

Worst-case analyses for first-order optimization methods

Adrien Taylor*, Baptiste Goujaud†

Current version: July 7, 2022

1	Introduction	2
2	Getting familiar with base performance estimation problems	2
3	Further exercises	10
4	Background material and useful facts	13
4.1	Standard definitions	13
4.2	Interpolation/extension theorems	13

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, whereas Section 3 contains exercises for going further. Background material that might be used for the exercises is provided in Section 4.

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, \dots, x_N . There are typically different ways of doing so, which might or might not be relevant 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.

So, our goal is to assess the quality of x_k by providing hopefully meaningful upper bounds on at least one of those quantities. 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 exemplifying 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 $\langle \cdot, \cdot \rangle : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ the standard Euclidean inner product, and by $\|\cdot\|^2 : \mathbb{R}^d \rightarrow \mathbb{R}$ the standard Euclidean norm, that is, the induced norm: for any $x \in \mathbb{R}^d$: $\|x\|^2 = \langle x, x \rangle$. 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 [4] (alternatively, for Matlab, PESTO [5]). For users planning to keep using performance estimation, we advise to install a good SDP solver such as MOSEK [6]. In any case, the Python package relies on CVXPY [7] (see [PEPit installation](#)) and the Matlab one relies on YALMIP [8] (see [PESTO installation](#)). Some exercises might be simpler to approach using some symbolic computations, such as Sympy [9].

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 exemplified 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 step sizes $\{\gamma_k\}_k$; 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

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

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

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

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

2. Show that

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

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

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

4. Using the change of variables

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

(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

$$\begin{aligned} \sup_{G, F} \frac{G_{1,1} + \gamma_k^2 G_{2,2} - 2\gamma_k G_{1,2}}{G_{1,1}} \\ \text{s.t. } F + \frac{L\mu}{2(L-\mu)} G_{1,1} + \frac{1}{2(L-\mu)} G_{2,2} - \frac{L}{L-\mu} G_{1,2} \leq 0 \\ -F + \frac{L\mu}{2(L-\mu)} G_{1,1} + \frac{1}{2(L-\mu)} G_{2,2} - \frac{\mu}{L-\mu} G_{1,2} \leq 0 \\ G \succcurlyeq 0. \end{aligned} \quad (6)$$

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

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

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

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

(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 .)

6. Let $L \geq \mu > 0$ 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).
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 .
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.
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?
10. Update your code for computing (numerically) the worst-case ratio $\frac{f(x_{k+1}) - f(x_*)}{f(x_k) - f(x_*)}$ 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?
11. Update your code for computing (numerically) the worst-case ratio $\frac{\|x_N - x_*\|^2}{\|x_0 - x_*\|^2}$ as a function of L , μ , and a sequence $\{\gamma_k\}_{0 \leq k \leq N-1}$. How would you update the SDP formulation for taking this change into account?
12. Using previous points: assume you computed an optimal solution to the SDP formulation for the ratio $\frac{\|x_N - x_*\|^2}{\|x_0 - x_*\|^2}$ what is the link between the rank of the Gram matrix G and the dimension d in which this counter-example lives?
13. For obtaining “low-dimensional” worst-case instances, it is often useful to use heuristics. One of them is known as the “trace heuristic”, which consists in solving a second optimization problem, whose solution hopefully forces G to have a lower rank. For (7), it consists in solving

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

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?

-
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 ?
 15. Same question with $\gamma_k = \frac{1}{L}$, $\mu = 0$, and the ratio $\frac{\|\nabla f(x_N)\|^2}{\|x_0 - x_\star\|^2}$.
 16. Same question with $\gamma_k = \frac{1}{L}$, $\mu = 0$, and the ratio $\frac{\|x_N - x_\star\|^2}{\|\nabla f(x_0)\|^2}$.
 17. Based on your current experience, what are, according to you, the key elements which allowed casting the worst-case analysis as an SDP?

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 do on Exercise 3.

The goal of the following 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.

