Worst-case analyses for first-order optimization methods

Adrien Taylor[*], Baptiste Goujaud[†]

Current version: July 20, 2022

1 Introduction **2**

2 Getting familiar with base performance estimation problems **2**

3 Further exercises **12**

4 Slightly more advanced techniques **19**

5 Background material and useful facts **20**

5.1 Standard definitions 20

5.2 Interpolation/extension theorems 20

Foreword & Acknowledgements

This document provides a series of exercises for getting familiar with "performance estimation problems" and the use of semidefinite programming for analyzing worst-case behaviors of optimization methods. An informal introduction can be found in this blog post. Exercises summarizing the main ingredients of the approach are provided in Section 2 (Exercise 1 focuses on the primal PEP formulation—i.e., on finding worst-case scenarios—, whereas Exercise 2 focuses on the dual formulation—i.e., on finding rigorous worst-case guarantees). Section 3 contains exercises for going further. Background material that might be used for the exercises is provided in Section 5.

Those notes were written for accompanying the TraDE-OPT workshop on algorithmic and continuous optimization. If you have any comment, remark, or if you found a typo/mistake, please don't hesitate to feedback the authors!

Funding. A. Taylor acknowledges support from the European Research Council (grant SEQUOIA 724063). This work was partly funded by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). The work of B. Goujaud is partially supported by ANR-19-CHIA-0002-01/chaire SCAI, and Hi!Paris.

[*]INRIA, SIERRA project-team, and D.I. Ecole normale supérieure, Paris, France. Email: adrien.taylor@inria.fr

[†]CMAP, École Polytechnique, Institut Polytechnique de Paris, France. Email: baptiste.goujaud@gmail.com

1 Introduction

In short, considering problems of the form

$$\min_{x\in\mathbb{R}^d} F(x),$$

where we denote an optimal solution by $x_\star \in \operatorname{argmin}_{x\in\mathbb{R}^d} F(x)$, our goal here is to assess "a priori" the quality of the output of some "black-box" iterative algorithm, whose iterates are denoted by $x_0, x_1, \ldots, x_N$. There are typically different ways of doing so, which might or might not be relevent depending on the target applications of a particular optimization method. In first-order optimization, we often want to upper bound the quality of an iterate x_k in one of the following terms (which we all ideally would like to be as small as possible, and decreasing functions of $k \in \mathbb{N}$): $\|x_k - x_\star\|^2$, $\|\nabla f(x_k)\|^2$, or $f(x_k) - f(x_\star)$. There are of course other possibilities, including combinations of them (see examples below).

So, our goal is to assess the quality of x_k by providing hopefully meaningfull upper bounds on at least one such quantity. For doing so, we consider classes of problems (i.e., sets of assumptions on F), and perform worst-case analyses (i.e., we want the bound to be valid for all F satisfying the set of assumptions at hand). The following exercises try to shed a bit of light on this topic, by examplifying a principled approach to construct such worst-case convergence bounds using the so-called "performance estimation framework", introduced by Drori and Teboulle in [1]. This document presents the performance estimation problem using the formalism from Taylor, Hendrickx, and Glineur [2, 3].

Notations. We denote by $\langle \cdot, \cdot \rangle : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ the standard Euclidean inner product, and by $\|\cdot\|^2 : \mathbb{R}^d \to \mathbb{R}$ the standard Euclidean norm, that is, the induced norm: for any $x \in \mathbb{R}^d$: $\|x\|^2 = \langle x, x \rangle$ (the results and techniques below apply more broadly (see, e.g., [2, 4]), but it is not needed for understanding the exercises below). Other notations are defined throughout the text.

Packages The numerical parts of the exercises were mostly done for using Python (or alternatively Matlab), but the reader can use other languages if he is proficient with semidefinite programming with it (although there is currently no interface for easing the access to performance estimation beyond Python and Matlab). A few exercises were designed for using the PEPit package [5] (alternatively, for Matlab, PESTO [6]). For users planning to keep using performance estimation, we advise to install a good SDP solver such as MOSEK [7]. In any case, the Python package relies on CVXPY [8] (see PEPit installation) and the Matlab one relies on YALMIP [9] (see PESTO installation). Some exercises might be simpler to approach using some symbolic computations, such as Sympy [10] (which is used in some of the Python notebooks).

Getting corrected exercises? Just recompile this latex file after setting the second line of the document to `\def\includeCorrections{1}` (`\def\includeCorrections{0}` to remove corrections).

2 Getting familiar with base performance estimation problems

In this section, we introduce the main base ingredients underlying the performance estimation technique. Those ingredients are all examplified for the analysis of gradient descent.

The goal of this first exercise is to (i) get familiar with the concept of performance estimation problem, that is, how can we cast, and solve, the problem of looking for worst-case scenarios in the context of first-order optimization; and (ii) get an idea on the applicability of the methodology for standard settings.

Exercise 1 (Gradient method—"primal performance estimation") *For this exercise, consider the problem of "black-box" minimization of a smooth strongly convex function:*

$$f_\star \triangleq \min_{x\in\mathbb{R}^d} f(x), \tag{1}$$

where f is L-smooth and μ-strongly convex (see Definition 2), and where $x_\star \triangleq \operatorname{argmin}_x f(x)$ and $f_\star \triangleq f(x_\star)$ its optimal value. For minimizing (1) we use gradient descent with a pre-determined sequence of stepsizes $\{\gamma_k\}_k$; that is, we iterate $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$. The goal of this exercise is to compute $\tau(\mu, L, \gamma_k)$, a.k.a. a convergence rate, the smallest value such that the inequality

$$\|x_{k+1} - x_\star\|^2 \leqslant \tau(\mu, L, \gamma_k)\|x_k - x_\star\|^2 \tag{2}$$

is valid for any $d \in \mathbb{N}$, for any L-smooth μ-strongly convex function f (notation $f \in \mathcal{F}_{\mu,L}$) and for all $x_k, x_{k+1} \in \mathbb{R}^d$ such that $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$, and any $x_\star \in \operatorname{argmin}_x f(x)$.

1. *Assuming $x_\star \neq x_k$ (without loss of generality), show that*

$$\begin{aligned}
\tau(\mu, L, \gamma_k) = \sup_{\substack{d,f\\ x_k, x_{k+1}, x_\star}} \quad & \frac{\|x_{k+1} - x_\star\|^2}{\|x_k - x_\star\|^2} \\
\text{s.t.} \quad & f \in \mathcal{F}_{\mu,L} \\
& x_{k+1} = x_k - \gamma_k \nabla f(x_k) \\
& \nabla f(x_\star) = 0,
\end{aligned} \tag{3}$$

where f, x_k, x_{k+1}, $x_\star$, and d are the variables of the optimization problem.

Correction: We trivially have (2) $\Leftrightarrow \frac{\|x_{k+1}-x_\star\|^2}{\|x_k - x_\star\|^2} \leqslant \tau(\mu, L, \gamma_k)$ for any $d \in \mathbb{N}$, for any $f \in \mathcal{F}_{\mu,L}$, and for all $x_k, x_{k+1} \in \mathbb{R}^d$ such that $x_{k+1} = x_k - \gamma_k \nabla f(x_k)$, and $x_\star \in \operatorname{argmin}_x f(x)$ (with $x_\star \neq x_k$). It trivially follows that $\tau(\mu, L, \gamma_k)$ upper bounds the RHS of (3). By definition of (2), this is even the smallest one, hence the equality.

2. *Show that*

$$\begin{aligned}
\tau(\mu, L, \gamma_k) = \sup_{\substack{d\\ x_k, x_{k+1}, x_\star\\ g_k, g_\star\\ f_k, f_\star}} \quad & \frac{\|x_{k+1} - x_\star\|^2}{\|x_k - x_\star\|^2} \\
\text{s.t.} \quad & \exists f \in \mathcal{F}_{\mu,L} \text{ such that } \begin{cases} f_i = f(x_i) & i = k, \star \\ g_i = \nabla f(x_i) & i = k, \star \end{cases} \\
& x_{k+1} = x_k - \gamma_k g_k \\
& g_\star = 0.
\end{aligned} \tag{4}$$

Correction: This operation does not change the optimal value of the problems. From any feasible point to (3) one can create a feasible point to (4) with the same objective value, and vice-versa.

3. *Using Theorem 2, show that $\tau(\mu, L, \gamma_k)$ is also equal to*

$$\begin{aligned}
\sup_{\substack{d\\ x_k, x_{k+1}, x_\star\\ g_k, g_\star\\ f_k, f_\star}} \quad & \frac{\|x_{k+1} - x_\star\|^2}{\|x_k - x_\star\|^2} \\
\text{s.t.} \quad & f_\star \geqslant f_k + \langle g_k, x_\star - x_k\rangle + \tfrac{1}{2L}\|g_\star - g_k\|^2 + \tfrac{\mu}{2(1-\mu/L)}\left\|x_\star - x_k - \tfrac{1}{L}(g_\star - g_k)\right\|^2 \\
& f_k \geqslant f_\star + \langle g_\star, x_k - x_\star\rangle + \tfrac{1}{2L}\|g_k - g_\star\|^2 + \tfrac{\mu}{2(1-\mu/L)}\left\|x_k - x_\star - \tfrac{1}{L}(g_k - g_\star)\right\|^2 \\
& x_{k+1} = x_k - \gamma_k g_k \\
& g_\star = 0.
\end{aligned} \tag{5}$$

Correction: From Theorem 2, there exists an L-smooth μ-strongly convex function satisfying $g_k = \nabla f(x_k)$, $g_\star = \nabla f(x_\star)$, $f_k = f(x_k)$ and $f_\star = f(x_\star)$ if and only if those inequalities are satisfied. Hence any feasible point to (5) can be converted to a feasible point to (4) and vice-versa (Theorem 2 provides necessary and sufficient conditions).

4. *Using the change of variables*

$$G \triangleq \begin{bmatrix} \|x_k - x_\star\|^2 & \langle g_k, x_k - x_\star\rangle \\ \langle g_k, x_k - x_\star\rangle & \|g_k\|^2 \end{bmatrix}, \quad F \triangleq f_k - f_\star, \tag{6}$$

(note that $G = [x_k - x_\star \quad g_k]^\top [x_k - x_\star \quad g_k] \succcurlyeq 0$*), show that* $\tau(\mu, L, \gamma_k)$ *can be computed as*

$$\begin{aligned} \sup_{G,\,F} \quad & \frac{G_{1,1} + \gamma_k^2 G_{2,2} - 2\gamma_k G_{1,2}}{G_{1,1}} \\ \text{s.t.} \quad & F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{L}{L-\mu} G_{1,2} \leqslant 0 \\ & -F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{\mu}{L-\mu} G_{1,2} \leqslant 0 \\ & G \succcurlyeq 0. \end{aligned} \tag{7}$$

Hint: you can use the following fact ("Cholesky factorization"). Let $X \in \mathbb{S}^n$ *(symmetric matrix of size* $n \times n$*), we have:*

$$X \succcurlyeq 0 \text{ with } \operatorname{rank}(X) \leqslant d \Leftrightarrow \exists P \in \mathbb{R}^{d \times n} \text{ such that } X = P^\top P.$$

Correction: We show that any feasible point to (5) can be converted to a feasible point of (7) with the same optimal value, and vice-versa.

