable situations.

Proposition 1 *Consider a class of composite objective functions $\mathcal{F}_K(\mathbb{E})$ with n components, a first-order method $\mathcal{M}$, a performance measure $\mathcal{P}$ and an initial condition $\mathcal{I}$ which are all Gram-representable.*

Computing the worst-case for criterion $\mathcal{P}$ of method $\mathcal{M}$ after N iterations on objective functions in class $\mathcal{F}_K(\mathbb{E})$ with initial condition $\mathcal{I}$ can be formulated as a convex program when dimension of the space $\mathbb{E}$ satisfies $d \geq (n+1)(N+2)$. Otherwise, it can be formulated as a convex program plus an additional non-convex rank constraint Rank $G_N \leq d$.

If in addition $\mathcal{F}_K(\mathbb{E})$, $\mathcal{M}$, $\mathcal{P}$ and $\mathcal{I}$ are linearly Gram-representable, then the corresponding convex problem is a semidefinite program of the form (SDP-PEP)*, whose variables are $F_N \in \mathbb{R}^{n(N+2)}$ and $G_N \in \mathbb{S}^{(n+1)(N+2)}$.*

Proof It directly follows from Remark 1 and from the definitions of (linear) Gram-representability for the class of functions, first-order methods, performance measures, optimality condition of a solution and initialization conditions: any solution to the corresponding optimization problem can be transformed into a particular instance of (CM), and vice versa.

Remark 2 The optimal value of (PEP) increases with dimension d. When (PEP) with Gram-representable elements attains a finite optimal value, Proposition 1 implies the existence of a function with dimension at most $(n+1)(N+2)$ that achieves the worst-case value.

Remark 3 The assumption $d \geq (n+1)(N+2)$ is referred to as the *large-scale assumption* in the sequel. In terms of performance estimation problems, this assumption allows to discard the non-convex rank constraint and lead to a tractable semidefinite programming problem, which can be solved to global optimality efficiently (see e.g., [45]). Without that assumption, our performance estimation problem is a nonconvex rank-constrained semidefinite program, equivalent to a quadratic programming problem that is NP-hard in general (e.g., it has max-cut [16] and other non-convex quadratic programs [34,40] as particular cases). Approaches to handle rank constraints exist (e.g., via augmented Lagrangian techniques [6], via manifold optimization [20] or via Newton-like methods [33]), but in general only guarantee convergence to stationary points. This is not useful for in the case of (SDP-PEP), as this only provides lower bounds on the worst-case performance.

Remark 4 Under the large-scale assumption, we obtain dimension-free guarantees (i.e. valid for any dimension, and tight as soon as $d \geq (n+1)(N+2)$), as is commonly found in the literature about first-order methods. In addition, we note that the dimension bound $(n+1)(N+2)$ is in fact (very) conservative for most standard algorithms — that is, the bound in the large-scale assumption can typically be significantly reduced, see Corollary 1 in the sequel.

Remark 5 The worst-case results provided by the SDP from Proposition 1 provide a tight worst-case achievable for any operator B and any dual pairing $\langle ., . \rangle$.

2.3 Linearly Gram-representable first-order methods

This class of first-order methods contains as particular cases what we call in the following the class of *fixed-step linear first-order methods* (FSLFOM), whose iterations are defined by a linear equation (with known constant coefficients) involving the iterates and the corresponding (sub)gradients.

Definition 6 A fixed-step linear first-order method (FSLFOM) is a method which computes iterate x_i as the solution of [7]

$$t_{i,i}Bx_i + \sum_{k\in K} h_{i,i}^{(k)} g_i^{(k)} = \sum_{j=0}^{i-1}\left[t_{i,j}Bx_j + \sum_{k\in K} h_{i,j}^{(k)} g_j^{(k)}\right], \tag{FSLFOM}$$

where all coefficients $h_{i,j}^{(k)}, t_{i,j} \in \mathbb{R}$ $(0 \le j \le i$ and $k \in K)$ are fixed beforehand.

Note the class of FSLFOM is exactly the class of methods whose iterations can be written in the form (using first-order optimality conditions, and convexity of $F^{(k)}$):

$$\begin{aligned} x_i = \operatorname*{argmin}_{x\in\mathbb{E}} \Bigg\{ & \frac{t_{i,i}}{2}\|x\|_{\mathbb{E}}^2 + \sum_{k\in K} h_{i,i}^{(k)} F^{(k)}(x) \\ & - \left\langle \sum_{j=0}^{i-1}\left[t_{i,j}Bx_j + \sum_{k\in K} h_{i,j}^{(k)} \nabla F^{(k)}(x_j)\right], x \right\rangle \Bigg\}; \end{aligned}$$

which, in some sense, describes the most general method our framework can deal with. The computation of iterate x_i can also be written as the following linear equation, that involves a linear combination of the columns of matrix P_N (which contain the harvested first-order information about the problem so far) using a constant vector of coefficients $m_i \in \mathbb{R}^{(n+1)(N+2)}$:

$$P_N m_i = 0.$$

Note that coefficients in m_i corresponding to columns describing subsequent iterates (Bx_j and $g_j^{(k)}$ for all $j > i$ and $k \in K$) must naturally be equal to zero, as well as those of columns related to the optimal solution (Bx_* and $g_*^{(k)}$ for all $k \in K$). Therefore, any FSLFOM is linearly Gram-representable using the following formulation:

$$0 = P_N m_i \Leftrightarrow 0 = \|P_N m_i\|_{\mathbb{E}^*}^2 = \left\langle P_N m_i, B^{-1} P_N m_i \right\rangle = m_i^\top G_N m_i, \tag{3}$$

which is clearly linear in terms of the Gram matrix G_N. This can also easily be extended to cope with more general classes of linearly Gram-representable first-order methods, such as the following

$$c_i^{(\text{low})\top} F_N + b_i^{(\text{low})} \le m_i^\top G_N m_i \le c_i^{(\text{up})\top} F_N + b_i^{(\text{up})}, \tag{4}$$

where $c_i^{(\text{low})}, b_i^{(\text{low})}$ and $c_i^{(\text{up})}, b_i^{(\text{up})}$ are some fixed parameters. Those could for example be used in order to require a sufficient decrease condition (involving function values in F_N), or to consider methods that perform an inexact computation of the next iterate in (FSLFOM), such as in:

$$\|P_N m_i\|_{\mathbb{E}^*} \le \epsilon_i \Leftrightarrow m_i^\top G_N m_i \le \epsilon_i^2, \tag{Inexact FSLFOM}$$

where $\epsilon_i \ge 0$ is some tolerance on the accuracy of the computation of x_i in (FSLFOM).

[7] The iteration is written as an equality on $\mathbb{E}$, but it is possible and totally equivalent to write it on $\mathbb{E}^*$ using the operator B^{-1}.

Examples of FSLFOM Before going into the details of the performance estimation problems for our class of linear fixed-step methods over different classes of convex functions, let us give several examples of methods fitting into the model provided by (FSLFOM) and (Inexact FSLFOM).

- *Fixed-step subgradient and gradient algorithms.*
  Minimizing a convex function F using a fixed-step subgradient method is naturally described as $x_i = x_{i-1} - \alpha_i B^{-1} g_{i-1}$, with α_i some step size and $g_{i-1} \in \partial F(x_{i-1})$. The method clearly belongs to the class of FSLFOM, and its linear Gram matrix representation can be obtained using formulation (3).
- *Proximal methods and proximal gradient methods.*
  Consider a composite objective $F^{(1)} + F^{(2)}$ where $F^{(2)}$ admits a computable proximal operator. Minimizing this objective with a fixed-step proximal gradient method is usually described as performing an explicit (sub)gradient step on $F^{(1)}$ followed by a (proximal) minimization step involving $F^{(2)}$:

$$\begin{aligned} x_i &= \text{prox}_{\alpha_i F^{(2)}} \left(x_{i-1} - \alpha_i B^{-1} \tilde{\nabla} F^{(1)}(x_{i-1}) \right) \\ &= \operatorname*{argmin}_{x \in \mathbb{E}} \left\{ \alpha_i F^{(2)}(x) + \frac{1}{2} \left\| x_{i-1} - \alpha_i B^{-1} \tilde{\nabla} F^{(1)}(x_{i-1}) - x \right\|_{\mathbb{E}}^2 \right\}. \end{aligned}$$

  Optimality conditions on this last term allow writing each iteration as

$$B x_i + \alpha_i \tilde{\nabla} F^{(2)}(x_i) = B x_{i-1} - \alpha_i \tilde{\nabla} F^{(1)}(x_{i-1}),$$

  with some $\tilde{\nabla} F^{(2)}(x_i) \in \partial F^{(2)}(x_i)$, which is an implicit equation in x_i. This method is clearly a FSLFOM and therefore fits in our framework. Projected gradient methods are obtained using the same technique, but on the particular class of convex indicator functions $F^{(2)}$, whereas proximal point algorithms correspond to the case where $F^{(1)} = 0$.
- *Conditional gradient methods.*
  Consider an objective function $F^{(1)}$ to be minimized over a closed convex set Q, whose indicator function is $F^{(2)}$. Conditional gradient methods for this problem also fit the FSLFOM model. Indeed, their iterations take the following form (given a starting point z_0):

$$\begin{aligned} y_i &= \operatorname*{argmin}_{z \in \mathbb{E}} \left\{ \left\langle z - z_i, \tilde{\nabla} F^{(1)}(z_i) \right\rangle + F^{(2)}(z) \right\}, \\ z_{i+1} &= (1 - \lambda_i) z_i + \lambda_i y_i, \end{aligned}$$

  with coefficient $\lambda_i \in [0, 1]$ chosen beforehand. Using first-order necessary and sufficient optimality conditions on the intermediate optimization problem, we obtain that y_i can be defined by the following equation

$$\tilde{\nabla} F^{(1)}(z_i) = -\tilde{\nabla} F^{(2)}(y_i).$$

  This algorithm can also clearly be written as a FSLFOM ; one only needs to merge the two sequences of iterates y_i and z_i into a single sequence, defining for example the iterates using $x_{2i} = z_i$ and $x_{2i+1} = y_i$ for every $i = 0, 1, \ldots$
- *Inexact (sub)gradient methods.*
  Consider a convex function $F^{(1)}$ on which inexact steps are performed according to $x_{i+1} = x_i - \alpha_i B^{-1}(\tilde{\nabla} F^{(1)}(x_i) + \varepsilon_i)$, where errors ε_i on the computation of the subgradients are bounded. More precisely, given some tolerance $\epsilon_i \geq 0$, we assume that $\|\varepsilon_i\|_{\mathbb{E}^*} \leq \epsilon_i$. This can be written in the inexact FSLFOM format:

$$\left\| \alpha_i^{-1} B \left(x_{i+1} - x_i \right) + \tilde{\nabla} F^{(1)}(x_i) \right\|_{\mathbb{E}^*}^2 \leq \epsilon_i^2.$$

  Other noise models can also easily be used in the framework, such as the one proposed by d'Aspremont [8]. However, the inexact (δ, L)-oracles developed by Devolder et al. [10] do not seem to easily fit into the approach[8].

