A stochastic approximation method for approximating the efficient frontier of chance-constrained nonlinear programs[*]

Rohit Kannan[1] and James Luedtke[2]

[1]Wisconsin Institute for Discovery, University of Wisconsin-Madison, Madison, WI, USA.
E-mail: rohit.kannan@wisc.edu
[2]Department of Industrial & Systems Engineering and Wisconsin Institute for Discovery,
University of Wisconsin-Madison, Madison, WI, USA. E-mail: jim.luedtke@wisc.edu

Version 3 (this document): May 28, 2020

Version 2: November 8, 2019

Version 1: December 17, 2018

Abstract

We propose a stochastic approximation method for approximating the efficient frontier of chance-constrained nonlinear programs. Our approach is based on a bi-objective viewpoint of chance-constrained programs that seeks solutions on the efficient frontier of optimal objective value versus risk of constraint violation. To this end, we construct a reformulated problem whose objective is to minimize the probability of constraints violation subject to deterministic convex constraints (which includes a bound on the objective function value). We adapt existing smoothing-based approaches for chance-constrained problems to derive a convergent sequence of smooth approximations of our reformulated problem, and apply a projected stochastic subgradient algorithm to solve it. In contrast with exterior sampling-based approaches (such as sample average approximation) that approximate the original chance-constrained program with one having finite support, our proposal converges to stationary solutions of a smooth approximation of the original problem, thereby avoiding poor local solutions that may be an artefact of a fixed sample. Our proposal also includes a tailored implementation of the smoothing-based approach that chooses key algorithmic parameters based on problem data. Computational results on four test problems from the literature indicate that our proposed approach can efficiently determine good approximations of the efficient frontier.

Key words: stochastic approximation, chance constraints, efficient frontier, stochastic subgradient

1 Introduction

Consider the following chance-constrained nonlinear program (NLP):

$$\nu_\alpha^* := \min_{x \in X} \ f(x) \tag{CCP}$$
$$\text{s.t. } \mathbb{P}\{g(x,\xi) \le 0\} \ge 1 - \alpha,$$

where $X \subset \mathbb{R}^n$ is a nonempty closed convex set, ξ is a random vector with probability distribution $\mathcal{P}$ supported on $\Xi \subset \mathbb{R}^d$, $f : \mathbb{R}^n \to \mathbb{R}$ is a continuous quasiconvex function, $g : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^m$ is continuously differentiable, and the constant $\alpha \in (0,1)$ is a user-defined acceptable level of constraint violation. We explore a stochastic approximation algorithm for approximating the efficient frontier of optimal objective value ν_α^* for varying levels of the risk tolerance α in (CCP), i.e., to find a collection of solutions which have varying probability of constraint violation, and where for each such solution the objective function value is (approximately) minimal among solutions having that probability of constraint violation or

1

smaller (see also [40, 52]). Most of our results only require mild assumptions on the data defining the above problem, such as regularity conditions on the functions g and the ability to draw independent and identically distributed (i.i.d.) samples of ξ from $\mathcal{P}$, and do not assume Gaussianity of the distribution $\mathcal{P}$ or make structural assumptions on the functions g (see Section 3). Our result regarding 'convergence to stationary solutions' of (CCP) additionally requires ξ to be a continuous random variable satisfying some mild distributional assumptions. We note that Problem (CCP) can model joint nonconvex chance constraints, nonconvex deterministic constraints (by incorporating them in g), and even some recourse structure (by defining g through the solution of auxiliary optimization problems, see Appendix C).

Chance-constrained programming was introduced as a modeling framework for optimization under uncertainty by Charnes et al. [15], and was soon after generalized to the joint chance-constrained case by Miller and Wagner [42] and to the nonlinear case by Prékopa [49]. Apart from a few known tractable cases, e.g., see Prékopa [50] and Lagoa et al. [37], solving chance-constrained NLPs to optimality is in general hard since the feasible region is not guaranteed to be convex and evaluating feasibility of the probabilistic constraint involves multi-dimensional integration. Motivated by a diverse array of applications [11, 32, 39, 64], many numerical approaches for chance-constrained NLPs attempt to determine good-quality feasible solutions in reasonable computation times (see Section 2 for a brief review).

Rather than solving (CCP) for a fixed α, we are interested in approximating the efficient frontier between risk of constraint violation and objective value since this efficient frontier provides decision-makers with a tool for choosing the parameter α in (CCP). An added advantage of estimating the efficient frontier of (CCP) is that we automatically obtain an estimate of the efficient frontier of a family of distributionally robust counterparts [36]. In this work, we investigate the potential of a stochastic subgradient algorithm [20, 30, 47] for approximating the efficient frontier of (CCP). While our proposal is more naturally applicable for approximating the efficient frontier, we also present a bisection strategy for solving (CCP) for a fixed risk level α.

We begin by observing that the efficient frontier can be recovered by solving the following stochastic optimization problem [52]:

$$\min_{x \in X} \ \mathbb{P}\left\{g(x, \xi) \not\leq 0\right\} \tag{1}$$
$$\text{s.t. } f(x) \leq \nu,$$

where the above formulation determines the minimum probability of constraints violation given an upper bound ν on the objective function value. The reformulation in (1) crucially enables the use of stochastic subgradient algorithms for its approximate solution. Assuming that (CCP) is feasible for risk levels of interest, solving (1) using varying values of the bound ν yields the same efficient frontier as solving (CCP) using varying values of α [52].

Let $\mathbb{1} : \mathbb{R} \to \{0, 1\}$, defined by $\mathbb{1}[z] = 1$ if $z > 0$ and $\mathbb{1}[z] = 0$ if $z \leq 0$, denote the characteristic function of the set $(0, +\infty)$, and $\max[\mathbb{1}[g(x, \xi)]]$ denote $\max_{j \in \{1, \cdots, m\}} \mathbb{1}[g_j(x, \xi)]$. Note that the following problem is equivalent to (1):

$$\min_{x \in X_\nu} \ \mathbb{E}\left[\max\left[\mathbb{1}[g(x, \xi)]\right]\right],$$

where X_ν denotes the closed convex set $\{x \in X : f(x) \leq \nu\}$. The above formulation is almost in the form that stochastic gradient-type algorithms can solve [21], but poses two challenges that prevent immediate application of such algorithms: the step function $\mathbb{1}[\cdot]$ is discontinuous and the max function is nondifferentiable. We propose to solve a partially-smoothened approximation of the above formulation using the projected stochastic subgradient algorithm of Nurminskii [47] and Ermoliev and Norkin [26]. In particular, we use the procedure outlined in Davis and Drusvyatskiy [20] to solve our approximation (also see the closely-related proposals of Ermoliev [25], Ghadimi et al. [30], and the references in these works). To enable this, we replace the discontinuous step functions in the above formulation using smooth approximations $\phi_j : \mathbb{R} \to \mathbb{R}$, $j \in \{1, \cdots, m\}$, to obtain the following approximation to (1):

$$\min_{x \in X_\nu} \ \mathbb{E}\left[\max\left[\phi(g(x, \xi))\right]\right], \tag{APP$_\nu$}$$

where $\max[\phi(g(x, \xi))]$ is shorthand for the composite function $\max\{\phi_1(g_1(x, \xi)), \cdots, \phi_m(g_m(x, \xi))\}$. Section 3.2 presents some options for the smooth approximations ϕ_j.

Our first primary contribution is to analyze the convergence of optimal solutions and stationary points of (APP$_\nu$) to those of problem (1). This analysis closely follows previous work on the use of

similar approximations for the chance-constrained problem (CCP) (cf. Hong et al. [34], Hu et al. [35], Norkin [46], Shan et al. [56, 57], and Geletu et al. [29]). Our analysis is required because: i. our approximations (APP$_\nu$) involve a convergent sequence of approximations of the objective function of (1) rather than a convergent sequence of approximations of the feasible region of (CCP), ii. we handle joint chance-constrained NLPs directly without reducing them to the case of individual chance-constrained programs, and iii. our computational results rely on a slight generalization of existing frameworks for smooth approximation of the step function. Related to the first point, Norkin [46] analyzes smooth approximations in the probability maximization context, but their analysis is restricted to the case when the probability function being maximized is concave (see also Ermoliev et al. [28, Example 5.6], Shapiro et al. [58, Section 4.4.2], Lepp [38], and the references in Norkin [46] for related work).

Our second primary contribution is to propose a practical implementation of a method based on the projected stochastic subgradient algorithm of Nurminskii [47] and Ermoliev and Norkin [26] (we use the specific version proposed in Davis and Drusvyatskiy [20]) to obtain stationary points for problem (APP$_\nu$) for varying levels of ν, and thus obtain an approximation of the efficient frontier. For each fixed ν this method yields convergence to a neighborhood of an approximately stationary solution since the objective function of (APP$_\nu$) is weakly convex under mild assumptions [23, 47]. A similar approach has been proposed in Norkin [46]. In particular, Norkin [46] considers the case when the function g is jointly convex with respect to x and ξ, uses Steklov-Sobolev averaging for constructing (APP$_\nu$) from (1), assumes that the negative of the objective of (1) and the negative of the objective of (APP$_\nu$) are 'α-concave functions' (see Definition 2.3 of Norkin [46] or Definition 4.7 of Shapiro et al. [58]), and establishes the rate of convergence of the stochastic quasigradient method of Ermoliev [25], Nemirovsky and Yudin [45] for solving (APP$_\nu$) to global optimality in this setting. Our contributions differ from those of Norkin [46] in two main respects: we present theory for the case when (APP$_\nu$) may not be solved to global optimality (see Lepp [38] for a related approach), and we design and empirically test a practical implementation of a stochastic approximation algorithm for approximating the solution of (1).

A major motivation for this study is the fact that sample average approximations [41] of (CCP) may introduce "spurious local optima" as a byproduct of sampling as illustrated by the following example from Section 5.2 of Curtis et al. [19] (see also the discussion in Ermoliev and Norkin [26, 27]).

Example 1. Consider the instance of (CCP) with $n = 2$, $d = 2$, $X = \mathbb{R}^n$, $\xi_1 \sim U[-12, 12]$, $\xi_2 \sim U[-3, 3]$, $f(x) = x_2$, and $g(x, \xi) = \frac{x_1^4}{4} - \frac{x_1^3}{3} - x_1^2 + 0.2x_1 - 19.5 + \xi_2 x_1 + \xi_1 \xi_2 - x_2$ with $\alpha = 0.05$. Figure 1 plots the (lower) boundary of the feasible region defined by the chance constraint in a neighborhood of the optimal solution $x^* \approx (1.8, -0.1)$ along with the boundaries of the feasible region generated by a ten thousand sample average approximation (SAA) of the chance constraint and the feasible region generated by the smooth approximation of the step function adopted in this work. The points highlighted by triangles in Figure 1 are local minima of the SAA approximation, but are not near any local minimia of the true problem. We refer to these as spurious local minima.

Example 1 illustrates that obtaining a local optimal solution of an accurate SAA of (CCP) may not necessarily yield a good solution even when ξ is a 'well-behaved' continuous random variable that makes the probability function $p(x) := \mathbb{P}\{g(x, \xi) \not\leq 0\}$ 'well behaved' (even though it might be nonconvex). While smoothing the chance constraint helps mitigate this issue (cf. Figure 1), existing algorithms [29, 57] still use SAA to solve the resulting smooth approximation. By employing a stochastic subgradient algorithm, we do not restrict ourselves to a single fixed batch of samples of the random variables, and can therefore hope to converge to truly locally optimal solutions of (APP$_\nu$). Our proposal leverages well-known advantages of stochastic subgradient-type approaches, including a low per-iteration cost, low memory requirement, good scaling with number of random variables and the number of joint chance constraints, and reasonable scaling with the number of decision variables (cf. the discussion in Nemirovski et al. [44]).

This article is organized as follows. Section 2 reviews related approaches for solving chance-constrained NLPs. Section 3 establishes consistency of the smoothing approach in the limit of its parameter values, and discusses some choices for smooth approximation of the step function. Section 4 outlines our proposal for approximating the efficient frontier of (CCP) and lists some practical considerations for its implementation. Section 5 summarizes implementation details and presents computational experiments on four test problems that illustrate the strength of our approach. We close with a summary of our contributions and some avenues for future work in Section 6. The appendices present auxiliary algorithms,

Figure 1: Comparison of the boundary of the feasible region (in a neighborhood of the optimal solution) of a SAA of Example 1 (with sample size 10000) with that of the smooth approximation of the step function adopted in this work (the function ϕ is defined in Example 4 and the smoothing parameter $\tau = 1.085$).
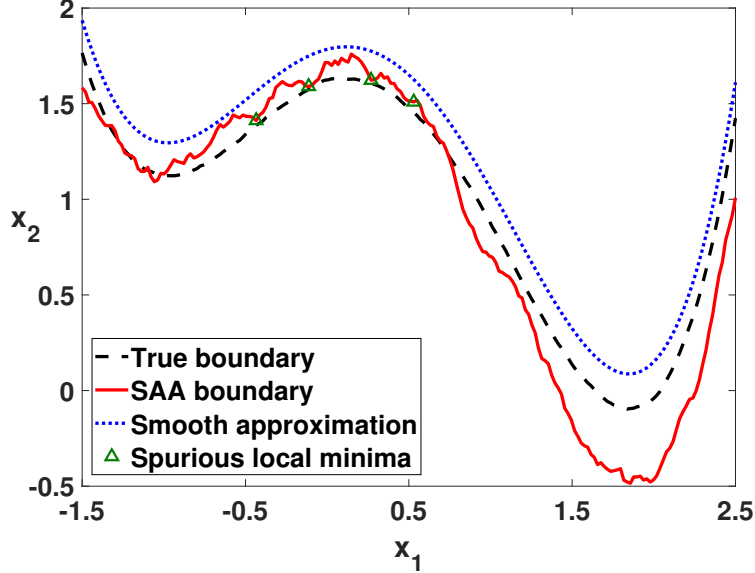
omitted proofs, and computational results, and provide technical and implementation-related details for problems with recourse structure.

Notation. We use e to denote a vector of ones of suitable dimension, $\mathbb{1}[\cdot]$ to denote the l.s.c. step function, $X_\nu := \{x \in X : f(x) \leq \nu\}$, $p(x) := \mathbb{P}\{g(x, \xi) \nleq 0\}$, $B_\delta(x) := \{z \in \mathbb{R}^n : \|z - x\| < \delta\}$, where $\|\cdot\|$ denotes the Euclidean norm, and 'a.e.' for the quantifier 'almost everywhere' with respect to the probability measure $\mathbb{P}$. Given functions $h : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^m$ and $\phi_j : \mathbb{R} \to \mathbb{R}$, $j = 1, \cdots, m$, we write $\max[h(x, \xi)]$ to denote $\max\{h_j(x, \xi) : j \in \{1, \ldots, m\}\}$ and $\phi(h(x, \xi))$ to denote the element-wise composition $(\phi_1(h_1(x, \xi)), \cdots, \phi_m(h_m(x, \xi)))$. Given a set $Z \subset \mathbb{R}^n$, we let co$\{Z\}$ denote its convex hull, $I_Z(\cdot)$ denote its indicator function, i.e., $I_Z(z) = +\infty$ if $z \notin Z$, and zero otherwise, $N_Z(z)$ denote its normal cone at a vector $z \in \mathbb{R}^n$ and proj(z, Z) to denote the projection of z onto Z (we abuse notation to write $y = \text{proj}(z, Z)$ when Z is a nonempty closed convex set), and $|Z|$ denote its cardinality when it is a finite set. Given a twice differentiable function $f : \mathbb{R} \to \mathbb{R}$, we write $df(x)$ and $d^2f(x)$ to denote its first and second derivatives at $x \in \mathbb{R}$. Given a locally Lipschitz continuous function $f : \mathbb{R}^n \to \mathbb{R}$, we write $\partial f(x)$ to denote its Clarke generalized gradient at a point $x \in \mathbb{R}^n$ [17] and $\partial_B f(x)$ to denote its B-subdifferential (note: $\partial f(x) = \text{co}\{\partial_B f(x)\}$). We assume measurability of random functions throughout.