First, by defining $P \triangleq [x_k - x_\star \quad g_k] \in \mathbb{R}^{2\times d}$ *we have* $G \triangleq P^\top P$ *with* $\operatorname{rank}(G) \leq d$ *by construction. Further, by substituting* $x_{k+1} = x_k - \gamma_k g_k$ *in the two inequalities, we obtain that they can both be expressed linearly in terms of* G *and* F*. Similarly, the objective function can be expressed using the elements of* G*. Therefore, any feasible point to (5) can be translated to a feasible point of*

$$\begin{aligned} \sup_{d,\,G,\,F} \quad & \frac{G_{1,1} + \gamma_k^2 G_{2,2} - 2\gamma_k G_{1,2}}{G_{1,1}} \\ \text{s.t.} \quad & F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{L}{L-\mu} G_{1,2} \leqslant 0 \\ & -F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{\mu}{L-\mu} G_{1,2} \leqslant 0 \\ & G \succcurlyeq 0 \\ & \operatorname{rank}(G) \leqslant d, \end{aligned}$$

with the same optimal value. Obviously, any feasible point to the last problem is feasible (with the same objective value) for (7), and the optimal value of (7) is therefore at least equal to the optimal value of (5) (any feasible point to (5) can be translated to a feasible point of (7) with the same objective value).

In the other direction, from any feasible point (G, F) *of (7):* $G \succcurlyeq 0$ *with* $\operatorname{rank}(G) \leq d$ *(for some* $d \in \mathbb{N}$*, with* $d \leqslant 2$ *due to the fact* $G \in \mathbb{S}^2$*), one can obtain some* $P \in \mathbb{R}^{d\times 2}$ *with* $G = P^\top P$ *(note:* P *is usually not unique). Choosing* $x_\star = 0 \in \mathbb{R}^d$*,* $x_k = P_{:,1}$ *(first column of* P*) and* $g_k = P_{:,2}$*, as well as* $x_{k+1} = x_k - \gamma_k g_k$*,* $f_\star = 0$ *and* $f_k = F$*, we obtain a feasible point to (5) (the inequalities of (5) in terms of* (P, F) *are exactly those of (7) expressed in terms of* (G, F)*) with the same objective value.*

5. *Show that* $\tau(\mu, L, \gamma_k)$ *can also be computed as a semidefinite program (SDP):*

$$\begin{aligned} \sup_{G,\,F} \quad & G_{1,1} + \gamma_k^2 G_{2,2} - 2\gamma_k G_{1,2} \\ \text{s.t.} \quad & F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{L}{L-\mu} G_{1,2} \leqslant 0 \\ & -F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{\mu}{L-\mu} G_{1,2} \leqslant 0 \\ & G_{1,1} = 1 \\ & G \succcurlyeq 0 \end{aligned} \tag{8}$$

(note that at this stage, it is actually simpler to show that the supremum is attained and that we can write max *instead of* sup.*)*

Correction: This statement relies on an homogeneity argument: remark that for any $\alpha > 0$ and any feasible point (G, F) for (7) the point $(\alpha G, \alpha F)$ is also feasible for (7) without affecting the objective value. Thereby, for any feasible point (G, F) for which $G_{1,1} \neq 0$ one can normalize the pair by picking $\alpha = \frac{1}{G_{1,1}}$, which is equivalent to restrict ourselves to solutions satisfying $G_{1,1} = 1$. Since the optimum is attained on points for which $G_{1,1} \neq 0$, we reach the desired conclusion.

6. *Let $L \geqslant \mu > 0$ and define $h_k \triangleq \gamma_k L$ and $\kappa \triangleq L/\mu$. Show that $\tau(\mu, L, \gamma_k) = \tau(1/\kappa, 1, h_k)$ (in other words: we can study the case $L = 1$ only and deduce the dependence of τ on L afterwards).*

   Correction: One can rewrite the problem in terms of h_k:

$$\begin{aligned} \sup_{G,F} \quad & G_{1,1} + h_k^2 L^2 G_{2,2} - 2h_k L G_{1,2} \\ s.t. \quad & F + \tfrac{L}{2(\kappa-1)} G_{1,1} + \tfrac{1}{2\mu(\kappa-1)} G_{2,2} - \tfrac{\kappa}{\kappa-1} G_{1,2} \leqslant 0 \\ & -F + \tfrac{L}{2(\kappa-1)} G_{1,1} + \tfrac{1}{2\mu(\kappa-1)} G_{2,2} - \tfrac{1}{\kappa-1} G_{1,2} \leqslant 0 \\ & G_{1,1} = 1 \\ & G \succcurlyeq 0, \end{aligned}$$

   rewritting the problem in terms of $\tilde{G}$ and $\tilde{F}$ as

$$\tilde{G} \triangleq \begin{bmatrix} \|x_k - x_\star\|^2 & \langle g_k, x_k - x_\star\rangle / L \\ \langle g_k, x_k - x_\star\rangle / L & \|g_k\|^2 / L^2 \end{bmatrix}, \quad \tilde{F} \triangleq (f_k - f_\star)/L,$$

   i.e., $\tilde{G} \triangleq [x_k - x_\star \quad g_k/L]^\top [x_k - x_\star \quad g_k/L] \succcurlyeq 0$); we arrive to

$$\begin{aligned} \sup_{\tilde{G},\tilde{F}} \quad & \tilde{G}_{1,1} + h_k^2 \tilde{G}_{2,2} - 2h_k \tilde{G}_{1,2} \\ s.t. \quad & \tilde{F} + \tfrac{1}{2(\kappa-1)} \tilde{G}_{1,1} + \tfrac{\kappa}{2(\kappa-1)} \tilde{G}_{2,2} - \tfrac{\kappa}{\kappa-1} \tilde{G}_{1,2} \leqslant 0 \\ & -\tilde{F} + \tfrac{1}{2(\kappa-1)} \tilde{G}_{1,1} + \tfrac{\kappa}{2(\kappa-1)} \tilde{G}_{2,2} - \tfrac{1}{\kappa-1} \tilde{G}_{1,2} \leqslant 0 \\ & G_{1,1} = 1 \\ & \tilde{G} \succcurlyeq 0, \end{aligned}$$

   where the two inequalities were obtained by dividing the two corresponding inequalities of the previous formulation by L. This last formulation is a function of h_k and κ only, and the conclusion easily follows.

   Note that there are other ways (not relying on the sampled version of the problem) to arrive to this conclusion.

7. *Complete the PEPit code (alternative in Matlab: PESTO code) for computing $\tau(\mu, L, \gamma_k)$ and compute its value for a few numerical values of μ and γ_k.*

   Correction: See notebook.

8. *Set $L = 1$ and compute the optimal value of γ_k numerically for a few values of μ. Similarly, compute the range of γ_k for which the ratios is smaller than 1 in the worst-case.*

   Correction: The optimal value for γ_k should match $2/(L + \mu)$. The range of acceptable stepsizes for convergence is $\gamma_k \in (0, 2/L)$.

9. *Update your code for computing (numerically) the worst-case ratio $\frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2}$ as a function of L, μ, and γ_k. For what values of γ_k do you observe that this ratio is smaller than 1? How would you update the SDP formulation for taking this change into account?*

Correction: The optimal stepsize should still match $2/(L+\mu)$ and the acceptable stepsize range is still $\gamma_k \in (0, 2/L)$. The code needs to be updated by changing the performance measure and initial condition.

There are several natural ways to adapt the problem to the ratios of consecutive gradients, as follows: first, (3) should simply be updated by changing its objective. Second, (4) can be updated to incorporate the gradient of g_{k+1} (which is in the objective), perhaps naturally leading to

$$\begin{aligned} \sup_{\substack{d \\ x_k, x_{k+1}, x_\star \\ g_k, g_{k+1}, g_\star \\ f_k, f_{k+1}, f_\star}} \quad & \|g_{k+1}\|^2 \\ \text{s.t. } \exists f \in \mathcal{F}_{\mu,L} \text{ such that } & \begin{cases} f_i = f(x_i) & i = k, k+1, \star \\ g_i = \nabla f(x_i) & i = k, k+1, \star \end{cases} \\ & x_{k+1} = x_k - \gamma_k g_k \\ & \|g_k\|^2 = 1 \\ & g_\star = 0. \end{aligned}$$

However, this formulation naturally translates to a 3×3 SDP with

$$G \triangleq \begin{bmatrix} \|x_k - x_\star\|^2 & \langle g_k, x_k - x_\star \rangle & \langle g_{k+1}, x_k - x_\star \rangle \\ \langle x_k - x_\star, g_k \rangle & \|g_k\|^2 & \langle g_{k+1}, g_k \rangle \\ \langle x_k - x_\star, g_{k+1} \rangle & \langle g_k, g_{k+1} \rangle & \|g_{k+1}\|^2 \end{bmatrix}, \quad F \triangleq [f_k - f_\star \quad f_{k+1} - f_\star].$$

along with 6 interpolation inequalities (two inequalities per pair of points in the discrete representation). A simpler representation is as follows using only two points (no need to incorporate $x_\star$)

$$\begin{aligned} \sup_{\substack{d \\ x_k, x_{k+1} \\ g_k, g_{k+1} \\ f_k, f_{k+1}}} \quad & \|g_{k+1}\|^2 \\ \text{s.t. } \exists f \in \mathcal{F}_{\mu,L} \text{ such that } & \begin{cases} f_i = f(x_i) & i = k, k+1 \\ g_i = \nabla f(x_i) & i = k, k+1 \end{cases} \\ & x_{k+1} = x_k - \gamma_k g_k \\ & \|g_k\|^2 = 1, \end{aligned}$$

which can be encoded as a 2×2 SDP using

$$G \triangleq \begin{bmatrix} \|g_k\|^2 & \langle g_{k+1}, g_k \rangle \\ \langle g_k, g_{k+1} \rangle & \|g_{k+1}\|^2 \end{bmatrix}, \quad F \triangleq f_{k+1} - f_k,$$

and hence only 2 inequalities arising from interpolation constraints.

10. *Update your code for computing (numerically) the worst-case ratio $\frac{f(x_{k+1}) - f(x_\star)}{f(x_k) - f(x_\star)}$ as a function of L, μ, and γ_k. For what values of γ_k do you observe that this ratio is smaller than 1? How would you update the SDP formulation for taking this change into account?*

    Correction: The optimal stepsize should still match $2/(L+\mu)$ and the acceptable stepsize range is still $\gamma_k \in (0, 2/L)$. The SDP follows the same changes as for the previous questions (with objective functions instead of gradients). Results in a 3×3 SDP involving 6 inequalities from interpolation. The worst-case convergence rate in terms of this ratio should also match that in terms of the squared distance and that in terms of squared gradient norms; namelly $\tau(\mu, L, \gamma_k) = \max\{(1-\gamma_k L)^2, (1-\gamma_k \mu)^2\}$.

