



**HAL**  
open science

## **Chance-constrained programs with convex underlying functions: a bilevel convex optimization perspective**

Yassine Laguel, Jérôme Malick, Wim Van Ackooij

► **To cite this version:**

Yassine Laguel, Jérôme Malick, Wim Van Ackooij. Chance-constrained programs with convex underlying functions: a bilevel convex optimization perspective. *Computational Optimization and Applications*, 2024, 88, pp.819-847. <10.1007/s10589-024-00573-9>. <hal-03174589v2>

**HAL Id: hal-03174589**

**<https://hal.science/hal-03174589v2>**

Submitted on 16 Nov 2025

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization

# Chance-constrained programs with convex underlying functions: a bilevel convex optimization perspective

Yassine Laguel\*

Univ. Côte d’Azur, LJAD

Jérôme Malick

Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK

Wim van Ackooij

EDF R&D

## Abstract

Chance constraints are a valuable tool for the design of safe decisions in uncertain environments; they are used to model satisfaction of a constraint with a target probability. However, because of possible non-convexity and non-smoothness, optimizing over a chance constrained set is challenging. In this paper, we consider chance constrained programs where the objective function and the constraints are convex with respect to the decision parameter. We establish an exact reformulation of such a problem as a bilevel problem with a convex lower-level. Then we leverage this bilevel formulation to propose a tractable penalty approach, in the setting of finitely supported random variables. The penalized objective is a difference-of-convex function that we minimize with a suitable bundle algorithm. We release an easy-to-use open-source python toolbox implementing the approach, with a special emphasis on fast computational subroutines.

Stochastic programming Bilevel optimization DC programming Chance constraints Convex optimization

## 1 Introduction

Chance constraints appear as a versatile way to model the exposure to uncertainty in optimization. Introduced in [7], they have been used in many applications, such as in energy management [33, 39], in telecommunications [25] or for reinforcement learning [8], to name of few of them. We refer to the seminal book [32], the book chapter [10] for introduction to the theory and to the recent article [38] for a discussion covering recent developments.

We consider a chance-constrained optimization problem of the following form. For a fixed safety probability level  $p \in [0, 1)$ , we write

$$\begin{aligned} \min_{x \in \mathcal{X}} \quad & f(x) \\ \text{s.t.} \quad & \mathbb{P}[g(x, \xi) \leq 0] \geq p, \end{aligned} \tag{1}$$

---

\*Corresponding author : [yassine.laguel@univ-cotedazur.fr](mailto:yassine.laguel@univ-cotedazur.fr)

where  $f: \mathbb{R}^d \rightarrow \mathbb{R}$  and  $g: \mathbb{R}^d \times \mathbb{R}^m \rightarrow \mathbb{R}$  are two given functions,  $\xi$  is a random vector valued in  $\mathbb{R}^m$  and  $\mathcal{X} \subset \mathbb{R}^d$  is a closed constraint set. We make the following blanket assumptions:

(A1)  $f$  and  $g$  are convex with respect to  $x$ ,

(A2) for all  $x \in X$ ,  $|\mathbb{E}(g(x, \xi))| < \infty$ .

We consider the case of underlying convexity: we assume that  $f$  and  $g$  are convex (with respect to  $x$ ). Even with this underlying convexity, uncertainty may make the chance constrained set non-convex (see e.g. [14]) Though solving chance-constrained problems is difficult, several computational methods have been proposed, regardless of any considerations of convexity and smoothness, and under various assumptions on uncertainty. Let us mention: sample average approximation [30, 23], scenario approximation [6], convex approximation [26, 17],  $p$ -efficient points [11], quantile-based reformulation [31] or computation of the efficient frontier [18] ; see e.g. [38] for an overview.

In this paper, we propose an original approach for solving chance-constrained optimization problems. First, we present an exact reformulation of (nonconvex) chance-constrained problems as (convex) bilevel optimization problems. This reformulation is simple and natural, involving superquantiles (also called conditional value-at-risk); see e.g., the tutorial [34]. Second, exploiting this bilevel reformulation, we propose a general algorithm for solving chance-constrained problems, and we release an open-source python toolbox implementing it. In the case where we make no assumption on the underlying uncertainty and have only samples of  $\xi$ , we propose and analyze a double penalization method, leading to an unconstrained single level DC (Difference of Convex) program. Thus our work mixes a variety of techniques coming from different subdomains of optimization: penalization, error bounds, DC programming, bundle algorithm, Nesterov's smoothing; relevant references are given along the presentation. The resulting algorithm enables to deal with a fairly large sample of data-points in comparison with state-of-the-art methods based on mixed-integer reformulations, e.g. [2].

This paper is structured as follows. In Section 2, we leverage the known link with (super)quantiles and chance-constraint to establish a novel bilevel reformulation of general chance constrained problems. In Section 3, we propose and analyze a penalty approach revealing the underlying DC structure. In Section 4, we discuss the implementation of this approach in a publicly available toolbox. In Section 5, we provide illustrative numerical experiments, as a proof of concept, showing the interest of the method. Technical details on secondary theoretical points and on implementation issues are postponed to appendices.

## 2 Chance constrained problems seen as bilevel problems

In this section, we derive the reformulation of a chance constraint as a bilevel program wherein both the upper and lower level problems, when taken individually, are convex. We first recall in Section 2.1 useful definitions. Our terminology and notations closely follow those of [34].

### 2.1 Recalls: cumulative distributions functions, quantiles, and superquantiles

In what follows, We consider a probability space and integrable real random variables. Given a random variable  $X$ , its *cumulative distribution function* is

$$F_X(t) = \mathbb{P}[X \leq t] \quad \forall t \in \mathbb{R}. \quad (2)$$

This function is known to be non-decreasing and right-continuous. Its jumps occur exactly at the values  $t \in \mathbb{R}$  at which  $\mathbb{P}[X = t] > 0$ . These properties allow one to define the *quantile function*  $p \mapsto Q_p(X)$  as the generalized inverse:

$$Q_p(X) = \inf\{t \in \mathbb{R} : F_X(t) \geq p\}, \quad \forall p \in [0, 1]. \quad (3)$$

If  $X$  is assumed to belong to  $\mathcal{L}^1$ , we can additionally define for any  $p \in [0, 1)$  its  *$p$ -superquantile* (also called *conditional value-at-risk*),  $\bar{Q}_p(X)$  as follows:

$$\bar{Q}_p(X) = \frac{1}{1-p} \int_{p'=p}^1 Q_{p'}(X) dp'. \quad (4)$$

As a consequence of [35, Th. 2], one can get from  $F_X$ , both the  $p$ -quantile and the  $p$ -superquantile functions as functions of  $p$  and reciprocally, knowing either the  $p$ -quantile (or the  $p$ -superquantile) for all  $p \in [0, 1]$  suffices to recover  $F_X$ .

From a statistical viewpoint, these three objects are also equally consistent [34, Th. 4] in the sense that convergence in distribution for a sequence of random variables  $(X_n)_{n \geq 0}$  is equivalent to the pointwise convergence of the two sequences of functions  $p \mapsto Q_p(X_n)$ ,  $p \mapsto \bar{Q}_p(X_n)$ . This result is particularly relevant when the distributions are observed through data sampling. We can use the empirical cumulative distribution functions, quantiles and super-quantiles all while upholding asymptotic convergence as the sample size grows.

From an optimization viewpoint though, these three objects are very different. In contrast with the two others, the superquantile has several good properties (including convexity [4, 13, 37]), useful with respect to numerical computation and optimization. In our developments, we use the following key result [36, Th. 1] linking quantiles and superquantile through a one-dimensional problem. For an integrable random variable  $X$  and a probability level  $p$ , the superquantile  $\bar{Q}_p(X)$  is the optimal value of the convex problem

$$\inf_{\eta \in \mathbb{R}} \eta + \frac{1}{1-p} \mathbb{E}[\max(X - \eta, 0)]. \quad (5)$$

Moreover, the quantile  $Q_p(X)$  is the left end-point of the solution interval.

Note finally that we consider in this paper with a single inequality system, but the generality of our approach allows to capture several constraint functions with the usual trick formalized in the next remark. [Joint chance constraints] The extension of our approach to joint chance-constrained problems is straightforward with the usual trick: a chance constraint with  $h : \mathbb{R}^d \times \mathbb{R}^m \rightarrow \mathbb{R}^k$  such that all its components  $h_i$  ( $1 \leq i \leq k$ ) are convex with respect to  $x$  can be written with convex function  $g(x, \xi) = \max_{1 \leq i \leq k} h_i(x, \xi)$ , since

$$\mathbb{P}[h(x, \xi) \leq 0] \geq p \iff \mathbb{P}\left[\max_{1 \leq i \leq k} h_i(x, \xi) \leq 0\right] \geq p.$$

## 2.2 Reformulation as a bilevel problem

By definition, the chance constraint in (1) involves the cumulative distribution function: we have, for  $x \in \mathbb{R}^d$ ,  $\mathbb{P}[g(x, \xi) \leq 0] \geq p \iff F_{g(x, \xi)}(0) \geq p$ . Following the discussion of the previous section, we easily rewrite this constraint using quantiles, as formalized in the next lemma. This result is well known (see e.g. [26]) and has been used recently in e.g. [31]. We provide here a short proof for completeness. For any  $x \in \mathbb{R}^d$  and  $p \in [0, 1)$ , we have:

$$\mathbb{P}[g(x, \xi) \leq 0] \geq p \iff Q_p(g(x, \xi)) \leq 0.$$

By definition of the quantile and the right-continuity of the cumulative function, we have  $p \leq \mathbb{P}[g(x, \xi) \leq Q_p(g(x, \xi))]$ . Since the cumulative function is increasing,  $Q_p(g(x, \xi)) \leq 0$  implies  $\mathbb{P}[g(x, \xi) \leq Q_p(g(x, \xi))] \leq \mathbb{P}[g(x, \xi) \leq 0]$  which implies in turn  $p \leq \mathbb{P}[g(x, \xi) \leq 0]$ . Conversely, since  $Q_p(g(x, \xi))$  is the infimum of  $\{t \in \mathbb{R} : \mathbb{P}[g(x, \xi) \leq t] \geq p\}$ , if  $Q_p(g(x, \xi)) > 0$ , then it holds  $\mathbb{P}[g(x, \xi) \leq 0] < p$ .

Compared to the empirical cumulative distribution function, the  $p$ -quantile function  $x \mapsto Q_p(g(x, \xi))$  is continuous with respect to  $x$ , whenever  $g$  is. However, it remains non-convex and non-smooth in general. So we propose to use yet another reformulation in terms of superquantiles, as follows.