[8] This is due to the fact no necessary and sufficient interpolation conditions for functions admitting such an inexact oracle are known — that is, standard conditions are only necessary to guarantee interpolability. Using necessary conditions that are not sufficient still allows obtaining upper bounds on the worst-case behavior, but those may not be tight.

An even broader class of methods can be obtained by combining some of the above examples and/or restricting the functions to specific classes. For example, alternate projection-type algorithms are special cases of proximal methods applied to sums of convex indicator functions, hence can be represented in the FSLFOM format.

2.4 Simplified performance estimation problems

Note that for standard algorithms such as the above examples of FSLFOM, the SDP resulting from Proposition 1 can typically be further simplified, leading to a reduction in its size.

Corollary 1 *Consider a class of composite objective functions $\mathcal{F}_K(\mathbb{E})$ with n components, a performance measure $\mathcal{P}$ and an initialization condition $\mathcal{I}$ which are linearly Gram-representable, and a FSLFOM $\mathcal{M}$ whose iterations are linearly independent, meaning that the constant vectors m_i used to define iterates x_i in (3) are linearly independent*[9]*.*

In addition, assume there are p points $(g_i^{(k)}, f_i^{(k)})$ such that neither $g_i^{(k)}$ nor $f_i^{(k)}$ are used in the performance measure $\mathcal{P}$, the initial condition $\mathcal{I}$ and the method $\mathcal{M}$. Then, the performance estimation problem can be written as a convex SDP using variables $F_N \in \mathbb{R}^{n(N+2)-p}$ and $G_N \in \mathbb{S}^{(n+1)(N+2)-N-p-1}$, with the possible additional rank constraint rank $G_N \leq d$.

Proof One can remove from the original SDP formulation those p unnecessary points corresponding to p function values in variable F_N, and to p rows/columns in the Gram matrix variable G_N. Furthermore, the N equations defining the iterations allows to further substitute N variables, i.e. to remove N columns from P_N and hence N rows/columns from the Gram matrix variable G_N. The dimension of G_N can finally be decreased by one, using the fact that one of the $g_*^{(k)}$ may also be discarded, by substituting it using the optimality condition defining x_*.

Under the assumptions of Corollary 1, the large-scale assumption becomes $d \geq (n+1)(N+2)-N-p-1$. For example, when considering methods where only the output from a single oracle (among the n possible $F^{(k)}$) is used at each iteration, we have that $p = (n-1)(N+1)$, which leads to $d \geq N+n+2$.

Furthermore, for many standard performance measures such as objective function accuracy $F_N - F_*$ or distance to optimality $\|x_N - x_*\|_{\mathbb{E}}^2$, one arbitrary point x_i may be fixed to zero because solutions to the SDP are be invariant with respect to translations. This results then in the large-scale assumption $d \geq N+n+1$. For $n = 1$, we recover the standard $d \geq N+2$ appearing in the case of a single component in the objective function [43].

The original SDP from Proposition 1 may be challenging to solve in practice, because of its potentially large size on the one hand, and because it may lack an interior on the other hand. We observe that the simplified performance estimation problem described above typically improves the situation for both issues, reducing the size of the problem and solving in a lot of cases the issue of a lack of interior points.

3 Convex interpolation

In this section, we study convex interpolation problems for different standard classes of convex functions. The underlying motivation is to obtain discrete characterization of convex functions commonly arising in the context of convex optimization via first-order methods. More specifically, the classes of convex functions of interest for this section are all linearly Gram-representable (see Definition 2). Therefore, using those classes within the performance estimation framework will lead to tractable formulations providing tightness guarantees.

The main technical tools from this section are borrowed from convex analysis, we refer to the seminal references [1, 18, 38, 39] for details.

[9] This is a reasonable assumption, as every method using new information at each iteration will necessarily satisfy it. This is does not imply that the points x_i themselves are linearly independent.

3.1 Functional characteristics

Consider a proper, closed and convex function f. The main characteristics of interest for us are the following, all commonly appearing in the context of first-order convex optimization.

(a) **Smoothness**: there exists some $L \in \mathbb{R}^{++} \cup \{\infty\}$ such that $\frac{1}{L}\|g_1 - g_2\|_{\mathbb{E}^*} \leq \|x_1 - x_2\|_{\mathbb{E}}$ holds for all pairs $x_1, x_2 \in \mathbb{E}$ and corresponding subgradients $g_1, g_2 \in \mathbb{E}^*$ (i.e. such that $g_1 \in \partial f(x_1)$ and $g_2 \in \partial f(x_2)$).
(b) **Strong convexity**: there exists some $\mu \in \mathbb{R}^+$ such that the function $f(x) - \frac{\mu}{2}\|x\|^2_{\mathbb{E}}$ is convex.
(c) **Gradient boundedness**: there exists some $M \in \mathbb{R}^+ \cup \{\infty\}$ such that $\|g\|_{\mathbb{E}^*} \leq M$ holds for all subgradients $g \in \mathbb{E}^*$ (i.e., such that $\exists x: \; g \in \partial f(x)$).
(d) **Domain boundedness**: there exists some $D \in \mathbb{R}^+ \cup \{\infty\}$ such that $\|x\|_{\mathbb{E}} \leq D$ holds for all x belonging to the domain $\{x \in \mathbb{E} : f(x) < \infty\}$.

Alternatively, domain and gradient boundedness can be specified in terms of diameters instead of radii.

(c') **Gradient boundedness**: there exists some $M \in \mathbb{R}^+ \cup \{\infty\}$ such that $\|g_1 - g_2\|_{\mathbb{E}^*} \leq M$ holds for all subgradients $g_1, g_2 \in \mathbb{E}^*$ (i.e., such that $\exists x_1, x_2 : \; g_1 \in \partial f(x_1)$ and $g_2 \in \partial f(x_2)$).
(d') **Domain boundedness**: there exists $D \in \mathbb{R}^+ \cup \{\infty\}$ such that $\|x_1 - x_2\|_{\mathbb{E}} \leq D$ holds for all pairs x_1, x_2 belonging to the domain $\{x \in \mathbb{E} : f(x) < \infty\}$.

As some characteristics are incompatible with each other (e.g., gradient boundedness is incompatible with strong convexity, domain boundedness is incompatible with smoothness), we define the following three classes of functions combining specific pairs of properties.

Definition 7 Let $f : \mathbb{E} \to \mathbb{R} \cup \{\infty\}$ be a proper, closed and convex function, denoted by $f \in \mathcal{F}_{0,\infty}$. We say that:

- $f \in \mathcal{F}_{\mu,L}(\mathbb{E})$ (L-smooth μ-strongly convex functions) if it satisfies conditions (a) and (b) with $\mu < L$.
- $f \in \mathcal{C}_{M,L}(\mathbb{E})$ (L-smooth M-Lipschitz convex functions) if it satisfies conditions (a) and (c); alternatively, $f \in \mathcal{C}'_{M,L}(\mathbb{E})$ if it satisfies (a) and (c').
- $f \in \mathcal{S}_{D,\mu}(\mathbb{E})$ (D-bounded μ-strongly convex functions) if it satisfies conditions (b) and (d); alternatively $f \in \mathcal{S}'_{D,\mu}(\mathbb{E})$ if it satisfies (b) and (d').

Note that boundedness and smoothness constants are allowed to take the value ∞, in order to allow the use of unbounded (domain or gradient) and non-smooth functions as well. We handle those using conventions $1/\infty = 0$ and $\infty - c = \infty$ for any $c \in \mathbb{R}$. By assuming $\mu < L$, we exclude the classes $\mathcal{F}_{L,L}(\mathbb{E})$ for $L \geq 0$. Those only contain quadratic functions of the form $f(x) = \frac{L}{2}\|x\|^2_{\mathbb{E}} + \langle b, x\rangle + c$ for some $b \in \mathbb{E}^*$ and $c \in \mathbb{R}$, for which it would be straightforward to obtain interpolation conditions.

A basic building block for the smooth convex interpolation conditions proposed in [43] comes from Fenchel-Legendre conjugation. In particular, when considering functions f in the class $\mathcal{F}_{0,\infty}(\mathbb{E})$, the duality correspondence $f \in \mathcal{F}_{\mu,L}(\mathbb{E}) \Leftrightarrow f^* \in \mathcal{F}_{1/L,1/\mu}(\mathbb{E}^*)$ was intensively used to require smoothness of the convex interpolant. In the following, we additionally use for functions f in $\mathcal{F}_{0,\infty}(\mathbb{E})$ the duality correspondences $f \in \mathcal{C}_{M,L}(\mathbb{E}) \Leftrightarrow f^* \in \mathcal{S}_{M,1/L}(\mathbb{E}^*)$ and its variant $f \in \mathcal{C}'_{M,L}(\mathbb{E}) \Leftrightarrow f^* \in \mathcal{S}'_{M,1/L}(\mathbb{E}^*)$, in order to include boundedness properties in the convex interpolating functions, along with smoothness.

Theorem 1 *Consider a function $f \in \mathcal{F}_{0,\infty}(\mathbb{E})$. We have $f \in \mathcal{C}_{M,\infty}(\mathbb{E})$ (resp. $f \in \mathcal{C}'_{M,\infty}(\mathbb{E})$) if and only if $f^* \in \mathcal{S}_{M,0}(\mathbb{E}^*)$ (resp. $f^* \in \mathcal{S}'_{M,0}(\mathbb{E}^*)$).*

Proof This follows from the equivalence: $g \in \partial f(x) \Leftrightarrow x \in \partial f^*(g) \Leftrightarrow f(x) + f^*(g) = \langle g, x\rangle$ that holds for every function f in $\mathcal{F}_{0,\infty}(\mathbb{E})$.

3.2 Interpolation conditions

In this section, we provide interpolation conditions for the previously introduced three classes of functions. We start by recalling the following known interpolation result [43, Theorem 6][10].

[10] Theorem 2 is formally proven in [43] for the case of the standard inner product $\langle x, y\rangle = x^\top y$ (and therefore also only for $\|.\|_2^2$). However, its proof can be rewritten in a completely straightforward manner to obtain the desired result for general inner products on $\mathbb{E}$ and self-adjoint positive definite linear operators B, and the corresponding induced primal and dual Euclidean norms.