11. *Update your code for computing (numerically) the worst-case ratio $\frac{\|x_N - x_\star\|^2}{\|x_0 - x_\star\|^2}$ as a function of L, μ, and a sequence $\{\gamma_k\}_{0 \leqslant k \leqslant N-1}$. How would you update the SDP formulation for taking this change into account?*

Correction: Following the same steps, we arrive to

$$\begin{aligned}
\sup_{\substack{d \\ x_0,x_1,\ldots,x_N,x_\star \\ g_0,g_1,\ldots,g_{N-1},g_\star \\ f_k,f_{k+1},f_\star}} \quad & \|x_N - x_\star\|^2 \\
\text{s.t. } \exists f \in \mathcal{F}_{\mu,L} \text{ such that } & \begin{cases} f_i = f(x_i) & i = 0,1,\ldots,N-1,\star \\ g_i = \nabla f(x_i) & i = 0,1,\ldots,N-1,\star \end{cases} \\
& x_{i+1} = x_i - \gamma_i g_i \quad i = 0,1,\ldots,N-1 \\
& \|x_0 - x_\star\|^2 = 1 \\
& g_\star = 0.
\end{aligned}$$

This formulation naturally translates to a $(N+1)\times(N+1)$ SDP with

$$P \triangleq [x_0 - x_\star,\, g_0,\, g_1,\, \ldots,\, g_{N-1}],\ G \triangleq P^\top P \succcurlyeq 0,$$
$$F \triangleq [f_0 - f_\star,\, f_1 - f_\star,\, f_{N-1} - f_\star].$$

along with $N(N+1)$ interpolation inequalities (two inequalities per pair of points in the discrete representation). As a validation test for your code, using any constant stepsize rule $\gamma_k = \gamma$ should lead to a worst-case ratio equal to $\left(\max\{(1-\gamma_k L)^2, (1-\gamma_k\mu)^2\}\right)^n$

12. *Using previous points: assume you computed an optimal solution to the SDP formulation for the ratio $\frac{\|x_N - x_\star\|^2}{\|x_0 - x_\star\|^2}$ what is the link between the rank of the Gram matrix G and the dimension d in which this counter-example lives?*

    Correction: The dimension of the counter-example corresponds to $\operatorname{rank}(G)$.

13. *For obtaining "low-dimensional" worst-case instances, it is often useful to use heuristics. One of them is known as the "trace heuristic" (alternatively, the "logdet heuristic" is sometimes more efficient; see Exercise 7), which consists in solving a second optimization problem, whose solution hopefully forces G to have a lower rank. The trace heuristic for (8) consists in solving*

$$\begin{aligned}
\min_{G,F} \quad & \operatorname{Trace}(G) \\
\text{s.t.} \quad & F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{L}{L-\mu} G_{1,2} \leqslant 0 \\
& -F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{\mu}{L-\mu} G_{1,2} \leqslant 0 \\
& G_{1,1} = 1 \\
& G_{1,1} + \gamma_k^2 G_{2,2} - 2\gamma_k G_{1,2} = \tau(\mu, L, \gamma_k) \\
& G \succcurlyeq 0,
\end{aligned} \tag{9}$$

for which one first needs to obtain a precise approximation of $\tau(\mu, L, \gamma_k)$.

The trace heuristic is already implemented within PEPit and PESTO ; see this PEPit code (alternative in Matlab: PESTO code). Can you plot the resulting worst-case function(s) for a few different values of N (e.g., $N = 1, 2, 5, 10$) with $\gamma_k = \frac{1}{L}$. Does it correspond to a simple function? (Hint: two extremely simple examples of L-smooth μ-strongly convex functions are $\frac{\mu}{2}x^2$ and $\frac{L}{2}x^2$). Can you use this conclusion for "guessing" an expression for $\tau(\mu, L, 1/L)$ based on the behavior of gradient descent on the worst-case function that you identified?

Correction: Experiments should lead to the observation that $\frac{\mu}{2}x^2$ is a low-dimensional worst-case function for all N (for the particular choice of stepsize $1/L$).

If the worst-case behavior is achieved (among others) on this function, the convergence rate is given by $\tau(\mu, L, 1/L) = (1-\mu/L)^2$, since $x_{k+1} - x_\star = x_k - \gamma_k \nabla f(x_k) - x_\star = (1-\mu\gamma_k)(x_0 - x_\star)$ on this simple quadratic function.

14. *Set $\gamma_k = \frac{1}{L}$ and $\mu = 0$. What worst-case ratio $\frac{\|x_N - x_\star\|^2}{\|x_0 - x_\star\|^2}$ do you observe numerically? A classical way to avoid such a pathological behavior (in the analyses) is to study different types of ratios. Modify your code for studying the ratio $\frac{f(x_N) - f_\star}{\|x_0 - x_\star\|^2}$. Can you guess the dependence of the worst-case ratio on N based on a few numerical trials? and the dependence on L?*

    Correction: The worst-case ratio $\frac{\|x_N - x_\star\|^2}{\|x_0 - x_\star\|^2}$ is equal to 1 (i.e., no guaranteed convergence); the worst-case ratio $\frac{f(x_N) - f_\star}{\|x_0 - x_\star\|^2}$ should match exactly $\frac{L}{4N+2}$.

15. *Same question with $\gamma_k = \frac{1}{L}$, $\mu = 0$, and the ratio $\frac{\|\nabla f(x_N)\|^2}{\|x_0 - x_\star\|^2}$.*

    Correction: The worst-case ratio should match exactly $\frac{L^2}{(N+1)^2}$.

16. *Same question with $\gamma_k = \frac{1}{L}$, $\mu = 0$, and the ratio $\frac{\|x_N - x_\star\|^2}{\|\nabla f(x_0)\|^2}$.*

    Correction: The worst-case ratio should be unbounded (PEPit returns an error for telling this). A possibility for finding low-dimensional worst-case examples illustrating this is to fix some threshold ϵ and to solve the trace minimization problem with $\|\nabla f(x_0)\|^2 = 1$ and $\|x_N - x_\star\|^2 = \epsilon$ for a few relatively large values of ϵ. For example, set $\|\nabla f(x_0)\|^2 = 1$, $L = 1$, $\gamma_k = 1$; the following Huber function

$$f(x) = \begin{cases} |x| - \frac{1}{2} & \text{if } \|x\| \geqslant 1 \\ \frac{1}{2}x^2 & \text{otherwise,} \end{cases} \tag{10}$$

    allows obtaining arbitrarily large $\|x_N - x_\star\|^2$ simply by taking x_0 arbitrarily far away from $x_\star = 0$. That is, pick $x_0 = N + \epsilon$ (with $\epsilon > 1$), we get $x_N = \epsilon$. As a conclusion, this type of bounds is not appropriate (it allows for arbitrarily bad guarantees by construction).

17. *Based on your current experience, what are, according to you, the key elements which allowed casting the worst-case analysis as an SDP?*

    Correction: Key elements:

    (a) *the class of functions admits simple quadratic interpolation conditions that can be represented within an SDP,*

    (b) *the class of algorithm is simple: the stepsizes are fixed before hand and do not depend on the function class,*

    (c) *the "performance measure" and "initial conditions" can be represented linearly in terms of the Gram matrix.*

At this stage, the reader is familiar with necessary ingredients for attacking most exercises from Section 3. Before going further, we advise the reader to go through Exercise 3.

The goal of the next exercise is to link what we have seen so far with "classical convergence proofs". That is, we want to illustrate how to create rigorous worst-case convergence bounds using the previous concept together with SDP duality. Once such a proof is found, one can convert it to a certificate that is easy to verify without resorting on any semidefinite programming.

Exercise 2 (Gradient method—"dual performance estimation") *This exercise uses the same problem formulations and notation as Exercise 1.*

1. *Using Lagrangian duality with the following primal-dual pairing ($\tau, \lambda_1, \lambda_2$ are dual variables)*

$$\begin{aligned} &F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{L}{L-\mu} G_{1,2} \leqslant 0 && : \lambda_1 \\ &-F + \tfrac{L\mu}{2(L-\mu)} G_{1,1} + \tfrac{1}{2(L-\mu)} G_{2,2} - \tfrac{\mu}{L-\mu} G_{1,2} \leqslant 0 && : \lambda_2 \\ &G_{1,1} = 1 && : \rho, \end{aligned} \tag{11}$$

one can show that (we will do it in the sequel):

$$\begin{aligned} &\min_{\rho,\lambda_1,\lambda_2\geqslant 0} \rho \\ &\text{s.t. } S=\begin{bmatrix} \rho-1+\frac{\lambda_1 L\mu}{L-\mu} & \gamma_k-\frac{\lambda_1(\mu+L)}{2(L-\mu)} \\ \gamma_k-\frac{\lambda_1(\mu+L)}{2(L-\mu)} & \frac{\lambda_1}{L-\mu}-\gamma_k^2\end{bmatrix}\succcurlyeq 0 \\ &0=\lambda_1-\lambda_2, \end{aligned} \tag{12}$$

is a standard Lagrangian dual for the problem of computing $\tau(\mu,L,\gamma_k)$. Note that equality holds due to strong duality (for going further: prove strong duality using a Slater condition).

What is the relationship between feasible points to (12) and $\tau(\mu,L,\gamma_k)$?

Correction: This a standard SDP dual (see following exercises for obtaining it).

As $\tau(\mu,L,\gamma_k)$ corresponds to the feasible point with the smallest value of ρ, all feasible points to the dual problem are upper bounds on $\tau(\mu,L,\gamma_k)$. That is, for any triplet

$$(\rho(\mu,L,\gamma_k),\lambda_1(\mu,L,\gamma_k),\lambda_2(\mu,L,\gamma_k))$$

that is feasible for (12) for a certain range $(\mu,L,\gamma_k)\in\Lambda$, one has $\tau(\mu,L,\gamma_k)\leqslant\rho(\mu,L,\gamma_k)$ for all $(\mu,L,\gamma_k)\in\Lambda$. In other words, for all $(\mu,L,\gamma_k)\in\Lambda$ it holds that

$$\|x_{k+1}-x_\star\|^2\leqslant\rho(\mu,L,\gamma_k)\|x_k-x_\star\|^2$$

for all $d\in\mathbb{N}$, all $f\in\mathcal{F}_{\mu,L}(\mathbb{R}^d)$, and all $x_k,x_{k+1},x_\star\in\mathbb{R}^d$ such that $x_{k+1}=x_k-\gamma_k\nabla f(x_k)$ and $x_\star=\operatorname{argmin}_{x\in\mathbb{R}^d} f(x)$.