2 Review of related work

We restrict our attention in this brief review to algorithms that attempt to generate good-quality *feasible* solutions to chance-constrained *nonlinear* programs.

In the *scenario approximation* approach, a fixed sample $\{\xi^i\}_{i=1}^N$ of the random variables from the distribution $\mathcal{P}$ is taken and the sampled constraints $g(x, \xi^i) \leq 0$, $\forall i \in \{1, \cdots, N\}$, are enforced in lieu of the chance constraint $\mathbb{P}\{g(x, \xi) \leq 0\} \geq 1 - \alpha$. The advantage of this approach is that the scenario problem is a standard NLP that can be solved using off-the-shelf solvers. Calafiore, Campi, and Garatti [10, 12] present upper bounds on the sample size N required to ensure that the solution of the scenario problem is *feasible* for (CCP) with a given probability when the functions $g(\cdot, \xi)$ are convex for each fixed $\xi \in \Xi$. When the functions g do not possess the above convexity property but instead satisfy alternative Lipschitz assumptions, Luedtke and Ahmed [41, Section 2.2.3] determine an upper bound on N that provides

similar theoretical guarantees for a slightly perturbed version of the scenario problem. A major limitation of scenario approximation is that it lacks strong guarantees on solution quality [41, 52]. A related approach for obtaining feasible solutions to (CCP) constructs convex approximations of the feasible region using conservative convex approximations of the step function [16, 43, 53]. Such approximations again may result in overly conservative solutions, e.g., see Hong et al. [34] and Cao and Zavala [13].

There has been a recent surge of interest in iterative smoothing-based NLP approaches in which the chance constraint $p(x) \leq \alpha$ is replaced by a convergent sequence of smooth approximations. To illustrate these approaches, consider first the case when we only have an individual chance constraint (i.e., $m = 1$), and suppose $p(x) = \mathbb{E}\big[\mathbb{1}\left[g(x,\xi)\right]\big] \leq \alpha$ is approximated by $\mathbb{E}\big[\phi_k\left(g(x,\xi)\right)\big] \leq \alpha$, where $\{\phi_k\}$ is a monotonic sequence of smooth approximations converging to the step function. If each element of $\{\phi_k\}$ is chosen to overestimate the step function, then solving the sequence of smooth approximations furnishes feasible solutions to (CCP) of improving quality. Due to the difficulty in evaluating high-dimensional expectations in general, the smooth approximation is typically applied to a scenario-based approximation. Joint chance constraints have been accommodated either by replacing the max function in $\mathbb{E}\big[\max\left[\phi_k\left(g(x,\xi)\right)\right]\big]$ with its own convergent sequence of smooth approximations [35, 56], or by conservatively approximating (CCP) using a Bonferroni-type approximation [43].

Lepp (see [38] and the references therein) proposes a stochastic version of a modified Lagrangian algorithm for solving progressively refined smooth approximations of individual chance-constrained NLPs with a differentiable probability function p, and establishes almost sure convergence to stationary solutions of (CCP). A key difference between our proposal and Lepp's [38] is that the latter requires a diverging sequence of mini-batch sizes to ensure convergence. Andrieu et al. [5] propose smoothing and finite difference-based stochastic Arrow-Hurwicz algorithms for individual chance-constrained NLPs with continuous random variables, and establish local convergence to stationary solutions under relatively strong assumptions.

Hong et al. [34] propose a sequence of conservative nonsmooth difference-of-convex (DC) approximations of the step function and use it to solve joint chance-constrained convex programs. They establish convergence of any stationary/optimal solution of the DC approximation scheme to the set of stationary/optimal solutions of (CCP) in the limit under certain assumptions. Hu et al. [35] build on this work by developing a sequence of conservative smooth approximations of joint chance-constrained convex programs (using log-sum-exp approximations of the max function), and use a sequential convex approximation algorithm to solve their smooth approximation of a SAA of (CCP). Shan et al. [56, 57] develop a family of conservative smooth DC approximations of the step function, also employ conservative log-sum-exp approximations of the max function to construct smooth approximations of (CCP), and provide similar theoretical guarantees as Hong et al. [34] under slightly weaker assumptions. Geletu et al. [29] propose a framework for conservative analytic approximations of the step function, provide similar theoretical guarantees as these works, and illustrate the applicability of their framework for individual chance-constrained NLPs using a class of sigmoid-like smoothing functions. They also propose smooth outer-approximations of (CCP) for generating lower bounds. Adam et al. [1, 3] develop a continuous relaxation of SAAs of (CCP), and propose to use smooth approximations of the step function that are borrowed from the literature on mathematical programs with complementarity constraints along with Benders' decomposition [7] to determine stationary points. Cao and Zavala [13] propose a nonsmooth sigmoidal approximation of the step function, and use it to solve a smooth approximation of a SAA of individual chance-constrained NLPs. They propose to initialize their algorithm using the solution of a SAA of a conservative approximation of (CCP) obtained by replacing the step function with a convex overestimate [53]. Finally, Peña-Ordieres et al. [48] also consider using a smooth approximation on an SAA of (CCP), analyze the joint convergence with respect to sample size and sequence of smooth approximations, propose to solve a quantile-based approximation, and devise a trust-region method to solve the approximation when applied to a joint chance constraint.

An alternative to smoothing the chance constraint is to solve a SAA problem directly using mixed-integer NLP techniques, which may be computationally challenging, especially when the constraint functions g are not convex with respect to the decisions x. Several tailored approaches [40, 61] have been proposed for solving SAAs of chance-constrained convex programs. Curtis et al. [19] attempt to directly solve a SAA of (CCP) using NLP techniques. They develop an exact penalty function for the SAA, and propose a trust region algorithm that solves quadratic programs with linear cardinality constraints to converge to stationary points of (CCP).

Another important body of work by van Ackooij and Henrion [59, 60] establishes (sub)gradient

formulae for the probability function p when the function g possesses special structure and $\mathcal{P}$ is a Gaussian or Gaussian-like distribution. These approaches employ internal sampling to numerically evaluate (sub)gradients, and can be used within a NLP framework for computing stationary point solutions to (CCP). When Problem (CCP) possesses certain special structures, NLP approaches based on 'p-efficient points' [22, 62] can also be employed for its solution.

The recent independent work of Adam and Branda [2] also considers a stochastic approximation method for chance-constrained NLPs. They consider the case when the random vector ξ has a discrete distribution with finite (but potentially large) support, rewrite the chance constraint in (CCP) using a quantile-based constraint, develop a penalty-based approach to transfer this constraint to the objective, and employ a mini-batch projected stochastic subgradient method to determine an approximately stationary solution to (CCP) for a given risk level. This work does not include a analysis that shows the method converges to a stationary solution.

3 The smoothing approach

In the first part of this section, we establish conditions under which global/stationary solutions of convergent sequences of smooth approximations (APP_ν) converge to a global/stationary solution of (1). We then present some concrete examples of sequences of smooth approximations that can be accommodated within this framework.

3.1 Consistency

We establish sufficient conditions under which solving a sequence of smooth approximations (APP_ν) of the stochastic program (1) yields a point on the efficient frontier of (CCP). The techniques in this section follow closely those in previous work that analyzed similar convergence for (CCP) (cf. Hong et al. [34], Hu et al. [35], Shan et al. [56, 57], and Geletu et al. [29]). We use the following assumptions on Problem (1) in our analysis. Not all results we present require all these assumptions; the required assumptions are included in the statements of our results.

Assumption 1. The convex set $X_\nu := \{x \in X : f(x) \leq \nu\}$ is nonempty and compact.

Assumption 2. For each $x \in X_\nu$, $\mathbb{P}\{\max[g(x, \xi)] = 0\} = 0$.

Assumption 3. The following conditions on the functions g_j hold for each $\xi \in \Xi$ and $j \in \{1, \cdots, m\}$:

    A. The function $g_j(\cdot, \xi)$ is Lipschitz continuous on X_ν with a nonnegative measurable Lipschitz constant $L_{g,j}(\xi)$ satisfying $\mathbb{E}[L_{g,j}^2(\xi)] < +\infty$.

    B. The gradient function $\nabla g_j(\cdot, \xi)$ is Lipschitz continuous on X_ν with a measurable Lipschitz constant $L_{g,j}'(\xi) \in \mathbb{R}_+$ satisfying $\mathbb{E}[L_{g,j}'(\xi)] < +\infty$.

    C. There exist positive constants $\sigma_{g,j}$ satisfying $\mathbb{E}[\|\nabla_x g_j(\bar{x}, \xi)\|^2] \leq \sigma_{g,j}^2$, $\forall \bar{x} \in X_\nu$.

Assumption 1 is used to ensure that Problem (1) is well defined (see Lemma 1). Assumption 2 guarantees that the family of approximations of the function p considered converge pointwise to it on X_ν in the limit of their parameter values (see Proposition 1). This assumption, when enforced, essentially restricts ξ to be a continuous random variable (see Lemma 1 for a consequence of this assumption). Assumption 3 is a standard assumption required for the analysis of stochastic approximation algorithms that serves two purposes: it ensures that the approximations of the function p possess important regularity properties (see Lemma 2), and it enables the use of the projected stochastic subgradient algorithm for solving the sequence of approximations (APP_ν). Note that Assumption 2 is implied by Assumption 6B, which we introduce later, and that Assumption 3A implies $\mathbb{E}[L_{g,j}(\xi)] < +\infty$.

Lemma 1. The probability function $p : \mathbb{R}^n \to [0, 1]$ is lower semicontinuous. Under Assumption 2, we additionally have that p is continuous on X_ν.

Proof. Follows from Theorem 10.1.1 of Prékopa [50]. $\qquad\square$

A natural approach to approximating the solution of Problem (1) is to construct a sequence of approximations (APP_ν) based on an associated sequence of smooth approximations that converge to the step function (cf. Section 2). In what follows, we use $\{\phi_k\}$ to denote a sequence of smooth approximations of the step function, where ϕ_k corresponds to a vector of approximating functions $(\phi_{k,1}, \cdots, \phi_{k,m})$ for the m constraints defined by the function g. Since we are mainly interested in employing sequences of approximations $\{\phi_k\}$ that depend on associated sequences of smoothing parameters $\{\tau_k\}$ (with $\phi_{k,j}$ 'converging' to the step function $\mathbb{1}[\cdot]$ as the smoothing parameter $\tau_{k,j}$ converges to zero), we sometimes write $\phi(\cdot; \tau_{k,j})$, where $\phi : \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}$, instead of $\phi_{k,j}(\cdot)$ to make this parametric dependence explicit. For ease of exposition, we make the following (mild) blanket assumptions on each element of the sequence of smoothing functions $\{\phi_k\}$ throughout this work. Section 3.2 lists some examples that satisfy these assumptions.

Assumption 4. The following conditions hold for each $k \in \mathbb{N}$ and $j \in \{1, \cdots, m\}$:

  A. The functions $\phi_{k,j} : \mathbb{R} \to \mathbb{R}$ are continuously differentiable.

  B. Each function $\phi_{k,j}$ is nondecreasing, i.e., $y \geq z \implies \phi_{k,j}(y) \geq \phi_{k,j}(z)$.

  C. The functions $\phi_{k,j}$ are equibounded, i.e., there exists a universal constant $M_\phi > 0$ (independent of indices j and k) such that $|\phi_{k,j}(y)| \leq M_\phi, \forall y \in \mathbb{R}$.

  D. Each approximation $\phi_{k,j}(y)$ converges pointwise to the step function $\mathbb{1}[y]$ except possibly at $y = 0$, i.e., $\lim_{k\to\infty} \phi_{k,j}(y) = \mathbb{1}[y], \forall y \in \mathbb{R}\backslash\{0\}$, or, equivalently, $\lim_{k\to\infty} \phi(y; \tau_{k,j}) = \mathbb{1}[y], \forall y \in \mathbb{R}\backslash\{0\}$.

We say that 'the *strong form* of Assumption 4' holds if Assumption 4D is replaced with the stronger condition of pointwise convergence *everywhere*, i.e., $\lim_{k\to\infty} \phi_{k,j}(y) = \mathbb{1}[y], \forall y \in \mathbb{R}$. Note that a sequence of conservative smooth approximations of the step function (which overestimate the step function everywhere) cannot satisfy the strong form of Assumption 4, whereas sequences of underestimating smooth approximations of the step function may satisfy it (see Section 3.2). In the rest of this paper, we use $\hat{p}_k(x)$ to denote the approximation $\mathbb{E}[\max[\phi_k(g(x, \xi))]]$ (again, we sometimes write $\hat{p}(x; \tau_k)$ instead of $\hat{p}_k(x)$ to make its dependence on the smoothing parameters τ_k explicit). Note that $\hat{p}_k : \mathbb{R}^n \to \mathbb{R}$ is continuous under Assumption 4, see Shapiro et al. [58, Theorem 7.43]. The following result establishes sufficient conditions under which $\hat{p}_k(x) \to p(x)$ pointwise on X_ν.

Proposition 1. Suppose Assumptions 2 and 4 hold, or the strong form of Assumption 4 holds. Then $\lim_{k\to\infty} \hat{p}_k(x) \equiv \lim_{k\to\infty} \hat{p}(x; \tau_k) = p(x), \forall x \in X_\nu$.

Proof. Define $h_k : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ by $h_k(x, \xi) := \max[\phi_k(g(x, \xi))]$ for each $k \in \mathbb{N}$, and note that h_k is continuous by virtue of Assumption 4A. Additionally, Assumption 4C implies that $|h_k(x, \xi)| \leq M_\phi$ for each $x \in \mathbb{R}^n, \xi \in \mathbb{R}^d$, and $k \in \mathbb{N}$. By noting that for each $x \in X_\nu$,

$$\lim_{k\to\infty} h_k(x, \xi) = \max[\mathbb{1}[g(x, \xi)]]$$

for a.e. $\xi \in \Xi$ due to Assumptions 2 and 4 (or just the strong form of Assumption 4), we obtain the stated result by Lebesgue's dominated convergence theorem [58, Theorem 7.31]. $\square$

The next two results show that a global solution of (1) can be obtained by solving a sequence of approximations (APP_ν) to global optimality. To achieve this, we rely on the concept of epi-convergence of sequences of extended real-valued functions (see Rockafellar and Wets [54, Chapter 7] for a comprehensive introduction).

Proposition 2. Suppose Assumptions 2 and 4 hold. Then, the sequence of functions $\{\hat{p}_k(\cdot) + I_{X_\nu}(\cdot)\}_k$ epi-converges to $p(\cdot) + I_{X_\nu}(\cdot)$ for each $\nu \in \mathbb{R}$.

Proof. Define $h_k : \mathbb{R}^n \to \mathbb{R}$ and $h : \mathbb{R}^n \to \mathbb{R}$ by $h_k(x) := \hat{p}_k(x) + I_{X_\nu}(x)$ and $h(x) := p(x) + I_{X_\nu}(x)$, respectively. From Proposition 7.2 of Rockafellar and Wets [54], we have that $\{h_k\}$ epi-converges to h if and only if at each $x \in \mathbb{R}^n$

$$\liminf_{k\to\infty} h_k(x_k) \geq h(x), \text{ for } \textit{every} \text{ sequence } \{x_k\} \to x, \text{ and}$$

$$\limsup_{k\to\infty} h_k(x_k) \leq h(x), \text{ for } \textit{some} \text{ sequence } \{x_k\} \to x.$$

Consider the constant sequence with $x_k = x$ for each $k \in \mathbb{N}$. We have

$$\limsup_{k \to \infty} h_k(x_k) = I_{X_\nu}(x) + \limsup_{k \to \infty} \hat{p}_k(x) = I_{X_\nu}(x) + p(x) = h(x)$$

as a result of Proposition 1, which establishes the latter inequality.