Together with (5), we obtain from the previous lemma a bilevel formulation of the general chance-constrained problem (7). The idea is simple: introducing an auxiliary variable  $\eta \in \mathbb{R}^d$  to recast the potentially non-convex chance constraint of (1) as two constraints, a simple bound constraint and a difficult optimality constraint, forming a lower subproblem. Introducing the jointly convex function  $G : \mathcal{X} \times \mathbb{R} \rightarrow \mathbb{R}$

$$G(x, s) = s + \frac{1}{1-p} \mathbb{E}[\max(g(x, \xi) - s, 0)], \quad (6)$$

we indeed have the following exact reformulation.

Under assumptions (A1) and (A2), Problem (1) is equivalent to the bilevel problem:

$$\begin{cases} \min_{x \in \mathcal{X}, \eta \in \mathbb{R}} & f(x) \\ \text{s.t.} & \eta \leq 0 \\ & \eta \in S(x) = \arg \min_{s \in \mathbb{R}} G(x, s). \end{cases} \quad (7)$$

More precisely, if  $x^*$  is an optimal solution of (1), then  $(x^*, Q_p(g(x^*, \xi)))$  is an optimal solution of the above bilevel problem, and conversely.

By Lemma 2.2, we have that (1) is equivalent to

$$\begin{cases} \min_{x \in \mathcal{X}, \eta \in \mathbb{R}} & f(x) \\ \text{s.t.} & \eta \leq 0 \\ & \eta = Q_p(g(x, \xi)). \end{cases} \quad (8)$$

By Lemma 2.1,  $Q_p(g(x, \xi)) \in S(x)$  for any  $x \in \mathbb{R}^d$ . Hence, any solution  $(x, \eta)$  of (8) is feasible for (7). Conversely, any solution  $(x^*, \eta^*)$  of (7) satisfies:  $Q_p(g(x^*, \xi)) \leq \eta^* \leq 0$  which implies that  $(x^*, Q_p(g(x^*, \xi)))$  is a feasible point of (8). Since both problems have the same objective, they are equivalent.

The first constraint  $\eta \leq 0$  is an easy one-dimensional bound constraint which does not involve the decision variable  $x$ . The second constraint, which constitutes the lower level problem is more difficult; when this constraint is satisfied,  $\eta$  is an upper-bound on the  $p$ -quantile of  $g(x, \xi)$ .

This bilevel reformulation is nice, natural, and seemingly new; we believe that it opens the door to new approaches for solving chance-constrained problems with underlying convexity. We propose such an approach, in the next section.

**Bilevel formulation vs sampled approximation** Let us emphasize that Theorem 2.2 provides an exact reformulation of the problem (1). This is in contrast with sampling methods, such as [6], which approximate the chance constraint through sampling. Note also that, though [6] provides theoretical guarantees, it also presents limitations when the sampled constraint remains infeasible,

as explained in [6, Section 4.2]. This infeasibility for the sampled problem occurs frequently in practice (see e.g. [1]), as well as in the illustrative examples of Section 5.

[Bilevel formulation vs bilevel approximation] Our bilevel formulation is exact, as opposed to the recent bilevel approach of [17]. This paper proposes indeed a reformulation that approximates the chance constraint with better guarantees than the CVaR approximation from [26]. In contrast, we leverage here the variational properties of the CVaR to enforce exactly the probabilistic constraint.

### 3 A double penalization scheme for chance constrained problems

In this section, we explore one possibility offered by the bilevel formulation of chance-constrained problems, presented in the previous section. We propose a (double) penalization approach for solving the bilevel optimization problem, with a different treatment of the two constraints: a basic penalization of the easy constraint together with an exact penalization of the hard constraint formalized as the lower problem.

We first derive in Section 3.1, some growth properties of the lower problem. We then show in Section 3.2 to what extent these properties help to provide an exact penalization of the “hard” constraint. We finally present the double penalty scheme in Section 3.3.

From the bilevel problem (7), we derive the two following penalized problems, associated with two penalization parameters  $\mu, \lambda > 0$  and

$$(P_\mu) \quad \begin{cases} \min_{(x,\eta) \in \mathcal{X} \times \mathbb{R}} & f(x) + \mu \max(\eta, 0) \\ \text{s.t.} & \eta \in \arg \min_{s \in \mathbb{R}} G(x, s) \end{cases} \quad (9)$$

and

$$(P_{\lambda,\mu}) \quad \min_{(x,\eta) \in \mathcal{X} \times \mathbb{R}} f(x) + \lambda \left( G(x, \eta) - \min_{s \in \mathbb{R}} G(x, s) \right) + \mu \max(\eta, 0). \quad (10)$$

We consider a general data-driven situation where the uncertainty  $\xi$  is just known through a sample (or, said alternatively, follows an equiprobable discrete distribution over arbitrary values):

**(A3)** We assume that there exists  $n \in \mathbb{N}^*$  and  $n$  different scenarios

$$\xi_1, \xi_2, \dots, \xi_n \in \mathbb{R}^m \quad \text{such that } \mathbb{P}[\xi = \xi_i] = \frac{1}{n} \text{ for all } i \in \{1, \dots, n\}. \quad (11)$$

Under this assumption, the objective (6) of the lower level problem writes:

$$G(x, s) = s + \frac{1}{n(1-p)} \sum_{i=1}^n \max(g(x, \xi_i) - s, 0). \quad (12)$$

We then define the set  $\mathcal{I}_n = \{\frac{i}{n}, i \in \{0, \dots, n-1\}\}$  which will play a special role in our developments. In particular, we denote the distance to  $\mathcal{I}_n$  by  $d_{\mathcal{I}_n}(p) = \inf\{|p - \frac{i}{n}|, 0 \leq i \leq n-1\}$ , and we define the key quantity

$$\delta = \begin{cases} \frac{1}{n(1-p)} & \text{if } p \in \mathcal{I}_n \\ \frac{d_{\mathcal{I}_n}(p)}{(1-p)} & \text{otherwise,} \end{cases} \quad (13)$$

which depends implicitly on the number of samples  $n$  and the fixed safety parameter  $p$ .

[Penalization choice] We study in this paper a double penalization approach with a penalization of the easy constraint (which is proved to be exact in forthcoming Proposition 3.2) followed by a penalization of the difficult constraint, which thus gives  $(P_{\lambda,\mu})$  in (10). Other approaches to solve the bilevel problem could be worth investigating as well, and are left for future research in this direction. Note that the alternative penalization that would consist in penalizing only the difficult constraint  $\eta \in \operatorname{argmin}_{s \in \mathbb{R}} G(x, s)$  and leave the easy bound constraint  $\eta \leq 0$  would suffer from the objective function not being Lipschitz. This extra difficulty might be tackled by more advanced penalization techniques, as e.g. [3] and [12] (in a framework of Nash Equilibrium). We use here the standard approach and focus on analyzing it and showing how it leads to an efficient algorithm.

### 3.1 Analysis of the value function

In view of the forthcoming exact penalization, we study here the value function  $h : \mathcal{X} \times \mathbb{R} \rightarrow \mathbb{R}$  defined, from  $G$  in (12), as

$$h(x, \eta) = G(x, \eta) - \min_{s \in \mathbb{R}} G(x, s). \quad (14)$$

Observe that, for a fixed  $x$ , the function  $h(x, \cdot)$  is a polyedral, in the data-driven situation (11). Similarly, the solution set  $S(x)$  of the lower level problems in (7) is polyedral as well. This yields that there is a steep increase of the value function with respect to  $\eta$  outside of  $S(x)$ . In accordance with the terminology of [5], the set  $S(x)$  is said to be a *weak sharp minima* for the lower-level problem. In terms of function values, this declines as  $h$  being lower-bounded by  $d_{S(x)}(\cdot)$ , the distance function to  $S(x)$ . In the next result, we establish this property by a direct proof, providing, as a by-product, a specific estimation of the lower-bound.

Let  $p \in [0, 1)$  be fixed but arbitrary. Under assumptions (A1), (A2), and (A3), the function  $h$  defined in (14) satisfies for any  $(x, \eta) \in \mathcal{X} \times \mathbb{R}$ :

$$h(x, \eta) \geq \delta d_{S(x)}(\eta) \quad (15)$$

with  $\delta$  and  $S(x)$  respectively defined by (13) and (7).

Let us fix  $x \in \mathcal{X}$ . Note first that, the function  $\psi : s \mapsto h(x, s)$  is continuous and convex, since so is  $G$ , as defined in 12. As a result  $S(x)$  is a closed interval. Since  $h$  is non-negative, (15) is immediate if  $S(x) = (-\infty, +\infty)$ . Furthermore it clearly appears that  $h(x, s) = 0$  for all  $s \in S(x)$ . We thus assume that  $S(x)$  is lower-bounded or upper-bounded. Let us introduce  $q_p^+ = \sup S(x)$ , and assume  $q_p^+ < \infty$ . By standard subdifferential calculus [15, 4.4.2], we have

$$\begin{aligned} \partial\psi(s) &= \left\{ 1 - \frac{1}{n(1-p)} \sum_{i=1}^n \mathbb{1}_{s < g(x, \xi_i)} + \alpha_i \mathbb{1}_{s = g(x, \xi_i)}, \text{ s.t. } (\alpha_1, \dots, \alpha_n) \in [0, 1]^n \right\} \\ &= \left[ 1 - \frac{1}{1-p} \mathbb{P}[g(x, \xi) \geq s], 1 - \frac{1}{1-p} \mathbb{P}[g(x, \xi) > s] \right] \\ &= \left[ \frac{\mathbb{P}[g(x, \xi) < s]}{1-p}, \frac{\mathbb{P}[g(x, \xi) \leq s]}{1-p} \right] - \frac{p}{1-p}. \end{aligned} \quad (16)$$

Therefore,

$$0 \in \partial\psi(s) \Leftrightarrow p \in [\mathbb{P}[g(x, \xi) < s], \mathbb{P}[g(x, \xi) \leq s]] \Leftrightarrow s \in S(x).$$

By the convexity of  $\psi$ , for any  $s, u$  and  $g_u \in \partial\psi(u)$ :

$$h(x, s) - h(x, u) \geq g_u^\top (s - u). \quad (17)$$

As a consequence for any  $u \notin S(x)$ , since  $0 \notin \partial\psi(u)$ , we obtain from (16) that  $\mathbb{P}(g(x, \xi) \leq u) \neq p$ . Moreover, since  $\mathbb{P}(g(x, \xi) \leq u) \in \mathcal{I}_n$ , we necessarily have

$$|\mathbb{P}(g(x, \xi) \leq u) - p| \geq \begin{cases} \frac{1}{n} & \text{if } p \in \mathcal{I}_n \\ d_{\mathcal{I}_n}(p) & \text{otherwise.} \end{cases}$$

Hence, for any  $s > u > q_p^+$ , we use the subgradient inequality (17) with  $g_u = \frac{\mathbb{P}(g(x, \xi) \leq u) - p}{1 - p}$  to obtain:

$$h(x, s) - h(x, u) \geq \frac{\mathbb{P}(g(x, \xi) \leq u) - p}{1 - p} (s - u) \geq \delta (s - u),$$

where we have used  $g_u > 0$ , since  $u > q_p^+$ . Now, by letting  $u$  approach  $q_p^+$  from above, upon using continuity of  $h(x, \cdot)$  in  $u$ , together with  $h(x, q_p^+) = 0$  (since  $q_p^+ \in S(x)$ ), we derive  $h(x, s) \geq \delta (s - q_p^+)$ . Since clearly  $(s - q_p^+) \geq d_{S(x)}(s)$  the result follows. Similarly, if  $q_p \triangleq \inf S(x) > -\infty$ , then for any  $s < u < q_p$ , we may leverage the subgradient inequality to derive:

$$\begin{aligned} h(x, s) - h(x, u) &\geq \frac{\mathbb{P}(g(x, \xi) \leq u) - p}{1 - p} (s - u) \\ &= \frac{p - \mathbb{P}(g(x, \xi) \leq u)}{1 - p} (u - s) = \frac{|\mathbb{P}(g(x, \xi) \leq u) - p|}{1 - p} (u - s) \\ &\geq \delta (u - s) \end{aligned}$$

since  $u < q_p \implies \mathbb{P}(g(x, \xi) \leq u) < p$ . Now letting  $u$  approach  $q_p$  from below, using continuity of  $h$  in the second argument,  $q_p \in S(x)$ ,  $h(x, q_p) = 0$  and  $(q_p - s) \geq d_{S(x)}(s)$ , we get  $h(x, s) \geq \delta d_{S(x)}(s)$  as desired.