2. *(Lagrangian dual) Show that (12) is a dual for (8) using the primal-dual pairing (11).*

Correction: For convenience, let us rewrite the constraints using more standard notations:

$$\begin{aligned} a_1F+\operatorname{Trace}(A_1G)\triangleq{}&F+\tfrac{L\mu}{2(L-\mu)}G_{1,1}+\tfrac{1}{2(L-\mu)}G_{2,2}-\tfrac{L}{L-\mu}G_{1,2}\leqslant 0\\ a_2F+\operatorname{Trace}(A_2G)\triangleq{}&-F+\tfrac{L\mu}{2(L-\mu)}G_{1,1}+\tfrac{1}{2(L-\mu)}G_{2,2}-\tfrac{\mu}{L-\mu}G_{1,2}\leqslant 0\\ a_0F+\operatorname{Trace}(A_0G)\triangleq{}&G_{1,1}=1, \end{aligned}$$

as well as the the objective as $a_{obj}F+\operatorname{Trace}(A_{obj}G)\triangleq G_{1,1}+\gamma_k^2G_{2,2}-2\gamma_kG_{1,2}$. Those choices are valid with (this choice is unique for making all A_i's symmetric):

$$a_0=0,\ a_1=1,\ a_2=-1,\ a_{obj}=0,$$

and

$$A_0=\begin{pmatrix}1&0\\0&0\end{pmatrix},\ A_1=\frac{1}{2(L-\mu)}\begin{pmatrix}L\mu&-L\\-L&L\mu\end{pmatrix},\ A_2=\frac{1}{2(L-\mu)}\begin{pmatrix}L\mu&-\mu\\-\mu&L\mu\end{pmatrix},\ A_{obj}=\begin{pmatrix}1&-\gamma_k\\-\gamma_k&\gamma_k^2\end{pmatrix}.$$

The standard Lagrangian relaxation of the problem is then given by

$$\min_{\lambda_1,\lambda_2,\rho\geqslant 0}\ \max_{G\succeq 0,F}\ \rho+\operatorname{Trace}\left(G(A_{obj}-\rho A_0-\lambda_1A_1-\lambda_2A_2)\right)+F(a_{obj}-\rho a_0-\lambda_1a_1-\lambda_2a_2), \tag{13}$$

and is an upper bound on $\tau(\mu,L,\gamma_k)$. For this upper bound to be nontrivial, we need (i) the factor multiplying F to be equal to 0 and (ii) the factor of G to be negative semidefinite. Under those conditions, the Lagrangian dual corresponds to

$$\begin{aligned} &\min_{\lambda_1,\lambda_2,\rho\geqslant 0}\rho\\ &\text{s.t. } S\triangleq-(A_{obj}-\rho A_0-\lambda_1A_1-\lambda_2A_2)\succcurlyeq 0\\ &s\triangleq-(a_{obj}-\rho a_0-\lambda_1a_1-\lambda_2a_2)=0. \end{aligned}\tag{14}$$

Substituting the expressions for A_i's and a_i's above, one recovers (12) (the expression of S is also obtained by substituting the expression of λ_2).

3. *(Trick) The intent of this second point is to provide a shortcut to obtain a symbolical formulation of the dual problem. In short, consider the following variation of* (5) *with* $d = 1$*:*

$$\begin{aligned} \sup_{\substack{x_k, x_{k+1}, x_\star \in \mathbb{R} \\ g_k, g_\star \in \mathbb{R} \\ f_k, f_\star \in \mathbb{R}}} \quad & \|x_{k+1} - x_\star\|^2 \\ \text{s.t. } & f_\star \geqslant f_k + \langle g_k, x_\star - x_k \rangle + \tfrac{1}{2L}\|g_\star - g_k\|^2 + \tfrac{\mu}{2(1-\mu/L)}\left\|x_\star - x_k - \tfrac{1}{L}(g_\star - g_k)\right\|^2 \\ & f_k \geqslant f_\star + \langle g_\star, x_k - x_\star \rangle + \tfrac{1}{2L}\|g_k - g_\star\|^2 + \tfrac{\mu}{2(1-\mu/L)}\left\|x_k - x_\star - \tfrac{1}{L}(g_k - g_\star)\right\|^2 \\ & x_{k+1} = x_k - \gamma_k g_k \\ & \|x_k - x_\star\|^2 = 1 \\ & g_\star = 0. \end{aligned} \tag{15}$$

After substitution of x_{k+1} *and* $g_\star$ *(one can also set* $x_\star = f_\star = 0$ *wlog) in the objective and the constraints, write the Lagrangian* $\mathcal{L}(x_k, g_k, f_k, x_\star, f_\star; \lambda_1, \lambda_2, \rho)$ *(where* $\lambda_1, \lambda_2, \rho \geqslant 0$ *are associated as in* (11)*). Then, show that the expression of* S *in* (12) *(i.e., the LMI in* (12)*) corresponds to* $S = -\frac{1}{2}\nabla^2_{x_k, g_k}\mathcal{L}(x_k, g_k, f_k, x_\star, f_\star; \lambda_1, \lambda_2, \rho)$ *and that the linear constraint in* (12) *corresponds to requiring* $s = \nabla_{f_k}\mathcal{L}(x_k, g_k, f_k, x_\star, f_\star; \lambda_1, \lambda_2, \rho)$ *to be zero.*

An advantage of this approach is that it allows to easily obtain the dual formulations using symbolic computation (see, e.g., Python this notebook, or this Matlab file).

Correction: Because all expressions in the Lagrangian are quadratics in $(x_k - x_\star, g_k)$ *and linear in* $f_k - f_\star$*, the gradient and Hessian of the Lagrangian respectively with respect to* f_k *and to* (x_k, g_k) *actually corresponds to the matrices* s *and* S *such that*

$$\mathrm{Trace}(-SG) + sF = \mathcal{L}(x_k, g_k, f_k, x_\star, f_\star; \lambda_1, \lambda_2, \rho),$$

where G *is the previously defined Gram matrix* (6)*.*

4. *Solve* (12) *numerically. Do those values match those obtained in the first exercise?*

   For doing that, you can complete the following Python code (alternative: *Matlab code*)*.*

5. *Is there a simple closed-form expression for* $\tau(\mu, L, \gamma_k)$*? Does it match the numerical values obtained using the previous codes for computing* $\tau(\mu, L, \gamma_k)$ *numerically?*

   Hint: can you solve (12) *in closed-form? Some help relying on symbolic computations is provided in the same notebook as in the previous exercise (alternative: Matlab code).*

   Correction: The expression is $\tau(\mu, L, \gamma_k) = \max\{(1 - \gamma_k L)^2, (1 - \gamma_k \mu)^2\}$*. Because the problem has a linear objective, the optimal* ρ *(i.e.,* τ*) must be such that the solution is on the boundary of the PSD cone. That is, one can find an expression of* ρ *as a function of the other variables and parameters by solving* $\det(S) = 0$*. There are then two possibilities:*

   (a) either the other eigenvalue of S *is zero (but one can observe, numerically, that the rank of* S *is empirically equal to 1 for non-optimal stepsizes)*

   (b) or λ_1 *can be chosen for minimizing the expression of* ρ*.*

   The notebook provides a path through this using simple symbolic computations.

6. *Consider* $\gamma_k = \frac{1}{L}$ *for simplicity. Given the optimal value of the multipliers* $\rho, \lambda_1, \lambda_2$ *in* (12)*, can you write a "direct" proof for the linear convergence in terms of distance to an optimal point without resorting on any SDP formulation?*

   Correction: A link might be drawn directly from writing the Lagrangian of (8)*; but let us first start with a concrete answer, and explain it afterwards.*

   Concretely, sum up the following inequalities with their corresponding weights:

$$\begin{aligned} f_\star &\geqslant f_k + \langle g_k; x_\star - x_k \rangle + \tfrac{1}{2L}\|g_k - g_\star\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_k - x_\star - \tfrac{1}{L}(g_k - g_\star)\|^2 : \lambda_1, \\ f_k &\geqslant f_\star + \langle g_\star; x_k - x_\star \rangle + \tfrac{1}{2L}\|g_k - g_\star\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_k - x_\star - \tfrac{1}{L}(g_k - g_\star)\|^2 : \lambda_2. \end{aligned}$$

We use the following values for the multipliers: $\lambda_1 = \lambda_2 = 2\gamma_k\tau(\mu, L, \gamma_k) \geqslant 0$ After appropriate substitutions of $g_\star$, x_{k+1}, and $\tau(\mu, L, \gamma_k)$, using respectively $g_\star = 0$, $x_{k+1} = x_k - \gamma_k g_k$ and $\tau(\mu, L, \gamma_k) = (1 - \gamma_k\mu)$, and with little effort, one can check that the previous weighted sum of inequalities can be written in the form:

$$\|x_{k+1} - x_\star\|^2 \leqslant \quad (1 - \gamma_k\mu)^2 \|x_k - x_\star\|^2 - \tfrac{\gamma_k(2-\gamma_k(L+\mu))}{L-\mu}\|\mu(x_k - x_\star) - g_k\|^2.$$

This statement can be checked simply by expanding both expressions (i.e., the weighted sum and its reformulation) and verifying that all terms indeed match. Note that the term

$$-\frac{\gamma_k(2 - \gamma_k(L + \mu))}{L - \mu}\|\mu(x_k - x_\star) - g_k\|^2$$

is equal to $-\mathrm{Trace}(SG)$.

Now, why is that that this simple weighted sum can be reformulated as is? Well, it turns out that this is a simple consequence of Lagrangian duality and our constructions above. First, our weighted sum can be rewritten as:

$$\begin{aligned}
0 \geqslant &\lambda_1 \left[f_k - f_\star + \langle g_k; x_\star - x_k\rangle + \frac{1}{2L}\|g_k - g_\star\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}(g_k - g_\star)\|^2\right] \\
&+ \lambda_2 \left[f_\star - f_k + \langle g_\star; x_k - x_\star\rangle + \frac{1}{2L}\|g_k - g_\star\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}(g_k - g_\star)\|^2\right] \\
= &\lambda_1 \left[a_1 F + \mathrm{Trace}(A_1 G)\right] + \lambda_2 \left[a_2 F + \mathrm{Trace}(A_2 G)\right],
\end{aligned}$$

where the equality follows from the construction of G and F and from the notations from Exercise 2 §2.