To see the former inequality, consider an arbitrary sequence $\{x_k\}$ in $\mathbb{R}^n$ converging to $x \in \mathbb{R}^n$. By noting that the indicator function $I_{X_\nu}(\cdot)$ is lower semicontinuous (since X_ν is closed) and

$$\liminf_{k \to \infty} h_k(x_k) \geq \liminf_{k \to \infty} \hat{p}_k(x_k) + \liminf_{k \to \infty} I_{X_\nu}(x_k),$$

it suffices to show $\liminf_k \hat{p}_k(x_k) \geq p(x)$ to establish $\{h_k\}$ epi-converges to h. In fact, it suffices to show the above inequality holds for any $x \in X_\nu$ since the former inequality holds trivially for $x \notin X_\nu$.

Define $q_k : \mathbb{R}^d \to \mathbb{R}$ by $q_k(\xi) := \max\left[\phi_k\left(g(x_k, \xi)\right)\right]$, and note that q_k is $\mathbb{P}$-integrable by virtue of Assumption 4. By Fatou's lemma [58, Theorem 7.30], we have that

$$\liminf_k \mathbb{E}\left[q_k(\xi)\right] = \liminf_k \hat{p}_k(x_k) \geq \mathbb{E}\left[q(\xi)\right],$$

where $q : \mathbb{R}^d \to \mathbb{R}$ is defined as

$$q(\xi) := \liminf_k q_k(\xi) = \liminf_k \max\left[\phi_k\left(g(x_k, \xi)\right)\right].$$

Therefore, it suffices to show that $q(\xi) \geq \max\left[\mathbb{1}\left[g(x, \xi)\right]\right]$ a.e. for each $x \in X_\nu$ to establish the former inequality. (Note that this result holds with Assumption 2 if the strong form of Assumption 4 is made.)

Fix $\xi \in \Xi$. Let $l \in \{1, \cdots, m\}$ denote an index at which the maximum in $\max\left[\mathbb{1}\left[g(x, \xi)\right]\right]$ is attained, i.e., $\max\left[\mathbb{1}\left[g(x, \xi)\right]\right] = \mathbb{1}\left[g_l(x, \xi)\right]$. Consider first the case when $g_l(x, \xi) > 0$. By the continuity of the functions g, for any $\varepsilon > 0$ there exists $N_\varepsilon \in \mathbb{N}$ such that $j \geq N_\varepsilon \implies g(x_j, \xi) > g(x, \xi) - \varepsilon e$. Therefore

$$q(\xi) = \liminf_k \max\left[\phi_k\left(g(x_k, \xi)\right)\right] \geq \liminf_k \max\left[\phi_k\left(g(x, \xi) - \varepsilon e\right)\right] \geq \liminf_k \phi_{k,l}\left(g_l(x, \xi) - \varepsilon\right),$$

where the first inequality follows from Assumption 4B. Choosing $\varepsilon < g_l(x, \xi)$ yields

$$q(\xi) = \liminf_k \max\left[\phi_k\left(g(x_k, \xi)\right)\right] \geq \liminf_k \phi_{k,l}\left(g_l(x, \xi) - \varepsilon\right) = 1 = \max\left[\mathbb{1}\left[g(x, \xi)\right]\right]$$

by virtue of Assumption 4D. The case when $g_l(x, \xi) < 0$ follows more directly since

$$q(\xi) = \liminf_k \max\left[\phi_k\left(g(x_k, \xi)\right)\right] \geq \liminf_k \phi_{k,l}\left(g_l(x, \xi) - \varepsilon\right) = \mathbb{1}\left[g_l(x, \xi) - \varepsilon\right] = 0 = \max\left[\mathbb{1}\left[g(x, \xi)\right]\right],$$

where the second equality follows from Assumption 4. Since $\max\left[g(x, \xi)\right] \neq 0$ a.e. $\forall x \in X_\nu$ by Assumption 2, we have that $q(\xi) \geq \max\left[\mathbb{1}\left[g(x, \xi)\right]\right]$ a.e. for each $x \in X_\nu$, which concludes the proof. $\square$

A consequence of the above proposition is the following key result (cf. Hong et al. [34, Theorem 2], Shan et al. [57, Theorem 4.1], Geletu et al. [29, Corollary 3.7], and Cao and Zavala [13, Theorem 5]), which establishes convergence of the optimal solutions and objective values of the sequence of approximating problems (APP$_\nu$) to those of the true problem (1).

Theorem 1. Suppose Assumptions 1, 2, and 4 hold. Then

$$\lim_{k \to \infty} \min_{x \in X_\nu} \hat{p}(x; \tau_k) = \min_{x \in X_\nu} p(x) \quad \text{and} \quad \limsup_{k \to \infty} \arg\min_{x \in X_\nu} \hat{p}(x; \tau_k) \subset \arg\min_{x \in X_\nu} p(x).$$

Proof. Follows from Proposition 2 and Theorem 7.33 of Rockafellar and Wets [54]. $\square$

Theorem 1 has a couple of practical limitations. First, it is not applicable to situations where ξ is a discrete random variable since it relies crucially on Assumption 2. Second, it only establishes that any accumulation point of a sequence of *global* minimizers of (APP$_\nu$) is a *global* minimizer of (1). Since (APP$_\nu$) involves minimizing a nonsmooth nonconvex expected-value function, obtaining a global minimizer of (APP$_\nu$) is itself challenging. The next result circumvents the first limitation when the smoothing function is chosen judiciously (cf. the outer-approximations of Geletu et al. [29] and Section 3.2). Proposition 4 shows that strict local minimizers of (1) can be approximated using sequences of local minimizers of (APP$_\nu$). While this doesn't fully address the second limitation, it provides some hope that strict local minimizers of (1) may be approximated by solving a sequence of approximations (APP$_\nu$) to local optimality. Theorem 2 establishes that accumulation points of sequences of stationary solutions to the approximations (APP$_\nu$) yield stationary solutions to (1) under additional assumptions.

Proposition 3. Suppose the approximations $\{\phi_k\}$ satisfy the strong form of Assumption 4. Then the sequence of functions $\{\hat{p}_k(\cdot) + I_{X_\nu}(\cdot)\}_k$ epi-converges to $p(\cdot) + I_{X_\nu}(\cdot)$. Moreover, the conclusions of Theorem 1 hold if we further make Assumption 1.

Proof. Follows by noting that Assumption 2 is no longer required for the proof of Proposition 2 when the strong form of Assumption 4 is made. $\qquad\square$

In fact, the proof of Proposition 3 becomes straightforward if we impose the additional (mild) condition (assumed by Geletu et al. [29]) that the sequence of smooth approximations $\{\phi_k\}$ is monotone nondecreasing, i.e., $\phi_{k+1,j}(y) \geq \phi_{k,j}(y)$, $\forall y \in \mathbb{R}, k \in \mathbb{N}$, and $j \in \{1, \cdots, m\}$. Under this additional assumption, the conclusions of Proposition 3 follow from Proposition 7.4(d) and Theorem 7.33 of Rockafellar and Wets [54].

The next result, in the spirit of Proposition 3.9 of Geletu et al. [29], shows that strict local minimizers of (1) can be approximated using local minimizers of (APP_ν).

Proposition 4. Suppose the assumptions of Theorem 1 (or those of Proposition 3) hold. If x^* is a strict local minimizer of (1), then there exists a sequence of local minimizers $\{\hat{x}_k\}$ of $\min_{x \in X_\nu} \hat{p}_k(x)$ with $\hat{x}_k \to x^*$.

Proof. Since x^* is assumed to be a strict local minimizer of (1), there exists $\delta > 0$ such that $p(x) > p(x^*)$, $\forall x \in X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$. Since $X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$ is compact, $\min_{x \in X_\nu \cap \mathrm{cl}(B_\delta(x^*))} \hat{p}_k(x)$ has a global minimum, say $\hat{x}_k$, due to Assumption 4A. Furthermore, $\{\hat{x}_k\}$ has a convergent subsequence in $X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$.

Assume without loss of generality that $\{\hat{x}_k\}$ itself converges to $\hat{x} \in X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$. Applying Theorem 1 to the above sequence of restricted minimization problems yields the conclusion that $\hat{x}$ is a global minimizer of p on $X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$, i.e., $p(x) \geq p(\hat{x})$, $\forall x \in X_\nu \cap \mathrm{cl}\,(B_\delta(x^*))$. Since x^* is a strict local minimizer of (1), this implies $\hat{x} = x^*$. Since $\{\hat{x}_k\} \to x^*$, this implies that x_k belongs to the open ball $B_\delta(x^*)$ for k sufficiently large. Therefore, $\hat{x}_k$ is a local minimizer of $\min_{x \in X_\nu} \hat{p}_k(x)$ for k sufficiently large since it is a global minimizer of the above problem on $B_\delta(x^*)$. $\qquad\square$

We are unable to establish the more desirable statement that a convergent sequence of local minimizers of approximations (APP_ν) converges to a local minimizer of (1) without additional assumptions (cf. Theorem 2).

The next few results work towards establishing conditions under which a convergent sequence of stationary solutions to the sequence of approximating problems (APP_ν) converges to a stationary point of (1). We make the following additional assumptions on each element of the sequence of smoothing functions $\{\phi_k\}$ for this purpose.

Assumption 5. The following conditions hold for each $k \in \mathbb{N}$ and $j \in \{1, \cdots, m\}$:

A. The derivative mapping $d\phi_{k,j}(\cdot)$ is bounded by $M'_{\phi,k,j}$ on $\mathbb{R}$, i.e., $|d\phi_{k,j}(y)| \leq M'_{\phi,k,j}$, $\forall y \in \mathbb{R}$.

B. The derivative mapping $d\phi_{k,j}(\cdot)$ is Lipschitz continuous on $\mathbb{R}$ with Lipschitz constant $L'_{\phi,k,j} \in \mathbb{R}_+$.

The above assumptions are mild since we let the constants $M'_{\phi,k,j}$ and $L'_{\phi,k,j}$ depend on the sequence index k. Note that in light of Assumption 4A, Assumption 5A is equivalent to the assumption that $\phi_{k,j}(\cdot)$ is Lipschitz continuous on $\mathbb{R}$ with Lipschitz constant $M'_{\phi,k,j}$. We make the following assumptions on the cumulative distribution function of a 'scaled version' of the constraint functions g and on the sequence $\{\phi_{k,j}\}$ of smoothing functions to establish Proposition 6 and Theorem 2.

Assumption 6. The following conditions on the constraint functions g, the distribution of ξ, and the sequences $\{\phi_{k,j}\}$ of smoothing functions hold for each $k \in \mathbb{N}$ and $\nu \in \mathbb{R}$ of interest:

A. The sequence of smoothing parameters $\{\tau_k\}$ satisfies

$$\tau_{k+1,j} = \tau_c \tau_{k,j} \text{ with } \tau_{1,j} > 0, \quad \forall j \in \{1, \cdots, m\}, \tag{2}$$

for some constant factor $\tau_c \in (0, 1)$, i.e., for each $j \in \{1, \cdots, m\}$, the smoothing parameters are chosen from a monotonically decreasing geometric sequence (with limit zero). Furthermore, the underlying smoothing function ϕ is homogeneous of degree zero, i.e.,

$$\phi(\lambda y; \lambda z) = \phi(y; z), \quad \forall \lambda > 0, y \in \mathbb{R}, z \in \mathbb{R}_+.$$

9

B. For each $j \in \{1, \cdots, m\}$, let $\bar{g}_j : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ be defined by $\bar{g}_j(x, \xi) := \frac{g_j(x, \xi)}{\tau_{1,j}}$, $\forall (x, \xi) \in \mathbb{R}^n \times \mathbb{R}^d$. Let $F : \mathbb{R}^n \times R \to [0, 1]$ denote the cumulative distribution function of $\max[\bar{g}(x, \xi)]$, i.e., $F(x, \eta) := \mathbb{P}\{\max[\bar{g}(x, \xi)] \leq \eta\}$. There exists a constant $\theta > 0$ such that distribution function F is continuously differentiable on $\bar{X} \times (-\theta, \theta)$, where $\bar{X} \supset X_\nu$ is an open subset of $\mathbb{R}^n$. Furthermore, for each $\eta \in \mathbb{R}$, $F(\cdot, \eta)$ is Lipschitz continuous on X_ν with a measurable Lipschitz constant $L_F(\eta) \in \mathbb{R}_+$ that is also Lebesgue integrable, i.e., $\int L_F(\eta) d\eta < +\infty$.

C. There exists a sequence of positive constants $\{\varepsilon_k\} \downarrow 0$ such that

$$\lim_{k \to \infty} \int_{|\eta| \geq \varepsilon_k} L_F(\eta) d\phi(\eta; \tau_c^{k-1}) d\eta = 0 \quad \text{and} \quad \lim_{k \to \infty} \phi(\varepsilon_k; \tau_c^{k-1}) - \phi(-\varepsilon_k; \tau_c^{k-1}) = 1.$$

Assumption 6A is a mild assumption on the choice of the smoothing functions ϕ_k that is satisfied by all three examples in Section 3.2. When this assumption is made in conjunction with Assumption 4B, the approximation $\hat{p}_k(x) := \mathbb{E}[\max[\phi_k(g(x, \xi))]]$ can be rewritten as $\hat{p}_k(x) = \mathbb{E}[\phi(\max[\bar{g}(x, \xi)]; \tau_c^{k-1})]$, which lends the following interpretation: the sequence of approximations $\{\hat{p}_k\}$ can be thought of as having been constructed using the same sequence of smoothing parameter values $\{\tau_{k,j}\} = \{\tau_c^{k-1}\}$ for each constraint j on a rescaled version $\bar{g}_j$ of the constraint function g_j. The Lipschitz continuity assumption in Assumption 6B is mild (and is similar to Assumption 4 of Shan et al. [57]). The assumption of local continuous differentiability of the distribution function F in Assumption 6B is quite strong (this assumption is similar to Assumption 4 of Hong et al. [34]). A consequence of this assumption is that the function $p(x) \equiv 1 - F(x, 0)$ is continuously differentiable on $\bar{X}$. Finally, note that Assumption 6C is mild, see Section 3.2 for examples that (trivially) satisfy it. When made along with Assumption 4, the first part of this assumption ensures that the mapping $d\phi(\cdot; \tau_c^{k-1})$ approaches the 'Dirac delta function' sufficiently rapidly (cf. Remark 3.14 of Ermoliev et al. [28]).

The following result ensures that the Clarke gradient of the approximation $\hat{p}_k$ is well defined.

Lemma 2. Suppose Assumptions 3, 4, and 5 hold. Then for each $k \in \mathbb{N}$:

1. $\phi_{k,j}(g_j(\cdot, \xi))$ is Lipschitz continuous on X_ν with Lipschitz constant $M'_{\phi,k,j} L_{g,j}(\xi)$ for each $\xi \in \Xi$ and $j \in \{1, \cdots, m\}$.

2. $\hat{p}_k$ is Lipschitz continuous on X_ν with Lipschitz constant $\mathbb{E}\left[\max_{j \in \{1, \cdots, m\}} \left[M'_{\phi,k,j} L_{g,j}(\xi)\right]\right]$.