Theorem 2 *The set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{F}_{\mu,L}$-interpolable if and only if the following set of conditions holds for every pair of indices $i \in I$ and $j \in I$*

$$f_i - f_j - \langle g_j, x_i - x_j\rangle \geq \frac{1}{2(1-\mu/L)}\left(\frac{1}{L}\|g_i - g_j\|^2_{\mathbb{E}^*} + \mu\|x_i - x_j\|^2_{\mathbb{E}} - 2\frac{\mu}{L}\langle g_j - g_i, x_j - x_i\rangle\right).$$

In particular, the simpler interpolation conditions for closed, convex proper functions (i.e. $\mathcal{F}_{0,\infty}(\mathbb{E})$ interpolation) are

$$f_i - f_j - \langle g_j, x_i - x_j\rangle \geq 0 \ \forall i, j \in I, \tag{5}$$

which will serve to develop our next interpolation conditions. We start with $\mathcal{S}_{D,\mu}(\mathbb{E})$-interpolability, and later obtain $\mathcal{C}_{M,L}(\mathbb{E})$-interpolation conditions using conjugation.

Theorem 3 *The set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{S}_{D,\mu}$- (D-bounded, μ-strongly convex) (resp. $\mathcal{S}'_{D,\mu}$-) interpolable if and only if the following set of conditions holds for every pair of indices $i \in I$ and $j \in I$*

$$\begin{aligned} &f_i - f_j - \langle g_j, x_i - x_j\rangle \geq \frac{\mu}{2}\|x_i - x_j\|^2_{\mathbb{E}}, \\ &\|x_j\|_{\mathbb{E}} \leq D \quad \textit{(resp. } \|x_j - x_i\|_{\mathbb{E}} \leq D\textit{).} \end{aligned}$$

Proof Every function $f \in \mathcal{S}_{D,\mu}(\mathbb{E})$ (resp. $f \in \mathcal{S}'_{D,\mu}(\mathbb{E})$) satisfies the conditions. To prove that they are sufficient, consider the following construction:

$$f(x) = \begin{cases} \max_{i\in I}\left\{f_i + \langle g_i, x - x_i\rangle + \frac{\mu}{2}\|x - x_i\|^2_{\mathbb{E}}\right\} & \text{if } x \in \operatorname{conv}\left(\{x_i\}_{i\in I}\right), \\ \infty & \text{elsewhere.} \end{cases}$$

Observe that f is μ-strongly convex (convex domain, and maximum of μ-strongly convex functions), and that it does interpolate the set $\{(x_i, g_i, f_i)\}_{i\in I}$. First, we have

$$\begin{aligned} f(x_j) &= \max_{i\in I}\left\{f_i + \langle g_i, x_j - x_i\rangle + \frac{\mu}{2}\|x_j - x_i\|^2_{\mathbb{E}}\right\}, \\ &\leq f_j, \end{aligned}$$

using interpolation conditions. By noting that the maximum is bigger than taking individually the component j, we also have that

$$\max_{i\in I}\left\{f_i + \langle g_i, x_j - x_i\rangle + \frac{\mu}{2}\|x_j - x_i\|^2_{\mathbb{E}}\right\} \geq f_j,$$

which allows to conclude that $f(x_j) = f_j$. To obtain that $g_j \in \partial f(x_j)$, let us write

$$\begin{aligned} f(x) &= \max_{i\in I}\left\{f_i + \langle g_i, x - x_i\rangle + \frac{\mu}{2}\|x - x_i\|^2_{\mathbb{E}}\right\}, \\ &\geq \max_{i\in I}\left\{f_i + \langle g_i, x - x_i\rangle\right\}, \\ &\geq f_j + \langle g_j, x - x_j\rangle. \end{aligned}$$

Finally, note that $\operatorname{conv}\left(\{x_i\}_{i\in I}\right) \subseteq B_{\mathbb{E}}(0, D)$, where $B_{\mathbb{E}}(0, D)$ is the ball centered at the origin with radius D according to norm $\|.\|_{\mathbb{E}}$. Indeed, choose $z = \sum_{i\in I}\lambda_i x_i$ with $\lambda_i \geq 0$ and $\sum_{i\in I}\lambda_i = 1$, we have $\|z\|_{\mathbb{E}} \leq \sum_{i\in I}\lambda_i\|x_i\|_{\mathbb{E}} \leq D$, and f has a bounded domain of radius D. Hence $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{S}_{D,\mu}$-interpolable, which concludes the proof for the $\mathcal{S}_{D,\mu}$ part.

To obtain the same result for $\mathcal{S}'_{D,\mu}$, note that $\forall y, z \in \operatorname{conv}(\{x_i\}_{i\in I})$, we can write $y = \sum_i \lambda_i x_i$ and $z = \sum_i \gamma_i x_i$ with $\lambda_i, \gamma_i \geq 0$ and $\sum_i \lambda_i = \sum_i \gamma_i = 1$. Hence, $\|y - z\|_{\mathbb{E}} \leq \sum_i \lambda_i \sum_j \gamma_j \|x_i - x_j\|_{\mathbb{E}} \leq D$.

This interpolation result can be used immediately to develop interpolation conditions for the class of convex functions with bounded gradient, using the conjugate duality between smoothness and strong convexity on the one hand, and gradient and domain boundedness on the other hand.

Theorem 4 *The set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{C}_{M,L}$ (L-smooth with M-bounded subgradients) (resp $\mathcal{C}'_{M,L}$) interpolable if and only if the following set of conditions holds for every pair of indices $i \in I$ and $j \in I$*

$$f_i - f_j - \langle g_j, x_i - x_j\rangle \geq \frac{1}{2L}\|g_i - g_j\|^2_{\mathbb{E}^*}, \tag{6}$$

$$\|g_j\|_{\mathbb{E}^*} \leq M \quad \text{(resp. } \|g_j - g_i\|_{\mathbb{E}^*} \leq M\text{)}. \tag{7}$$

Proof Note that a function $f \in \mathcal{C}_{M,L}(\mathbb{E})$ (resp. $f \in \mathcal{C}'_{M,L}(\mathbb{E})$) interpolates the set $\{(x_i, g_i, f_i)\}_{i\in I}$ if and only if there exists a corresponding conjugate function $f^* \in \mathcal{S}_{M,1/L}(\mathbb{E}^*)$ (resp. $f^* \in \mathcal{S}'_{M,1/L}(\mathbb{E}^*)$) interpolating the set $\{(g_i, x_i, \langle g_i, x_i\rangle - f_i)\}_{i\in I} = \{(\tilde{x}_i, \tilde{g}_i, \tilde{f}_i)\}_{i\in I}$ (see Section 3.1). Using interpolation conditions from Theorem 3, such a conjugate function f^* exists if and only if

$$\tilde{f}_i - \tilde{f}_j - \langle \tilde{x}_i - \tilde{x}_j, \tilde{g}_j\rangle \geq \frac{1}{2L}\|\tilde{x}_i - \tilde{x}_j\|^2_{\mathbb{E}^*},$$

$$\|\tilde{x}_j\|_{\mathbb{E}^*} \leq M \quad \text{(resp. } \|\tilde{x}_j - \tilde{x}_i\|_{\mathbb{E}^*} \leq M\text{)},$$

which are respectively equivalent to conditions (6) and (7).

3.3 Indicator and support functions

The use of projection (to deal with constraints) and regularization is so recurrent in optimization that we dedicate the next lines to interpolation procedures specifically tailored to deal with them.

Indicator functions In our setting, an indicator function is a closed convex function taking only values 0 and ∞, for which it can be shown that the domain must be a closed convex set. As explained earlier, this class of functions is particularly interesting when considering projection operators in the context of performance estimation, as a proximal step over an indicator function is equivalent to a projection on its domain.

Given such a proper and closed convex function $i : \mathbb{E} \to \{0, \infty\}$, we say that it is a D-bounded indicator function (which we denote by $f \in \mathcal{I}_D(\mathbb{E})$ — resp. $f \in \mathcal{I}'_D(\mathbb{E})$) if there exists a radius (resp. a diameter) $0 \leq D \leq \infty$ such that $\|x\|_{\mathbb{E}} \leq D$ (resp. $\|x_1 - x_2\|_{\mathbb{E}} \leq D$) holds for all x belonging to the domain $\{x : i(x) = 0\}$ (resp. for all x_1, x_2 belonging to the domain $\{x : i(x) = 0\}$).

This corresponds to a particular case of the $\mathcal{S}_{D,\mu}$- (or $\mathcal{S}'_{D,\mu}$)-interpolation problem with $\mu = 0$. Note however that indicator function interpolation is not completely straightforward from $\mathcal{S}'_{D,\mu}$-interpolation, as for example requiring the corresponding interpolation constraints in addition to $f_i = 0$ would not a priori guarantee that the interpolated function from Theorem 3 would satisfy $f(x) = 0$ on $\operatorname{dom} f$.

Theorem 5 *The set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{I}_D$ (resp. $\mathcal{I}'_D$)-interpolable, i.e. interpolable by a D-bounded indicator, if and only if the following inequalities hold for every pair of indices $i \in I$ and $j \in I$:*

$$\begin{aligned} &f_i = 0,\\ &\langle g_j, x_i - x_j\rangle \leq 0,\\ &\|x_i\|_{\mathbb{E}} \leq D \quad \text{(resp. } \|x_j - x_i\|_{\mathbb{E}} \leq D\text{)}. \end{aligned} \tag{8}$$

Proof Any function $f \in \mathcal{I}_D(\mathbb{E})$ (resp. $f \in \mathcal{I}'_D(\mathbb{E})$) satisfies those conditions. To prove that they are sufficient, let us construct a convex set whose indicator function interpolates the set $\{(x_i, g_i, 0)\}$. That is, we construct a closed convex set Q containing all x_i's, for which $\|x\|_{\mathbb{E}} \leq D$ holds for all $x \in Q$ (resp. $\|x - y\|_{\mathbb{E}} \leq D$ for all $x, y \in Q$), and such that $\langle g_i, x - x_i\rangle \leq 0$ holds for all $x \in Q$.
We start with the simpler case $D = \infty$, by considering the polyhedral set

$$Q = \{x \in \mathbb{E} \mid \langle a_j, x\rangle \leq b_j \; \forall j \in I\},$$

with $a_j = g_j$ and $b_j = \langle g_j, x_j\rangle$. The construction guarantees that $x_i \in Q$. Indeed, by Condition (8) we have $\langle g_j, x_i\rangle \le \langle g_j, x_j\rangle$, which is equivalent to $\langle a_j, x_i\rangle \le b_j$ using the definitions of a_j and b_j, and therefore guarantees that $x_i \in Q$.
In order to add the boundedness requirement, we replace the set Q by the following $\tilde{Q} = Q \cap \operatorname{conv}(\{x_i\}_{i\in I})$. This new set $\tilde{Q}$ is still convex (intersection of two convex sets), it also trivially still satisfies inclusions $x_i \in \tilde{Q}$ (which are by construction both contained in Q and $\operatorname{conv}(\{x_i\}_i)$) and conditions $\langle g_i, x - x_i\rangle \le 0$ for all $x \in \tilde{Q}$ (since $\tilde{Q} \subseteq Q$). In addition, $\tilde{Q}$ has a radius bounded above by D, because D is an upper bound on the radius (resp. diamater) of $\operatorname{conv}(\{x_i\}_{i\in I})$. It is therefore clear that the indicator function $I_{\tilde{Q}} \in \mathcal{I}_D(\mathbb{E})$ (resp. $\mathcal{I}'_D(\mathbb{E})$) interpolates $\{(x_i, g_i, 0)\}_{i\in I}$.