Following the terminology of [41], this theorem shows that  $h$  is a uniform parametric error bound and provides the modulus of uniformity. We note here that the quality of the lower bound depends on the number  $n$  of data points considered, as  $\delta$  depends on  $d$ . This dependence is a drawback which turns out to “pass to the limit”, in the sense that the lower bound fails to be a uniform parametric error bound when  $\xi$  follows a continuous distribution; this is an interesting but secondary result that we prove in Appendix A.1.

### 3.2 Exact penalization for the hard constraint

We show here that  $(P_{\lambda, \mu})$  is an exact penalization of  $(P_\mu)$ , when  $\lambda$  is large enough. The proof of this result follows usual rationale (see e.g., [9, Prop. 2.4.3]); the main technicality is the sharp growth of  $h$  established in Theorem 3.1. Let  $\mu > 0$  be given and assume that there is a solution to  $(P_\mu)$  defined in (9). Then, under assumptions (A1), (A2), and (A3), for any  $\lambda > \mu/\delta$  with  $\delta$  defined in (13), the solution set of  $(P_\mu)$  coincides with the one of  $(P_{\lambda, \mu})$  defined in (10).

Take  $\mu > 0$ , define  $\lambda_\mu = \mu/\delta$ , and take  $\lambda > \lambda_\mu$  arbitrary but fixed. Let us first take a solution  $(x^*, \eta^*) \in \mathcal{X} \times \mathbb{R}$  of  $(P_\mu)$  and show by contradiction that it is also a solution of  $(P_{\lambda, \mu})$ . Indeed, to the contrary, assume there exists some  $\varepsilon > 0$  and  $(x', \eta') \in \mathcal{X} \times \mathbb{R}$  such that:

$$f(x') + \mu \max(0, \eta') + \lambda h_{x'}(\eta') \leq f(x^*) + \mu \max(0, \eta^*) + \lambda h_{x^*}(\eta^*) - \varepsilon.$$

Let then  $\eta'_p \in S(x')$  be such that :  $|\eta'_p - \eta'| \leq d_{S(x')}(\eta') + \frac{\varepsilon}{2\mu}$ . Then the point  $(x', \eta'_p)$  is a feasible for  $P_\mu$  (recall  $\eta'_p \in S(x')$ ) and since  $\eta \mapsto \mu \max(0, \eta)$  is  $\mu$ -Lipschitz, we first have

$$\begin{aligned} f(x') + \mu \max(0, \eta'_p) &\leq f(x') + \mu \max(\eta', 0) + \mu |\eta'_p - \eta'| \\ &\leq f(x') + \mu \max(\eta', 0) + \mu \left( d_{S(x')}(\eta') + \frac{\varepsilon}{2\mu} \right). \end{aligned}$$

Using Theorem 3.1, we then have

$$\begin{aligned} f(x') + \mu \max(0, \eta'_p) &\leq f(x') + \mu \max(\eta', 0) + \mu \left( \frac{1}{\delta} h(x', \eta') + \frac{\varepsilon}{2\mu} \right) \\ &\leq f(x') + \mu \max(\eta', 0) + \lambda_\mu h(x', \eta') + \frac{\varepsilon}{2} \\ &\leq f(x^*) + \mu \max(\eta^*, 0) - \frac{\varepsilon}{2} \end{aligned}$$

which gives the contradiction. Hence any solution of  $(P_\mu)$  is also a solution to problem  $(P_{\lambda, \mu})$ .

Let now  $(\bar{x}, \bar{\eta})$  be a solution of  $(P_{\lambda, \mu})$  and let us show that it is actually a solution for  $P_\mu$ . Let again  $(x^*, \eta^*)$  be an arbitrary solution of  $(P_\mu)$ . We first note that, by the optimality result of  $(\bar{x}, \bar{\eta})$  for  $(P_{\lambda, \mu})$ , we have:

$$f(\bar{x}) + \mu \max(0, \bar{\eta}) + \underbrace{\lambda h(\bar{x}, \bar{\eta})}_{\geq 0} \leq f(x^*) + \mu \max(0, \eta^*) + \underbrace{\lambda h(x^*, \eta^*)}_{=0},$$

which by positivity of the function  $h$  and feasibility for  $(P_\mu)$ , i.e.,  $h(x^*, \eta^*) = 0$  of  $(x^*, \eta^*)$  yields:

$$f(\bar{x}) + \mu \max(0, \bar{\eta}) \leq f(x^*) + \mu \max(0, \eta^*).$$

It remains to show that  $(\bar{x}, \bar{\eta})$  is a feasible point for  $(P_\mu)$ . By the first point,  $(x^*, \eta^*)$  is both a solution of  $(P_{\lambda, \mu})$  and  $(P_{\frac{\lambda + \lambda_\mu}{2}, \mu})$ . Hence, we have:

$$\begin{aligned} f(\bar{x}) + \mu \max(0, \bar{\eta}) + \lambda h(\bar{x}, \bar{\eta}) &\leq f(x^*) + \mu \max(0, \eta^*) \\ &= f(x^*) + \mu \max(0, \eta^*) + \frac{\lambda + \lambda_\mu}{2} h(x^*, \eta^*) \\ &\leq f(\bar{x}) + \mu \max(0, \bar{\eta}) + \frac{\lambda + \lambda_\mu}{2} h(\bar{x}, \bar{\eta}) \end{aligned}$$

But since  $\lambda > \lambda_\mu$  we necessarily have:  $h(\bar{x}, \bar{\eta}) = 0$  which implies by the properties of the value function that  $(\bar{x}, \bar{\eta})$  is a feasible point for  $(P_\mu)$ .

We note that the above result is a consequence, in our specific case, of [41, Theorem 2.6] which is meant for generalized bilevel programs. Based on the terminology of [41] and references therein, we have that  $P_\mu$  satisfies the partial calmness property, as the value function  $h$  is a uniform parametric error bound.

### 3.3 Double penalization scheme

From the previous results, we get that solving the sequence of penalized problems gives approximations of the solution of the initial problem. We formalize this in the next proposition suited for our

context of double penalization. The proof of this result follows standard arguments; see e.g. [24, Ch. 13.1].

Assume (A1), (A2), and (A3) are satisfied, and Problem (7) has a non-empty feasible set. Let  $(\mu_k)_{k \geq 0}$  be an increasing sequence such that  $\mu_k \nearrow \infty$ , and  $(\lambda_k)_{k \geq 0}$  be taken such that  $\lambda_k > \frac{\mu_k}{\delta}$  with  $\delta$  as defined in (13). If, for all  $k$ , there exists a solution of  $(P_{\lambda_k, \mu_k})$  (denoted by  $(x_k, \eta_k)$ ), then any cluster point of the sequence  $(x_k, \eta_k)$  is an optimal solution of (1).

The fact that  $(x_k, \eta_k)$  is an optimal solution of  $(P_{\lambda_k, \mu_k})$  implies that

$$\begin{aligned} f(x_k) + \mu_k \max(0, \eta_k) + \lambda_k h(x_k, \eta_k) \\ \leq f(x_{k+1}) + \mu_k \max(0, \eta_{k+1}) + \lambda_k h(x_{k+1}, \eta_{k+1}) \end{aligned} \quad (18)$$

Similarly for  $(x_{k+1}, \eta_{k+1})$ , we get

$$\begin{aligned} f(x_{k+1}) + \mu_{k+1} \max(0, \eta_{k+1}) + \lambda_{k+1} h(x_{k+1}, \eta_{k+1}) \\ \leq f(x_k) + \mu_{k+1} \max(0, \eta_k) + \lambda_{k+1} h(x_k, \eta_k). \end{aligned} \quad (19)$$

By Proposition 3.2,  $\eta_k$  (resp.  $\eta_{k+1}$ ) is feasible for  $(P_{\mu_k})$  (resp.  $(P_{\mu_{k+1}})$ ); in other words, we have  $h(x_k, \eta_k) = h(x_{k+1}, \eta_{k+1}) = 0$ . Hence summing up these two inequalities yields

$$\max(\eta_k, 0) \geq \max(\eta_{k+1}, 0).$$

Using this last inequality with (18) gives:

$$f(x_k) - f(x_{k+1}) \leq \mu_k (\max(\eta_{k+1}, 0) - \max(\eta_k, 0)) \leq 0,$$

and as a consequence the sequence  $\{f(x_k)\}_{k \geq 0}$  increases. Let  $(x', \eta')$  be an arbitrary feasible solution for  $(P)$ . By definition of the sequence  $(x_k, \eta_k)$ , for any  $k \in \mathbb{N}$ , we have:

$$f(x_k) \leq f(x_k) + \mu_k \max(\eta_k, 0) \leq f(x') + \mu_k \max(\eta', 0) \leq f(x'). \quad (20)$$

Therefore for any cluster point  $(\bar{x}, \bar{\eta})$  of the sequence  $\{(x_k, \eta_k)\}_{k \geq 0}$ , we have  $f(\bar{x}) \leq f(x')$ . In order to show that  $(\bar{x}, \bar{\eta})$  is a solution of (7), it remains to show its feasibility. With the right hand side inequality of (20), we obtain

$$\max(\eta_k, 0) \leq \frac{f(x') - f(x_k)}{\mu_k} \leq \frac{f(x') - f(x_0)}{\mu_k} \xrightarrow{k \rightarrow \infty} 0,$$

so that we may deduce that,  $\bar{\eta} \leq 0$ . Moreover, continuity of  $h$  ensures that  $h(\bar{x}, \bar{\eta}) = 0$  which completes the proof.