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):

$$\begin{aligned} F + \frac{L\mu}{2(L-\mu)}G_{1,1} + \frac{1}{2(L-\mu)}G_{2,2} - \frac{L}{L-\mu}G_{1,2} &\leq 0 & : \lambda_1 \\ -F + \frac{L\mu}{2(L-\mu)}G_{1,1} + \frac{1}{2(L-\mu)}G_{2,2} - \frac{\mu}{L-\mu}G_{1,2} &\leq 0 & : \lambda_2 \\ G_{1,1} &= 1 & : \tau \end{aligned}$$

one can show that

$$\begin{aligned} \tau(\mu, L, \gamma_k) &= \min_{\tau, \lambda_1, \lambda_2 \geq 0} \tau \\ \text{s.t. } S &= \begin{bmatrix} \tau - 1 + \frac{\lambda_1 L \mu}{L - \mu} & \gamma_k - \frac{\lambda_1(\mu + L)}{2(L - \mu)} \\ \gamma_k - \frac{\lambda_1(\mu + L)}{2(L - \mu)} & \frac{\lambda_1}{L - \mu} - \gamma_k^2 \end{bmatrix} \succcurlyeq 0 \\ 0 &= \lambda_1 - \lambda_2. \end{aligned} \tag{9}$$

Note that equility holds due to strong duality (for going further: obtain this dual formulation and prove strong duality using a Slater condition).

What is the relationship between feasible points to $(\tau, \lambda_1, \lambda_2)$ to (10) and $\tau(\mu, L, \gamma_k)$?

2. Solve (10) numerically. Do those values match those obtained in the first exercise?

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

3. Is there a simple closed-form expression for $\tau(\mu, L, \gamma_k)$? You can complete the same notebook as in the previous exercise (alternative: [Matlab code](#)).

Does it match the numerical values obtained using the previous codes for computing $\tau(\mu, L, \gamma_k)$ numerically?

4. Consider $\gamma_k = \frac{1}{L}$ for simplicity. Given the optimal value of the multipliers $\tau, \lambda_1, \lambda_2$ in (10), can you write a “direct” proof for the linear convergence in terms of distance to an optimal point without resorting on any SDP formulation?

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

$$\begin{aligned} \min_{\tau, \lambda_1, \lambda_2 \geq 0} \quad & \tau \\ \text{s.t.} \quad & S = \begin{bmatrix} \tau + \lambda_1 \frac{(1-\gamma_k L)(1-\gamma_k \mu)}{L-\mu} & -\lambda_1 \frac{2-\gamma_k(L+\mu)}{2(L-\mu)} \\ -\lambda_1 \frac{2-\gamma_k(L+\mu)}{2(L-\mu)} & \frac{\lambda_1}{L-\mu} - 1 \end{bmatrix} \succcurlyeq 0 \\ & 0 = \lambda_1 - \lambda_2. \end{aligned} \quad (10)$$

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

6. Using the following primal-dual pairing

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

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

$$\begin{aligned} \min_{\tau, \lambda_1, \lambda_2 \geq 0} \quad & \tau \\ \text{s.t.} \quad & \begin{bmatrix} \frac{\mu L(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)}{L-\mu} & \frac{-L(\lambda_2 + \gamma\mu(\lambda_3 + \lambda_4)) + \mu\lambda_1}{L-\mu} & \frac{-L\lambda_4 + \mu\lambda_3}{L-\mu} \\ * & \frac{\gamma\mu(\gamma L(\lambda_3 + \lambda_4 + \lambda_5 + \lambda_6) - 2\lambda_5) - 2\gamma L\lambda_6 + \lambda_1 + \lambda_2 + \lambda_5 + \lambda_6}{L-\mu} & \frac{\gamma L\lambda_4 + \lambda_5(\gamma L - 1) + \gamma\mu(\lambda_3 + \lambda_6) - \lambda_6}{L-\mu} \\ * & * & \frac{\lambda_3 + \lambda_4 + \lambda_5 + \lambda_6}{L-\mu} \end{bmatrix} \succcurlyeq 0 \\ & 0 = \tau - \lambda_1 + \lambda_2 - \lambda_5 + \lambda_6 \\ & 1 = -\lambda_3 + \lambda_4 + \lambda_5 - \lambda_6, \end{aligned} \quad (11)$$

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](#))

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

3 Further exercises

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 algorithms, which respectively iterate:

- a gradient method:

$$x_{k+1} = x_k - \frac{1}{L} \nabla f(x_k),$$

- an heavy-ball method:

$$x_{k+1} = x_k + \frac{k}{k+2}(x_k - x_{k-1}) - \frac{1}{k+2} \frac{1}{L} \nabla f(x_k),$$

- an accelerated (or fast) gradient method:

$$\begin{aligned} x_{k+1} &= y_k - \frac{1}{L} \nabla f(y_k) \\ y_{k+1} &= x_{k+1} + \frac{k-1}{k+2}(x_{k+1} - x_k) \end{aligned}$$

1. Consider the following ratios:

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

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

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?