Proof.   1. For any $x, y \in X_\nu$, $j \in \{1, \cdots, m\}$, and $\xi \in \Xi$, we have

$$|\phi_{k,j}(g_j(y, \xi)) - \phi_{k,j}(g_j(x, \xi))| \leq M'_{\phi,k,j}|g_j(y, \xi) - g_j(x, \xi)| \leq M'_{\phi,k,j} L_{g,j}(\xi)\|y - x\|,$$

where the first step follows from Assumptions 4A and 5A and the mean-value theorem, and the second step follows from Assumption 3A.

2. For any $x, y \in X_\nu$, we have

$$
\begin{aligned}
|\hat{p}_k(y) - \hat{p}_k(x)| &= \left|\mathbb{E}[\max[\phi_k(g(y, \xi))] - \max[\phi_k(g(x, \xi))]]\right| \\
&\leq \mathbb{E}[\max[|\phi_k(g(y, \xi)) - \phi_k(g(x, \xi))|]] \\
&\leq \mathbb{E}\left[\max_j \left[M'_{\phi,k,j} L_{g,j}(\xi)\right]\right]\|y - x\|,
\end{aligned}
$$

which completes the proof. $\square$

The next result characterizes the Clarke generalized gradient of the approximating objectives $\hat{p}_k$.

Proposition 5. Suppose Assumptions 3, 4, and 5 hold. Then the Clarke generalized gradient of $\hat{p}_k$ can be expressed as

$$\partial \hat{p}_k(x) = \mathbb{E}[\text{co}\{\Gamma_k(x, \xi)\}],$$

where $\Gamma_k : \mathbb{R}^n \times \mathbb{R}^d \rightrightarrows \mathbb{R}^n$ is defined as $\Gamma_k(x, \xi) := \{\nabla_x \phi_{k,l}(g_l(x, \xi)) : l \in \mathcal{A}(x, \xi)\}$, $\mathcal{A}(x, \xi)$ denotes the set of active indices $l \in \{1, \cdots, m\}$ at which $\max[\phi_k(g(x, \xi))] = \phi_{k,l}(g_l(x, \xi))$, and the expectation above is to be interpreted in the sense of Definition 7.39 of Shapiro et al. [58].

Proof. From the corollary to Proposition 2.2.1 of Clarke [17], we have that $\phi_k\left(g(\cdot,\xi)\right)$ is strictly differentiable. Theorem 2.3.9 of Clarke [17] then implies that $\max\left[\phi_k\left(g(\cdot,\xi)\right)\right]$ is Clarke regular. The stronger assumptions of Theorem 2.7.2 of Clarke [17] are satisfied because of Assumption 4, Lemma 2, and the regularity of $\max\left[\phi_k\left(g(\cdot,\xi)\right)\right]$, which then yields

$$\partial\hat{p}_k(x) = \partial\mathbb{E}\big[\max\left[\phi_k\left(g(x,\xi)\right)\right]\big] = \mathbb{E}\big[\partial_x \max\left[\phi_k\left(g(x,\xi)\right)\right]\big].$$

Noting that $\phi_k\left(g(\cdot,\xi)\right)$ is Clarke regular from Proposition 2.3.6 of Clarke [17], the stated equality then follows from Proposition 2.3.12 of Clarke [17]. □

The next result, similar to Lemma 4.2 of Shan et al. [57], helps characterize the accumulation points of stationary solutions to the sequence of approximations (APP_ν).

Proposition 6. Suppose Assumptions 3, 4, 5, and 6 hold. Then

$$\limsup_{\substack{x\to\bar{x}\\k\to\infty}} \partial\hat{p}_k(x) + N_{X_\nu}(x) \subset \{\nabla p(\bar{x})\} + N_{X_\nu}(\bar{x}).$$

Proof. See Appendix B. □

The following key result is an immediate consequence of the above proposition (cf. Shan et al. [57, Theorem 4.2]), and establishes convergence of Clarke stationary solutions of the sequence of approximating problems (APP_ν) to those of the true problem (1).

Theorem 2. Suppose Assumptions 1, 3, 4, 5, and 6 hold. Let $\{x_k\}$ be a sequence of stationary solutions to the sequence of approximating problems $\min_{x\in X_\nu}\hat{p}_k(x)$. Then every accumulation point of $\{x_k\}$ is an stationary solution to $\min_{x\in X_\nu} p(x)$.

Proof. From the corollary to Proposition 2.4.3 of Clarke [17], we have $0 \in \partial\hat{p}_k(x_k) + N_{X_\nu}(x_k)$ by virtue of the stationarity of x_k for $\min_{x\in X_\nu}\hat{p}_k(x)$. The stated result follows from Proposition 6 since $0 \in \{\nabla p(\bar{x})\} + N_{X_\nu}(\bar{x})$ for any accumulation point $\bar{x}$ of $\{x_k\}$ (see Theorem 5.37 of Rockafellar and Wets [54]). □

3.2 Examples of smooth approximations

We present a few examples of sequences of smooth approximations of the step function that satisfy Assumptions 4, 5, and 6. Throughout this section, we let $\{\tau_{k,j}\}$, $j\in\{1,\cdots,m\}$, denote sequences of *positive* reals satisfying (2).

Example 2. This example is from Example 5.1 of Shan et al. [57]. The sequence of approximations

$$\phi_{k,j}(y) = \begin{cases} 0 & \text{if } y < -\tau_{k,j}, \\ 1 - 2\left(\frac{y}{\tau_{k,j}}\right)^3 - 3\left(\frac{y}{\tau_{k,j}}\right)^2 & \text{if } -\tau_{k,j} \le y \le 0, \\ 1 & \text{if } y > 0 \end{cases}$$
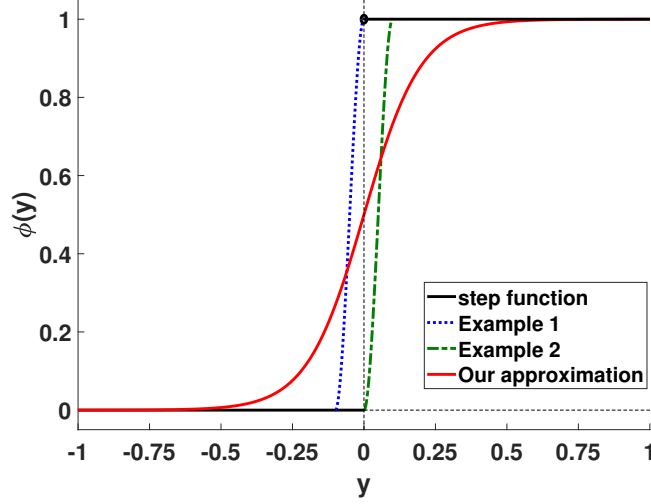
of the step function satisfy Assumptions 4 and 5. Assumption 6A is easily verified, and Assumption 6C trivially holds with $\varepsilon_k = \tau_c^{k-1}$.

Example 3. This example is based on the above example and the outer-approximation framework of Geletu et al. [29]. The sequence of approximations

$$\phi_{k,j}(y) = \begin{cases} 0 & \text{if } y < 0, \\ 1 - 2\left(\frac{y-\tau_{k,j}}{\tau_{k,j}}\right)^3 - 3\left(\frac{y-\tau_{k,j}}{\tau_{k,j}}\right)^2 & \text{if } 0 \le y \le \tau_{k,j}, \\ 1 & \text{if } y > \tau_{k,j} \end{cases}$$

of the step function satisfy the strong form of Assumption 4 and Assumption 5. Assumption 6A is easily verified, and Assumption 6C trivially holds with $\varepsilon_k = \tau_c^{k-1}$.

Figure 2: Illustration of the examples of smoothing functions in Section 3.2.

The next example is the sequence of smooth approximations $\{\phi_k\}$ that we adopt in this work.

Example 4. The sequence of approximations

$$\phi_{k,j}(y) = \frac{1}{1 + \exp\left(-\frac{y}{\tau_{k,j}}\right)}$$

of the step function satisfy Assumptions 4 and 5 (see Proposition 7). Assumption 6A is easily verified, and Assumption 6C could be satisfied with $\varepsilon_k = \tau_c^{\beta(k-1)}$ for some constant $\beta \in (0,1)$ (depending on the distribution function F, cf. Remark 3.14 of Ermoliev et al. [28]).

Figure 2 illustrates the above smoothing functions with the common smoothing parameter $\tau = 0.1$. We refer the reader to the works of Shan et al. [57], Geletu et al. [29], and Cao and Zavala [13] for other examples of 'smooth' approximations that can be accommodated within our framework. The next result estimates some important constants related to our smooth approximation in Example 4.

Proposition 7. The sequence of approximations $\{\phi_k\}$ in Example 4 satisfies Assumption 5 with constants $M'_{\phi,k,j} \leq 0.25\tau_{k,j}^{-1}$ and $L'_{\phi,k,j} \leq 0.1\tau_{k,j}^{-2}$, $\forall k \in \mathbb{N}$, $j \in \{1, \cdots, m\}$.

Proof. The bounds follow by noting that for each $y \in \mathbb{R}$:

$$d\phi_k(y) = \frac{\tau_{k,j}^{-1}}{\left[\exp(0.5\tau_{k,j}^{-1}y) + \exp(-0.5\tau_{k,j}^{-1}y)\right]^2} \leq 0.25\tau_{k,j}^{-1}, \text{ and}$$

$$\left|d^2\phi_{k,j}(y)\right| = \tau_{k,j}^{-2} \left| \frac{\exp\left(\frac{y}{2\tau_{k,j}}\right) - \exp\left(-\frac{y}{2\tau_{k,j}}\right)}{\left[\exp\left(\frac{y}{2\tau_{k,j}}\right) + \exp\left(-\frac{y}{2\tau_{k,j}}\right)\right]^3} \right| \leq 0.1\tau_{k,j}^{-2},$$

where the final inequality is obtained by maximizing the function $x \mapsto \left| \frac{x - \frac{1}{x}}{\left(x + \frac{1}{x}\right)^3} \right|$ on $\mathbb{R}$. $\qquad \square$

4   Proposed algorithm

Our proposal for approximating the efficient frontier of (CCP) involves solving a sequence of approximations (APP$_\nu$) constructed using the sequence of smooth approximations of the step function introduced

in Example 4. We use the projected stochastic subgradient algorithm of Davis and Drusvyatskiy [20] to obtain an approximately stationary solution to (APP$_\nu$) for each value of the objective bound ν and each element of the sequence of smoothing parameters $\{\tau_k\}$ (note that we suppress the constraint index j in our notation from here on unless necessary). Section 4.1 outlines a conceptual algorithm for approximating the efficient frontier of (CCP) using the projected stochastic subgradient algorithm, and Sections 4.2 and 4.3 outline our proposal for estimating some parameters of the algorithm in Section 4.1 that are important for a good practical implementation.

4.1 Outline of the algorithm

Algorithm 1 presents a prototype of our approach for constructing an approximation of the efficient frontier of (CCP). It takes as input an initial guess $\hat{x}^0$, a sequence of objective bounds $\{\nu^i\}_{i=1}^M$, the maximum number of 'runs', R, of the projected stochastic subgradient method, and the maximum number of iterations of the method in each run, N_{max}. For each objective bound ν^i, we first determine a sequence of smoothing parameters and step lengths $\{(\tau_k, \gamma_k)\}_{k=1}^K$ for the sequence of approximations (APP$_\nu$). The middle loop of Algorithm 1 then iteratively solves the sequence of approximations (APP$_\nu$) with $\nu = \nu^i$ and smoothing parameters $\{\tau_k\}$ by using the solution from the previous approximation as its initial guess. The innermost loop of Algorithm 1 monitors the progress of the stochastic subgradient method over several 'runs'. Each run generates a candidate solution $\bar{x}^{i,k,r}$ with risk level $\bar{\alpha}^{i,k,r}$ (we omit the superscript r in Algorithm 1 for simplicity), and we move on to the next approximation in the sequence either if insufficient progress is made over the last few runs, or if the number of runs exceeds its limit. Since the choice of the smoothing parameters $\{\tau_k\}$, step lengths $\{\gamma_k\}$, and the termination criteria employed within Algorithm 1 are important for its practical performance, we present our proposal for estimating/setting these algorithmic parameters from the problem data in Section 4.2. First, we split Algorithm 1 into several component algorithms so that it is easier to parse.

Algorithm 2 outlines our proposal for approximating the efficient frontier of (CCP). This algorithm takes as its inputs an initial guess $\hat{x}^0 \in X$, an initial objective bound $\bar{\nu}^0$ of interest, an objective spacing $\tilde{\nu}$ for discretizing the efficient frontier, and a lower bound α_{low} on risk levels of interest that is used as a termination criterion. Section 4.2 prescribes ways to determine a good initial guess $\hat{x}^0$ and initial bound $\bar{\nu}^0$, whereas Section 5.1 lists our setting for $\tilde{\nu}$, α_{low}, and other algorithmic parameters. The preprocessing step uses Algorithm 4 to determine a suitable initial sequence of smoothing parameters $\{\tau_k\}$ based on the problem data, and uses this choice of $\{\tau_k\}$ to determine an initial sequence of step lengths for the stochastic subgradient algorithm (see Section 4.2). Good choices of both parameters are critical for our proposal to work well. The optimization phase of Algorithm 2 uses Algorithm 3 to solve the sequence of approximations (APP$_\nu$) for different choices of the objective bound $\nu = \bar{\nu}^i$ (note that Algorithm 3 rescales its input smoothing parameters $\{\bar{\tau}_k\}$ at the initial guess $\bar{x}^i$ for each bound $\bar{\nu}^i$). Finally, Algorithm 8 in the appendix adapts Algorithm 2 to solve (CCP) approximately for a given risk level $\hat{\alpha}$.

Algorithm 3 solves a sequence of approximations (APP$_\nu$) for a given objective bound ν using the projected stochastic subgradient algorithm of Davis and Drusvyatskiy [20] and an adaptation of the two-phase randomized stochastic projected gradient method of Ghadimi et al. [30]. This algorithm begins by rescaling the smoothing parameters $\{\bar{\tau}_k\}$ at the initial guess $\bar{x}^0$ using Algorithm 4 to ensure that the sequence of approximations (APP$_\nu$) are 'well-scaled'[1] at $\bar{x}^0$. The optimization phase then solves each element of the sequence of approximations (APP$_\nu$) using two loops: the inner loop employs a mini-batch version[2] of the algorithm of Davis and Drusvyatskiy [20] to solve (APP$_\nu$) for a given smoothing parameter τ_k, and the outer loop assesses the progress of the inner loop across multiple 'runs/replicates' (the initialization strategy for each run is based on `Algorithm 2-RSPG-V` of Ghadimi et al. [30]). Finally, the initial guess for the next approximating problem (APP$_\nu$) is set to be a solution corresponding to the smallest estimate of the risk level determined thus far. This initialization step is important for the next approximating problem in the sequence to be well-scaled at its initial guess - this is why we do not solve a 'single tight approximating problem' (APP$_\nu$) that can be hard to initialize (cf. Figure 2), but instead solve a sequence of approximations to approximate the solution of (1). Note that while the number of iterations N of the inner projected stochastic subgradient loop is chosen at random as required by

---

[1] We make this notion more precise in Section 4.2.

[2] Although Davis and Drusvyatskiy [20] establish that mini-batching is not necessary for the convergence of the projected stochastic subgradient algorithm, a small mini-batch can greatly enhance the performance of the algorithm in practice.