In words, cluster points of a sequence of solutions obtained as  $\mu$  grows to  $+\infty$  are feasible solutions of the initial chance-constrained problem. In practice though, we have observed that taking a fixed  $\mu$  is enough for reaching good approximations of the solution with increasing  $\lambda$ 's; see the numerical experiments of Section 5. In the next section, we discuss further the practical implementation of the conceptual double penalization scheme.

We finish this section by a note about the assumption of existence of a solution to  $(P_{\lambda, \mu})$ , in Proposition 3.3. Asserting the existence of a (global) solution for a difference of convex program is obviously difficult in general. For our doubly penalized problem, we can still get a specific result,

under standard “compactness” assumptions giving existence of a solution for the problem without the chance-constraint. Assume that the constraint function  $g(\cdot, z)$  is continuous for all  $z \in \mathbb{R}^m$ . Assume also that the objective function  $f$  is coercive or that the constraint set  $\mathcal{X}$  is compact. Then, for any  $\lambda, \mu > 0$ , the penalized problem  $(P_{\lambda, \mu})$  admits a global solution.

For  $\lambda, \mu \geq 0$  fixed, introduce  $\varphi$  the DC objective function of  $(P_{\lambda, \mu})$

$$\varphi(x, \eta) = f(x) + \lambda h(x, \eta) + \mu \max(\eta, 0) \quad \text{as well as} \quad \varphi^* = \inf_{x \in \mathcal{X}, \eta \in \mathbb{R}} \varphi(x, \eta),$$

and observe that we have

$$\varphi(x, \eta) \geq f(x) + \mu \max(\eta, 0) \geq f(x). \quad (21)$$

Consider  $(x_n, \eta_n)_{n \geq 0}$  to be such that  $\varphi(x_n, \eta_n) \searrow \varphi^*$ . By either the compactness of  $\mathcal{X}$  or (21) combined with the coercivity of  $f$ , we have that  $(x_n)$  necessarily admits a finite cluster point  $\bar{x}$ . We assume without loss of generality that  $x_n \rightarrow \bar{x}$ . Let us show the existence of some  $\bar{\eta}$  such that  $(\bar{x}, \bar{\eta})$  is a solution of  $(P_{\lambda, \mu})$ . Note that, if  $\eta_n$  is bounded, then the proof is done. So the interesting cases are  $\eta_n \rightarrow +\infty$  and  $\eta_n \rightarrow -\infty$  (up to a subsequence extraction)

- If  $\eta_n \rightarrow +\infty$ , then, by (21)

$$\varphi(x_n, \eta_n) \geq f(x_n) + \mu \max(0, \eta_n) \geq \min_x f(x) + \mu \max(0, \eta_n) \rightarrow +\infty$$

which is absurd.

- If  $\eta_n \rightarrow -\infty$ , the sequence becomes eventually negative, and we can write

$$\varphi(x_n, \eta_n) = f(x_n) + \lambda h(x_n, \eta_n) \geq f(x_n) + \lambda h(x_n, Q_p(g(x_n, \xi))) \geq \varphi^*.$$

By continuity of  $g(\cdot, \xi)$  and  $Q_p$  we have convergence of  $Q_p(g(x_n, \xi))$  toward  $\bar{\eta} = Q_p(g(\bar{x}, \xi))$ . We conclude that  $(\bar{x}, \bar{\eta})$  is a minimum of  $\varphi$ .

## 4 Double penalization in practice

In this section, we propose a practical version of the double penalization scheme for solving chance-constrained optimization problems. First, we present in Section 4.1 how to tackle the inner penalized problem  $(P_{\lambda, \mu})$  by leveraging its difference-of-convex (DC) structure. Then we quickly describe, in Section 4.2, the python toolbox that we release, implementing this bundle algorithm and efficient oracles within the double penalization method.

### 4.1 Solving penalized problems by a bundle algorithm

We discuss an algorithm for solving  $(P_{\lambda, \mu})$  by revealing the DC structure of the objective function. Notice indeed that, introducing the convex functions

$$\varphi_1(x, \eta) = f(x) + \lambda G(x, \eta) + \mu \max(\eta, 0) \quad \text{and} \quad \varphi_2(x, \eta) = \lambda \min_{s \in \mathbb{R}} G(x, s)$$

we can write  $(P_{\lambda,\mu})$  as the DC problem

$$\min_{(x,\eta) \in \mathcal{X} \times \mathbb{R}} \varphi(x, \eta) = \varphi_1(x, \eta) - \varphi_2(x, \eta). \quad (22)$$

We then propose to solve this problem by the bundle algorithm of [28], which has shown to be a method of choice for DC problems. This bundle algorithm interacts with first-order oracles for  $\varphi_1$  and  $\varphi_2$ ; in our situation, there exist computational procedures to compute subgradients of  $\varphi_1$  and  $\varphi_2$  from output of oracles of  $f$  and  $g$ , as formalized in the next proposition. The proof of this proposition is deferred to Appendix A.2. Note that at the price of more heavy expressions, we could derive the whole subdifferential.

Let  $(x, \eta) \in \mathcal{X} \times \mathbb{R}$  be fixed. Let  $s_f$  be a subgradient of  $f$  at  $x$  and  $s_{g_1}, \dots, s_{g_n}$  be respective subgradients of  $g(\cdot, \xi_1), \dots, g(\cdot, \xi_n)$  at  $x$ . For a given  $t \in \mathbb{R}$ , denote by  $I_{>t}$  the set of indices such that  $g(x, \xi_i) > t$  and by  $I_{=t}$  the set of indices such that  $g(x, \xi_i) = t$ . Let finally  $\alpha = \frac{\mathbb{P}[g(x, \xi) \leq Q_p(g(x, \xi))] - p}{\#(I_{=Q_p(g(x, \xi))})}$ . Then,  $s_{\varphi_1}$  and  $s_{\varphi_2}$  defined as:

$$s_{\varphi_1} = \left( s_f + \frac{\lambda}{n(1-p)} \sum_{i \in I_{>\eta}} s_{g_i}, 1 + \mu \mathbb{1}_{\eta > 0} - \lambda \frac{\#(I_{>\eta})}{n(1-p)} \right)$$

$$s_{\varphi_2} = \left( \frac{\lambda}{n(1-p)} \left( \sum_{i \in I_{>Q_p(g(x, \xi))}} s_{g_i} + \alpha \sum_{i \in I_{=Q_p(g(x, \xi))}} s_{g_i} \right), 0 \right)$$

are respectively subgradients of  $\varphi_1$  and  $\varphi_2$  at  $(x, \eta)$ .

Thus we can use the bundle algorithm to tackle  $(P_{\lambda,\mu})$  written as (22). Notice, though, that this algorithm is not guaranteed to converge towards optimal solutions: the convergence result [28, Th. 1] establishes convergence towards a point  $\bar{u} = (\bar{x}, \bar{\eta})$  satisfying

$$\partial\varphi_2(\bar{u}) \cap \partial\varphi_1(\bar{u}) \neq \emptyset, \quad (23)$$

which is a weak notion of criticality. We then follow the suggestion of [29] to replace  $\varphi_2$  in (22) by a smooth approximation of it, denoted by  $\tilde{\varphi}_2$ . The reason is that the bundle method minimizing  $\tilde{\varphi} = \varphi_1 - \tilde{\varphi}_2$  then reaches a Clarke-stationary point: indeed, (23) reads  $\nabla\tilde{\varphi}_2(\bar{u}) \subset \partial\varphi_1(\bar{u})$ , which in turn gives that

$$0 \in \partial\varphi(\bar{u}) = \partial\varphi_1(\bar{u}) - \nabla\tilde{\varphi}_2(\bar{u}),$$

i.e., that  $\bar{u}$  is Clarke-stationary (for the smoothed problem). Quantifying the approximation of the true optimal solutions by the obtained points is completely out of the scope of this paper; we refer to preliminary discussions in [28, 29]. In particular, [29] mentions that this smoothing technique helps in computing (approximate) critical points of good quality.

In practice, to smooth  $\varphi_2$ , we use the efficient smoothing procedure of [20] for superquantile-based functions (implementing the Nesterov's smoothing technique [27]). More precisely, [20, Prop. 2.2] reads as follows.

Assume that  $g$  is differentiable. For a smoothing parameter  $\rho > 0$ , the function

$$\tilde{\varphi}_2(x, \eta) = \lambda \sup_{\substack{0 \leq q_i \leq \frac{1}{n(1-p)} \\ q_1 + \dots + q_n = 1}} \sum_{i=1}^n \left\{ q_i g(x, \xi_i) - \frac{\rho}{2} (q_i - \frac{1}{n})^2 \right\} \quad (24)$$

is a global approximation of  $\varphi_2$ , such that  $\tilde{\varphi}_2(x, \eta) \leq \varphi_2(x, \eta) \leq \tilde{\varphi}_2(x, \eta) + \frac{\lambda \rho}{2}$  for all  $(x, \eta) \in \mathbb{R}^{d+1}$ . Moreover, the function is differentiable and its gradient writes, with  $S = (s_{g_i})_{1 \leq i \leq n}$  the Jacobian of  $x \mapsto (g(x, \xi_1), \dots, g(x, \xi_n))$ , as

$$\nabla \tilde{\varphi}_2(x, \eta) = (\lambda S \tilde{q}, 0)$$

where  $\tilde{q}$  is the (unique) optimal solution<sup>1</sup>. of (24).

Let us conclude about our approach to tackle  $(P_{\lambda, \mu})$ . We implement an existing advanced bundle algorithm, together with efficient subroutines, to solve a smooth approximation of  $(P_{\lambda, \mu})$ . There is a convergence result of this algorithm towards a critical point of the approximated problem, but no convergence guarantee towards the optimal solution of the approximated problem or of  $(P_{\lambda, \mu})$  itself. In practice, though, we observe satisfying empirical convergence to near optimal solutions, confirming the good results presented in [28, Sec. 6]. We illustrate the correct empirical convergence of the resulting algorithm in Section 5.2 on problems for which we have explicit optimal solutions.

## 4.2 A python toolbox for chance constrained optimization

We release TACO, an open-source python toolbox for solving chance constrained optimization problems (1). The toolbox implements the penalization approach outlined in section 3 together with the bundle method [28] for the inner penalized subproblems. TACO routines rely on just-in-time compilation supported by Numba [22]. The routines are optimized to provide fast performances on reasonably large datasets. Documentation is available at:

<https://yassine-laguel.github.io/taco>

We provide here basic information on TACO; for further information, we refer to section B in appendix and the online documentation.

The python class `Problem` wraps up all information about the problem to be solved. This class possesses an attribute `data` which contains the values of  $\xi$  and is formatted as a numpy array in 64-bit float precision. The class also implements two methods giving first-order oracles: `objective_func` and `objective_grad` for the objective function  $f$ , and `constraint_func` and `constraint_grad` for the constraint function  $g$ .