Support functions It is a standard observation that support functions are convex conjugates of indicator functions. Indeed, the support function for the closed convex set $Q \subseteq \mathbb{E}$ is defined as

$$\sigma_Q(s) = \sup_{x\in Q} \langle s, x\rangle = \sup_{x\in\mathbb{E}} \langle s, x\rangle - I_Q(x).$$

Support functions are very commonly used in applications. In particular, all norms, which are used for regularization, are support functions (e.g. the l_1 norm is the support function of the unit ball for $\|.\|_\infty$).

Denote the set of support functions with an M-Lipschitz condition by $\mathcal{I}^*_M(\mathbb{E})$ (resp. $\mathcal{I}'^*_M(\mathbb{E})$). Using conjugacy, interpolation conditions for indicator functions immediately give us the equivalent result for support functions. Indeed, requiring a set $S = \{(x_i, g_i, f_i)\}_{i\in I}$ to be $\mathcal{I}^*_M$ (resp. $\mathcal{I}'^*_M$)-interpolable is equivalent to requiring the set $\tilde{S} = \{(g_i, x_i, \langle g_i, x_i\rangle - f_i)\}_{i\in I}$ to be $\mathcal{I}_M$ (resp. $\mathcal{I}'_M$)-interpolable, and we obtain the following consequence of Theorem 5.

Corollary 2 *The set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{I}^*_M$ (resp. $\mathcal{I}'^*_M$)-interpolable, i.e. interpolable by a support function with M-bounded subgradients, if and only if the following inequalities hold for every pair of indices $i \in I$ and $j \in I$:*

$$\begin{aligned} \langle g_i, x_i\rangle - f_i &= 0, \\ \langle g_i - g_j, x_j\rangle &\le 0, \\ \|g_i\|_{\mathbb{E}^*} &\le M, \quad \textit{(resp. } \|g_i - g_j\|_{\mathbb{E}^*} \le M\textit{).} \end{aligned}$$

3.4 Smooth non-convex interpolation

In this short section, we derive interpolation conditions for smooth, not necessarily convex, functions. Those conditions are also linearly Gram-representable, and can be used to obtain tight versions of (f-PEP) for non-convex optimization.

Definition 8 Let $L \in \mathbb{R}^+$. A differentiable function $f : \mathbb{E} \to \mathbb{R} \cup \{\infty\}$ is L-smooth, denoted by $f \in \mathcal{F}_{-L,L}(\mathbb{E})$), if it satisfies the following condition for all $x, y \in \mathbb{E}$:

$$\left|f(x) + \langle \nabla f(x), y - x\rangle - f(y)\right| \le \frac{L}{2}\|x - y\|^2_{\mathbb{E}}.$$

The following lemma will be used to derive interpolation conditions for $f \in \mathcal{F}_{-L,L}(\mathbb{E})$ from the smooth convex case.

Lemma 1 *Let $L \in \mathbb{R}^+$, and consider a function $f : \mathbb{E} \to \mathbb{R}\cup\{\infty\}$. We have the equivalence $f \in \mathcal{F}_{-L,L}(\mathbb{E}) \Leftrightarrow f + \frac{L}{2}\|x\|^2_{\mathbb{E}} \in \mathcal{F}_{0,2L}(\mathbb{E})$.*

Proof Let $f : \mathbb{E} \to \mathbb{R} \cup \{\infty\}$ and define $h(x) = f(x) + \frac{L}{2}\|x\|^2_{\mathbb{E}}$. Since we have that $\nabla h(x) = \nabla f(x) + LBx$, it follows that, for all $x, y \in \mathbb{E}$:

$$\begin{aligned} f(x) + \langle \nabla f(x), y - x\rangle - f(y) \leq \frac{L}{2}\|x-y\|^2_{\mathbb{E}} &\Leftrightarrow h(y) \geq h(x) + \langle \nabla h(x), y-x\rangle, \\ -f(x) - \langle \nabla f(x), y - x\rangle + f(y) \leq \frac{L}{2}\|x-y\|^2_{\mathbb{E}} &\Leftrightarrow h(y) \leq h(x) + \langle \nabla h(x), y-x\rangle \\ &\qquad + L\|x-y\|^2_{\mathbb{E}}, \end{aligned}$$

where the equivalences are obtained by expressing f and ∇f in terms of h and ∇h (or reciprocally), which proves our statement.

From Lemma 1 and Theorem 2, it is now straightforward to establish the desired interpolation conditions.

Theorem 6 *Let $L \in \mathbb{R}^{++}$, the set $\{(x_i, g_i, f_i)\}_{i\in I}$ is $\mathcal{F}_{-L,L}$ (L-smooth) - interpolable if and only if the following inequality holds $\forall i, j \in I$:*

$$f_i \geq f_j - \frac{L}{4}\|x_i - x_j\|^2_{\mathbb{E}} + \frac{1}{2}\langle g_i + g_j, x_j - x_i\rangle + \frac{1}{4L}\|g_i - g_j\|^2_{\mathbb{E}^*}.$$

Proof As L is positive and finite, the statement follows from the equivalence between $\mathcal{F}_{-L,L}$-interpolability of $\{(x_i, g_i, f_i)\}_{i\in I}$ and $\mathcal{F}_{0,2L}$-interpolability of $\{(\, x_i, g_i + LBx_i, f_i + \frac{L}{2}\|x_i\|^2_{\mathbb{E}}\,)\}_{i\in I}$.

4 Algorithm analysis

In this section, we analytically and numerically study different algorithms for solving variants of (CM), and compare our results with standard guarantees from the literature[11]. We begin with an analytical study of a proximal point algorithm (Section 4.1). This is followed by a comparison between several standard variants of fast proximal gradient methods (Section 4.2) using the PEP approach. On the way, we propose an extension of the optimized gradient method (OGM) proposed by Kim and Fessler [22]. Finally, we conclude by applying our framework to a conditional gradient method (Section 4.4) and to two alternate projections schemes (Section 4.5). Those choices illustrate the applicability of the approach for studying a large variety of methods and performance measures.

4.1 A proximal point algorithm

Consider a simple model with only one convex (possibly non-smooth) term in the objective function,

$$\min_{x\in\mathbb{E}} F(x),$$

with $F \in \mathcal{F}_{0,\infty}(\mathbb{E})$. In this first example, we assume that the following proximal operation is easy to compute for F, and defines the next iterate (using a given step size α_{k+1}):

$$x_{k+1} = \text{prox}_{\alpha_{k+1}F}(x_k) = \operatorname*{argmin}_{x\in\mathbb{E}} \left\{ \alpha_{k+1}F(x) + \frac{1}{2}\|x_k - x\|^2_{\mathbb{E}} \right\}.$$

Using an observation made in Section 2.3, we see that iterations can also be written in the form of an implicit method $x_{k+1} = x_k - \alpha_{k+1}B^{-1}g_{k+1}$, for some $g_{k+1} \in \partial F(x_{k+1})$, and hence belong to the class (FSLFOM).

For a recent overview and motivations concerning proximal algorithms, we refer the reader to the work of Combettes and Pesquet[12] [7], and to the review works of Bertsekas [3] and Parikh and Boyd [35]. For a historical point of view on those methods, we refer to the pioneer works of Moreau [25], Rockafellar [37] and the analysis of Guler [17].

[11] Note that most of the literature results are presented when B is the identity operator (and hence $\mathbb{E} = \mathbb{E}^*$). We will nevertheless compare our slightly more general results with the standard bounds from the literature (thus even when they are officially valid only for B being the identity) — we recall that our results are valid for any self-adjoint positive definite linear operator $B : \mathbb{E} \to \mathbb{E}^*$ (see Remark 5).

[12] This work among others features a large list of known proximal operators.

```
Proximal Point Algorithm (PPA)
    Input: $F \in \mathcal{F}_{0,\infty}(\mathbb{E})$, $x_0 \in \mathbb{E}$. Parameters: $\{\alpha_k\}_{k\geq 1}$ with $\alpha_k > 0$.
    For $k = 1 : N$
        $x_k = \operatorname{prox}_{\alpha_k F}(x_{k-1})$
```

4.1.1 Convergence of PPA in function and gradient values

The standard convergence result for the proximal point algorithm is provided by Guler in [17, Theorem 2.1] :

$$F(x_N) - F_* \leq \frac{R^2}{2\sum_{k=1}^N \alpha_k},$$

for any initial condition x_0 satisfying $\|x_0 - x_*\|_{\mathbb{E}} \leq R$. We are able to divide this bound by 2 using the PEP approach.

Theorem 7 *Let $\{\alpha_k\}_k$ be a sequence of positive step sizes and x_0 some initial iterate satisfying $\|x_0 - x_*\|_{\mathbb{E}} \leq R$ for some optimal point x_*. Any sequence $\{x_k\}_k$ generated by the proximal point algorithm with step sizes $\{\alpha_k\}_k$ applied to a function $F \in \mathcal{F}_{0,\infty}(\mathbb{E})$ satisfies*

$$F(x_N) - F_* \leq \frac{R^2}{4\sum_{k=1}^N \alpha_k}$$

and this bound cannot be improved, even in dimension one ($\dim \mathbb{E} = \dim \mathbb{E}^* = 1$*).*

Proof We first prove that the bound is tight. For given N, R and step sizes $\{\alpha_k\}_{1\leq k\leq N}$, we consider the l_1-shaped one-dimensional function

$$F(x) = \frac{\sqrt{B}R|x|}{2\sum_{k=1}^N \alpha_k} = \frac{R\|x\|_{\mathbb{E}}}{2\sum_{k=1}^N \alpha_k},$$

for which $x_* = 0$ and $F_* = 0$. Applying N iterations of PPA with step sizes $\{\alpha_k\}_{1\leq k\leq N}$ to this one-dimensional function, starting from $x_0 = -\frac{R}{\sqrt{B}}$ (which satisfies $\|x_0 - x_*\|_{\mathbb{E}} \leq R$), leads to a sequence whose last iterate satisfies

$$F(x_N) - F_* = \frac{R^2}{4\sum_{k=1}^N \alpha_k}.$$

Indeed, note that for $x \neq 0$, we have $\nabla F(x) = \operatorname{sign}(x)\frac{\sqrt{B}R}{2\sum_{k=1}^N \alpha_k}$. Hence,

$$x_N = x_0 + B^{-1}\sum_{k=1}^N \alpha_k \frac{\sqrt{B}R}{2\sum_{k=1}^N \alpha_k} = -\frac{R}{2\sqrt{B}}$$

which implies the desired result.