Algorithm 1 Prototype of algorithm for approximating the efficient frontier of (CCP)

1: **Input**: initial point $\hat{x}^0 \in X$, sequence of objective bounds $\{\nu^i\}_{i=1}^M$, maximum number of projected stochastic subgradient iterations, N_{max}, and number of 'runs' of the stochastic subgradient method, R.

2: **Output**: pairs $\{(\nu^i, \hat{\alpha}^i)\}_{i=1}^M$ of objective values and risk levels on the approximation of the efficient frontier along with associated solutions $\{\hat{x}^i\}_{i=1}^M$.

3: **for** iteration $i = 1$ to M **do**   ▷ Loop through the objective bounds $\{\nu^i\}$

4:     Set initial guess $\bar{x}^{i,0} := \hat{x}^{i-1}$.

5:     Determine sequence of smoothing parameters and step lengths $\{(\tau_k, \gamma_k)\}_{k=1}^K$.

6:     **for** approximation $k = 1$ to K **do**   ▷ Loop through the sequence of smooth approximations

7:         Set initial guess $\tilde{x}^{k,0} := \bar{x}^{i,k-1}$ for approximation k.

8:         **for** run $r = 1$ to R **do**   ▷ Runs of the stochastic subgradient method

9:             Initialize the first iterate $x^1 := \tilde{x}^{k,0}$.

10:             Choose number of iterations N for this run uniformly at random from $\{1, \cdots, N_{max}\}$.

11:             Execute N iterations of the projected stochastic subgradient algorithm of [20] for solving (APP$_\nu$) with
                $\nu = \nu^i$ and smoothing parameter τ_k using step length γ_k.

12:             Set $\bar{x}^{i,k}$ to be the last iterate of the stochastic subgradient method, and estimate its risk level $\bar{\alpha}^{i,k}$.

13:             Update incumbents $(\hat{\alpha}^i, \hat{x}^i)$ with $(\bar{\alpha}^{i,k}, \bar{x}^{i,k})$ if needed. Update $\tilde{x}^{k,0} = \bar{x}^{i,k}$.

14:             If there is insufficient improvement in the incumbent $\hat{\alpha}^i$ over the last few runs, exit the 'runs' loop.

15:         **end for**

16:         Update initial guess for the next approximation in the sequence to the incumbent solution, i.e., set $\bar{x}^{i,k} = \hat{x}^i$.

17:     **end for**

18: **end for**

Algorithm 2 Approximating the efficient frontier of (CCP)

1: **Input**: initial point $\hat{x}^0 \in X$, initial objective bound $\bar{\nu}^0$, objective spacing $\tilde{\nu}$, and lower bound on risk levels of interest $\alpha_{low} \in (0, 1)$.

2: **Set algorithmic parameters**: mini-batch size M, maximum number of iterations N_{max}, minimum and maximum number of 'runs', R_{min} and R_{max}, parameters for determining and updating step length, termination criteria, fixed sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$ from $\mathcal{P}$ for estimating risk levels of candidate solutions, and sequence of smoothing parameters $\{\bar{\tau}_k\}_{k=1}^K$ for some $K \in \mathbb{N}$.

3: **Output**: set of pairs $\{(\bar{\nu}^i, \bar{\alpha}^i)\}$ of objective values and risk levels that can be used to approximate the efficient frontier and associated solutions $\{\hat{x}^i\}$.

4: **Preprocessing**: determine smoothing parameters $\{\tau_k\}_{k=1}^K$ scaled at proj$(\hat{x}^0, X_{\bar{\nu}^0})$ using Algorithm 4, and an initial sequence of step lengths $\{\bar{\gamma}_k\}_{k=1}^K$ for this sequence of approximating problems (APP$_\nu$) with $\nu = \bar{\nu}^0$ using Algorithm 5.

5: **Optimization Phase**:

6: Initialize iteration count $i = 0$.

7: **repeat**

8:     Update iteration count $i \leftarrow i + 1$ and set objective bound $\bar{\nu}^i = \bar{\nu}^0 - (i-1)\tilde{\nu}$.

9:     Obtain $(\bar{\alpha}^i, \hat{x}^i)$ by solving the sequence of approximations (APP$_\nu$) using Algorithm 3 with objective bound $\nu = \bar{\nu}^i$,
        scaling parameters $\{\bar{\tau}_k\}_{k=1}^K$, the above algorithmic parameter settings, and $\bar{x}^i := $ proj$(\hat{x}^{i-1}, X_{\bar{\nu}^i})$ as the
        initial guess.

10: **until** $\bar{\alpha}^i \leq \alpha_{low}$

---

the theory in Davis and Drusvyatskiy [20], a practical alternative is to choose a deterministic number of iterations N and set $\bar{x}^i$ to be the solution at the final iterate. Line 17 of Algorithm 3 heuristically updates the step length to ensure good practical performance. Additionally, in contrast with the proposal of `Algorithm 2-RSPG-V` of Ghadimi et al. [30] that chooses the solution over the multiple runs that is closest to stationarity, motivated by practical application, we return the solution corresponding to the smallest found risk level. Because of these heuristic steps, the algorithm would require slight modification (removal of the heuristic step-size update method and different choice of the solution from multiple runs) to be assured to have the convergence guarantees of Davis and Drusvyatskiy [20]. However, as our emphasis in this paper is on deriving a practical algorithm and testing it empirically, we keep the algorithm description matching what we have implemented.

Algorithm 3 Generating a point approximately on the efficient frontier

---

1: **Input**: objective bound ν, initial guess $\bar{x}^0 \in X_\nu$, mini-batch size M, maximum number of iterations N_{max}, minimum and maximum number of 'runs', R_{min} and $R_{max} \geq R_{min}$, initial step lengths $\{\bar{\gamma}_k\}_{k=1}^K$, parameters for updating step length, run termination criteria, sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$ from $\mathcal{P}$ for estimating risk levels, and sequence of smoothing parameters $\{\bar{\tau}_k\}_{k=1}^K$.
2: **Output**: Smallest estimate of the risk level $\hat{\alpha}$ and corresponding solution $\hat{x} \in X_\nu$.
3: **Preprocessing**: determine smoothing parameters $\{\tau_k\}_{k=1}^K$ scaled at $\bar{x}^0$ using Algorithm 4.
4: **Initialize**: Best found solution $\hat{x} = \bar{x}^0$ and estimate of its risk level $\hat{\alpha}$ using the sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$.
5: **Optimization Phase**:
6: **for** approximation $k = 1$ to K **do**
7:     Initialize run count $i = 0$ and step length $\gamma_k = \bar{\gamma}_k$.
8:     **repeat**   ▷ Begin outer loop of 'runs/replicates'
9:         Update $i \leftarrow i + 1$, and initialize the first iterate $x^1 := \bar{x}^{i-1}$.
10:         Choose number of iterations N uniformly at random from $\{1, \cdots, N_{max}\}$.
11:         **for** iteration $l = 1$ to $N - 1$ **do**   ▷ Inner projected stochastic subgradient loop
12:             Draw an i.i.d. sample $\{\xi^{l,q}\}_{q=1}^M$ of ξ from $\mathcal{P}$.
13:             Estimate an element $G(x^l) \in \mathbb{R}^n$ of the subdifferential of the objective of (APP_ν) with smoothing
                    parameters τ_k at x^l using the mini-batch $\{\xi^{l,q}\}_{q=1}^M$.
14:             Update $x^{l+1} := \text{proj}\left(x^l - \gamma_k G(x^l), X_\nu\right)$.
15:         **end for**
16:         Set $\bar{x}^i = x^N$, and estimate its risk level $\bar{\alpha}^i$ using the sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$.
17:         Update step length γ_k using Algorithm 6, and the incumbents $(\hat{\alpha}, \hat{x})$ with $(\bar{\alpha}^i, \bar{x}^i)$ if $\hat{\alpha} > \bar{\alpha}^i$.
18:     **until** run termination criteria are satisfied (see Algorithm 6)
19:     Update initial guess $\bar{x}^0 = \hat{x}$ for the next approximation.
20: **end for**

In the remainder of this section, we verify that the assumptions of Davis and Drusvyatskiy [20] hold to justify the use of their projected stochastic subgradient method for solving the sequence of approximations (APP_ν). First, we verify that the objective function $\hat{p}_k$ of (APP_ν) with smoothing parameter τ_k is a weakly convex function on X_ν. Next, we verify that Assumption (A3) of Davis and Drusvyatskiy [20] holds. Note that a function $h : \mathbb{R}^d \to \overline{\mathbb{R}}$ is said to be ρ_h-weakly convex (for some $\rho_h \geq 0$) if $h + 0.5\rho_h \|\cdot\|^2$ is convex on $\mathbb{R}^d$ (see Definition 4.1 of Drusvyatskiy and Paquette [23] and the surrounding discussion).

Proposition 8. Suppose Assumptions 3, 4 and 5 hold. Then $\hat{p}_k(\cdot)$ is $\bar{L}_k$-weakly convex on X_ν, where $\bar{L}_k$ is the Lipschitz constant of the Jacobian $\mathbb{E}\left[\nabla \phi_k\left(g(\cdot, \xi)\right)\right]$ on X_ν.

Proof. Follows from Lemma 4.2 of Drusvyatskiy and Paquette [23]. □

The following result is useful for bounding $\hat{p}_k$'s weak convexity parameter in terms of known constants.

Proposition 9. Suppose Assumptions 3, 4 and 5 hold. Then

1. For any $\xi \in \Xi$, the Lipschitz constant $L_{k,j}(\xi)$ of $\nabla \phi_{k,j}\left(g_j(\cdot, \xi)\right)$ on X_ν satisfies

$$L_{k,j}(\xi) \leq M'_{\phi,k,j} L'_{g,j}(\xi) + L'_{\phi,k,j} L^2_{g,j}(\xi).$$

2. The Lipschitz constant $\bar{L}_k$ of the Jacobian $\nabla\mathbb{E}\big[\phi_k\left(g(\cdot,\xi)\right)\big]$ on X_ν satisfies

$$\bar{L}_k \leq \mathbb{E}\Big[\big(\sum_{j=1}^m L_{k,j}^2(\xi)\big)^{\frac{1}{2}}\Big].$$

Proof. 1. Follows by using the chain rule $\nabla\phi_{k,j}\left(g_j(\cdot,\xi)\right) = d\phi_{k,j}\left(g_j(\cdot,\xi)\right)\nabla g_j(\cdot,\xi)$ and bounding the Lipschitz constant of the above product on X_ν using Assumptions 3A, 3B, and 5 and standard arguments.

2. From part one of Lemma 2 and Theorem 7.44 of Shapiro et al. [58], we have $\nabla\mathbb{E}\big[\phi_k\left(g(x,\xi)\right)\big] = \mathbb{E}\big[\nabla_x\phi_k\left(g(x,\xi)\right)\big]$, $\forall x \in X_\nu$. The stated result then follows from the first part of this proposition and the fact that the Frobenius norm of a matrix provides an upper bound on its spectral norm, where the existence of the quantity $\mathbb{E}\big[\big(\sum_{j=1}^m L_{k,j}^2(\xi)\big)^{\frac{1}{2}}\big]$ follows from the fact that $\sqrt{\sum_j L_{k,j}^2(\xi)} \leq \sum_j L_{k,j}(\xi)$, Assumptions 3A and 3B, and Lebesgue's dominated convergence theorem. $\qquad\square$

We derive alternative results regarding the weak convexity parameter of the approximation $\hat{p}_k$ when (CCP) is used to model recourse formulations in Appendix C. We now establish that Assumption (A3) of Davis and Drusvyatskiy [20] holds.

Proposition 10. Suppose Assumptions 3, 4, and 5 hold. Let $G(x,\xi)$ be a stochastic B-subdifferential element of $\hat{p}_k(\cdot)$ at $x \in X_\nu$, i.e., $\mathbb{E}\big[G(x,\xi)\big] \in \partial_B\mathbb{E}\big[\max\left[\phi_k\left(g(x,\xi)\right)\right]\big]$. Then

$$\mathbb{E}\big[\|G(x,\xi)\|^2\big] \leq \max_{j\in\{1,\cdots,m\}} \left(M'_{\phi,k,j}\right)^2 \sigma_{g,j}^2.$$

Proof. Follows from Assumptions 3C and 5 and the fact that $G(x,\xi) = d\phi_{k,j}(g_j(x,\xi))\nabla_x g_j(x,\xi)$ for some active constraint index $j \in \{1,\cdots,m\}$ such that $\max\left[\phi_k\left(g(x,\xi)\right)\right] = \phi_{k,j}\left(g_j(x,\xi)\right)$. $\qquad\square$

4.2 Estimating the parameters of Algorithm 2

This section outlines our proposal for estimating some key parameters of Algorithm 2. We present pseudocode for determining suitable smoothing parameters and step lengths, and for deciding when to terminate Algorithm 3 with a point on our approximation of the efficient frontier.

Algorithm 4 rescales the sequence of smoothing parameters $\{\bar{\tau}_k\}$ based on the values assumed by the constraints g at a reference point $\bar{x}$ and a Monte Carlo sample of the random vector ξ. The purpose of this rescaling step is to ensure that the initial approximation (APP$_\nu$) with smoothing parameter τ_1 is 'well-scaled' at the initial guess $\bar{x}$. By 'well-scaled', we mean that the smoothing parameter τ_1 is chosen (neither too large, nor too small) such that the realizations of the scaled constraint functions $\bar{g}_j(\bar{x},\xi) := \frac{g_j(\bar{x},\xi)}{\tau_{1,j}}$, $j \in \{1,\cdots,m\}$, are mostly supported on the interval $[-1,1]$ but not concentrated at zero. This rescaling ensures that stochastic subgradients at $\bar{x}$ are not too small in magnitude. If $\bar{x}$ is 'near' a stationary solution, then we can hope that the approximation remains well-scaled over a region of interest and that our 'first-order algorithm' makes good progress in a reasonable number of iterations. Note that the form of the scaling factor β_j in Algorithm 4 is influenced by our choice of the smoothing functions in Example 4. Algorithm 5 uses the step length rule in Davis and Drusvyatskiy [20, Page 6] with the trivial bound '$R = 1$' and sample estimates of the weak convexity parameter of $\hat{p}_k$ and the parameter σ_k^2 related to its stochastic subgradient. Since stochastic approximation algorithms are infamous for their sensitivity to the choice of step lengths (see Section 2.1 of Nemirovski et al. [44], for instance), Algorithm 6 prescribes heuristics for updating the step length based on the progress of Algorithm 3 over multiple runs. These rules increase the step length if insufficient progress has been made, and decrease the step length if the risk level has increased too significantly. Algorithm 6 also suggests the following heuristic for terminating the 'runs loop' within Algorithm 3: terminate either if the upper limit on the number of runs is hit, or if insufficient progress has been made over the last few runs.

Algorithm 4 Scaling the smoothing parameters

1: **Input**: reference point $\bar{x} \in X$, and sequence of smoothing parameters $\{\bar{\tau}_k\}_{k=1}^K$.
2: **Set algorithmic parameters**: number of samples N_{scale}, scaling tolerance $s_{tol} > 0$, and scaling factor $\omega > 0$.
3: **Output**: smoothing parameters $\{\tau_k\}_{k=1}^K$ scaled at $\bar{x}$.
4: Draw a sample $\{\xi^i\}_{i=1}^{N_{scale}}$ of ξ from $\mathcal{P}$.
5: For each $j \in \{1, \cdots, m\}$ and k, set $\tau_{k,j} = \beta_j \bar{\tau}_k$, where $\beta_j := \omega \max \left\{ \text{median} \left\{ |g_j(\bar{x}, \xi^i)| \right\}, s_{tol} \right\}$.