Let us take a simple quadratic problem in  $\mathbb{R}^2$  to illustrate the instantiation of a problem. We consider

$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & \|x - a\|^2 & a &= [1.0, 2.0]^\top \\ \text{s.t.} \quad & \mathbb{P}[x^\top \xi \leq 0] \geq 0.9, & & \text{with 1000 samples of } \xi \sim \mathcal{N}(0, 1). \end{aligned}$$

The instance of `Problem` is in this case:

---

```
import numpy as np
class Problem:
    def __init__(self, dim=2, sample_size=1000):
        self.data = np.random.normal(size=(sample_size, dim), dtype=np.float64)
        self.a = np.array([1.0, 2.0], dtype=np.float64)
    def objective_fun(self, x):
```

---

<sup>1</sup>The computation of  $\tilde{q}$  is performed with the fast computational procedures of [20]

```

        return np.dot(x-self.a,x-self.a)
def objective_grad(self, x):
    return x
def constraint_func(self, x, z):
    return np.dot(x,z)
def constraint_grad(self, x, z)
    return z
problem = Problem()

```

---

TACO handles the optimization process with a python class named `Optimizer`. Given an instance of `Problem` and hyper-parameters provided by the user, the class `Optimizer` runs an implementation of the bundle method of [28] on the penalized problem (10). The toolbox gives the option to update the penalization parameters  $\mu, \lambda$  along the running process to escape possible stationary points for the DC objective that are non-feasible for the chance constraint.

---

```

from taco import Optimizer
problem = Problem()
optimizer = Optimizer(problem, p=0.9, starting_point=np.zeros(2, dtype=np.float64),
    pen1=1.0, pen2=10.0)
sol = optimizer.run()

```

---

Customizable parameters are stored in a python dictionary, called `params`, and designed as an attribute of the class `Optimizer`. The main parameters to tune are: the safety level of probability `p`, the starting penalization parameters  $\mu = \text{pen1}$  and  $\lambda = \text{pen2}$ , the starting point of the algorithm and the starting value for the proximal parameter of the bundle method. We underline in particular the importance of the two starting penalization parameters: an initial tuning of  $\mu = \text{pen1}$  and  $\lambda = \text{pen2}$  is often required to get a satisfying solutions for the problem considered. See for instance the experimental setup of our numerical illustrations in B.3 and codes on the toolbox website for the specific tuning used there. Others parameters are filled with default values when instantiating an `Optimizer`; for instance:

---

```

custom_options = {
    'p': 0.9,
    'pen1': 1.0,
    'pen2': 10.0,
    'bund_mu_start': 50.0,
    'bund_max_size_bundle_set': 30,
}
custom_optimizer = Optimizer(problem, params=custom_options)

```

---

Some important parameters (such as the safety probability level, or the starting penalization parameters) may also be given directly to the constructor of the class `Optimizer`, when instantiating the object; as in the first example.

## 5 Numerical illustrations

We illustrate our double penalisation approach implemented in the toolbox TACO on three problems: a 2-dimensional quadratic problem with a non-convex chance constraint (in Section 5.1), a family

of problems with explicit solutions (in Section 5.2), and a case study from [26] (in Section 5.3). These proof-of-concept experiments are not meant to be extensive but to show that our approach is viable. These experiments are reproducible: the experimental framework is available on the toolbox’s website.

### 5.1 Visualization of convergence on a 2d problem

We consider a two-dimensional toy quadratic problem in order to track the convergence of the iterates on the sublevel sets. We take [21, Ex. 4.1] which considers an instance of problem (1) with

$$\begin{aligned}
 f(x) &= \frac{1}{2}(x - a)^\top Q(x - a) \quad \text{with } a = \begin{pmatrix} 2. \\ 2. \end{pmatrix}, Q = \begin{pmatrix} 5.5 & 4.5 \\ 4.5 & 5.5 \end{pmatrix} \\
 g(x, z) &= z^\top W(x)z + w^\top z \quad \text{with } W(x) = \begin{pmatrix} x_1^2 + 0.5 & 0. \\ 0. & |x_2 - 1|^3 + 1 \end{pmatrix} \\
 \xi &\sim \mathcal{N}(\mu, \Sigma) \quad 10^4 \text{ samplings with } \mu = \begin{pmatrix} 1. \\ 1. \end{pmatrix}, \Sigma = \begin{pmatrix} 20. & 0. \\ 0. & 20. \end{pmatrix}.
 \end{aligned} \tag{25}$$

For this example, [21] shows that the chance constraint is convex for large enough probability levels, but here we take a low probability level  $p = 0.008$  to have a non-convex chance-constraint. We can see this on Figure 1, plotting the level sets of the objective function and the constraint function: the chance-constrained region for  $p = 0.008$  is delimited by a black dashed line; the optimal value of this problem is located at the star.

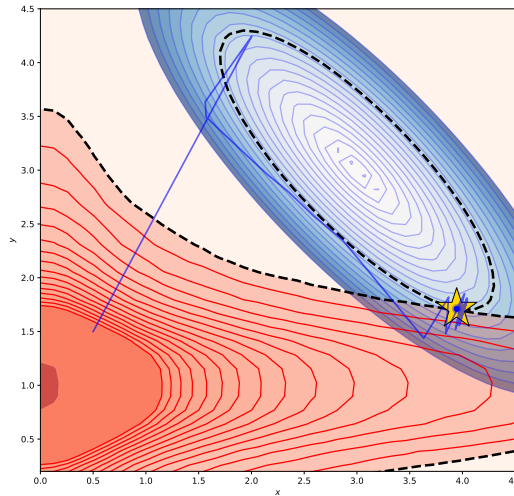


Figure 1: Trajectory of the iterates (in blue) on the plot of the level sets of the chance-constraint and the objective for the 2d problem with data (25).

We apply our double penalization method to solve this problem, with the setting described in Appendix B.3 and available on the TACO website. We plot on the sublevel sets of Figure 1 the path (in deep blue) taken by the sequence of iterates starting from the point  $[0.5, 1.5]$  moving towards the solution. We observe that the sequence of iterates, after a exploration of the functions landscape, gets rapidly close to the optimal solution. At the end of the convergence, we also see a zigzag behaviour around the frontier of the chance constraint. This can be explained by the penalization

term which is activated asymptotically whenever the sequence leaves the feasible region of the chance constraint.

## 5.2 Experiments on a family of problems with explicit solutions

We consider the family of  $d$ -dimensional problems of [16, section 5.1]. For a given dimension  $d$ , the problem writes as an instance of (1) with

$$f(x) = -\sum_{i=1}^d x_i, \quad \mathcal{X} = \mathbb{R}_+^d, \quad g(x, Z) = \max_{i \in \{1, \dots, 10\}} \sum_{j=1}^d Z_{i,j}^2 x_j^2 - 100 \quad (26)$$

and  $\xi$  is random  $10 \times d$ -matrix satisfying for all  $i, j$ ,  $\xi_{i,j} \sim \mathcal{N}(0, 1)$ . The interest of this family of problems is that they have explicit solutions: for given  $d$ , the optimal value is

$$f^* = -\frac{10d}{\sqrt{F_{\chi_d^2}^{(-1)}(p^{\frac{1}{10}})}}$$

where  $F_{\chi_d^2}$  is  $\chi^2$  cumulative distribution with  $d$  degrees of freedom. We consider four instances of this problem with dimension  $d$  from 2 to 200 and the safety probability threshold  $p$  set to 0.8. We consider a rich information on uncertainty:  $\xi$  is sampled 10000 times. In this case, a direct approach consisting in solving the standard mixed-integer quadratic reformulations (see e.g. [2]) with efficient MINLP solvers (we used `Juniper` [19]) does not provide reasonable solutions; see basic information in Appendix B.3.

We solve these instances with our double penalization approach, parameterized as described in Appendix B.3. Figure 2 plots the relative suboptimality  $(f(x_k) - f^*)/|f^*|$  along iterations. The green (resp. red) regions represent iterates that, respectively, satisfy (resp. do not satisfy) the chance constraint.

In the four instances, we take an initial iterate well inside the feasible region. We then observe an initial decrease of the objective function down to global optimal value. The chance constraint starts to be violated later, only when the threshold is reached, and therefore the last part of convergence deals with local improvement of precision and feasibility. Let us underline that the convergence to the global optimal solution, observed here, is specific to this particular situation, and not guaranteed in the general setting.

Table 1 reports the final suboptimality and satisfaction of the probabilistic constraint. The probability constraint is evaluated for 100 sampled points out of the total  $N = 10000$  points. We give the resulting probability; the standard deviation is 0.004 for the four instances.

We observe that the algorithm reaches an accuracy of order of  $10^{-3}$ . Regarding satisfaction of the constraint  $\mathbb{P}[g(x, \xi) \leq 0] \geq 0.8$ , it is achieved to a  $10^{-4}$  precision for  $d = 2$  but it slightly degrades as the dimension grows.

## 5.3 Experiments on a classic case study

We consider the chance-constrained portfolio optimization problem from [26]. The problem writes as maximizing the superquantile/value-at-risk of a portfolio over  $d = 65$  assets (with respective

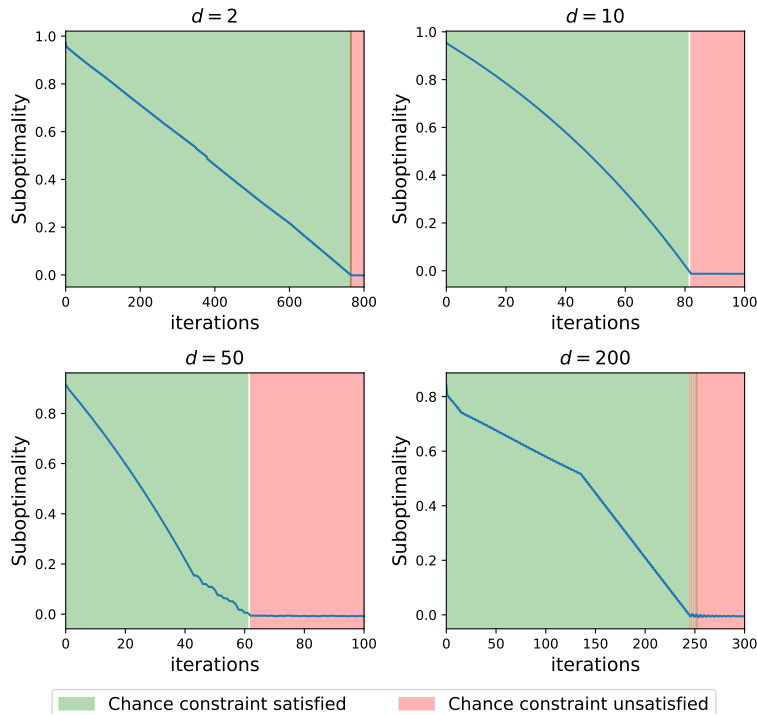


Figure 2: Convergence of the algorithm on four problems (26) with  $d = 2, 10, 50, 200$ .