The proof of the upper bound is based on considering a simplified formulation of (f-PEP) for the proximal point algorithm, computing its dual and exhibiting a feasible solution to that dual. As it is a little longer it is relegated to Appendix A.

Let us considering another convergence measure based on the residual subgradient norm. Studying a PEP similar to the one above, we obtained strong numerical evidence for the following conjecture.

Conjecture 1 Let $\{\alpha_k\}_k$ be a sequence of positive step sizes and x_0 some initial iterate satisfying $\|x_0 - x_*\|_{\mathbb{E}} \leq R$ for some optimal point x_*. For any sequence $\{x_k\}_k$ generated by the proximal point algorithm with step sizes $\{\alpha_k\}_k$ on a function $F \in \mathcal{F}_{0,\infty}(\mathbb{E})$, there exists for every iterate x_N a subgradient $g_N \in \partial F(x_N)$ such that

$$\|g_N\|_{\mathbb{E}^*} \leq \frac{R}{\sum_{k=1}^N \alpha_k}.$$

In particular, the choice $g_N = \frac{Bx_{N-1} - Bx_N}{\alpha_N}$ is a subgradient satisfying the inequality.

Observe that this bound cannot be improved, as it is attained on the (one-dimensional) l_1-shaped function $F(x) = \frac{\sqrt{B}R|x|}{\sum_{k=1}^{N}\alpha_k}$ started from $x_0 = -R/\sqrt{B}$. The particular choice of subgradient suggested in the theorem corresponds to the subgradient appearing in the proximal operation when written as an implicit subgradient step.

This sort of convergence results in terms of residual (sub)gradient norm is particularly interesting when considering dual methods. In that case, the dual residual gradient norm corresponds to the primal distance to feasibility (see e.g., [9]).

4.2 Fast gradient methods

In this section, we consider the two-term composite objective function

$$\min_{x\in\mathbb{E}} \left\{ F(x) \equiv F^{(1)}(x) + F^{(2)}(x) \right\}, \tag{9}$$

with $F^{(1)} \in \mathcal{F}_{0,L}(\mathbb{E})$ (smooth convex function) and $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$ (non-smooth convex function). We assume that gradients are easy to compute for $F^{(1)}$, and that the proximal operation is easy to compute for $F^{(2)}$:

$$\text{prox}_{\alpha F^{(2)}}(x) = \underset{y\in\mathbb{E}}{\text{argmin}} \left\{ \alpha F^{(2)}(y) + \frac{1}{2}\|x-y\|_{\mathbb{E}}^2 \right\}.$$

In order to approximatively solve (9), it is common to use different variants of fast proximal gradient methods (FPGM). We numerically investigate the worst-case guarantees of two variants using different step size policies, and propose new variants with slightly better worst-case behaviors. Also, we highlight differences in the worst-case performances obtained in the cases where $F^{(2)} = 0$ (unconstrained smooth convex minimization), $F^{(2)} \in \mathcal{I}_\infty(\mathbb{E})$ (constrained smooth convex minimization) and the general $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$ (non-smooth composite convex minimization).

In the following, we call the standard fast proximal gradient method FPGM1 (FISTA [2]) and introduce FPGM2, a variant with slightly better guarantees, and POGM, a novel proximal version of the optimized gradient method [22]. FPGM2 and POGM illustrate how performance estimation problems can be used in the development of new optimization algorithms ; their study in this paper remains however entirely numerical.

4.2.1 Standard Fast Proximal Gradient Methods (FPMG1)

The first variants of accelerated proximal methods we are considering use a standard proximal step after an explicit gradient step for generating the so-called *primary sequence* $\{y_k\}_k$.

Fast Proximal Gradient Method (FPGM1)

```
Input: $F^{(1)} \in \mathcal{F}_{0,L}(\mathbb{E})$, $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$ $x_0 \in \mathbb{E}$, $y_0 = x_0$.
For $k = 1 : N$
    $y_k = \text{prox}_{F^{(2)}/L}\left(x_{k-1} - \frac{1}{L}B^{-1}\nabla F^{(1)}(x_{k-1})\right)$
    $x_k = y_k + \alpha_k(y_k - y_{k-1})$
```

In this algorithm, we refer to coefficients α_k as *inertial parameters*. We use two standard variants: $\alpha_k^{(a)} = \frac{k-1}{k+2}$ — among others proposed in [42,44] — and $\alpha_k^{(b)} = \frac{\theta_{k-1}-1}{\theta_k}$ (with $\theta_k = \frac{1+\sqrt{4\theta_{k-1}^2+1}}{2}$ and $\theta_0 = 1$) — see [2, 28,44]. For both variants, the standard convergence result is (see e.g., [2,42]):

$$F(y_N) - F_* \leq \frac{2LR^2}{(N+1)^2}, \tag{10}$$

for any initial iterate x_0 such that $\|x_0 - x_*\|_{\mathbb{E}} \leq R$. We numerically compare those two variants of FPGM1 using (f-PEP) on Fig. 4.2.1 (left plot). After 100 iterations, both inertial parameter policies behave about the same way (parameters $\alpha_k^{(b)}$ perform only about 2% better than $\alpha_k^{(a)}$ in terms of worst-case performances). We also observe that the behavior of both variants of FPGM1 is well captured by the standard guarantee (10).

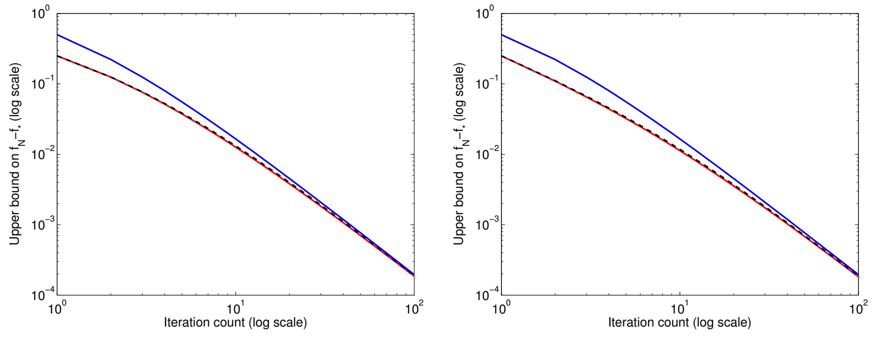


Fig. 1 Comparison of the worst-case convergence speed of the different variants of FPGM1 (left) and FPGM2 (right) for $N \in \{1,\ldots,100\}$, $L = 1$ and $R = 1$. Curves correspond to the different inertial coefficient, namely $\alpha_k^{(a)}$ (dashed, black) and $\alpha_k^{(b)}$ (red), and to the standard guarantee (10) (blue).

4.2.2 New Fast Proximal Gradient Methods (FPGM2)

Secondary sequences $\{x_k\}$ are usually converging slightly faster than primary sequences $\{y_k\}$ in the unconstrained case ($F^{(2)} = 0$), as observed in [22,43]. However, some issues may arise with the secondary sequences of FPGM1 when applied to constrained or proximal problems: iterates may in some cases become infeasible, or the objective may become unbounded (see Table 1 below). We therefore propose a new variant of a fast proximal gradient method called FPGM2, also with two different step size policies, that does not suffer from theses drawbacks. Part of the underlying motivation behind FPGM2 is also the ability to generalize it later to the optimized gradient method.

Remark 6 The design of FPGM2 is based on two ideas: on the one hand, it should be equivalent to the standard fast gradient method in the case of smooth unconstrained convex minimization, and on the other hand, it should not move after two consecutive iterates have reached the same optimal point for (9) (i.e., $x_{k-1} = x_k = x_*$ implies $x_{k+1} = x_*$).

Fast Proximal Gradient Method 2 (FPGM2)

```
Input: $F^{(1)} \in \mathcal{F}_{0,L}(\mathbb{E})$, $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$ $x_0 \in \mathbb{E}$, $z_0 = y_0 = x_0$.
For $k = 1 : N$
    $y_k = x_{k-1} - \frac{1}{L} B^{-1} \nabla F^{(1)}(x_{k-1})$
    $z_k = y_k + \alpha_k (y_k - y_{k-1}) + \frac{\alpha_k}{L\gamma_{k-1}} (z_{k-1} - x_{k-1})$
    $x_k = \mathrm{prox}_{\gamma_k F^{(2)}}(z_k)$
```

In this algorithm, we use the coefficients $\gamma_k = \frac{\alpha_k + 1}{L}$. Note that we introduced two intermediate sequences: on the one hand sequence $\{\gamma_k\}_k$, corresponding to the step sizes to be taken by the proximal steps, and on the other hand sequence $\{z_k\}_k$, which keeps track of the subgradient used in the proximal steps (note that $\frac{1}{\gamma_k}(z_k - x_k)$ corresponds to the subgradient used in the proximal step from z_k to x_k). Although FPGM2 may look more intricate than the classical FPGM1, it is in fact simpler, as it involves only one sequence on which both implicit (proximal) and explicit (gradient) steps are being taken. Indeed, explicit steps are taken using gradient values of $F^{(1)}$ at x_k, and subgradients used in the proximal steps are subgradients of $F^{(2)}$ also at x_k. This can also be seen by rewriting the iterations of FPGM2 using the secondary sequence $\{x_k\}_k$ only, in the following way

$$\begin{aligned} x_{k+1} = x_k &+ \alpha_{k+1}(x_k - x_{k-1}) \\ &+ \frac{\alpha_{k+1}}{L}B^{-1}\nabla F^{(1)}(x_{k-1}) - \frac{1}{L}B^{-1}\nabla F^{(1)}(x_k) - \frac{\alpha_{k+1}}{L}B^{-1}\nabla F^{(1)}(x_k) \\ &+ \frac{\alpha_{k+1}}{L}B^{-1}\tilde{\nabla} F^{(2)}(x_k) - \frac{1}{L}B^{-1}\tilde{\nabla} F^{(2)}(x_{k+1}) - \frac{\alpha_{k+1}}{L}B^{-1}\tilde{\nabla} F^{(2)}(x_{k+1}), \end{aligned}$$

with $\tilde{\nabla} F^{(2)}(x_k)$ the subgradient of $F^{(2)}$ used in the proximal operation generating x_k.

Comparing the different variants of FPGM2 on Fig. 4.2.1 (right plot) leads to the same conclusion as for FPGM1: inertial parameters $\alpha^{(b)}$ perform slightly better than $\alpha^{(a)}$.

In Table 1, we report the different worst-case performance guarantees obtained numerically for FPGM1 (for both sequences) and FPGM2 (for the better secondary sequence only). We consider three situations: $F^{(2)} = 0$ (unconstrained smooth convex minimization), $F^{(2)} \in \mathcal{I}_\infty(\mathbb{E})$ (constrained smooth convex minimization with projected methods) and $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$ (non-smooth composite convex minimization with proximal methods).