Algorithm 5 Determining initial step length

1: **Input**: objective bound ν, reference point $\bar{x} \in X_\nu$, mini-batch size M, maximum number of iterations N_{max}, minimum number of runs R_{min}, and smoothing parameters $\{\tau_k\}_{k=1}^K$.
2: **Set algorithmic parameters**: number of samples for estimating weak convexity parameter N_{wc}, sampling radius $r > 0$, number of samples for estimating 'variability' of stochastic gradients N_{var}, and batch size for computing expectations N_{batch}.
3: **Output**: initial step lengths $\{\bar{\gamma}_k\}_{k=1}^K$.
4: Estimate the weak convexity parameter ρ_k of the objective of (APP$_\nu$) with smoothing parameter τ_k using i. N_{wc} samples of pairs of points in $X_\nu \cap B_r(\bar{x})$, and ii. mini-batches of ξ for estimating stochastic gradients (see Section 4.3).
5: Estimate $\sigma_k^2 := \max_{x \in X_\nu} \mathbb{E}\left[\|G(x, \xi)\|^2\right]$ using i. N_{var} sample points $x \in X_\nu \cap B_r(\bar{x})$, and ii. stochastic B-subdifferential elements $G(x, \xi)$ of the subdifferential of the objective of (APP$_\nu$) with smoothing parameter τ_k and mini-batches of ξ (see Section 4.3).
6: Set $\bar{\gamma}_k := \dfrac{1}{\sqrt{\rho_k \sigma_k^2 (N_{max} + 1) R_{min}}}$.

4.3   Other practical considerations

We list some other practical considerations for implementing Algorithm 2 below.

Setting an initial guess: The choice of the initial point $\hat{x}^0$ for Algorithm 2 is important especially because Algorithm 3 does not guarantee finding global solutions to (APP$_\nu$). Additionally, a poor choice of the initial bound $\bar{\nu}^0$ can lead to excessive effort expended on uninteresting regions of the efficient frontier. We propose to initialize $\hat{x}^0$ and $\bar{\nu}^0$ by solving a set of tuned scenario approximation problems. For instance, Algorithm 7 in the appendix can be solved with $M = 1$, $R = 1$, and a tuned sample size N_1 (tuned such that the output $(\bar{\nu}^{1,1}, \bar{\alpha}^{1,1}, \bar{x}^{1,1})$ of this algorithm corresponds to an interesting region of the efficient frontier) to yield the initial guess $\hat{x}^0 = \bar{x}^{1,1}$ and $\bar{\nu}^0 = \bar{\nu}^{1,1}$. Note that it may be worthwhile to solve the scenario approximation problems to global optimality to obtain a good initialization $(\hat{x}^0, \bar{\nu}^0)$, e.g., when the number of scenarios N_1 is not too large (in which case they may be solved using off-the-shelf solvers/tailored decomposition techniques in acceptable computation times). Case study 4 in Section 5 provides an example where globally optimizing the scenario approximation problems to obtain an initial guess yields a significantly better approximation of the efficient frontier.

Computing projections: Algorithm 2 requires projecting onto the sets X_ν for some $\nu \in \mathbb{R}$ multiple times while executing Algorithm 3. Thus, it may be beneficial to implement tailored projection subroutines [14], especially for sets X_ν with special structures [18] (also see Case study 2 in Section 5). If the set X_ν is a polyhedron, then projection onto X_ν may be carried out by solving a quadratic program.

Computing stochastic subgradients: Given a point $\bar{x} \in X_\nu$ and a mini-batch $\{\xi^l\}_{l=1}^M$ of the random vector ξ, a stochastic element $G(\bar{x}, \xi)$ of the B-subdifferential of the objective of (APP$_\nu$) with smoothing parameter τ_k at $\bar{x}$ can be obtained as

$$G(\bar{x}, \xi) = \frac{1}{M} \sum_{l=1}^M d\phi_{k,j(l)} \left(g_{j(l)}(\bar{x}, \xi^l) \right) \nabla_x g_{j(l)}(\bar{x}, \xi^l),$$

Algorithm 6 Updating step length and checking termination

1: **Input**: current run number r, minimum and maximum number of 'runs', R_{min} and R_{max}, step length γ_k, estimated risk levels $\{\bar{\alpha}^i\}_{i=1}^r$, and the risk level at the initial guess $\hat{\alpha}^0$.

2: **Set algorithmic parameters**: step length checking frequency N_{check}, run termination checking parameter N_{term}, relative increase in risk level $\delta_1 > 0$ beyond which we terminate, relative increase in risk level $\delta_2 > \delta_1$ beyond which we decrease the step length by factor $\gamma_{decr} > 1$, and step length increase factor $\gamma_{incr} > 1$.

3: Construct the sequence $\{\hat{\alpha}^i\}_{i=0}^r$, with $\hat{\alpha}^i := \min\{\hat{\alpha}^{i-1}, \bar{\alpha}^i\}$, of smallest risk levels at the end of each run thus far.

4: **Updating step length**:

5: **if** run number is divisible by N_{check} **then**

6:     Determine maximum relative 'decrease' in the estimated risk level

$$\hat{\delta} := \max_{i \in \{1, \cdots, N_{check}\}} \frac{\hat{\alpha}^{r-N_{check}} - \bar{\alpha}^{r+1-i}}{\hat{\alpha}^{r-N_{check}}}$$

    over the past N_{check} runs, relative to the smallest known estimate of the risk level N_{check} runs ago.

7:     **if** $\hat{\delta} \geq -\delta_1$ **then**

8:         Increase step length by factor γ_{incr}.   ▷ Insufficient decrease in estimated risk level

9:     **else if** $\hat{\delta} \leq -\delta_2$ **then**

10:         Decrease step length by factor γ_{decr}.   ▷ Unacceptable increase in estimated risk level

11:     **end if**

12: **end if**

13: **Termination check**:

14: **if** number of runs equals R_{max} **then**

15:     Terminate.

16: **else if** number of runs is at least R_{min} **then**

17:     Determine maximum relative 'decrease' in the estimated risk level over the past N_{term} runs

$$\hat{\delta} := \max_{i \in \{1, \cdots, N_{term}\}} \frac{\hat{\alpha}^{r-N_{term}} - \bar{\alpha}^{r+1-i}}{\hat{\alpha}^{r-N_{term}}}.$$

    If $\hat{\delta} < -\delta_1$, terminate due to insufficient decrease in risk level.

18: **end if**

---

where for each $l \in \{1, \cdots, M\}$, $j(l) \in \{1, \cdots, m\}$ denotes a constraint index satisfying $\max\left[\phi_k\left(g(\bar{x}, \xi^l)\right)\right] = \phi_{k,j(l)}\left(g_{j(l)}(\bar{x}, \xi^l)\right)$. Note that sparse computation of stochastic subgradients can speed Algorithm 2 up significantly.

Estimating risk levels: Given a candidate solution $\bar{x} \in X_\nu$ and a sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$ of the random vector ξ, a stochastic upper bound on the risk level $p(\bar{x})$ can be obtained as

$$\bar{\alpha} = \max_{\alpha \in [0,1]} \left\{ \alpha : \sum_{i=0}^{N_{viol}} \binom{N_{MC}}{i} \alpha^i (1-\alpha)^{N_{MC}-i} = \delta \right\},$$

where N_{viol} is the cardinality of the set $\{l : g(\bar{x}, \bar{\xi}^l) \not\leq 0\}$ and $1 - \delta$ is the required confidence level, see Nemirovski and Shapiro [43, Section 4]. Since checking the satisfaction of N_{MC} constraints at the end of each run in Algorithm 3 may be time consuming (especially for problems with recourse structure), we use a smaller sample size (determined based on the risk lower bound α_{low}) to estimate risk levels during the course of Algorithm 3 and only use all N_{MC} samples of ξ to estimate the risk level $\{\bar{\alpha}^i\}$ of the final solutions $\{\hat{x}^i\}$ in Algorithm 2. If our discretization of the efficient frontier obtained from Algorithm 2 consists of N_{EF} points each of whose risk levels is estimated using a confidence level of $1 - \delta$, then we may conclude that our approximation of the efficient frontier is 'achievable' with a confidence level of at least $1 - \delta N_{EF}$ using Bonferroni's inequality.

Estimating weak convexity parameters: Proposition 8 shows that the Lipschitz constant $\bar{L}_k$ of the Jacobian $\mathbb{E}\left[\nabla\phi_k\left(g(\cdot,\xi)\right)\right]$ on X_ν provides a conservative estimate of the weak convexity parameter ρ_k of $\hat{p}_k$ on X_ν. Therefore, we use an estimate of the Lipschitz constant $\bar{L}_k$ as an estimate of ρ_k. To avoid overly conservative estimates, we restrict the estimation of $\bar{L}_k$ to a neighborhood of the reference point $\bar{x}$ in Algorithm 5 to get at the local Lipschitz constant of the Jacobian $\mathbb{E}\left[\nabla\phi_k\left(g(\cdot,\xi)\right)\right]$.

Estimating σ_k^2: For each point x sampled from X_ν, we compute multiple realizations of the mini-batch stochastic subdifferential element $G(x,\xi)$ outlined above to estimate $\mathbb{E}\left[\|G(x,\xi)\|^2\right]$. Once again, we restrict the estimation of σ_k^2 to a neighborhood of the reference point $\bar{x}$ in Algorithm 5 to get at the local 'variability' of the stochastic subgradients.

Estimating initial step lengths: Algorithm 5 estimates initial step lengths for the sequence of approximations (APP_ν) by estimating the parameters $\{\rho_k\}$ and $\{\sigma_k^2\}$, which in turn involve estimating subgradients and the Lipschitz constants of $\{\hat{p}_k\}$. Since the numerical conditioning of the approximating problems (APP_ν) deteriorates rapidly as the smoothing parameters τ_k approach zero (see Figure 2), obtaining good estimates of these constants via sampling becomes challenging when k increases. To circumvent this difficulty, we only estimate the initial step length $\bar{\gamma}_1$ for the initial approximation (APP_ν) with smoothing parameter τ_1 by sampling, and propose the conservative initialization $\bar{\gamma}_k := \left(\frac{\tau_k}{\tau_1}\right)^2 \bar{\gamma}_1$ for $k > 1$ (see Propositions 7, 8, 9, and 10 for a justification).

5 Computational study

We present implementation details and results of our computational study in this section. We use the abbreviation 'EF' for the efficient frontier throughout this section.

5.1 Implementation details

The following parameter settings are used for testing our stochastic approximation method:

- Algorithm 2: Obtain initial guess $\hat{x}^0$ and initial objective bound $\bar{\nu}^0$ using Algorithm 7 in Appendix A.1 with $M = 1$, $R = 1$, and $N_1 = 10$. Set $\tilde{\nu} = 0.005|\bar{\nu}^0|$ (we assume that $\bar{\nu}^0 \neq 0$ for simplicity) and $\alpha_{low} = 10^{-4}$ unless otherwise specified. In addition, set $M = 20$, $N_{max} = 1000$, $R_{min} = 10$, $R_{max} = 50$, $K = 3$, and $\{\bar{\tau}_k\} = \{(0.1)^{k-1}\}$ (i.e., $\tau_c = 0.1$ in Equation (2)).

- Algorithm 4: $N_{scale} = 10^4$, $s_{tol} = 10^{-6}$, and $\omega = 1$.

- Algorithm 5: $N_{wc} = N_{var} = 200$, $N_{batch} = 20$, and $r = 0.1\|\bar{x}\|$ (assuming that $\bar{x} \neq 0$).

- Algorithm 6: $N_{check} = 3$, $N_{term} = 5$, $\delta_1 = 10^{-4}$, $\delta_2 = 10^{-2}$, and $\gamma_{incr} = \gamma_{decr} = 10$.

- Projecting onto X_ν: Case study 1: using the algorithm of Condat [18]; Case study 2: numerical solution of the KKT conditions (see Appendix A.4 for details); Case studies 3 and 4 and Case study 5 in the appendix: by solving a quadratic program.

- Estimating risk level $p(x)$: Case studies 1 and 2: analytical solution; Case studies 3 and 4: Monte Carlo estimate using $N_{MC} = 10^5$ and reliability level $\delta = 10^{-6}$; Case study 5 in the appendix: numerical estimation of the exact risk level based on Ruben [55].

- Case study 2: $\tilde{\nu} = 0.02|\bar{\nu}^0|$

- Case studies 3 and 4: $\alpha_{low} = 5 \times 10^{-4}$.

We compare the results of our proposed approach with a tuned application of scenario approximation that enforces a predetermined number of random constraints, solves the scenario problem to local/global optimality, and determines an a posteriori estimate of the risk level at the solution of the scenario problem using an independent Monte Carlo sample (see Section 4.3) to estimate a point on the EF of (CCP). Algorithm 7 in Appendix A presents pseudocode for approximating the EF of (CCP) by solving many scenario approximation problems using an iterative (cutting-plane) approach.

Our codes are written in Julia 0.6.2 [8], use Gurobi 7.5.2 [33] to solve linear, quadratic, and second-order cone programs, use IPOPT 3.12.8 [63] to solve nonlinear scenario approximation problems (with

MUMPS [4] as the linear solver), and use SCIP 6.0.0 [31] to solve nonlinear scenario approximation problems to global optimality (if necessary). The above solvers were accessed through the JuMP 0.18.2 modeling interface [24]. All computational tests[3] were conducted on a Surface Book 2 laptop running Windows 10 Pro with a 1.90 GHz four core Intel i7 CPU, 16 GB of RAM.

5.2 Numerical experiments

We tested our approach on four test cases from the literature. We present basic details of the test instances below. The Julia code and data for the test instances are available at https://github.com/rohitkannan/SA-for-CCP. For each test case, we compared the EFs generated by the stochastic approximation method against the solutions obtained using the tuned scenario approximation method and, if available, the analytical EF. Because the output of our proposed approach is random, we present enclosures of the EF generated by our proposal over ten different replicates in Appendix D for each case study. These results indicate that the output of the proposed method does not vary significantly across the replicates.

Case study 1. This portfolio optimization instance is based on Example 2.3.6 in Ben-Tal et al. [6], and includes a single individual linear chance constraint:

$$\max_{t,\, x \in \Delta_N} \quad t$$
$$\text{s.t.} \quad \mathbb{P}\left\{\xi^{\mathrm{T}} x \geq t\right\} \geq 1 - \alpha,$$

where $\Delta_N := \left\{y \in \mathbb{R}_+^N : \sum_i y_i = 1\right\}$ is the standard simplex, x_i denotes the fraction of investment in stock $i \in \{1, \cdots, N\}$, $\xi \sim \mathcal{P} := \mathcal{N}(\mu, \Sigma)$ is a random vector of returns with joint normal probability distribution, $\mu_i = 1.05 + 0.3\frac{N-i}{N-1}$ and $\sigma_i = \frac{1}{3}\left(0.05 + 0.6\frac{N-i}{N-1}\right)$, $i \in \{1, \cdots, N\}$, and $\Sigma = \text{diag}(\sigma_1^2, \cdots, \sigma_N^2)$. We consider the instance with number of stocks $N = 1000$. We assume the returns ξ are normally distributed so that we can benchmark our approach against an analytical solution for the true EF when the risk level $\alpha \leq 0.5$ (e.g., see Prékopa [50, Theorem 10.4.1]).

Figure 3 compares a typical EF obtained using our approach against the analytical EF and the solutions generated by the tuned scenario approximation algorithm. Our proposal is able to find a very good approximation of the true EF, whereas the scenario approximation method finds solutions that can be improved either in objective or risk. Our proposed approach took 388 seconds on average (and a maximum of 410 seconds) to approximate the EF using 26 points, whereas the tuned scenario approximation method took a total of 6900 seconds to generate its 1000 points in Figure 3. Note that even though the above instance of (CCP) does not satisfy Assumption 1 as written (because $X_\nu := \{(x,t) : x \in \Delta_N, t \geq \nu\}$ is not compact), we can essentially incorporate the upper bound $t \leq \nu$ within the definition of the set X_ν because the solution to Problem (APP$_\nu$) always satisfies $t^* = \nu$.