Now, since λ_1, λ_2, and ρ are constructed as feasible points to the dual problem (12), or equivalently (14). Thereby, using the constraints in (14) (which are satisfied by $(\lambda_1, \lambda_2, \rho)$), we get

$$\begin{aligned}
0 \geqslant &\lambda_1 \left[f_k - f_\star + \langle g_k; x_\star - x_k\rangle + \frac{1}{2L}\|g_k - g_\star\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}(g_k - g_\star)\|^2\right] \\
&+ \lambda_2 \left[f_\star - f_k + \langle g_\star; x_k - x_\star\rangle + \frac{1}{2L}\|g_k - g_\star\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_k - x_\star - \frac{1}{L}(g_k - g_\star)\|^2\right] \\
= &\lambda_1 \left[a_1 F + \mathrm{Trace}(A_1 G)\right] + \lambda_2 \left[a_2 F + \mathrm{Trace}(A_2 G)\right] \\
= &(s + a_{obj} - \rho a_0)F + \mathrm{Trace}((S + A_{obj} - \rho A_0)G).
\end{aligned}$$

Thefore, it suffices to reorganize the terms to reach

$$a_{obj}F + \mathrm{Trace}(A_{obj}G) \leqslant \rho(a_0 F + \mathrm{Trace}(A_0 G)) - sF - \mathrm{Trace}(SG),$$

and finally

$$a_{obj}F + \mathrm{Trace}(A_{obj}G) \leqslant \rho(a_0 F + \mathrm{Trace}(A_0 G))$$

as $S \succcurlyeq 0$ and $s = 0$. Note that this reasoning is still valid for other objectives of the original problem (i.e., other values of $a_0, A_0, a_{obj}, A_{obj}$). For this exercise, the conclusion is reached by noting that $a_{obj} = a_0 = 0$.

7. *A corresponding dual problem for the worst-case ratio $\frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2}$ given by*

$$\begin{aligned}
\min_{\rho,\lambda_1,\lambda_2 \geqslant 0} \; &\rho \\
s.t. \; S = &\begin{bmatrix} \rho + \lambda_1 \frac{(1-\gamma_k L)(1-\gamma_k\mu)}{L-\mu} & -\lambda_1 \frac{2-\gamma_k(L+\mu)}{2(L-\mu)} \\ -\lambda_1 \frac{2-\gamma_k(L+\mu)}{2(L-\mu)} & \frac{\lambda_1}{L-\mu} - 1 \end{bmatrix} \succcurlyeq 0 \\
0 = &\lambda_1 - \lambda_2.
\end{aligned} \qquad (16)$$

Is there a simple closed-form solution for this problem? Alternatively: solve this problem numerically and try to guess a solution.

Correction: The answer for the convergence rate is the same as that for the previous ratio in terms of distances, but the multiplier are different. The interested reader can modify the previous codes for obtaining the corresponding expressions.

8. *Using the following primal-dual pairing*

$$\begin{aligned}
&f(x_0) \geqslant f(x_\star) + \tfrac{1}{2L}\|\nabla f(x_0)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_0 - x_\star - \tfrac{1}{L}\nabla f(x_0)\|^2 && :\lambda_1\\
&f(x_\star) \geqslant f(x_0) + \langle \nabla f(x_0), x_\star - x_0\rangle + \tfrac{1}{2L}\|\nabla f(x_0)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_0 - x_\star - \tfrac{1}{L}\nabla f(x_0)\|^2 && :\lambda_2\\
&f(x_1) \geqslant f(x_\star) + \tfrac{1}{2L}\|\nabla f(x_1)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_1 - x_\star - \tfrac{1}{L}\nabla f(x_1)\|^2 && :\lambda_3\\
&f(x_\star) \geqslant f(x_1) + \langle \nabla f(x_1), x_\star - x_1\rangle + \tfrac{1}{2L}\|\nabla f(x_1)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_1 - x_\star - \tfrac{1}{L}\nabla f(x_1)\|^2 && :\lambda_4\\
&f(x_0) \geqslant f(x_1) + \langle \nabla f(x_1), x_0 - x_1\rangle + \tfrac{1}{2L}\|\nabla f(x_0) - \nabla f(x_1)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_1 - x_0 - \tfrac{1}{L}(\nabla f(x_1) - \nabla f(x_0))\|^2 && :\lambda_5\\
&f(x_1) \geqslant f(x_0) + \langle \nabla f(x_0), x_1 - x_0\rangle + \tfrac{1}{2L}\|\nabla f(x_0) - \nabla f(x_1)\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|x_1 - x_0 - \tfrac{1}{L}(\nabla f(x_1) - \nabla f(x_0))\|^2 && :\lambda_6\\
&f(x_0) - f(x_\star) = 1 && :\rho
\end{aligned}$$

a corresponding dual problem for the worst-case ratio $\frac{f(x_{k+1})-f_\star}{f(x_k)-f_\star}$ *is given by*

$$\begin{aligned}
&\min_{\rho,\lambda_1,\lambda_2\geqslant 0} \rho\\
&\text{s.t.} \begin{bmatrix} \frac{\mu L(\lambda_1+\lambda_2+\lambda_3+\lambda_4)}{L-\mu} & -\frac{L(\lambda_2+\gamma\mu(\lambda_3+\lambda_4))+\mu\lambda_1}{L-\mu} & -\frac{L\lambda_4+\mu\lambda_3}{L-\mu}\\ * & \frac{\gamma\mu(\gamma L(\lambda_3+\lambda_4+\lambda_5+\lambda_6)-2\lambda_5)-2\gamma L\lambda_6+\lambda_1+\lambda_2+\lambda_5+\lambda_6}{L-\mu} & \frac{\gamma L\lambda_4+\lambda_5(\gamma L-1)+\gamma\mu(\lambda_3+\lambda_6)-\lambda_6}{L-\mu}\\ * & * & \frac{\lambda_3+\lambda_4+\lambda_5+\lambda_6}{L-\mu}\end{bmatrix} \succcurlyeq 0\\
&\quad 0 = \rho - \lambda_1 + \lambda_2 - \lambda_5 + \lambda_6\\
&\quad 1 = -\lambda_3 + \lambda_4 + \lambda_5 - \lambda_6,
\end{aligned} \tag{17}$$

where "$*$" denotes symmetrical elements in the PSD matrix. Is there a simple closed-form solution for this problem? Note that this SDP is already coded here (alternative in Matlab: here)

Trick #1: solve the problem numerically and plot some values for the multipliers; trick #2: pick $\lambda_1 = \lambda_3 = \lambda_6 = 0$*; does the problem simplify?*

Trick #3: one can use trace norm of logdet minimization (on S), or $\|.\|_1$ norm minimization on λ's on the dual side for trying to get simpler/sparser proofs. That being said, when there are only few inequalities involved, it is often easier to simply greedily try different combinations of λ_i's to be set to 0.

Correction: Two difficulties here: (i) the LMI is larger, and (ii) contrary to the previous cases, the solution is not unique, which makes it complicated for the symbolic solvers. A simple approach for overcoming this is to "clean" the problem by hand, e.g., by removing unnecessary multipliers manually.

9. *Obtain the dual formulations for the problems of computing the worst-case ratios* $\frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2}$ *and* $\frac{f(x_{k+1})-f_\star}{f(x_k)-f_\star}$*, either using symbolic computations or by hand.*
Correction: See notebooks!

3 Further exercises

The goal of this second set of exercises is to get familiar with the approach and to understand to what extend it applies beyond a single iteration of gradient descent. There main points of attention below can be categorized within three types:

- what type of algorithms can it study?
- What classes of problems can it study?
- What types of guarantees can it study?

The corrections are much lighter and are mostly contained in a few notebooks.

Exercise 3 (Sublinear convergence of gradient descent and acceleration) *For this exercise, we consider the problem of minimizing*

$$\min_x f(x),$$

where f is an L-smooth convex function (see Definition 1). We consider three algorthms, which respectively iterate:

- *a gradient method:*
$$x_{k+1} = x_k - \tfrac{1}{L}\nabla f(x_k),$$
- *an heavy-ball method:*
$$x_{k+1} = x_k + \tfrac{k}{k+2}(x_k - x_{k-1}) - \tfrac{1}{k+2}\tfrac{1}{L}\nabla f(x_k),$$
- *an accelerated (or fast) gradient method:*
$$x_{k+1} = y_k - \tfrac{1}{L}\nabla f(y_k)$$
$$y_{k+1} = x_{k+1} + \tfrac{k-1}{k+2}(x_{k+1} - x_k)$$

1. *Consider the following ratios:*

   (a) $\frac{f(x_N)-f_\star}{\|x_0-x_\star\|^2}$ *and* $\frac{\min_{0\leqslant i\leqslant N}\{f(x_i)-f_\star\}}{\|x_0-x_\star\|^2}$,

   (b) $\frac{\|\nabla f(x_N)\|^2}{\|x_0-x_\star\|^2}$ *and* $\frac{\min_{0\leqslant i\leqslant N}\{\|\nabla f(x_i)\|^2\}}{\|x_0-x_\star\|^2}$.

   Can we formulate the worst-case analyses for those methods and those ratios as SDPs? What is the influence of N on its size and on the number of inequalities under consideration?

   Correction: Compared to Exercise 1, it suffices to introduce a discrete version of f sampled at $x_\star$ as well as at all iterates where a gradient is evaluated (i.e., $x_0, x_1, \ldots, x_N$). The discrete version of the function is handled via a larger Gram matrix $G = P^T P \succcurlyeq 0$ and a larger F of the type:

$$P \triangleq [x_0 - x_\star,\ g_0,\ g_1,\ \ldots,\ g_N], \quad F \triangleq [f_0 - f_\star,\ f_1 - f_\star,\ \ldots,\ f_N - f_\star]$$

   (note that $x_1, \ldots, x_N$ are not necessary in P and in the Gram matrix as they are obtained by fixed linear combinations of x_0 and $g_0, \ldots, g_{N-1}$). Hence $G \in \mathbb{R}^{(N+2)\times(N+2)}$ and the number of interpolation inequalities is $(N+1)(N+2) = O(N^2)$ (two inequalities per pair of points in $\{x_\star, x_0, x_1, \ldots, x_N\}$).

   Then, for incorporating performance measures of type $\min_{0\leqslant i\leqslant N}\{\|\nabla f(x_i)\|^2\}$ it suffices to introduce a slack variable t (which is the objective of the performance estimation problem, which is maximized) and to impose that $t \leqslant \|\nabla f(x_i)\|^2 \triangleq G_{i+2,i+2}$ for $i = 0, \ldots, N$. Similarly, for $\min_{0\leqslant i\leqslant N}\{f(x_i) - f_\star\}$ we introduce the slack variable t (to be maximized) and impose $t \leqslant f(x_i) - f_\star \triangleq F_{i+1}$ for $i = 0, \ldots, N$.

2. *Using PEPit or PESTO compare those methods in terms of those ratios(for $L = 1$ and as a function of N few values of $N = 0, 1, \ldots, 30$).*

   Correction: See this notebook.