Type	$F(y_N) - F_*$ (FPGM1)	$F(x_N) - F_*$ (FPGM1)	$F(x_N) - F_*$ (FPGM2)
Unconstrained ($F^{(2)} = 0$)	$\frac{LR^2}{2}\frac{4}{N^2+5N+6}$	$\frac{LR^2}{2}\frac{4}{N^2+7N+4}$	$\frac{LR^2}{2}\frac{4}{N^2+7N+4}$
Constrained ($F^{(2)} \in \mathcal{I}_\infty$)	$\frac{LR^2}{2}\frac{4}{N^2+5N+2}$	Infeasible	$\frac{LR^2}{2}\frac{4}{N^2+7N}$
Non-smooth ($F^{(2)} \in \mathcal{F}_{0,\infty}$)	$\frac{LR^2}{2}\frac{4}{N^2+5N+2}$	Unbounded	$\frac{LR^2}{2}\frac{4}{N^2+7N}$

Table 1 Worst-case obtained for FPGM1 and FPGM2 with inertial coefficient $\alpha_k = \frac{k-1}{k+2}$ and $N \geq 1$.

All finite convergence results reported in the table actually correspond to specific worst-case functions that we could identify numerically, which means that they provide rigorous lower bounds. After solving the corresponding PEPs numerically (for $L = R = 1$ and $1 \leq N \leq 100$), we conjecture them to be equal to the exact worst-case guarantees.

We observe that the worst-case guarantees for FPGM2 are slightly better than for FPGM1. Guarantees for the unconstrained case are slightly better than those for the constrained and proximal cases, which are equal. Note that the secondary sequence of FPGM1 is not guaranteed to be feasible in the constrained case, and that the corresponding objective value may be unbounded in the proximal case (for any $N \geq 1$).

The worst-case functions identified numerically for the unconstrained case are Huber-shaped functions [43]. In the constrained case, we identified one-dimensional linear optimization problems of the form $\min_{x\geq 0} cx$ as worst-cases, where c is a constant defined by

$$c = \frac{\sqrt{B}R}{2\sum_{j=0}^{N-1} h_{N,j}^{(1)}}$$

revd(following the notations from (FSLFOM) with $t_{N,N} = 1$, and $t_{N,j} = 0$ for $0 \leq j \leq N-1$). Finally, for the proximal case, our worst-case has function $F^{(1)}(x) = cx$ with the same c as above, and function $F^{(2)}(x)$ may be chosen equal to zero for $x \geq 0$ and to sx for $x < 0$, for any negative value of the slope $s < 0$.

4.3 A proximal optimized gradient method

In this section, we consider again the nonsmooth composite convex minimization problem (9). In particular, we are investigate the possibility of obtaining an optimized method that is able to handle this sort of problem (i.e., a method whose worst-case performance the best possible).

Our proposal consists in extending the optimized gradient method (OGM) developed by Kim and Fessler in [22], which is originally tailored for smooth unconstrained minimization ($F^{(2)} = 0$). In the unconstrained smooth minimization setting, this first-order method was recently shown in [12] to have the best achievable worst-case guarantee for the criterion $F_N - F_*$.

The new method we propose, called POGM, has been obtained by combining ideas obtained from the original OGM [22] and the non-standard placement of the proximal operator used for speeding up the convergence of fast proximal gradient methods (FPGM2). It was designed using the same two principles as FPGM2 (see Remark 6): on the one hand, it is equivalent to OGM when applied to smooth unconstrained convex minimization problems, and on the other hand, it remains at an optimal point when it reaches one.

Proximal Optimized Gradient Method (POGM)

```
Input: $F^{(1)} \in \mathcal{F}_{0,L}(\mathbb{E})$, $F^{(2)} \in \mathcal{F}_{0,\infty}(\mathbb{E})$, $x_0 \in \mathbb{E}$, $y_0 = x_0$, $\theta_0 = 1$.
For $k = 1 : N$
    $y_k = x_{k-1} - \frac{1}{L} B^{-1} \nabla F^{(1)}(x_{k-1})$
    $z_k = y_k + \frac{\theta_{k-1} - 1}{\theta_k}(y_k - y_{k-1}) + \frac{\theta_{k-1}}{\theta_k}(y_k - x_{k-1}) + \frac{\theta_{k-1} - 1}{L\gamma_{k-1}\theta_k}(z_{k-1} - x_{k-1})$
    $x_k = \mathrm{prox}_{\gamma_k F^{(2)}}(z_k)$
```

In this algorithm, we use the sequence $\gamma_k = \frac{1}{L}\frac{2\theta_{k-1}+\theta_k-1}{\theta_k}$ and the inertial coefficients proposed in [22]:

$$\theta_k = \begin{cases} \frac{1+\sqrt{4\theta_{k-1}^2+1}}{2}, & i \leq N-1 \\ \frac{1+\sqrt{8\theta_{k-1}^2+1}}{2}, & i = N \end{cases}$$

Simply trying to generalize OGM using the standard proximal step on the primary sequence $\{y_i\}$ (as for FPGM1) does not lead to a converging algorithm. We obtained numerical evidence, i.e. worst-case functions showing that the worst-case bound for this candidate algorithm does not decrease after each iteration (in other words, its worst-case rate is not converging to zero). Therefore we have to introduce the same idea used in FPGM2 concerning the place of the proximal operator.

We compare POGM to FPGM with inertial coefficients $\alpha_k^{(b)}$ on Fig. 2. We obtain worst-case performances about twice better for POGM when compared to both FPGM1 and FPGM2 between 1 and 100 iterations. Also, we observe that the bound for POGM (equivalent to OGM when $F^{(2)} = 0$) is approximately 12% worse than that for OGM [22] in the worst-case.

Of course, POGM suffers from the drawback of requiring the knowledge of the number of iterations in advance (because the rule to compute the last coefficient θ_N differs from the rule to compute all the previous ones). This practical disadvantage is not easily solved: if the last θ_N is updated with the same rule as all the previous coefficients, performance is degraded by a non-negligible factor, rendering it even slower than FPGM (note that this is already the case for smooth unconstrained minimization [21]).

4.4 A conditional gradient method

Consider the constrained smooth convex optimization problem

$$\min_{x \in Q} F(x),$$

with $F \in \mathcal{F}_{0,L}(\mathbb{E})$ and $Q \subset \mathbb{E}$ a bounded and closed convex set. In that setting, different ways exist for treating the constraint set Q. In the previous section, we proposed to use fast gradient methods, which require the

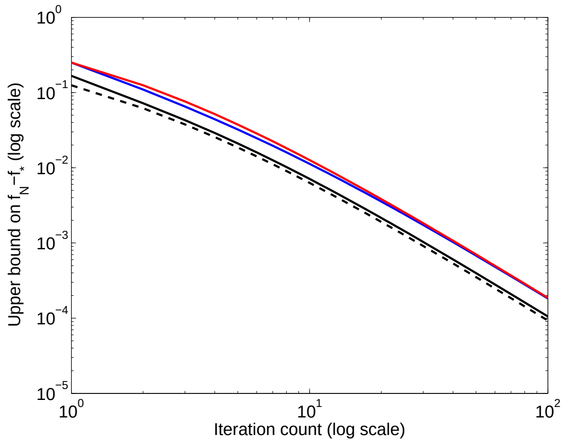


Fig. 2 Comparison between the worst-case performances of FPGM1 (with inertial coefficients $\alpha_k^{(b)}$) (red), FPGM2 (with inertial coefficients $\alpha_k^{(b)}$) (blue), POGM (black) and OGM (dashed, black) for $N \in \{1, \ldots, 100\}$, $L = 1$ and $R = 1$.

ability to project onto the closed convex set Q. In this section, we rather consider the standard conditional gradient method (CGM, also sometimes referred to as the Frank-Wolfe method), which originates from [15]. This algorithm has the advantage of avoiding projections onto Q, and performs instead linear optimization on this set (which is typically easier when Q is a polyhedral set).

```
Conditional Gradient Method (CGM)
    Input: $F \in \mathcal{F}_{0,L}(\mathbb{E})$, closed convex $Q \subset \mathbb{E}$ with $\|x-y\|_{\mathbb{E}} \leq D \ \forall x, y \in Q$, $x_0 \in Q$.
    For $k = 1 : N$
        $y_k = \underset{y \in Q}{\operatorname{argmin}} \{\langle \nabla F(x_{k-1}), y - x_{k-1} \rangle\}$
        $\lambda_k = \frac{2}{1+k}$
        $x_k = (1-\lambda_k) x_{k-1} + \lambda_k y_k$
```

The standard global convergence guarantee for this method (see e.g., [19, Theorem 1]) is

$$F(x_N) - F_* \leq \frac{2LD^2}{N+2},$$

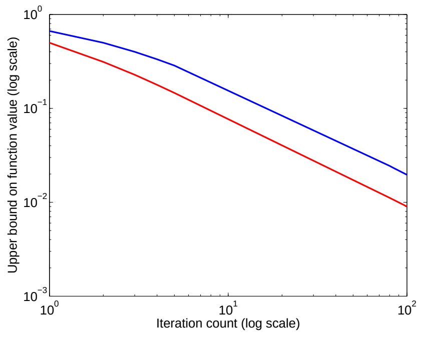
which we compare with the exact bound provided by PEP on Fig. 3(a) (see Section 2.3 which shows that CGM fits into the (FSLFOM) format). The numerical guarantees we obtained by solving the performance estimation problem for up to a hundred iterations are between two and three times better than the standard guarantee.

4.5 Alternate projection and Dykstra methods

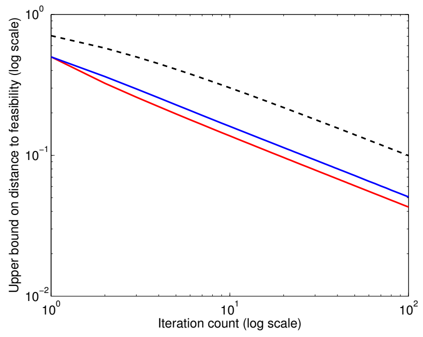
In this section, we numerically investigate the difference between the worst-case behaviors of the standard alternate projection method (APM) for finding a point in the intersection of two convex sets, and the Dykstra [5] method (DAPM) for finding the closest point in the intersection of two convex sets. APM is a particular instance of subgradient-type descent[13] applied to the problem

$$\min_{x \in \mathbb{E}} \{f(x) = \max_i \|x - \Pi_{Q_i}(x)\|_{\mathbb{E}}\}, \tag{11}$$

[13] It can be shown that $\frac{x - \Pi_{Q_k}(x)}{\|x - \Pi_{Q_k}(x)\|}$ is a subgradient of the function $f(x)$ (at x such that $f(x) = \|x - \Pi_{Q_k}(x)\|$). Therefore, in the case of two sets Q_1, Q_2, and assuming that x is feasible for one of the two sets (say, Q_1), a projection onto the other one corresponds to a subgradient step on f with step size $\|x - \Pi_{Q_2}(x)\|$. Hence, APM is an instance of a subgradient method for $k > 1$ (when x_k is feasible for one of the two sets).