Dimension	Suboptimality	$\mathbb{P}[g(x, \xi) \leq 0]$	Computation time (s)
$d = 2$	$8.9 \times 10^{-4}$	0.8005	59.67
$d = 10$	$5.0 \times 10^{-3}$	0.7960	12.51
$d = 50$	$5.6 \times 10^{-3}$	0.7776	16.54
$d = 200$	$1.8 \times 10^{-3}$	0.7696	96.61

Table 1: Final suboptimality and feasibility for (26) (where  $p = 0.8$ ).

random returns  $r_i$ ):

$$\begin{cases} \max_{x \geq 0, t \in \mathbb{R}} & t - 1 \\ \text{s.t.} & \mathbb{P}[\sum_{j=0}^d r_j x_j \leq t] \geq 0.95 \\ & \sum_{j=1}^d x_j = 1. \end{cases} \quad (27)$$

We follow [26] for the modeling of randomness and the generation of instances of this problem; see details in Appendix B.3. We consider several samplings of the  $r_i$ , yielding several instances of the problem.

Our reference instance uses a sampling of  $n = 14684$  points; this sample size corresponds to the theoretical size for which the solution obtained by a scenario approximation satisfies the intrinsic chance constraint in (27) with probability  $1 - 10^{-3}$ ; see [6, 26]. We also generate instances with smaller sample sizes ( $n \in \{500, 1000, 2000, 5000, 10000\}$ ) to illustrate the out-of-sample performance of solutions computed by our algorithm.

We report the computational results in Figure 3, highlighting three aspects. In the left-hand plot,

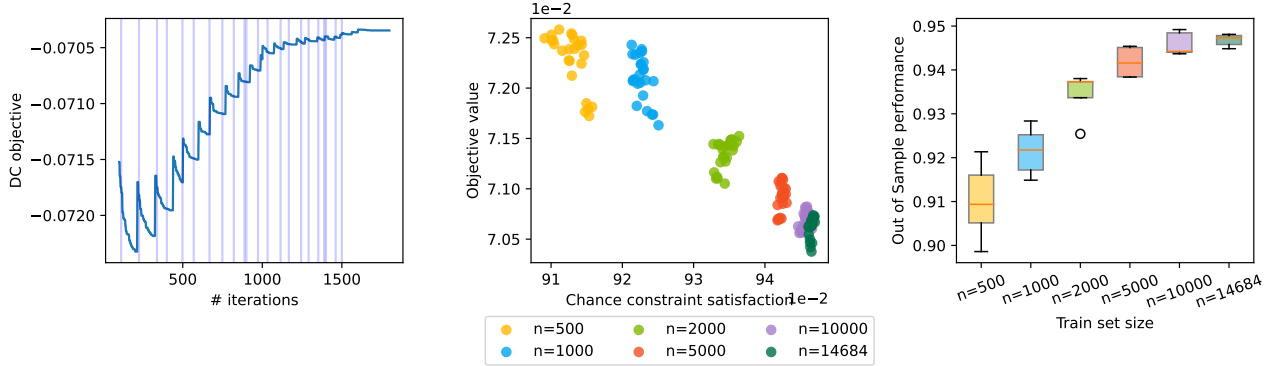


Figure 3: Convergence, trade-off quality, and out-of-sample performance of our algorithm for various parameters solving (27) with different sample size.

we illustrate the empirical convergence of our algorithm on the reference instance. We report the values of the DC objective (22) along the iterations of our algorithm. Vertical lines correspond to updates of the penalty parameters. Between two such updates, the DC objective decreases along the iterates of the inner bundle algorithm. In contrast, at each update, there is an increase, as expected in theory (see (19) in the proof of Proposition 3.3). After approximately 1600 iterations, no more updates on the penalization parameter occur and we observe an empirical convergence.

In the middle plot in Figure 3, we illustrate the trade-off between maximization of the objective and satisfaction of the approximated chance-constraint. More precisely, here is the set-up of this experiment. For each sample size, we generate independently 5 instances of the approximated problem, that we solve several times with our algorithm using different penalization parameters<sup>2</sup>. For each sample size (that has an associated color) and each run of the algorithm, we report the averages over the 5 instances of the final objective values and the chance-constraint satisfaction<sup>3</sup>. We observe, on the figure, a better satisfaction of the chance constraint as  $n$  grows, which comes at the price of lower objective values. We also note a bigger sensitivity to penalization parameters for the instances with small  $n$  (as cloud of colors points are more scattered).

Finally, in the right-hand plot of Figure 3, we illustrate fluctuations on the out-of-sample performances of computed solutions. More precisely, in the same set-up as above, we select the couple of starting penalization parameters providing the best average<sup>4</sup>. As in the previous illustration, we observe that the constraint satisfaction improves when the sample size grows. We also see that the variance of the constraint satisfaction significantly improve as  $n$  grows.

<sup>2</sup>We run our algorithm with penalization parameters  $\mu$  and  $\lambda$  (respectively in  $\{3.0, 3.5, 4.0, 4.5, 5\}$  and  $\{0.3, 0.4, 0.5, 0.6, 0.7\}$ ).

<sup>3</sup>Chance-constraint computed on a separate test set of maximal size, generated independently from the training sets.

<sup>4</sup>averaged performance of the computed solutions, averaged over the 5 different training sets of the same sample size  $n$ .

## 6 Conclusions, perspectives

In this paper, we bring to light a nice bilevel reformulation of chance-constrained optimization problems. We then leverage it to propose a double penalization algorithm, discuss its theoretical properties, and illustrate it on three non-convex case studies. We also release an open-source python toolbox implementing the algorithm and the numerical experiments.

This work opens the door to various possible extensions. First, the applicability of the approach can be widened. For instance, we consider here a convex objective function, but the approach generalizes to difference-of-convex functions, with the same algorithm. Further theoretical guarantees as well as numerical illustrations would then be necessary. Second, note that all ingredients of the algorithm could be improved. For instance, we could try another approach or a better tuning of hyperparameters, with the help of Lagrangian duality. We could also investigate the use of specific first-order methods, rather than a generic bundle method, to solve the inner penalized problems. Finally, beyond the preliminary computational experiments (illustrating the ease-of-use of the toolbox, the heuristic convergence of the algorithm, and the out-of-sample performance of the approach), a study of the sensitivity of the algorithm to the parameters, the adaptivity to special instances, and the comparison to other approaches (other than MINLP) would be of interest. All this is out of the scope of this paper and deserves thorough investigations.

### Data availability statement

We do not analyse or generate any datasets, because our work proceeds within a theoretical and mathematical approach.

### Acknowledgments

We are grateful to the three referees and the editors for their patience, numerous suggestions, and careful reading. In particular, a referee found a mistake in the initial proof of Theorem 3.1. Finally, we acknowledge the support of MIAI Grenoble Alpes (ANR-19-P3IA-0003).

## References

- [1] van Ackooij, W., Henrion, R., Möller, A., Zorgati, R.: Joint chance constrained programming for hydro reservoir management. *Optimization and Engineering* **15**(2), 509–531 (2014)
- [2] Ahmed, S., Shapiro, A.: Solving chance-constrained stochastic programs via sampling and integer programming. In: *State-of-the-art decision-making tools in the information-intensive age*, pp. 261–269. Informs (2008)
- [3] Ba, Q., Pang, J.S.: Exact penalization of generalized nash equilibrium problems. *Operations Research* **70**(3), 1448–1464 (2022)
- [4] Ben-Tal, A., Teboulle, M.: An old-new concept of convex risk measures: The optimized certainty equivalent. *Mathematical Finance* **17**(3), 449–476 (2007)

- [5] Burke, J.V., Ferris, M.C.: Weak sharp minima in mathematical programming. *SIAM Journal on Control and Optimization* **31**(5), 1340–1359 (1993)
- [6] Calafiore, G.C., Campi, M.C.: The scenario approach to robust control design. *IEEE Transactions on Automatic Control* **51**(5), 742–753 (2006)
- [7] Charnes, A., Cooper, W.W.: Chance-constrained programming. *Management science* **6**(1), 73–79 (1959)
- [8] Chow, Y., Ghavamzadeh, M., Janson, L., Pavone, M.: Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research* **18**(1), 6070–6120 (2017)
- [9] Clarke, F.H.: *Optimization and nonsmooth analysis*, vol. 5. Siam (1990)
- [10] Dentcheva, D.: Optimization models with probabilistic constraints. In: *Probabilistic and randomized methods for design under uncertainty*, pp. 49–97. Springer (2006)
- [11] Dentcheva, D., Prékopa, A., Ruszczyński, A.: Concavity and efficient points of discrete distributions in probabilistic programming. *Mathematical Programming* **89**(1) (2000)
- [12] Facchinei, F., Lampariello, L.: Partial penalization for the solution of generalized nash equilibrium problems. *Journal of Global Optimization* **50**, 39–57 (2011)
- [13] Föllmer, H., Schied, A.: Convex measures of risk and trading constraints. *Finance and stochastics* **6**(4), 429–447 (2002)
- [14] Henrion, R., Strugarek, C.: Convexity of chance constraints with independent random variables. *Computational Optimization and Applications* **41**, 263–276 (2008)
- [15] Hiriart-Urruty, J.B., Lemaréchal, C.: *Convex analysis and minimization algorithms I: Fundamentals*, vol. 305. Springer science & business media (2013)
- [16] Hong, L.J., Yang, Y., Zhang, L.: Sequential convex approximations to joint chance constrained programs: A monte carlo approach. *Operations Research* **59**(3) (2011)
- [17] Jiang, N., Xie, W.: Also- $x$  and also- $x+$ : Better convex approximations for chance constrained programs. *Operations Research* **70**(6), 3581–3600 (2022)
- [18] Kannan, R., Luedtke, J.: A stochastic approximation method for approximating the efficient frontier of chance-constrained nonlinear programs (2020)
- [19] Kröger, O., Coffrin, C., Hijazi, H., Nagarajan, H.: Juniper: An open-source nonlinear branch-and-bound solver in julia. In: *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Springer International Publishing (2018)
- [20] Laguel, Y., Malick, J., Harchaoui, Z.: First-order optimization for superquantile-based supervised learning. In: *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. IEEE (2020)
- [21] Laguel, Y., van Ackooij, W., Malick, J., Matiussi Ramalho, G.: On the convexity of level-sets of probability functions. *Journal of Convex Analysis* **29**(2), 1–32 (2022)