Exercise 4 (Primal proximal point method) *Consider the problem of minimizing a (closed, proper) convex function*

$$\min_{x\in\mathbb{R}^d} f(x),$$

with the proximal-point method (with stepsize γ):

$$x_{k+1} = \operatorname*{argmin}_{x\in\mathbb{R}^d}\{f(x) + \tfrac{1}{2\gamma}\|x - x_k\|^2\}.$$

1. *Note that using optimality conditions on the definition of the proximal-point iteration, one can rewrite the iteration as*
$$x_{k+1} = x_k - \gamma g_{k+1},$$
with $g_{k+1} \in \partial f(x_{k+1})$ (the subdifferential of f at x_{k+1}; i.e., g_{k+1} is a subgradient at x_{k+1}). Can you reformulate the problem of finding a worst-case example for the ratio $\frac{f(x_N)-f_\star}{\|x_0-x_\star\|^2}$ as an SDP? What are the key ingredients for arriving to it?

   Correction: Essentially just as for gradient descent, except that (i) the gradient that is used for the update is that of the upcoming iterate (implicit update), and (ii) that the interpolation conditons are simplified (just convexity): $f_i \geqslant f_j + \langle g_j, x_i - x_j\rangle$ for all $i, j \in \{\star, 1, 2, \ldots, N\}$ (no index 0).

2. *Provide a PEPit or PESTO code for computing the worst-case value for this ratio, and experiment with it.*

   Correction: See this notebook.

3. *Guess the dependence of the worst-case ratio on γ and on N; confirm your findings using numerics.*

   Correction: The best bound on this ratio is $\frac{1}{4\gamma N}$.

4. *Can you guess the shape of a worst-case function? You may want to use an heuristic for finding lower rank worst-case examples.*

   Correction: Using a rank minimization heuristic such as the trace norm minimization or the logdet heuristic (implemented within PEPit and described in [11]—see pdf), one can obtain that the worst-cases are achieved, among others, on some one-dimensional linear minimization problems of the form $\min_{x\geqslant 0}\, cx$ for properly chosen c (see, [2, Theorem 4.1]; pdf here.)

   The code is provided in this notebook.

Exercise 5 (Projected gradient and Frank-Wolfe) *We consider the minimization problem*
$$\min_{x\in Q} f(x),$$
where f is an L-smooth convex function and Q is a non-empty convex set with finite diameter (i.e., $\|x-y\| \leqslant D$ for all $x, y \in Q$ for some $D > 0$). Given some $x_0 \in Q$, we consider two possible first-order methods:

- *projected gradient:*
$$x_{k+1} = \mathrm{Proj}_Q\left[x_k - \tfrac{1}{L}\nabla f(x_k)\right]$$
- *conditional gradient (a.k.a., Frank-Wolfe):*
$$y_k = \operatorname*{argmin}_{y\in Q} \langle \nabla f(x_k), y\rangle$$
$$x_{k+1} = \tfrac{k}{k+2}x_k + \tfrac{1}{k+2}y_k.$$

1. *For each method, how can you write the problem of computing the worst-case for the value of $f(x_N) - f_\star$? (note that we do not need a denominator as the diameter $D < \infty$ is bounded).*

   Correction: Denote by $\mathcal{I}_D$ the set of indicator functions (see Definition 3) with bounded diameter D. The problem of finding the worst-case is then formulated as a problem of finding a worst couple (f, i_Q) with $f \in \mathcal{F}_{0,L}$ (f is an L-smooth convex function) and $i_Q \in \mathcal{I}_D$. In other words, we aim to solve
$$\begin{aligned} \sup_{\substack{d,f,i_D\\ x_0,\ldots,x_N,x_\star}} \quad & f(x_N) - f_\star \\ \text{s.t.} \quad & f \in \mathcal{F}_{0,L},\; i_D \in \mathcal{I}_D \\ & x_{k+1} = \Pi_Q\left(x_k - \gamma_k \nabla f(x_k)\right) \text{ for } k = 0, \ldots, N-1 \\ & \nabla f(x_\star) + s_\star = 0 \text{ with } s_\star \in \partial i_Q(x_\star) \\ & i_Q(x_0) = 0, \end{aligned} \tag{18}$$

for the projected gradient method. Note that we can rewrite the iteration as

$$x_{k+1} = x_k - \gamma_k \nabla f(x_k) - s_{k+1}$$

with $s_{k+1} \in \partial i_Q(x_{k+1})$; that is, the projection is an implicit subgradient method on $i_Q(\cdot)$; this can be seen by using the definition of projection on Q as $\mathrm{Proj}_Q(x) = \mathrm{argmin}_y \{ i_Q(y) + \frac{1}{2}\|y-x\|^2\}$. Note that we incorporated the constraint $i_Q(x_0) = 0$ for enforcing x_0 to be feasible. For Frank-Wolfe, we aim to solve

$$\begin{aligned} \sup_{\substack{d,f,i_D \\ x_0,\ldots,x_N,x_\star}} \quad & f(x_N) - f_\star \\ \text{s.t.} \quad & f \in \mathcal{F}_{0,L},\ i_D \in \mathcal{I}_D \\ & y_k = \operatorname*{argmin}_{y \in Q} \langle \nabla f(x_k), y \rangle \text{ for } k = 0, \ldots, N-1 \\ & x_{k+1} = \tfrac{k}{k+2} x_k + \tfrac{1}{k+2} y_k \text{ for } k = 0, \ldots, N-1 \\ & \nabla f(x_\star) + s_\star = 0 \text{ with } s_\star \in \partial i_Q(x_\star) \\ & i_Q(x_0) = 0, \end{aligned} \tag{19}$$

where the equality defining y_k's can also be described in terms of optimality conditions of the linear minimization problem: $y_k = \mathrm{argmin}_y \langle \nabla f(x_k), y \rangle + i_Q(y) \Leftrightarrow 0 = \nabla f(x_k) + s_k$ with $s_k \in \partial i_Q(y_k)$.

2. *For each method, how can we sample the problem? (at which points do we need to sample each function?)*

   Correction: For projected gradient: f is sampled in terms of triplets (x_k, g_k, f_k) for $k \in \{\star, 0, 1, \ldots, N\}$ and i_Q is sampled in terms of triplets $(x_k, s_k, 0)$ for $k \in \{\star, 0, 1, \ldots, N\}$ (the two functions are sampled at the same places).

   For Frank-Wolfe: f is sampled in terms of triplets (x_k, g_k, f_k) for $k \in \{\star, 0, 1, \ldots, N\}$ and i_Q is sampled at $x_\star$, x_0 and y_k for $k \in \{0, 1, \ldots, N-1\}$ (the two functions are not sampled at the same places).

3. *Write a PEPit or PESTO code and guess the dependence on the iteration counter N of the worst-case $f(x_N) - f_\star$ for each of those methods. What are the dependences on D and L?*

   Correction: See this notebook.

   Both bounds should scale with LD^2 and be of order $O(N^{-1})$. For easy reference, the reader can compare his numerical bounds with:

   - *for projected gradient with $N \geqslant 1$*

     $$f(x_N) - f_\star \leqslant \frac{LD^2}{4N},$$

     see, e.g., [12, Theorem 2.9],
   - *for Frank-Wolfe with $N \geqslant 1$*

     $$f(x_N) - f_\star \leqslant \frac{2LD^2}{N+2},$$

     see, e.g., [13, Theorem 1].

Exercise 6 (Proximal point method for monotone inclusions) *For this exercise, we consider the problem of solving a monotone inclusion problem:*

$$\text{find } x \in \mathbb{R}^d \text{ such that } 0 \in M(x),$$

where $M : \mathbb{R}^d \rightrightarrows \mathbb{R}^d$ (point to set mapping) is a (maximal) mononotone operator (see Definition 5). In this context, the proximal point method consists in iterating

$$x_{k+1} = \mathrm{prox}_{\alpha M}(x_k) = (I + \alpha M)^{-1} x_k,$$

(where I is the identity operator) or equivalently

$$x_{k+1} = x_k - \alpha M(x_{k+1}).$$

1. *Consider the ratio $\frac{\|x_N - x_{N+1}\|^2}{\|x_0 - x_\star\|^2}$, where $x_\star$ is such that $0 \in M(x_\star)$. Can you compute this worst-case ratio using an SDP? Hint: you can use Theorem 5.*

   Correction: As the discretization is the same as that for the proximal method in Exercise 4, except that no function values will be stored (because M is an operator, not specifically related to a function).

2. *Write a PEPit or PESTO code for studying this ratio as a function of N and α.*

   Correction: See this notebook.

3. *Using dimension reduction, there should exist a low-dimensional worst-case example. Find it (numerically) and plot it.*

   Hint: more information about the worst-case guarantee and a low-dimensional worst-case example can be found in [14] (see pdf). Are your observations compatible with the fact that M might encode a rotation matrix?

   Correction: See this notebook. After dimension reduction, you should be able to plot a nice two-dimensional example nicely spiraling around some $x_\star$. This worst-case indeed corresponds to that in [14].

Exercise 7 (Fixed-point iterations) *For this exercise, we consider the fixed point problem*

$$\text{find } x \in \mathbb{R}^d \text{ such that } x = F(x),$$

where $F : \mathbb{R}^d \to \mathbb{R}^d$ is a nonexpansive mapping (i.e., it is 1-Lipschitz; that is, for all $x, y \in \mathbb{R}^d$, $\|F(x) - F(y)\| \leqslant \|x - y\|$).

1. *The Halpern iteration is given by*

$$x_{k+1} = \left(1 - \frac{1}{k+2}\right) F(x_k) + \frac{1}{k+2} x_0$$

   Can you write an SDP formulation for computing the worst-case ratio $\frac{\|x_N - F(x_N)\|^2}{\|x_0 - x_\star\|^2}$ (for some $x_\star = F(x_\star)$)?

   Hint: Theorem 6 with $L = 1$.

   Correction: As in previous examples, the key point is to realize that the problem can be sampled properly; see Theorem 6 with $L = 1$. Then, the algorithm can be formulated within the SDP as before (only "interpolation inequalites" must be adapted).

2. *Write a PEPit or PESTO code for computing the worst-case value of this ratio. Can you guess the dependence in N of the worst-case behavior of the previous ratio?*

   Correction: See this notebook.

3. *Can you find low-dimensional worst-case examples for this method? (Hint: use a few iterations of the logdet heuristic (implemented within PEPit and described in [11] (see pdf)).*

   Correction: See this notebook. There are indeed one-dimensional worst-case examples; see, e.g., [15, Example 3.1], or this pdf.

4. *An alternative is the so-called Krasnolselskii-Mann iteration, which can be instantiated as*

$$x_{k+1} = \left(1 - \frac{1}{k+2}\right) F(x_k) + \frac{1}{k+2} x_k$$

   How does it compare to Halpern in terms of worst-case ratios?

   Correction: See this notebook. The Halpern has a better worst-case behavior in terms of the ratio defined above.