(a) Worst-case performance of CGM (red) and its theoretical guarante (4.4) (blue) for $N \in \{1,\dots,100\}$, $L = 1$ and $D = 1$.



(b) Worst-case performance of APM (red), DAPM (blue) and lower bound $\frac{R}{\sqrt{N+1}}$ valid for subgradient methods (dashed, black), for $N \in \{1,\dots,100\}$ and $R = 1$.
.

Fig. 3 Numerical analysis of a conditional gradient method (left) and of two variants of alternate projections algorithms (right).

whose objective function is convex and non-smooth (with Lipschitz constant $M = 1$). Therefore, its expected global convergence rate is $\mathcal{O}(\frac{1}{\sqrt{N}})$ (see [14, Theorem A.1]). We compare below the convergence of both APM and DAPM with the standard lower bound for subgradient schemes $\frac{MR}{\sqrt{N+1}}$ as a reference.

```
Alternate Projection Method (APM)
    Input: $x_0 \in \mathbb{E}$, convex sets $Q_1, Q_2 \subseteq \mathbb{E}$, $\|x_0 - x_*\|_{\mathbb{E}} \leq R$, for some $x_* \in Q_1 \cap Q_2$.
    For $k = 1 : N$
        $x_k = \Pi_{Q_2}(\Pi_{Q_1}(x_{k-1}))$
```

```
Dykstra Alternate Projection Method (DAPM)
    Input: $x_0 \in \mathbb{E}$, convex sets $Q_1, Q_2 \subseteq \mathbb{E}$, $\|x_0 - x_*\|_{\mathbb{E}} \leq R$, for some $x_* \in Q_1 \cap Q_2$.
    Initialize $p_0 = q_0 = 0$.
    For $k = 0 : N - 1$
        $y_k = \Pi_{Q_1}(x_k + p_k)$
        $p_{k+1} = x_k + p_k - y_k$
        $x_{i+1} = \Pi_{Q_2}(y_k + q_k)$
        $q_{k+1} = y_k + q_k - x_{k+1}$
```

The performance measure used is $\min_{x \in Q_1} \|x - x_N\|_{\mathbb{E}} = \|x_N - \Pi_{Q_1}(x_N)\|_{\mathbb{E}}$ (noting that $x_N \in Q_2$ always holds). We do not provide details on the corresponding performance estimation problem here, as it is very similar to the previous sections. The results for APM and DAPM are shown on Fig. 3(b), where the (expected) convergence in $\mathcal{O}(\frac{1}{\sqrt{N}})$ is clearly obtained. Interestingly, DAPM converges slightly slower than APM (more precisely, DAPM has a worst-case about 18% larger than APM), which is therefore more advisable for finding a point in the intersection of two convex sets (in terms of worst-case performance, when no additional structure is assumed). In addition, note that both APM and DAPM have a worst-case which is about twice better than the standard lower bound for explicit non-smooth schemes.

5 Conclusion

In this work, we presented a performance estimation approach to analyze first-order algorithms for composite optimization problems. The results of [43] were largely extended to handle both larger classes of (composite) objective functions and larger classes of first-order algorithms (also in a more general setting for handling pairs of conjugate norms).

Our contribution was essentially threefold: first, we developed specific interpolation conditions for different classes of convex and non-convex functions; then, we exploited those interpolation conditions to formulate the exact worst-case problem for fixed-step linear first-order methods and finally we applied that methodology to provide tight analyses for different first-order methods. Among others, we presented a new analytical guarantee for the proximal point algorithm that is twice better than previously known, and improved the standard worst-case guarantee for the conditional gradient method by more than a factor of two. On the way, we also proposed an extension of the optimized gradient method proposed by Kim and Fessler [22] that incorporates a projection or a proximal operator.

As further research, we believe this methodology should be applied to refine analyses of methods fitting in the context of fixed-step linear first-order methods, and possibly extended to handle dynamic step size rules. To this end, a possibility is to explore convex relaxations of the resulting possibly non-convex performance estimations problems. As an example, we believe it would be interesting to analyze algorithms involving line-search, such as backtracking or Armijo-Wolfe procedures. Moreover, it seems to us that the performance estimation approach could be used to refine the analyses of randomized coordinate descent-type algorithms [31]. Performance estimation problems also opened the door for looking towards optimized methods, as proposed by Kim and Fessler [22] for unconstrained smooth convex minimization. Such an optimized method in the proximal or conditional settings would be of great interest.

Finally, algorithmic analyses using performance estimation problems are intrinsically limited by our ability to solve semidefinite problems, both numerically (when the number of iterations is large) or analytically (to obtain results valid for any number of iterations). Therefore, any idea leading to (convex) programs that are easier to solve while maintaining reasonable guarantees would be very advantageous.

Software. MATLAB implementations of the performance estimation approach for different variants of gradient methods are available online at `http://perso.uclouvain.be/adrien.taylor`.

References

1. Bauschke, H.H., Combettes, P.: Convex analysis and monotone operator theory in Hilbert spaces. Springer (2011)
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences **2**(1), 183–202 (2009)
3. Bertsekas, D.P.: Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. Optimization for Machine Learning pp. 1–38 (2010)
4. Bertsekas, D.P.: Convex Optimization Algorithms. Athena Scientific (2015)
5. Boyle, J.P., Dykstra, R.L.: A method for finding projections onto the intersection of convex sets in hilbert spaces. In: Advances in order restricted statistical inference, pp. 28–47. Springer (1986)
6. Burer, S., Monteiro, R.: A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. Mathematical Programming **95**(2), 329–357 (2003)
7. Combettes, P.L., Pesquet, J.C.: Proximal splitting methods in signal processing. In: Fixed-point algorithms for inverse problems in science and engineering, pp. 185–212. Springer (2011)
8. d'Aspremont, A.: Smooth optimization with approximate gradient. SIAM Journal on Optimization **19**(3), 1171–1183 (2008)
9. Devolder, O., Glineur, F., Nesterov, Y.: Double smoothing technique for large-scale linearly constrained convex optimization. SIAM Journal on Optimization **22**(2), 702–727 (2012)
10. Devolder, O., Glineur, F., Nesterov, Y.: First-order methods of smooth convex optimization with inexact oracle. Mathematical Programming **146**(1-2), 37–75 (2014)
11. Drori, Y.: Contributions to the complexity analysis of optimization algorithms. Ph.D. thesis, Tel-Aviv University (2014)
12. Drori, Y.: The exact information-based complexity of smooth convex minimization. Journal of Complexity (2016)
13. Drori, Y., Teboulle, M.: Performance of first-order methods for smooth convex minimization: a novel approach. Mathematical Programming **145**(1-2), 451–482 (2014)
14. Drori, Y., Teboulle, M.: An optimal variant of kelleys cutting-plane method. Mathematical Programming **160**(1-2), 321–351 (2016)
15. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval research logistics quarterly **3**(1-2), 95–110 (1956)
16. Goemans, M., Williamson, D.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM (JACM) **42**(6), 1115–1145 (1995)
17. Güler, O.: On the convergence of the proximal point algorithm for convex minimization. SIAM Journal on Control and Optimization **29**(2), 403–419 (1991)
18. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms. Springer Verlag, Heidelberg (1996). Two volumes - 2nd printing

19. Jaggi, M.: Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In: Proceedings of the 30th International Conference on Machine Learning (ICML-13), pp. 427–435 (2013)
20. Journée, M., Bach, F., Absil, P.A., Sepulchre, R.: Low-rank optimization on the cone of positive semidefinite matrices. SIAM Journal on Optimization **20**(5), 2327–2351 (2010)
21. Kim, D., Fessler, J.A.: On the convergence analysis of the optimized gradient method. Journal of Optimization Theory and Applications pp. 1–19 (2016)
22. Kim, D., Fessler, J.A.: Optimized first-order methods for smooth convex minimization. Mathematical Programming **159**(1-2), 81–107 (2016)
23. Lessard, L., Recht, B., Packard, A.: Analysis and design of optimization algorithms via integral quadratic constraints. SIAM Journal on Optimization **26**(1), 57–95 (2016)
24. Löfberg, J.: YALMIP : A toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD Conference (2004)
25. Moreau, J.J.: Proximité et dualité dans un espace hilbertien. Bulletin de la Société mathématique de France **93**, 273–299 (1965)
26. Mosek, A.: The MOSEK optimization software. Online at http://www.mosek.com **54** (2010)
27. Nemirovsky, A., Yudin, D.: Problem complexity and method efficiency in optimization. Willey-Interscience, New York (1983)
28. Nesterov, Y.: A method of solving a convex programming problem with convergence rate O($1/k^2$)). Soviet Mathematics Doklady **27**, 372–376 (1983)
29. Nesterov, Y.: Introductory lectures on convex optimization : a basic course. Applied optimization. Kluwer Academic Publ. (2004)
30. Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical programming **103**(1), 127–152 (2005)
31. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM Journal on Optimization **22**(2), 341–362 (2012)
32. Nesterov, Y.: Gradient methods for minimizing composite functions. Mathematical Programming **140**(1), 125–161 (2013)
33. Orsi, R., Helmke, U., Moore, J.B.: A newton-like method for solving rank constrained linear matrix inequalities. Automatica **42**(11), 1875–1882 (2006)
34. Pardalos, P.M., Vavasis, S.A.: Quadratic programming with one negative eigenvalue is np-hard. Journal of Global Optimization **1**(1), 15–22 (1991)
35. Parikh, N., Boyd, S.: Proximal algorithms. Foundations and Trends in optimization **1**(3), 123–231 (2013)
36. Polyak, B.T.: Introduction to optimization. Optimization Software New York (1987)
37. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. SIAM journal on control and optimization **14**(5), 877–898 (1976)
38. Rockafellar, R.T.: Convex Analysis. Princeton University Press (1996)
39. Rockafellar, R.T., Wets, R.B.: Variational Analysis. Springer (1998)
40. Sahni, S.: Computationally related problems. SIAM Journal on Computing **3**(4), 262–279 (1974)
41. Sturm, J.F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optimization Methods and Software **11–12**, 625–653 (1999)
42. Su, W., Boyd, S., Candes, E.: A differential equation for modeling nesterovs accelerated gradient method: Theory and insights. In: Advances in Neural Information Processing Systems, pp. 2510–2518 (2014)
43. Taylor, A.B., Hendrickx, J.M., Glineur, F.: Smooth strongly convex interpolation and exact worst-case performance of first-order methods. Mathematical Programming **161**(1-2), 307–345 (2017)
44. Tseng, P.: On accelerated proximal gradient methods for convex-concave optimization. Submitted to SIAM Journal on Optimization (2008)
45. Vandenberghe, L., Boyd, S.: Semidefinite programming. SIAM Review **38**, 49–95 (1994)

A Proof of upper bound in Theorem 7

In order to express the corresponding PEP in the simplest form, we heavily rely on some straightforward simplifications of (SDP-PEP) (see Corollary 1 and Remark 2.4). Let us denote by P_N the matrix containing the information harvested after N iterations: $P_N = [g_1\ g_2\ \dots\ g_N\ Bx_0]$ (we use the notation g_i for subgradients $g_i \in \partial F(x_i)$), and by G_N its corresponding Gram matrix (see Section 2.1). Also, we introduce the step size vectors m_k that express each iterate x_k in terms of x_0 and the subgradients $\{g_i\}_{1\leq i\leq N}$, that is $x_k = P_N m_k$ $(k = 0, \dots, N)$. Using the standard notation e_i for the unit vector having a single 1 as its i^{th} component, this results in the following explicit expressions for m_k: $m_k = e_{N+1} - \sum_{i=1}^{k} \alpha_i e_i$, along with $m_0 = e_{N+1}$ and $m_* = 0$ (where we assumed without loss of generality that $x_* = 0$).