Since existing smoothing-based approaches provide the most relevant comparison, we compare our results with one such approach from the literature. We choose the smoothing-based approach of Cao and Zavala [13] for comparison because: i. they report encouraging computational results in their work relative to other such approaches, and ii. they have made their implementation available. Table 1 summarizes typical results of the sigmoidal smoothing approach of Cao and Zavala [13] when applied to the above instance with a specified risk level of $\alpha = 0.01$, with a varying number of scenarios, and with different settings for the scaling factor γ of the sigmoidal approximation method (see Algorithm SigVar-Alg of Cao and Zavala [13]). Appendix A.3 lists details of our implementation of this method. The second, third, and fourth columns of Table 1 present the overall solution time in seconds (or a failure status returned by IPOPT), the best objective value and the true risk level of the corresponding solution returned by the method over two replicates (we note that there was significant variability in solution times over the replicates; we report the results corresponding to the smaller solution times). Figure 3 plots the solutions returned by this method (using the values listed in Table 1). The relatively poor performance of Algorithm SigVar-Alg on this example is not surprising; even the instance of the sigmoidal approximation problem with only a hundred scenarios (which is small for $\alpha = 0.01$) has more than a thousand variables and a hundred thousand nonzero entries in the (dense) Jacobian. Therefore, tailored

---

[3]The scenario approximation problems solved to global optimality using SCIP in Case study 4 were run on a different laptop running Ubuntu 16.04 with a 2.6 GHz four core Intel i7 CPU, 8 GB of RAM due to interfacing issues on Windows.

approaches have to be developed for solving these problems efficiently. Since our implementation of the proposal of Cao and Zavala [13] failed to perform well on this instance even for a single risk level, we do not compare against their approach for the rest of the case studies.

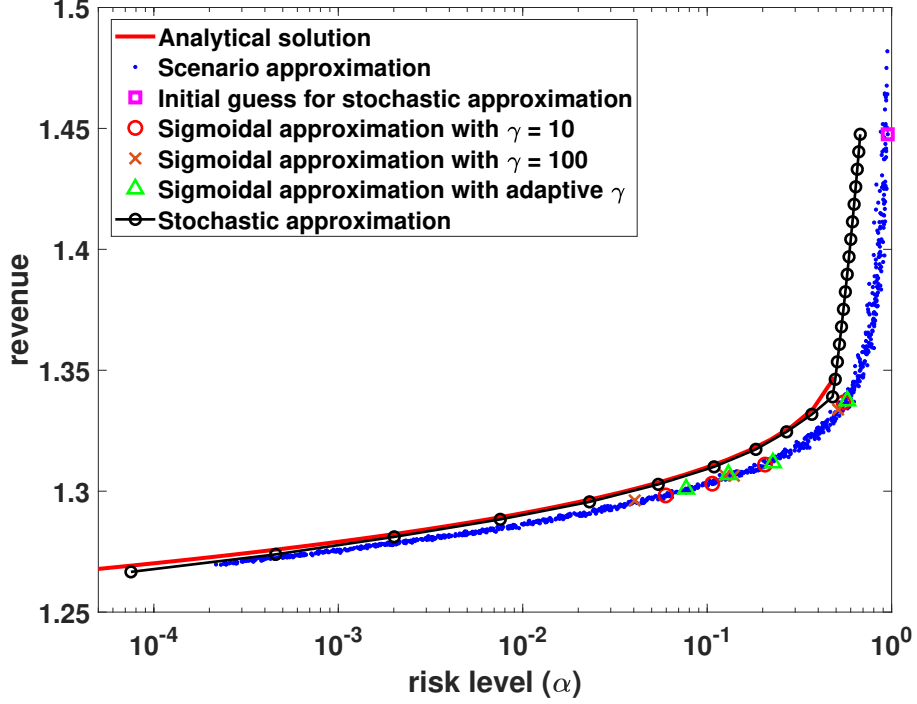Figure 3: Comparison of the efficient frontiers for Case study 1.

Table 1: Performance of the sigmoidal smoothing approach of Cao and Zavala [13] on Case study 1. The entry for each column of γ indicates the time in seconds (or IPOPT status)/best objective value/best risk level returned by the sigmoidal approximation method over two replicates. The entries `infeas` and `tle` denote infeasible and time limit exceeded statuses returned by IPOPT.

| Num. scenarios	$\gamma = 10$			$\gamma = 100$			γ adaptive		
	Time	Obj	Risk	Time	Obj	Risk	Time	Obj	Risk
100	34	1.337	0.548	23	1.334	0.513	29	1.337	0.575
500	508	1.311	0.206	106	1.306	0.139	59	1.312	0.227
1000	606	1.303	0.106	1203	1.306	0.125	1011	1.307	0.130
2000	1946	1.298	0.059	296	1.296	0.040	2787	1.301	0.077
5000	4050	1.226	2.4×10^{-8}	6289	1.292	0.025		infeas	
10000	2708	1.199	5.1×10^{-11}		tle			tle	

Case study 2. We consider the following variant of Case study 1 where the variance of the portfolio is minimized subject to the constraint that the return is larger than a fixed threshold at least with a given probability:

$$\min_{x \in \Delta_N} \quad x^{\mathrm{T}} \Sigma x$$
$$\text{s.t. } \mathbb{P}\left\{\xi^{\mathrm{T}} x \geq \bar{t}\right\} \geq 1 - \alpha,$$

21

where we require that the 100α percentile of the return is at least $\bar{t} = 1.2$. Once again, we consider the instance with number of stocks $N = 1000$, and assume the returns ξ are normally distributed as in Case study 1 so that we can benchmark our approach against the analytical solution for the true EF.

We found that using a general-purpose NLP solver to solve the quadratically-constrained quadratic program to compute projections onto the set $X_\nu := \{x \in \Delta_N : x^T \Sigma x \le \nu\}$ during the course of Algorithm 2 was a bottleneck of the algorithm in this case. Thus, we implemented a tailored projection routine that exploits the structure of this projection problem. Appendix A.4 presents details of the projection routine. Figure 4 compares a typical EF obtained using our approach against the analytical EF and the solutions generated by the tuned scenario approximation algorithm. Our proposal is able to generate solutions that lie on or very close to the true EF, whereas the scenario approximation method finds solutions that can be improved either in objective or risk. Our proposed approach took 1556 seconds on average (and a maximum of 1654 seconds) to approximate the EF using 20 points, whereas the tuned scenario approximation method took a total of 36259 seconds to generate its 1000 points in Figure 4.

Figure 4: Comparison of the efficient frontiers for Case study 2.
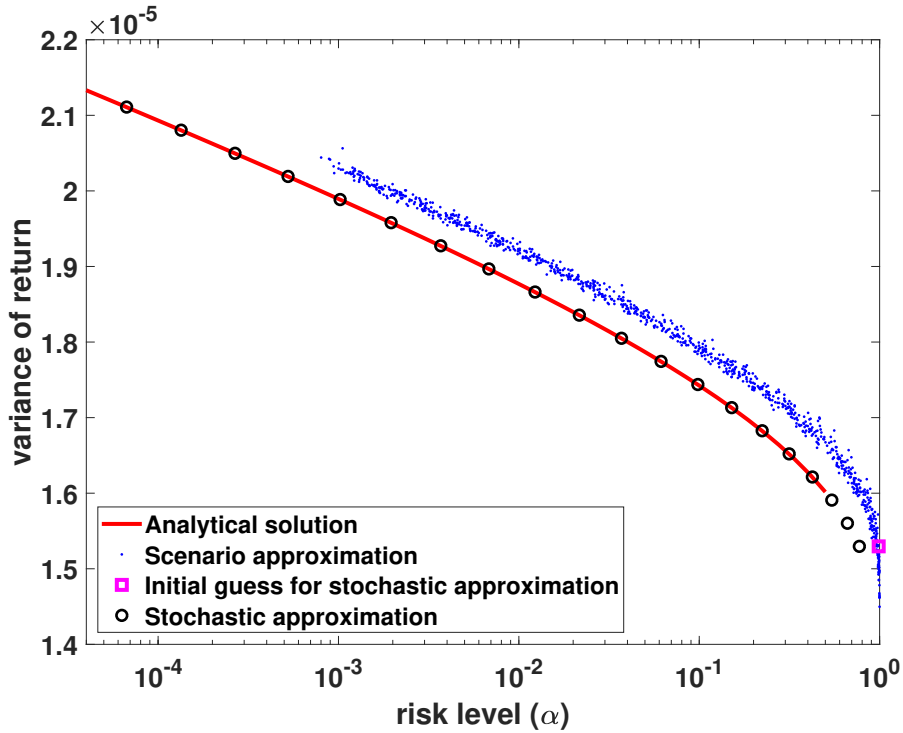
Case study 3. This norm optimization instance is based on Section 5.1.2 of Hong et al. [34], and includes a joint convex nonlinear chance constraint:

$$\min_{x \in \mathbb{R}_+^n} \quad -\sum_i x_i$$

$$\text{s.t.} \quad \mathbb{P}\left\{\sum_i \xi_{ij}^2 x_i^2 \le U^2, \ j = 1, \cdots, m\right\} \ge 1 - \alpha,$$

where ξ_{ij} are dependent normal random variables with mean $\frac{i}{d}$ and variance 1, and $\text{cov}(\xi_{ij}, \xi_{i'j}) = 0.5$ if $i \ne i'$, $\text{cov}(\xi_{ij}, \xi_{i'j'}) = 0$ if $j \ne j'$. We consider the instance with number of variables $n = 100$, number of constraints $m = 100$, and bound $U = 100$. Figure 5 compares a typical EF obtained using our approach against the solutions generated by the tuned scenario approximation algorithm. Once again, our proposed approach is able to find a significantly better approximation of the EF than the scenario

Figure 5: Comparison of the efficient frontiers for Case study 3.
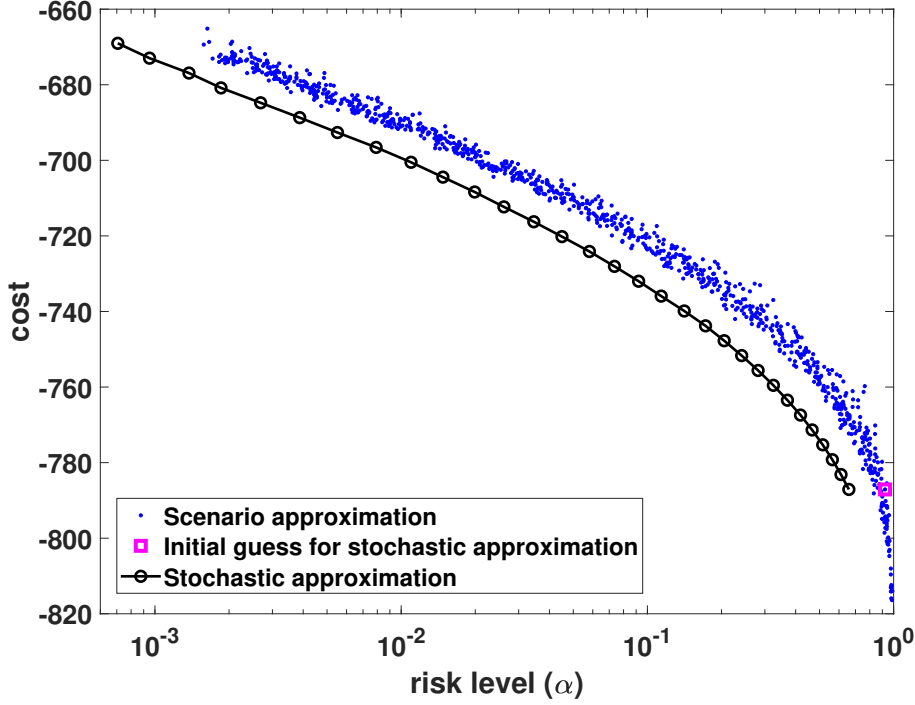
approximation method, although it is unclear how our proposal fares compared to the true EF since it is unknown in this case. Our proposal took 6742 seconds on average (and a maximum of 7091 seconds) to approximate the EF using 31 points, whereas it took tuned scenario approximation a total of 25536 seconds to generate its 1000 points in Figure 5. We note that more than 70% of the reported times for our method is spent in generating random numbers because the random variable ξ is high-dimensional and the covariance matrix of the random vector $\xi_{\cdot j}$ is full rank. A practical instance might have a covariance matrix rank that is at least a factor of ten smaller, which would reduce our overall computation times roughly by a factor of three (cf. the smaller computation times reported for the similar Case study 5 in the appendix). Appendix D benchmarks our approach against the true EF when the random variables ξ_{ij} are assumed to be i.i.d. in which case the analytical solution is known (see Section 5.1.1 of Hong et al. [34]). Once again, note that Assumption 1 does not hold because the set $X_\nu = \left\{ x \in \mathbb{R}_+^n : \sum_i x_i \geq -\nu \right\}$ is not compact; however, we can easily deduce an upper bound on each x_i, $i \in \{1, \cdots, n\}$, that is required for feasibility of the chance constraint for the largest (initial) risk level of interest and incorporate this upper bound within the definition of X_ν without altering the EF.

Case study 4. This probabilistic resource planning instance is based on Section 3 of Luedtke [40], and is modified to include a nonconvex recourse constraint:

$$\min_{x \in \mathbb{R}_+^n} \ c^{\mathrm{T}} x$$
$$\text{s.t.} \ \ \mathbb{P}\left\{ x \in R(\lambda, \rho) \right\} \geq 1 - \alpha,$$

where x_i denotes the quantity of resource i, c_i denotes the unit cost of resource i,

$$R(\lambda, \rho) = \left\{ x \in \mathbb{R}_+^n : \exists y \in \mathbb{R}_+^{nn_c} \text{ s.t. } \sum_{j=1}^{n_c} y_{ij} \leq \rho_i x_i^2, \ \forall i \in \{1, \cdots, n\}, \ \sum_{i=1}^{n} \mu_{ij} y_{ij} \geq \lambda_j, \ \forall j \in \{1, \cdots, n_c\} \right\},$$

n_c denotes the number of customer types, y_{ij} denote the amount of resource i allocated to customer type j, $\rho_i \in (0, 1]$ is a random variable that denotes the yield of resource i, $\lambda_j \geq 0$ is a random variable that

23

denotes the demand of customer type j, and $\mu_{ij} \geq 0$ is a deterministic scalar that denotes the service rate of resource i for customer type j. Note that the nonlinear term $\rho_i x_i^2$ in the definition of $R(\lambda, \rho)$ is a modification of the corresponding linear term $\rho_i x_i$ in Luedtke [40]. This change could be interpreted as a reformulation of the instance in Luedtke [40] with concave objective costs (due to economies of scale). We consider the instance with number of resources $n = 20$ and number of customer types $n_c = 30$. Details of how the parameters of the model are set (including details of the random variables) can be found in the electronic companion to Luedtke [40].

Figure 6 compares a typical EF obtained using our approach against the solutions generated by the tuned scenario approximation algorithm. The blue dots in the top part of Figure 6 correspond to the 1000 points obtained using the scenario approximation method when IPOPT is used to solve the scenario approximation problems. We mention that the vertical axis has been truncated for readability; the scenario approximation yields some solutions that are further away from the EF (with objective values up to 110 for the risk levels of interest). The 1000 red dots in the bottom part of Figure 6 correspond to the points obtained by solving the scenario approximation problems to global optimality using SCIP. The top black curve (with circles) corresponds to the EF obtained using the stochastic approximation method when it is initialized using the IPOPT solution of a scenario approximation problem. When a better initial point obtained by solving a scenario approximation problem using SCIP is used, the stochastic approximation method generates the bottom green curve (with squares) as its approximation of the EF.