Exercise 8 (Subgradient methods) *For this exercise, we consider the problem of minimizing an M-Lipschitz convex function (which is possibly nonsmooth; see Definition 4)*

$$\min_x f(x),$$

using a subgradient method $x_{k+1} = x_k - \gamma_k s_k$ with $s_k \in \partial f(x_k)$. We aim to compute worst-case bounds on $\min_{0 \leqslant i \leqslant N}\{f(x_i) - f_\star\}$ for certain values of N under the condition that $\|x_0 - x_\star\|^2 \leqslant R^2$ for some $x_\star \in \operatorname{argmin}_x f(x)$ and when $f \in \mathcal{C}_M$ (notation for closed, proper, convex, and M-Lipschitz).

1. *Write a code for computing a worst-case bound on $\min_{0\leqslant i\leqslant N}\{f(x_i) - f_\star\}$.*

   Correction: See this notebook.

2. *How does your bound compare to the known*

$$\min_{0\leqslant i\leqslant N}\{f(x_i) - f_\star\} \leqslant \frac{R^2 + M^2\sum_{k=0}^{N-1}\gamma_k^2}{\sum_{k=0}^{N-1}\gamma_k}$$

   for a few stepsize policies, including:

   - *fixed stepsize policy $\gamma_k = 1$,*
   - *fixed stepsize policy depending on the horizon $\gamma_k = \frac{R}{M\sqrt{N+1}}$,*
   - *decreasing stepsize policy $\gamma_k = \frac{R}{M\sqrt{k+1}}$,*
   - *decreasing stepsize policy $\gamma_k = \frac{R}{M(k+1)}$.*

   Correction: For all the rules, the numerically worst-case guarantee should be better than the known guarantee provided above. For some cases (e.g., fixed stepsize depending on the horizon), the guarantee is actually perfectly tight. See this notebook.

3. *Adapt your code for computing a worst-case bound on the last iterate $f(x_N) - f_\star$. Does the worst-case deteriorate?*

   Correction: Yes, the method is not monotonic and the worst-case behavior at the last iterate is worse than that at the best iterate, contrasting with gradient descent in the smooth convex setting.

4. *Evaluate the same two worst-case guarantees (best function value accuracy among the iterates and last one) for the quasi-monotone subgradient method: (from [16], or [17]):*

$$\begin{aligned}
s_k &\in \partial f(x_k)\\
d_k &= \frac{1}{k+2}\sum_{i=0}^{k} s_i\\
y_{k+1} &= \frac{k+1}{k+2}x_k + \frac{1}{k+2}x_0\\
x_{k+1} &= y_{k+1} - \frac{R}{M\sqrt{N+1}}d_k.
\end{aligned}$$

   Correction: See this notebook. The worst-case guarantees match $\frac{MR}{\sqrt{N+1}}$ for both the best and the last iterate.

Exercise 9 (Proximal gradient method) *We consider the composite convex minimization problem of the form*

$$\min_{x\in\mathbb{R}^d}\{F(x) \triangleq f(x) + h(x)\},$$

where f is L-smooth and μ-strongly convex and h is ccp (closed, convex, proper) so that its proximal operation is well-defined. We want to use the proximal gradient method

$$x_{k+1} = \operatorname*{argmin}_{x\in\mathbb{R}^d}\{h(x) + \langle\nabla f(x_k), x - x_k\rangle + \tfrac{L}{2}\|x - x_k\|^2\}$$

for solving the problem.

1. *Can we formulate the problem of computing the worst-case ratio $\frac{\|x_{k+1}-x_\star\|^2}{\|x_k-x_\star\|^2}$ as an SDP? Describe at which points each function needs to be sampled.*

   Correction: This exercise is similar to Exercise 5 (projected gradient method): f needs to be sampled at $x_\star$ and x_k, and h needs to be sampled at $x_\star$ and x_{k+1} (and evantually at x_k if we want to use the fact that x_k is feasible; but this is not necessary in the proof).

2. *Write a code for computing it, and compare the numerical values with those of Exercise 1.*

   Correction: See this notebook.

3. *Can you find low-dimensional worst-case instances?*

   Correction: Same worst-case instances as in Exercise 1: either $\frac{L}{2}x^2$ or $\frac{\mu}{2}x^2$ depending on the stepsize choice. See this notebook.

Exercise 10 (Douglas-Rachford) *Consider a composite convex minimization problem of the form*

$$\min_{x\in\mathbb{R}^d}\{F(x)\triangleq f(x)+h(x)\},$$

where f is L-smooth and μ-strongly convex and h is ccp (closed, convex, proper) so that both proximal operations are well-defined. We want to use the Douglas-Rachford (DR) method

$$\begin{aligned}x_k &= \mathrm{prox}_{\alpha h}(w_k),\\ y_k &= \mathrm{prox}_{\alpha f}(2x_k-w_k),\\ w_{k+1} &= w_k+\theta(y_k-x_k),\end{aligned}$$

for solving the problem.

1. *Let w_0 and w_0' be two different starting points. Formulate the problem of computing the worst-case ratio $\frac{\|w_1-w_1'\|^2}{\|w_0-w_0'\|^2}$ where w_1 and w_1' are the outputs of a DR iteration from respectively w_0 and w_0'. Where should the two functions be sampled?*

   Correction: This is similar to previous examples. h needs to be sampled at $x_0\triangleq\mathrm{prox}_{\alpha h}(w_0)$ and $x_0'\triangleq\mathrm{prox}_{\alpha h}(w_0')$; f needs to be sampled at $y_0\triangleq\mathrm{prox}_{\alpha f}(2x_0-w_0)$ and $y_0'\triangleq\mathrm{prox}_{\alpha f}(2x_0'-w_0')$.

2. *Write a code for computing it. As a reference when $\theta=1$: how do the numerics compare to the bound $\|w_1-w_1'\|^2\leqslant\left(\max\left\{\frac{1}{1+\alpha\mu},\frac{\alpha L}{1+\alpha L}\right\}\right)^2\|w_0-w_0'\|^2$ (see, e.g., [18, Theorem 2]).*

   Correction: See this notebook. The bounds should match.

3. *Can you find low-dimensional worst-case instances?*

   Correction: Yes, it is possible to find one-dimensional worst-case examples again.

   As follows from [18, Section 3.2], one such worst-case is achieved on either $f(x)=\frac{\mu}{2}x^2$ and $h(x)=0$ or on $f(x)=\frac{L}{2}x^2$ and $h(x)=i_{\{0\}}(x)$ (indicator function of the set $\{0\}$).

Exercise 11 (Alternate projections) *We consider the problem of finding a point in the intersection of two closed convex sets:*

$$\text{find } x\in Q_1\cap Q_2,$$

where $Q_1,Q_2\subseteq\mathbb{R}^d$ are two closed convex sets. We will compare, in terms of their worst-case behaviors, a few methods for solving this problem.

Write PEPit (or PESTO) codes for computing the worst-case ratios $\frac{\|\mathrm{Proj}_{Q_1}(x_N)-\mathrm{Proj}_{Q_2}(x_N)\|^2}{\|x_0-x_\star\|^2}$ (where $\mathrm{Proj}_Q(\cdot)$ denotes the projections onto a set Q, some $x_\star\in Q_1\cap Q_2$, and x_k denotes the iterate of the following methods; N denotes the "final" iteration count); you may experiment with other types of ratios. Can you guess the dependence on k of the worst-case value for this ratio?

1. *Alternate projections: set $y_0 = x_0$ and iterate*

$$\begin{aligned} x_{k+1} &= \mathrm{Proj}_{Q_1}(y_k), \\ y_{k+1} &= \mathrm{Proj}_{Q_2}(x_k), \end{aligned}$$

2. *averaged projections: iterate*

$$x_{k+1} = \frac{1}{2}\left(\mathrm{Proj}_{Q_1}(x_k) + \mathrm{Proj}_{Q_2}(x_k)\right),$$

3. *Dykstra: initialize $p_0 = q_0 = 0$ and iterate*

$$\begin{aligned} y_k &= \mathrm{Proj}_{Q_1}(x_k + p_k), \\ p_{k+1} &= x_k + p_k - y_k, \\ x_{k+1} &= \mathrm{Proj}_{Q_2}(y_k + q_k), \\ q_{k+1} &= y_k + q_k - x_{k+1}. \end{aligned}$$

Correction: See this notebook. The bounds should behave in $O(N^{-1})$.

Credits Detailed results for Exercise 2 (closed-form worst-case convergence rates for gradient descent in different performance measures) can be found in [2]; the analyses from Exercise 7 can be found in [15]; the closed-form worst-case analysis of the proximal minimization algorithm from Exercise 9 can be found in [19]; the worst-case analysis for the proximal point method in the context of monotone inclusions can be found in [14]; the analysis from Exercise 10 can be found in [18].

4 Slightly more advanced techniques

This section will be completed shortly. We currently only provide a few pointers to some relevant literature. The corresponding exercises will be provided in an upcoming version of this document.

Exercise 12 (Lyapunov analyses—linear convergence) *How to search for Lyapunov functions guaranteeing linear convergence? See, e.g., [20, 21].*

Exercise 13 (Lyapunov analyses—sublinear convergence) *How to verify Lyapunov functions guaranteeing sublinear convergence? See, e.g., [22].*

Exercise 14 (Designing methods—via line-searches) *How to design new methods using "idealistic methods" involving line-searches or span-searches? See, e.g., [17] or [23] for a simple example.*

Exercise 15 (Designing methods—by optimizing stepsizes) *How to design new methods by optimizing their stepsize coefficients? See, e.g., [24, 25, 26, 27, 28] (using convex relaxations), or the more recent [29] (using a more "to-the-point" approach, i.e., more heavy-duty nonlinear optimization).*

Exercise 16 (Relaxations and upper bounds) *How to study algorithms on problem classes for which we don't have "interpolation/extensions" theorems? See, e.g., [30] for a setting where no known interpolation theorem holds.*

Further pointers Mirror descent [4], decentralized/distributed optimization [31, 32, 33, 34], continuous-time approaches [35, 36], approximate first-order information [2, 37, 38], stochasticity [39, 22, 40] and adaptivity [37, 41] will also be part of future versions.

5 Background material and useful facts

5.1 Standard definitions

All convex functions under consideration in this exercise statements are closed and proper per assumption (i.e., they have a closed and non-empty epigraph).