- [22] Lam, S.K., Pitrou, A., Seibert, S.: Numba: A llvm-based python jit compiler. In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, LLVM '15. Association for Computing Machinery, New York, NY, USA (2015)
- [23] Luedtke, J., Ahmed, S.: A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization* **19**, 674–699 (2008)
- [24] Luenberger, D.G., Ye, Y.: *Linear and nonlinear programming*, vol. 2. Springer (1984)
- [25] Medova, E.: Chance-constrained stochastic programming for integrated services network management. *Annals of Operations Research* **81**, 213–230 (1998)
- [26] Nemirovski, A., Shapiro, A.: Convex approximations of chance constrained programs. *SIAM Journal on Optimization* **17**(4), 969–996 (2006)
- [27] Nesterov, Y.: Smooth minimization of non-smooth functions. *Mathematical programming* **103**(1), 127–152 (2005)
- [28] de Oliveira, W.: Proximal bundle methods for nonsmooth dc programming. *Journal of Global Optimization* (2019)
- [29] de Oliveira, W.: The abc of dc programming. *Set-Valued and Variational Analysis* **28**(4), 679–706 (2020)
- [30] Pagnoncelli, B.K., Ahmed, S., Shapiro, A.: Sample average approximation method for chance constrained programming: theory and applications. *Journal of optimization theory and applications* **142**(2), 399–416 (2009)
- [31] Peña-Ordieres, A., Luedtke, J.R., Wächter, A.: Solving chance-constrained problems via a smooth sample-based nonlinear approximation. *SIAM Journal on Optimization* **30**(3), 2221–2250 (2020)
- [32] Prékopa, A.: *Stochastic Programming*. Kluwer, Dordrecht (1995). DOI 10.1007/978-94-017-3087-7
- [33] Prékopa, A., Szántai, T.: Flood control reservoir system design using stochastic programming. In: *Mathematical programming in use*, pp. 138–151. Springer (1978)
- [34] Rockafellar, R.T., Royset, J.O.: Superquantiles and their applications to risk, random variables, and regression. In: *Theory Driven by Influential Applications*. INFORMS (2013)
- [35] Rockafellar, R.T., Royset, J.O.: Random variables, monotone relations, and convex analysis. *Mathematical Programming* **148**(1-2), 297–331 (2014)
- [36] Rockafellar, R.T., Uryasev, S.: Optimization of conditional value-at-risk. *Journal of risk* **2**, 21–42 (2000)
- [37] Ruszczyński, A., Shapiro, A.: Optimization of convex risk functions. *Mathematics of operations research* **31**(3), 433–452 (2006)

- [38] van Ackooij, W.: A discussion of probability functions and constraints from a variational perspective. *Set-Valued and Variational Analysis* **28**(4), 585–609 (2020). DOI 10.1007/s11228-020-00552-2
- [39] van Ackooij, W., Henrion, R., Möller, A., Zorgati, R.: Joint chance constrained programming for hydro reservoir management. *Optimization and Engineering* **15** (2014)
- [40] Vandenberghe, L.: The cvxopt linear and quadratic cone program solvers. Online: <http://cvxopt.org/documentation/coneprog.pdf> (2010)
- [41] Ye, J.J., Zhu, D., Zhu, Q.J.: Exact penalization and necessary optimality conditions for generalized bilevel programming problems. *SIAM Journal on optimization* **7**(2), 481–507 (1997)

## A Proofs of complementary results

### A.1 Uniform bound at the limit

We show here that the uniform error bound derived in Section 3.1 vanishes at the limiting case of continuous distributions. We assume that, for a fixed  $x \in \mathbb{R}^d$ , the random variable  $g(x, \xi)$  has a continuous density  $f_{x, \xi} : \mathbb{R} \rightarrow \mathbb{R}$  denoted by  $f_{x, \xi}$ : we have, for all  $a \leq b$ ,

$$\mathbb{P}[a \leq g(x, \xi) \leq b] = \int_a^b f_{x, \xi}(t) dt.$$

Fix  $x \in \mathbb{R}^d$  and denote by  $q_p$  the  $p$ -quantile of the distribution followed by the random variable  $g(x, \xi)$ . If  $g(x, \xi)$  has a continuous density, then the value function  $\eta \mapsto h(x, \eta)$  defined in (14) is differentiable at  $\eta = q_p$  (with  $h'(x, q_p) = 0$ ). We first note that the existence of a density ensures the continuity of the cumulative distribution function of  $g(x, \xi)$ , which in turns implies  $\mathbb{P}[g(x, \xi) \leq q_p] = p$ . Let us now observe that for any arbitrary fixed  $\eta \in \mathbb{R}$ , we have:

$$\begin{aligned} h(x, \eta) &= \eta + \frac{1}{1-p} \mathbb{E}[\max(g(x, \xi) - \eta, 0)] - \left( q_p + \frac{1}{1-p} \mathbb{E}[\max(g(x, \xi) - q_p, 0)] \right) \\ &= (\eta - q_p) + \frac{1}{1-p} \mathbb{E}[\max(g(x, \xi), \eta) - \eta - (\max(g(x, \xi), q_p) - q_p)] \\ &= (\eta - q_p) \left( 1 - \frac{1}{1-p} \right) + \frac{1}{1-p} \mathbb{E}[\max(g(x, \xi), \eta) - \max(g(x, \xi), q_p)]. \end{aligned}$$

Hence, for any  $\eta > q_p$ , we have

$$\begin{aligned} h(x, \eta) &= (\eta - q_p) \left( 1 - \frac{1}{1-p} \right) + \frac{1}{1-p} \mathbb{E}[(\eta - g(x, \xi)) \mathbb{1}_{q_p < g(x, \xi) \leq \eta} + (\eta - q_p) \mathbb{1}_{g(x, \xi) \leq q_p}] \\ &= (\eta - q_p) \left( 1 - \frac{1}{1-p} + \frac{1}{1-p} \mathbb{P}[g(x, \xi) \leq q_p] \right) + \frac{1}{1-p} \mathbb{E}[(\eta - g(x, \xi)) \mathbb{1}_{q_p < g(x, \xi) \leq \eta}] \\ &= \frac{(\eta - q_p)}{1-p} \left( \mathbb{P}[g(x, \xi) \leq q_p] - p + \mathbb{E} \left[ \frac{\eta - g(x, \xi)}{\eta - q} \mathbb{1}_{q_p < g(x, \xi) \leq \eta} \right] \right). \end{aligned}$$

Consequently,

$$\begin{aligned}
h(x, \eta) &= (\eta - q_p) \frac{1}{1-p} \left( \mathbb{P}[g(x, \xi) \leq q_p] - p + \mathbb{E} \left[ \frac{\eta - g(x, \xi)}{\eta - q_p} \mathbb{1}_{q_p < g(x, \xi) \leq \eta} \right] \right) \\
&= \frac{1}{1-p} \mathbb{E} [(\eta - g(x, \xi)) \mathbb{1}_{q_p < g(x, \xi) \leq \eta}] = \frac{1}{1-p} \int_{q_p}^{\eta} (\eta - t) f_{x, \xi}(t) \, dt \\
&= \frac{1}{1-p} \left( \eta \int_{q_p}^{\eta} f_{x, \xi}(t) \, dt - \int_{q_p}^{\eta} t f_{x, \xi}(t) \, dt \right).
\end{aligned} \tag{A.1}$$

By continuity of the above integrands, we can use the fundamental theorem of calculus to get that  $h(x, \cdot)$  admits a right derivative at  $\eta = q_p$  such that

$$\begin{aligned}
h'_+(x, \eta) &= \lim_{\substack{\eta \rightarrow q_p \\ \eta > q_p}} \frac{h(x, \eta) - h(x, q_p)}{\eta - q_p} \\
&= \lim_{\substack{\eta \rightarrow q_p \\ \eta > q_p}} \frac{1}{1-p} \left( \eta \frac{\int_{q_p}^{\eta} f_{x, \xi}(t) \, dt}{\eta - q_p} - \frac{\int_{q_p}^{\eta} t f_{x, \xi}(t) \, dt}{\eta - q_p} \right) \\
&= \lim_{\substack{\eta \rightarrow q_p \\ \eta > q_p}} \frac{1}{1-p} (\eta f_{x, \xi}(q_p) - q_p f_{x, \xi}(q_p)) = 0.
\end{aligned}$$

For the case  $\eta < q_p$ , we observe from (A.1), that

$$\begin{aligned}
h(x, \eta) &= (\eta - q_p) \left( 1 - \frac{1}{1-p} \right) + \frac{1}{1-p} \mathbb{E} [(g(x, \xi) - q_p) \mathbb{1}_{\eta < g(x, \xi) \leq q_p} + (\eta - q_p) \mathbb{1}_{g(x, \xi) \leq \eta}] \\
&= (\eta - q_p) \left( 1 - \frac{1}{1-p} + \frac{1}{1-p} \mathbb{P}[g(x, \xi) \leq \eta] \right) + \frac{1}{1-p} \mathbb{E} [(g(x, \xi) - q_p) \mathbb{1}_{\eta < g(x, \xi) \leq q_p}] \\
&= \frac{(q_p - \eta)}{1-p} \left( p - \mathbb{P}[g(x, \xi) \leq \eta] - \mathbb{E} \left[ \frac{q_p - g(x, \xi)}{q_p - \eta} \mathbb{1}_{\eta < g(x, \xi) \leq q_p} \right] \right).
\end{aligned}$$

Since  $\mathbb{P}[g(x, \xi) = q_p] = 0$ , we obtain

$$\begin{aligned}
h(x, \eta) &= (q_p - \eta) \frac{1}{1-p} \left( \mathbb{E} \left[ 1 - \frac{(q_p - g(x, \xi))}{q_p - \eta} \mathbb{1}_{\eta < g(x, \xi) < q_p} \right] + \mathbb{P}[g(x, \xi) = q_p] \right) \\
&= \frac{1}{1-p} \left( (\eta - q_p) \int_{\eta}^{q_p} f_{x, \xi}(t) \, dt - \int_{\eta}^{q_p} (q_p - t) f_{x, \xi}(t) \, dt \right).
\end{aligned}$$

Using again the fundamental theorem of calculus, we get that  $h(x, \cdot)$  admits a left derivative at  $\eta = q_p$  with:

$$\begin{aligned}
h'_-(x, \eta) &= \lim_{\substack{\eta \rightarrow q_p \\ \eta < q_p}} \frac{h(x, \eta) - h(x, q_p)}{\eta - q_p} \\
&= \lim_{\substack{\eta \rightarrow q_p \\ \eta < q_p}} \frac{1}{1-p} \left( (\eta - q_p) \frac{\int_{\eta}^{q_p} f_{x, \xi}(t) \, dt}{\eta - q_p} - \frac{\int_{\eta}^{q_p} (q_p - t) f_{x, \xi}(t) \, dt}{\eta - q_p} \right) = 0.
\end{aligned}$$

We can conclude that  $h(x, \cdot)$  is differentiable at  $q_p$  with zero as derivative.