In order to perform the wort-case analysis for PPA, we now formulate the performance estimation problem (f-PEP) as the following SDP, the simplified version of (SDP-PEP) where the x_k's $(k = 1, \dots, N)$ have been substituted using the equation defining the iterates $x_k = x_{k-1} - \alpha_k B^{-1} g_k$:

$$\max_{G_N\in\mathbb{S}^{N+1}, f_1,\dots,f_N,f_*\in\mathbb{R}^N} f_N - f_*, \text{ s.t. } f_j - f_i + \operatorname{Tr}(A_{ij}G_N) \leq 0, \quad i,j \in \{1,\dots,N,*\} \qquad \text{(PPA-PEP)}$$

$$\|x_0 - x_*\|_{\mathbb{E}}^2 \leq R^2,$$

$$G_N \succeq 0,$$

with matrices $2A_{ij} = e_j(m_i - m_j)^\top + (m_i - m_j)e_j^\top$ (where $e_* = 0$) coming from the non-smooth convex interpolation inequalities (see Condition (5)). In order to obtain an analytical upper bound for PPA, we consider the Lagrangian dual to (PPA-PEP), which is given by the following:

$$\min_{\lambda_{ij}\geq 0, \tau\geq 0} \tau R^2 \text{ s.t. } e_N - \sum_i \sum_{j\neq i} (\lambda_{ij} - \lambda_{ji})e_j = 0, \qquad \text{(PPA-dPEP)}$$
$$\sum_i \sum_{j\neq i} \lambda_{ij} A_{ij} + \tau m_0 m_0^\top \succeq 0$$

(where the constraint corresponding to f_* can be discarded since it is clear that letting $f_* = 0$ does not change the optimal solution of (PPA-PEP)). Note that the set of equality constraints can be assimilated to a set of *flow* constraints on a complete directed graph. That is, considering a graph where the optimum and each iterate correspond to nodes, each nonnegative λ_{ij} corresponds to the flow on the edge going from node j to node i (we choose this direction by convention). This flow constraint imposes that the outgoing flow equals the ingoing flow for every node, except at the node for final iterate N where the outgoing flow should be equal to 1, and at the optimum node, where the incoming flow should be equal to 1. We show that the following choice is a feasible point of the dual (PPA-dPEP).

$$\lambda_{i,i+1} = \frac{\sum_{k=1}^{i} \alpha_k}{2\sum_{k=1}^{N} \alpha_k - \sum_{k=1}^{i} \alpha_k}, \qquad i \in \{1,\ldots,N-1\}$$
$$\lambda_{*,i} = \frac{2\alpha_i \sum_{k=1}^{N} \alpha_k}{\left(2\sum_{k=1}^{N} \alpha_k - \sum_{k=1}^{i} \alpha_k\right)\left(2\sum_{k=1}^{N} \alpha_k - \sum_{k=1}^{i-1} \alpha_k\right)}, \qquad i \in \{1,\ldots,N\}$$
$$\tau = \frac{1}{4\sum_{k=1}^{N} \alpha_k},$$

and $\lambda_{ij} = 0$ otherwise. First, we clearly have $\lambda_{ij} \geq 0$ and some basic computations allow to verify that the equality constraints from (PPA-dPEP) are satisfied:

$$\lambda_{*,1} - \lambda_{1,2} = 0,\ \lambda_{*,i} + \lambda_{i-1,i} - \lambda_{i,i+1} = 0\ (i \in \{2,\ldots,N-1\}),\ \lambda_{*,N} + \lambda_{N-1,N} = 1.$$

It remains to show that the corresponding dual matrix S is positive semidefinite.

$$2S = \sum_{i=1}^{N-1} 2\alpha_{i+1}\lambda_{i,i+1} e_{i+1}e_{i+1}^\top + 2\tau e_{N+1}e_{N+1}^\top$$
$$+ \sum_{i=1}^{N} \lambda_{*,i}\left[e_i\left(-e_{N+1} + \sum_{k=1}^{i} \alpha_k e_k\right)^\top + \left(-e_{N+1} + \sum_{k=1}^{i} \alpha_k e_k\right)e_i^\top\right].$$

In order to reduce the number of indices to be used, we will note $\lambda_i = \lambda_{i,i+1}$ and $\mu_i = \lambda_{*,i}$. Then, using the equality constraints, we arrive at the following dual matrix:

$$2S = \begin{pmatrix} 2\alpha_1\lambda_1 & \alpha_1\mu_2 & \alpha_1\mu_3 & \ldots & \alpha_1\mu_{N-1} & \alpha_1\mu_N & -\mu_1 \\ \alpha_1\mu_2 & 2\alpha_2\lambda_2 & \alpha_2\mu_3 & \ldots & \alpha_2\mu_{N-1} & \alpha_2\mu_N & -\mu_2 \\ \alpha_1\mu_3 & \alpha_2\mu_3 & 2\alpha_3\lambda_3 & \ldots & \alpha_3\mu_{N-1} & \alpha_3\mu_N & -\mu_3 \\ \vdots & & \ddots & \ddots & & \vdots & \vdots \\ \alpha_1\mu_{N-1} & \alpha_2\mu_{N-1} & \alpha_3\mu_{N-1} & \ldots & 2\alpha_{N-1}\lambda_{N-1} & \alpha_{N-1}\mu_N & -\mu_{N-1} \\ \alpha_1\mu_N & \alpha_2\mu_N & \alpha_3\mu_N & \ldots & \alpha_{N-1}\mu_N & 2\alpha_N & -\mu_N \\ -\mu_1 & -\mu_2 & -\mu_3 & \ldots & -\mu_{N-1} & -\mu_N & 2\tau \end{pmatrix}.$$

In order to prove $S \succeq 0$, we first use a Schur complement and then show that the resulting matrix is diagonally dominant with positive diagonal elements. After taking the Schur complement (with respect to the lower right scalar component 2τ), we obtain the matrix $\tilde{S}$:

$$\tilde{S} = \begin{pmatrix} 2\alpha_1\lambda_1 & \alpha_1\mu_2 & \alpha_1\mu_3 & \ldots & \alpha_1\mu_{N-1} & \alpha_1\mu_N \\ \alpha_1\mu_2 & 2\alpha_2\lambda_2 & \alpha_2\mu_3 & \ldots & \alpha_2\mu_{N-1} & \alpha_2\mu_N \\ \alpha_1\mu_3 & \alpha_2\mu_3 & 2\alpha_3\lambda_3 & \ldots & \alpha_3\mu_{N-1} & \alpha_3\mu_N \\ \vdots & & \ddots & \ddots & & \vdots \\ \alpha_1\mu_{N-1} & \alpha_2\mu_{N-1} & \alpha_3\mu_{N-1} & \ldots & 2\alpha_{N-1}\lambda_{N-1} & \alpha_{N-1}\mu_N \\ \alpha_1\mu_N & \alpha_2\mu_N & \alpha_3\mu_N & \ldots & \alpha_{N-1}\mu_N & 2\alpha_N \end{pmatrix} - \frac{1}{2\tau}\begin{pmatrix}\mu_1\\ \mu_2\\ \vdots\\ \mu_N\end{pmatrix}\begin{pmatrix}\mu_1\\ \mu_2\\ \vdots\\ \mu_N\end{pmatrix}^\top.$$

The first step to show the diagonally dominant character of $\tilde{S}$ is to note that every non-diagonal element of $\tilde{S}$ is non-positive: $\alpha_j\mu_i - \frac{\mu_i\mu_j}{2\tau} \leq 0,\ \forall i \neq j$. Indeed, this is equivalent to write this in the following form ($\mu_i > 0$):

$$\alpha_j - \frac{\mu_j}{2\tau} = \alpha_j\left(\frac{\left(2\sum_{k=1}^{N}\alpha_k - \sum_{k=1}^{i}\alpha_k\right)\left(2\sum_{k=1}^{N}\alpha_k - \sum_{k=1}^{i-1}\alpha_k\right) - \left(2\sum_{k=1}^{N}\alpha_k\right)^2}{\left(2\sum_{k=1}^{N}\alpha_k - \sum_{k=1}^{i}\alpha_k\right)\left(2\sum_{k=1}^{N}\alpha_k - \sum_{k=1}^{i-1}\alpha_k\right)}\right) \leq 0,$$

since $\alpha_k \geq 0$ by assumption. This allows to discard the absolute values in the diagonal dominance criteria. Then, using the equality constraints, we obtain an expression for the sum of all non-diagonal elements of line i of $\tilde{S}$:

$$\mu_i \sum_{j=1}^{i-1} \alpha_j + \alpha_i \sum_{j=i+1}^{N} \mu_j - \frac{\mu_i}{2\tau} \sum_{j \neq i} \mu_j$$
$$= \begin{cases} \mu_i \sum_{j=1}^{i-1} \alpha_j + \alpha_i (1-\lambda_i) - \frac{1}{2\tau}\mu_i(1-\mu_i), & \text{if } i < N \\ \mu_N \sum_{j=1}^{N-1} \alpha_j - \frac{1}{2\tau}\mu_N(1-\mu_N) & \text{if } i = N \end{cases}$$

Using the values of μ_i, λ_i and τ along with elementary computations allows to verify that $\forall i \in \{1, \ldots, N\}$:

$$\begin{cases} -(\mu_i \sum_{j=1}^{i-1} \alpha_j + \alpha_i(1-\lambda_i) - \frac{1}{2\tau}\mu_i(1-\mu_i)) = 2\alpha_i\lambda_i - \frac{\mu_i^2}{2\tau} & \text{if } i = 1, \ldots, N-1, \\ -(\mu_i \sum_{j=1}^{i-1} \alpha_j - \frac{1}{2\tau}\mu_i(1-\mu_i)) = 2\alpha_i - \frac{\mu_i^2}{2\tau} & \text{if } i = N, \end{cases}$$

which implies diagonal dominance of $\tilde{S}$ (even more: the sum of the elements of each line equals 0). $\square$