Definition 1 *A differentiable convex function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth (with $L \in [0, \infty)$; we denote $f \in \mathcal{F}_{0,L}$) if it satisfies $\forall x, y \in \mathbb{R}^d$: $\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|$.*

Definition 2 *A convex differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth and μ-strongly convex ($0 \leqslant \mu < L < \infty$; we denote $f \in \mathcal{F}_{\mu,L}$) if it satisfies $\forall x, y \in \mathbb{R}^d$: $\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|$ and $\|\nabla f(x) - \nabla f(y)\| \geqslant \mu\|x - y\|$.*

Definition 3 *Let $Q \subseteq \mathbb{R}^d$ be a non-empty closed convex set. The convex indicator function for Q is defined as*

$$i_Q(x) \triangleq \begin{cases} 0 & \text{if } x \in Q, \\ +\infty & \text{otherwise.} \end{cases}$$

Definition 4 *A closed proper convex function $f : \mathbb{R}^d \to \mathbb{R}$ is M-Lipschitz (with $M \in [0, \infty)$; notation $f \in \mathcal{C}_M$) if it satisfies $\forall x \in \mathbb{R}^d$ and $s_x \in \partial f(x)$: $\|s_x\| \leqslant M$.*

Definition 5 *A point to set operator M (notation: $M : \mathbb{R}^d \rightrightarrows \mathbb{R}^d$) is monotone if for all $x, y \in \mathbb{R}^d$ and all $m_x \in M(x)$, $m_y \in M(y)$ it holds that*

$$\langle x - y, m_x - m_y \rangle \geqslant 0.$$

In addition, M is maximal if there is no monotone operator M' such that for all x: $M(x) \subset M'(x)$.

Definition 6 *A mapping $F : \mathbb{R}^d \to \mathbb{R}^d$ is L-Lipschitz if for all $x, y \in \mathbb{R}^d$*

$$\|F(x) - F(y)\| \leqslant L\|x - y\|.$$

5.2 Interpolation/extension theorems

This section gathers useful elements allowing to answer certain questions. Those results (or references to them) can be found in, e.g., [2, 42].

Theorem 1 *Let I be an index set and $S = \{(x_i, g_i, f_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$ be a set of triplets. There exists $f \in \mathcal{F}_{0,\infty}$ (a closed, proper and convex function) satisfying $f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for all $i \in I$ if and only if*

$$f_i \geqslant f_j + \langle g_j; x_i - x_j \rangle$$

holds for all $i, j \in I$.

Theorem 2 ($\mathcal{F}_{\mu,L}$-interpolation) *Let I be an index set and $S = \{(x_i, g_i, f_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$ be a set of triplets. There exists $f \in \mathcal{F}_{\mu,L}$ satisfying $f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for all $i \in I$ if and only if*

$$f_i \geqslant f_j + \langle g_j; x_i - x_j \rangle + \frac{1}{2L}\|g_i - g_j\|^2 + \frac{\mu}{2(1 - \mu/L)}\|x_i - x_j - \tfrac{1}{L}(g_i - g_j)\|^2$$

holds for all $i, j \in I$.

Theorem 3 (Indicator-interpolation) *Let I be an index set and $S = \{(x_i, s_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d$ be a set of pairs. There exists a (closed proper convex) indicator function $i_Q : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ (for a non-empty closed domain $Q \subseteq \mathbb{R}^d$ of diameter D) satisfying $s_i \in \partial i_Q(x_i)$ for all $i \in I$ if and only if*

$$\langle s_j; x_i - x_j \rangle \leqslant 0$$
$$\|x_i - x_j\|^2 \leqslant D^2,$$

holds for all $i, j \in I$.

Theorem 4 ($\mathcal{C}_M$-interpolation) *Let I be an index set and $S = \{(x_i, g_i, f_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$ be a set of triplets. There exists $f \in \mathcal{C}_M$ (a closed, proper, convex, and M-Lipschitz function) satisfying $f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for all $i \in I$ if and only if*

$$\begin{aligned} f_i &\geqslant f_j + \langle g_j; x_i - x_j\rangle \\ M^2 &\geqslant \|g_i\|^2 \end{aligned}$$

holds for all $i, j \in I$.

Theorem 5 (Monotone-interpolation) *Let I be an index set and $S = \{(x_i, s_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d$ be a set of pairs. There exists a maximal monotone operator $M : \mathbb{R}^d \rightrightarrows \mathbb{R}^d$ satisfying $s_i \in M(x_i)$ for all $i \in I$ if and only if*

$$\langle s_j - s_i; x_j - x_i\rangle \geqslant 0$$

holds for all $i, j \in I$.

Theorem 6 (Lipschitz-interpolation) *Let I be an index set and $S = \{(x_i, s_i)\}_{i\in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d$ be a set of pairs. There exists an L-Lipschitz mapping $F : \mathbb{R}^d \to \mathbb{R}^d$ satisfying $s_i = F(x_i)$ for all $i \in I$ if and only if*

$$\|s_i - s_j\|^2 \leqslant L^2\|x_i - x_j\|^2$$

holds for all $i, j \in I$.

References

[1] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: a novel approach. *Mathematical Programming*, 145(1-2):451–482, 2014.

[2] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Exact worst-case performance of first-order methods for composite convex optimization. *SIAM Journal on Optimization*, 27(3):1283–1313, 2017.

[3] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming*, 161(1-2):307–345, 2017.

[4] R.-A. Dragomir, A.B. Taylor, A. d'Aspremont, and J. Bolte. Optimal complexity and certification of bregman first-order methods. *Mathematical Programming*, pages 1–43, 2021.

[5] B. Goujaud, C. Moucer, F. Glineur, J. Hendrickx, A. Taylor, and A. Dieuleveut. PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python. *preprint arXiv:2201.04040*, 2022.

[6] A. Taylor, J. Hendrickx, and F. Glineur. Performance Estimation Toolbox (PESTO): automated worst-case analysis of first-order optimization methods. In *Proceedings of the 56th IEEE Conference on Decision and Control (CDC 2017)*, 2017.

[7] APS Mosek. The MOSEK optimization software. *Online at http://www.mosek.com*, 54, 2010.

[8] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[9] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, 2004.

[10] A. Meurer, C. P. Smith, M. Paprocki, O. Čertík, S. B. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. K. Moore, S. Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, 2017.

[11] M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 3, pages 2156–2162. IEEE, 2003.

[12] Y. Drori. *Contributions to the Complexity Analysis of Optimization Algorithms.* PhD thesis, Tel-Aviv University, 2014.

[13] M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning (ICML)*, pages 427–435, 2013.

[14] G. Gu and J. Yang. Tight sublinear convergence rate of the proximal point algorithm for maximal monotone inclusion problems. *SIAM Journal on Optimization*, 30(3):1905–1921, 2020.

[15] F. Lieder. On the convergence rate of the Halpern-iteration. *Optimization Letters*, 15(2):405–418, 2021.

[16] Y. Nesterov and V. Shikhman. Quasi-monotone subgradient methods for nonsmooth convex minimization. *Journal of Optimization Theory and Applications*, 165(3):917–940, 2015.

[17] Y. Drori and A.B. Taylor. Efficient first-order methods for convex minimization: a constructive approach. *Mathematical Programming*, 184(1):183–220, 2020.

[18] P. Giselsson and S. Boyd. Linear convergence and metric selection for Douglas-Rachford splitting and ADMM. *IEEE Transactions on Automatic Control*, 62(2):532–544, 2016.

[19] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Exact worst-case convergence rates of the proximal gradient method for composite convex minimization. *Journal of Optimization Theory and Applications*, 178(2):455–476, 2018.

[20] L. Lessard, B. Recht, and A. Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

[21] A. Taylor, B. Van Scoy, and L. Lessard. Lyapunov functions for first-order methods: Tight automated convergence guarantees. In *International Conference on Machine Learning (ICML)*, 2018.

[22] A. Taylor and F. Bach. Stochastic first-order methods: non-asymptotic and computer-aided analyses via potential functions. In *Conference on Learning Theory (COLT)*, 2019.

[23] B. Goujaud, A. Taylor, and A. Dieuleveut. Optimal first-order methods for convex functions with a quadratic upper bound. *preprint arXiv:2205.15033*, 2022.

[24] D. Kim and J. A. Fessler. Optimized first-order methods for smooth convex minimization. *Mathematical Programming*, 159(1-2):81–107, 2016.

[25] D. Kim and J.A. Fessler. Optimizing the efficiency of first-order methods for decreasing the gradient of smooth convex functions. *Journal of Optimization Theory and Applications*, 188(1):192–219, 2021.

[26] B. Van Scoy, R. A. Freeman, and K. M. Lynch. The fastest known globally convergent first-order method for minimizing strongly convex functions. *IEEE Control Systems Letters*, 2(1):49–54, 2018.

[27] D. Kim. Accelerated proximal point method for maximally monotone operators. *Mathematical Programming*, pages 1–31, 2021.

[28] A. Taylor and Y. Drori. An optimal gradient method for smooth strongly convex minimization. *Mathematical Programming*, pages 1–38, 2022.

[29] S. Das Gupta, B.P.G. Van Parys, and E. K. Ryu. Branch-and-bound performance estimation programming: A unified methodology for constructing optimal optimization methods. *preprint arXiv:2203.07305*, 2022.

[30] E. Gorbunov, A. Taylor, and G. Gidel. Last-iterate convergence of optimistic gradient method for monotone variational inequalities. *preprint arXiv:2205.08446*, 2022.

[31] A. Sundararajan, B. Hu, and L. Lessard. Robust convergence analysis of distributed optimization algorithms. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1206–1212, 2017.

[32] A. Sundararajan, B. Van Scoy, and L. Lessard. Analysis and design of first-order distributed optimization algorithms over time-varying graphs. *IEEE Transactions on Control of Network Systems*, 7(4):1597–1608, 2020.

[33] S. Colla and J. M. Hendrickx. Automated worst-case performance analysis of decentralized gradient descent. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2627–2633, 2021.

[34] S. Colla and J. M. Hendrickx. Automatic performance estimation for decentralized optimization. *preprint arXiv:2203.05963*, 2022.

[35] M. Fazlyab, A. Ribeiro, M. Morari, and V.M. Preciado. Analysis of optimization algorithms via integral quadratic constraints: Nonstrongly convex problems. *SIAM Journal on Optimization*, 28(3):2654–2689, 2018.

[36] C. Moucer, A. Taylor, and F. Bach. A systematic approach to lyapunov analyses of continuous-time models in convex optimization. *preprint arXiv:2205.12772*, 2022.

[37] E. de Klerk, F. Glineur, and A. B. Taylor. On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions. *Optimization Letters*, 11(7):1185–1199, 2017.

[38] M. Barré, A. Taylor, and F. Bach. Principled analyses and design of first-order methods with inexact proximal operators. *preprint arXiv:2006.06041*, 2020.

[39] B. Hu, P. Seiler, and A. Rantzer. A unified analysis of stochastic optimization methods using jump system theory and quadratic constraints. In *Conference on Learning Theory (COLT)*, 2017.

[40] B. Hu, P. Seiler, and L. Lessard. Analysis of biased stochastic gradient descent using sequential semidefinite programs. *Mathematical Programming*, 187(1):383–408, 2021.

[41] M. Barré, A. Taylor, and A. d'Aspremont. Complexity guarantees for Polyak steps with momentum. In *Conference on Learning Theory (COLT)*, 2020.

[42] E.K. Ryu, A.B. Taylor, C. Bergeling, and P. Giselsson. Operator splitting performance estimation: Tight contraction factors and optimal parameter selection. *SIAM Journal on Optimization*, 30(3):2251–2271, 2020.