## A.2 Proof of the explicit subgradient expressions

We provide here a direct proof of the subgradient expressions of Proposition 4.1. Let  $(x, \eta) \in \mathcal{X} \times \mathbb{R}$  be fixed, and consider first the case of  $\varphi_1$ . For  $i \in \{1, \dots, n\}$ , by successive applications of Theorems 4.1.1 and 4.4.2 from [15, Chap. D] to the functions

$$\varphi_1^{(i)} : (x, \eta) \mapsto \frac{1}{n} \left[ f(x) + \mu \max(\eta, 0) + \lambda \left( \eta + \frac{1}{1-p} \max(g(x, \xi_i) - \eta, 0) \right) \right]$$

we get for any  $i \in \{1, \dots, n\}$

$$\begin{aligned} \frac{1}{n} s_f + \frac{\lambda}{n(1-p)} \mathbb{1}_{g(x, \xi_i) > \eta} s_g &\in \partial_x \varphi_1^i(x, \eta) \\ \frac{\mu}{n} \mathbb{1}_{\eta > 0} + \frac{\lambda}{n} - \frac{\lambda}{n(1-p)} \mathbb{1}_{g(x, \xi_i) > \eta} &\in \partial_\eta \varphi_1^i(x, \eta). \end{aligned}$$

Since  $\varphi_1 = \sum_{i=1}^n \varphi_1^{(i)}$ , we thus have

$$\left( s_f + \frac{\lambda}{n(1-p)} \sum_{i \in I_{>\eta}} s_{g_i}, \mu \mathbb{1}_{\eta > 0} + \lambda - \lambda \frac{\#(I_{>\eta})}{n(1-p)} \right) \in \partial \varphi_1(x, \eta)$$

For  $\varphi_2$  we need first the whole subdifferential of the function  $G$ , which, using above mentioned properties, writes

$$\partial G(x, \eta) = \left\{ \left( \frac{1}{1-p} \sum_{i=1}^n \frac{s_{g_i}}{n} (\mathbb{1}_{g(x, \xi_i) > \eta} + \beta_i \mathbb{1}_{g(x, \xi_i) = \eta}), \right. \right. \\ \left. \left. 1 - \frac{1}{1-p} \sum_{i=1}^n \frac{1}{n} (\mathbb{1}_{g(x, \xi_i) > \eta} + \beta_i \mathbb{1}_{g(x, \xi_i) = \eta}) \right), \beta_i \in [0, 1], \quad \forall i \in \{1, \dots, n\} \right\}.$$

By taking  $\beta_i = \alpha$  (for all  $i \in \{1, \dots, n\}$ ) with the specific  $\alpha$  given in the statement, we can zero the second term in the above expression. Now since  $\varphi_2(x, \eta) = \lambda \min_{s \in \mathbb{R}} G(x, s)$  with  $Q_p(g(x, \xi)) \in \arg \min_{s \in \mathbb{R}} G(x, s)$ , we apply Corollary 4.5.3 of [15, Chap. D] to obtain a subgradient of  $\varphi_2$ :

$$s_{\varphi_2} = \left( \frac{\lambda}{n(1-p)} \left( \sum_{i \in I_{>Q_p(g(x, \xi))}} s_{g_i} + \alpha \sum_{i \in I_{=Q_p(g(x, \xi))}} s_{g_i} \right), 0 \right)$$

which completes the proof.

## B Implementation details on TACO

### B.1 Further customization

TACO relies on a set of hyperparameters to be provided by the user and specified in a single dictionary passed as an argument of the class `Optimizer`. There are two families of parameters to be specified. First, the parameters concerning the oracles  $\varphi_1$  and  $\varphi_2$ . These are the starting

penalization parameters  $\lambda$  and  $\mu$ , the multiplicative factors to increment them along the penalization process, and the smoothing parameter of  $\tilde{\varphi}_2$ . The second family of parameters concerns the bundle method. It gathers the proximal parameters of the bundle method, the precision targeted, the starting point of the algorithm, the maximal size of the bundle information, and parameters related used when restarting the bundle method (see more in the following section). Overall the most important parameters to specify are the starting penalization parameters  $\mu$  and  $\lambda$  with respective keys ‘pen1’ and ‘pen2’ and the starting proximal parameter of the bundle algorithm. In the toolbox, we provide the set of parameters used in our numerical experiments. In addition of the final solution, it is possible to log the iterates, function values and time values, by calling the method with the option `logs=True`. The `verbose=True` option also allows the user to observe in real time the progression of the algorithm along the iterations.

Finally we underline that TACO subroutines rely on just-in-time compilation supported by Numba, which consistently improves the running time. Further improvements can be achieved when the instance considered can be cast as a Numba `jitclass`. The parameter ‘numba’ in the input dictionary of the associated `Optimizer` object should then be set to `True`.

## B.2 On the bundle algorithm

Here are some information on our implementation of the bundle algorithm of [28] to tackle the double penalized problem  $(P_{\lambda,\mu})$  written as a DC problem. We discuss the parameters used at various steps of the procedure. We refer to [28] for more details.

- **Overall run:** The starting point, the maximum number of iterations as well as the precision tolerance for termination may be set by the user.
- **Subproblems:** Each iteration of the bundle algorithm requires solving a quadratic subproblem (see [28, Eq. (9)]), for which we use the solver `cvxopt` [40] by simplicity.
- **Stabilization center:** Whenever the solution of a subproblem satisfies a sufficient decrease in terms a function value, it is considered as a new stability center. The condition to qualify sufficient decrease is given in [28, Eq. (12)]. It involves a constant  $\kappa$  which may be tuned by the user.
- **Proximal parameters:** The initial value of the proximal parameter involved in quadratic subproblems can be set by the user. The user can also specify upper and lower acceptance bounds for it. After each iteration, the prox-parameter is updated: it is increased by a constant factor in case of serious step, and decreased otherwise. Both factors can be tuned by the user.
- **Bundle information:** The bundle of cutting-planes is augmented after each null step with new linearization, and emptied after each serious step. We fix a maximum size for the bundle: above this parameter, the bundle is emptied and proximal parameter is restarted to a specified restarting value. When the bundle is emptied, we have the chance of a specific improvement: if the stability center is feasible in the chance-constraint, we replace the coordinate playing the role of  $\eta$  by the  $p$ -quantile of  $g(x, \xi)$ , thus decrease the objective function.

- **Termination Criteria:** We use a simple stopping criteria: we stop when the euclidean distance between the current iterate and the current stability center falls below a certain threshold specified by the user.

### B.3 Experimental settings

**Setting of Section 5.1.** For this 2d problem, we use the starting point  $x = (0.5, 1.5)$  (and  $\eta = 0.01$ ) well-inside the chance-constraint. The initial penalization parameters  $\mu$  and  $\lambda$  are respectively initialized to 400 and 600. The initial proximal parameter is fixed to 38.0 with lower and upper acceptance bounds set to  $10^{-3}$  and  $10^3$ . Increasing and decreasing factors for this parameter are fixed to 1.05 and 0.95. The classification rule parameter is set to  $10^{-4}$ . The maximal size of the information bundle is set to 20 and the threshold of the termination criteria is set to  $10^{-7}$ . Running time of our algorithm on this problem is approximately a minute.

**Setting for Section 5.2.** For any fixed dimension  $d$  comprised in  $\{2, 10, 50, 200\}$ , the algorithm is run from the starting point  $(0.1, \dots, 0.1) \in \mathbb{R}^{d+1}$ . The starting penalization parameter  $\mu$ , constant for the 4 instances, is set to  $\mu = 10.0$ . We tuned the second penalization parameter  $\lambda$  along problems: we observed that  $\lambda = \{1.75, 1.25, 1.5, 2.0\}$  give good performances for the considered problems. The starting proximal parameters is fixed to 60.0 with lower and upper acceptance bounds set to  $10^{-4}$  and  $10^5$  respectively. Increasing and decreasing factors for the proximal parameter are fixed to 1.01 and 0.99. The classification rule parameter is set to  $10^{-4}$ . The maximal size of the information bundle is set to 300.

**Setting for Section 5.3.** We follow the portfolio optimization model of [26]. We consider 64 risky assets (for  $1 \leq i \leq 64$ ) and one deterministic one (with  $r_0 = 1$ ). We sample the returns  $r_i, (1 \leq i \leq 64)$  from distributions modeled as:

$$r_i = \eta_i + \sum_{k=1}^8 \gamma_{i,k} \zeta_k$$

where  $\eta_i \sim \text{LogNormal}(\mu_i, \sigma_i^2)$ ,  $\zeta_k \sim \text{LogNormal}(0, 0.1)$  and  $\gamma_{i,k}$  are deterministic constants. We used  $\mu_i = \sigma_i$  and took the constants  $\mu_i$  and  $\gamma_i$  such that :

$$\mathbb{E}\left[\sum_{k=1}^8 \gamma_{i,k} \zeta_k\right] = \rho_i/2 \quad \text{and} \quad \mathbb{E}[\eta_i] = 1 + \rho_i/2.$$

Specifically,  $\gamma_{i,k}$  is set as :  $\gamma_{i,k} = (1 + \frac{\epsilon_k}{4})\bar{\gamma}_i$  with

$$\bar{\gamma}_i = \frac{\rho_i}{16e^{0.005}}, \quad \rho_i = 0.1 \times \frac{i}{64}, \quad \epsilon_k = \begin{cases} 1 & \text{if } i \leq 4 \\ -1 & \text{if } i \geq 4 \end{cases}$$

and  $\mu_i$  is set as

$$\mu_i = -1 + \sqrt{1 + 2 \log\left(1 + \frac{\rho_i}{2}\right)}.$$

About the parameters of our algorithm. The first penalization parameter  $\mu$  is taken constant to  $\mu = 10.0$ ; the second penalization parameter  $\lambda$  is tuned over the grid  $\{0.3, 0.4, 0.5, 0.6, 0.7\}$ . The starting proximal parameter is set to 38.0, with lower and upper acceptance bounds set to  $10^{-3}$  and  $10^3$  respectively. Increasing and decreasing factors for the proximal parameter are set to 1.05 and 0.9. The computation time for a single seed and a couple of starting parameters is approximately a minute for this problem.

**Limitations of MINLP approach.** Mixed-integer reformulation approaches (see e.g. [2]) are often considered as the state-of-the-art to solve chance constrained optimization problems by sample average approximation. Applying directly such a reformulation to Problem (26) in Section 5.2 leads to the equivalent mixed integer quadratic program:

$$\begin{aligned}
& \min_{x \in \mathbb{R}^d, z \in \{0,1\}^N} && - \sum_{i=1}^d x_i \\
& \text{s.t.} && \sum_{k=1}^d (\xi_i)_{j,k}^2 x_k^2 - 100 \leq M z_i, \quad \forall i \in \llbracket 1, N \rrbracket, \forall j \llbracket 1, 10 \rrbracket \\
& && \sum_{i=1}^N z_i \leq pN, \quad x \geq 0.
\end{aligned}$$

where  $M$  is a large “big- $M$ ” constant. In our setting, such a formulation involves  $10 \times N = 100000$  quadratic constraints involving binary variables. We were not able to solve the resulting mixed-integer problem in reasonable time using the MINLP solver `Juniper` [19] (that is based on Ipopt and JuMP). This shows that a direct application of reformulation techniques combined with reliable software failed on this problem.