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 $N = 0, 1, \dots, 30$).

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):

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

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 ingredient for arriving to it?

2. Provide a PEPit or PESTO code for computing those ratios, and experiment with it.

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

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.

Exercise 5 (Frank-Wolfe and projected gradient) 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., $\|x - y\| \leq D$ for all $x, y \in Q$). We consider two first-order methods:

- projected gradient:

$$x_{k+1} = \operatorname{Proj}_D [x_k - \frac{1}{L} \nabla f(x_k)]$$

- conditional gradient (a.k.a., Frank-Wolfe):

$$y_k = \operatorname{argmin}_{y \in Q} \langle \nabla f(x_k), y \rangle$$

$$x_{k+1} = \frac{k}{k+2} x_k + \frac{1}{k+2} y_k.$$

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).

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

-
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 ?

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) monotone operator (see Definition 4). In this context, the proximal point method consists in iterating

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

(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?
2. Write a PEPit or PESTO code for studying this ratio as a function of N and α .
3. Using dimension reduction, there should exist a low-dimensional worst-case example. Find it and plot it.

More information about the worst-case guarantee and a low-dimensional worst-case example can be found in [10] (see [arXiv](#)). Are your observations compatible with the fact that M might encode a rotation matrix?

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

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 SDP formulations for studying the ratio $\frac{\|x_N - F(x_N)\|^2}{\|x_0 - x_\star\|^2}$ (for some $x_\star = F(x_\star)$)?

2. Can you find low-dimensional worst-case examples for this method? (Hint: use the logdet heuristic (implemented within PEPit and described in [11] (see [pdf](#)))
3. Can you guess the dependence in N of the worst-case behavior of the previous ratio?
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?

Exercise 8 (Proximal gradient method) We consider the composite convex minimization problem of the form

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

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

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

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; how do you arrive to such a formulation? Write a code for computing it, and compare the numerical values with those of Exercise 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 closed-form worst-case analysis of the proximal minimization algorithm can be found in [12]; the worst-case analysis for the proximal point method in the context of monotone inclusions can be found in [10]; and the analyses from Exercise 7 can be found in [13]

4 Background material and useful facts

4.1 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 $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is L -smooth (with $L \in [0, \infty)$) if it satisfies $\forall x, y \in \mathbb{R}^d$: $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$.

Definition 2 A convex differentiable function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is L -smooth and μ -strongly convex ($0 \leq \mu < L < \infty$) if it satisfies $\forall x, y \in \mathbb{R}^d$: $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ and $\|\nabla f(x) - \nabla f(y)\| \geq \mu\|x - y\|$.

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 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

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

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

Definition 5 A mapping $F : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is L -Lipschitz if for all $x, y \in \mathbb{R}^d$

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

4.2 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, ?].

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

$$f_i \geq f_j + \langle g_j, x_i - x_j \rangle$$

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

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

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 $i_Q : \mathbb{R}^d \rightarrow \mathbb{R} \cup \{\infty\}$ (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

$$\begin{aligned} \langle s_j; x_i - x_j \rangle &\leq 0 \\ \|x_i - x_j\|^2 &\leq D^2, \end{aligned}$$

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

Theorem 4 (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

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

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

Theorem 5 (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 $F : \mathbb{R}^d \rightarrow \mathbb{R}^d$ satisfying $s_i = F(x_i)$ for all $i \in I$ if and only if

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

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] 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. *arXiv preprint 2201.04040*, 2022.
- [5] 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.
- [6] APS Mosek. The MOSEK optimization software. *Online at <http://www.mosek.com>*, 54, 2010.
- [7] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.
- [8] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, 2004.
- [9] 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.
- [10] 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.

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