Several remarks are in order. The scenario approximation solutions generated using the local solver IPOPT are very scattered possibly because IPOPT gets stuck at suboptimal local minima. However, the best solutions generated using IPOPT provide a comparable approximation of the EF as the stochastic approximation method that is denoted by the black curve with circles (which appears to avoid the poor local minima encountered by IPOPT). We note that IPOPT finds a good local solution at one of the 1000 scenario approximation runs (indicated by the blue circle). Solving the scenario approximation problems to global optimality using SCIP yields a much better approximation of the EF than the local solver IPOPT. Additionally, when the proposed approach is initialized using the global solution obtained from SCIP for a single scenario approximation problem (which took less than 60 seconds to compute), it generates a significantly better approximation of the EF that performs comparably to the EF generated using the global solver SCIP. We also tried initializing the solution of IPOPT with the one global solution from SCIP, and this did not yield a better approximation of the EF. Our approach took 9148 seconds on average (and a maximum of 9439 seconds) to generate the green curve (with squares) approximation of the EF using 26 points, whereas it took the blue scenario approximations (solved using IPOPT) and the red scenario approximations (solved using SCIP) a total of 110291 seconds and 145567 seconds, respectively, to generate their 1000 points in Figure 6.

6   Conclusion and future work

We proposed a stochastic approximation algorithm for estimating the efficient frontier of chance-constrained NLPs. Our proposal involves solving a sequence of partially smoothened stochastic optimization problems to local optimality using a projected stochastic subgradient algorithm. We established that every limit point of the sequence of stationary/global solutions of the above sequence of approximations yields a stationary/global solution of the original chance-constrained program with an appropriate risk level. A potential advantage of our proposal is that it can find truly stationary solutions of the chance-constrained NLP unlike scenario-based approaches that may get stuck at spurious local optima generated by sampling. Our computational experiments demonstrated that our proposed approach is consistently able to determine good approximations of the efficient frontier in reasonable computation times.

Extensions of our proposal that can handle multiple sets of joint chance constraints merit further investigation. One option is to minimize the maximum risk of constraints violation over the various sets of joint chance constraints, which can be formulated as a minimax stochastic program (cf. Section 3.2 of Nemirovski et al. [44]). This can in turn be reformulated as a weakly convex-concave saddle point problem that can be solved to stationarity using existing techniques [51]. Since our proposal relies on computing projections efficiently, approaches for reducing the computational effort spent on projections, such as random constraint projection techniques, could be explored. Additionally, extensions that incorporate deterministic nonconvex constraints in a more natural fashion provide an avenue for future work.

Figure 6: Comparison of the efficient frontiers for Case study 4.
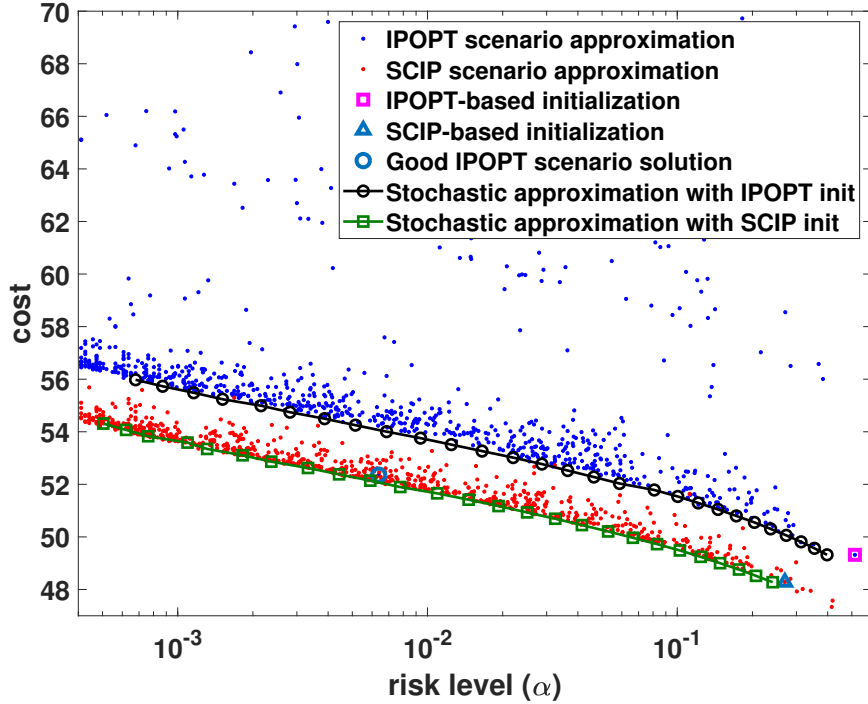
The projected stochastic subgradient method of Davis and Drusvyatskiy [20] has recently been extended to the non-Euclidean case [65], which could accelerate convergence of our proposal in practice. Finally, because stochastic approximation algorithms are an active area of research, several auxiliary techniques, such as adaptive step sizes, parallelization, acceleration, etc., may be determined to be applicable (and practically useful) to our setting.

Acknowledgements

References

[1] Lukáš Adam and Martin Branda. Nonlinear chance constrained problems: optimality conditions, regularization and solvers. *Journal of Optimization Theory and Applications*, 170(2):419–436, 2016.

[2] Lukáš Adam and Martin Branda. Machine learning approach to chance-constrained problems: An algorithm based on the stochastic gradient descent. http://www.optimization-online.org/DB_HTML/2018/12/6983.html (Last accessed: April 1, 2019), 2018.

[3] Lukáš Adam, Martin Branda, Holger Heitsch, and René Henrion. Solving joint chance constrained problems using regularization and Benders decomposition. *Annals of Operations Research*, pages 1–27, 2018. doi: 10.1007/s10479-018-3091-9.

[4] Patrick R Amestoy, Iain S Duff, Jean-Yves L'Excellent, and Jacko Koster. MUMPS: a general purpose distributed memory sparse solver. In *International Workshop on Applied Parallel Computing*, pages 121–130. Springer, 2000.

[5] Laetitia Andrieu, Guy Cohen, and Felisa Vázquez-Abad. Stochastic programming with probability constraints. *arXiv preprint arXiv:0708.0281*, 2007.

[6] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[7] Jacques F Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[8] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[9] Daniel Bienstock, Michael Chertkov, and Sean Harnett. Chance-constrained optimal power flow: Risk-aware network control under uncertainty. *SIAM Review*, 56(3):461–495, 2014.

[10] Giuseppe Calafiore and Marco C Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[11] Giuseppe C Calafiore, Fabrizio Dabbene, and Roberto Tempo. Research on probabilistic methods for control system design. *Automatica*, 47(7):1279–1293, 2011.

[12] Marco C Campi and Simone Garatti. A sampling-and-discarding approach to chance-constrained optimization: feasibility and optimality. *Journal of Optimization Theory and Applications*, 148(2):257–280, 2011.

[13] Yankai Cao and Victor Zavala. A sigmoidal approximation for chance-constrained nonlinear programs. http://www.optimization-online.org/DB_FILE/2017/10/6236.pdf (Last accessed: April 1, 2019), 2017.

[14] Yair Censor, Wei Chen, Patrick L Combettes, Ran Davidi, and Gabor T Herman. On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints. *Computational Optimization and Applications*, 51(3):1065–1088, 2012.

[15] Abraham Charnes, William W Cooper, and Gifford H Symonds. Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. *Management Science*, 4(3):235–263, 1958.

[16] Wenqing Chen, Melvyn Sim, Jie Sun, and Chung-Piaw Teo. From CVaR to uncertainty set: Implications in joint chance-constrained optimization. *Operations Research*, 58(2):470–485, 2010.

[17] Frank H Clarke. *Optimization and nonsmooth analysis*, volume 5. SIAM, 1990.

[18] Laurent Condat. Fast projection onto the simplex and the ℓ_1 ball. *Mathematical Programming*, 158(1-2):575–585, 2016.

[19] Frank E Curtis, Andreas Wachter, and Victor M Zavala. A sequential algorithm for solving nonlinear optimization problems with chance constraints. *SIAM Journal on Optimization*, 28(1):930–958, 2018.

[20] Damek Davis and Dmitriy Drusvyatskiy. Stochastic subgradient method converges at the rate $O\left(k^{-1/4}\right)$ on weakly convex functions. *arXiv preprint arXiv:1802.02988*, 2018.

[21] Damek Davis, Dmitriy Drusvyatskiy, Sham Kakade, and Jason D Lee. Stochastic subgradient method converges on tame functions. *arXiv preprint arXiv:1804.07795*, 2018.

[22] Darinka Dentcheva and Gabriela Martinez. Regularization methods for optimization problems with probabilistic constraints. *Mathematical Programming*, 138(1-2):223–251, 2013.

[23] Dmitriy Drusvyatskiy and Courtney Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Mathematical Programming*, pages 1–56, 2018. doi: 10.1007/s10107-018-1311-3.

[24] Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[25] Yuri M Ermoliev. Stochastic quasigradient methods. In Yuri M Ermoliev and Roger JB Wets, editors, *Numerical techniques for stochastic optimization*, chapter 6, pages 141–185. Springer, 1988.

[26] Yuri M Ermoliev and VI Norkin. Stochastic generalized gradient method for nonconvex nonsmooth stochastic optimization. *Cybernetics and Systems Analysis*, 34(2):196–215, 1998.

[27] Yuri M Ermoliev and Vladimir I Norkin. On nonsmooth and discontinuous problems of stochastic systems optimization. *European Journal of Operational Research*, 101(2):230–244, 1997.

[28] Yuri M Ermoliev, Vladimir I Norkin, and Roger JB Wets. The minimization of semicontinuous functions: mollifier subgradients. *SIAM Journal on Control and Optimization*, 33(1):149–167, 1995.

[29] Abebe Geletu, Armin Hoffmann, Michael Kloppel, and Pu Li. An inner-outer approximation approach to chance constrained optimization. *SIAM Journal on Optimization*, 27(3):1834–1857, 2017.

[30] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305, 2016.

[31] Ambros Gleixner, Michael Bastubbe, Leon Eifler, Tristan Gally, Gerald Gamrath, Robert Lion Gottwald, Gregor Hendel, Christopher Hojny, Thorsten Koch, Marco E. Lübbecke, Stephen J. Maher, Matthias Miltenberger, Benjamin Müller, Marc E. Pfetsch, Christian Puchert, Daniel Rehfeldt, Franziska Schlösser, Christoph Schubert, Felipe Serrano, Yuji Shinano, Jan Merlin Viernickel, Matthias Walter, Fabian Wegscheider, Jonas T. Witt, and Jakob Witzig. The SCIP Optimization Suite 6.0. Technical report, Optimization Online, July 2018. URL http://www.optimization-online.org/DB_HTML/2018/07/6692.html.

[32] Claudia Gotzes, Holger Heitsch, René Henrion, and Rüdiger Schultz. On the quantification of nomination feasibility in stationary gas networks with random load. *Mathematical Methods of Operations Research*, 84 (2):427–457, 2016.

[33] Gurobi Optimization LLC. Gurobi Optimizer Reference Manual, 2018. URL http://www.gurobi.com.

[34] L Jeff Hong, Yi Yang, and Liwei Zhang. Sequential convex approximations to joint chance constrained programs: A Monte Carlo approach. *Operations Research*, 59(3):617–630, 2011.

[35] Zhaolin Hu, L Jeff Hong, and Liwei Zhang. A smooth Monte Carlo approach to joint chance-constrained programs. *IIE Transactions*, 45(7):716–735, 2013.

[36] Ruiwei Jiang and Yongpei Guan. Data-driven chance constrained stochastic program. *Mathematical Programming*, 158(1-2):291–327, 2016.

[37] Constantino M Lagoa, Xiang Li, and Mario Sznaier. Probabilistically constrained linear programs and risk-adjusted controller design. *SIAM Journal on Optimization*, 15(3):938–951, 2005.

[38] Riho Lepp. Extremum problems with probability functions: Kernel type solution methods. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization*, pages 969–973. Springer, 2009. URL https://doi.org/10.1007/978-0-387-74759-0_170.

[39] Pu Li, Harvey Arellano-Garcia, and Günter Wozny. Chance constrained programming approach to process optimization under uncertainty. *Computers & Chemical Engineering*, 32(1-2):25–45, 2008.

[40] James Luedtke. A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. *Mathematical Programming*, 146(1-2):219–244, 2014.

[41] James Luedtke and Shabbir Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008.

[42] Bruce L Miller and Harvey M Wagner. Chance constrained programming with joint constraints. *Operations Research*, 13(6):930–945, 1965.

[43] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2006.

[44] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[45] Arkadii Semenovich Nemirovsky and David Borisovich Yudin. *Problem complexity and method efficiency in optimization*. Wiley, 1983.

[46] Vladimir I Norkin. The analysis and optimization of probability functions. Technical report, IIASA Working Paper, WP-93-6, January 1993.

[47] EA Nurminskii. The quasigradient method for the solving of the nonlinear programming problems. *Cybernetics*, 9(1):145–150, 1973.

[48] Alejandra Peña-Ordieres, James R Luedtke, and Andreas Wächter. Solving chance-constrained problems via a smooth sample-based nonlinear approximation, 2019.

[49] András Prékopa. On probabilistic constrained programming. In *Proceedings of the Princeton symposium on mathematical programming*, pages 113–138. Princeton, NJ, 1970.

[50] András Prékopa. *Stochastic programming*, volume 324. Springer Science & Business Media, 1995.

[51] Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. Non-convex min-max optimization: Provable algorithms and applications in machine learning. *arXiv preprint arXiv:1810.02060*, 2018.

[52] Tara Rengarajan and David P Morton. Estimating the efficient frontier of a probabilistic bicriteria model. In *Winter Simulation Conference*, pages 494–504, 2009.

[53] R Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2: 21–42, 2000.

[54] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[55] Harold Ruben. Probability content of regions under spherical normal distributions, IV: The distribution of homogeneous and non-homogeneous quadratic functions of normal variables. *The Annals of Mathematical Statistics*, 33(2):542–570, 1962.

[56] F Shan, XT Xiao, and LW Zhang. Convergence analysis on a smoothing approach to joint chance constrained programs. *Optimization*, 65(12):2171–2193, 2016.

[57] Feng Shan, Liwei Zhang, and Xiantao Xiao. A smoothing function approach to joint chance-constrained programs. *Journal of Optimization Theory and Applications*, 163(1):181–199, 2014.

[58] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.

[59] Wim van Ackooij and René Henrion. Gradient formulae for nonlinear probabilistic constraints with Gaussian and Gaussian-like distributions. *SIAM Journal on Optimization*, 24(4):1864–1889, 2014.

[60] Wim van Ackooij and René Henrion. (Sub-)Gradient formulae for probability functions of random inequality systems under Gaussian distribution. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):63–87, 2017.

[61] Wim van Ackooij, Antonio Frangioni, and Welington de Oliveira. Inexact stabilized Benders decomposition approaches with application to chance-constrained problems with finite support. *Computational Optimization and Applications*, 65(3):637–669, 2016.

[62] Wim van Ackooij, V Berge, Welington de Oliveira, and C Sagastizábal. Probabilistic optimization via approximate p-efficient points and bundle methods. *Computers & Operations Research*, 77:177–193, 2017.

[63] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[64] Hui Zhang and Pu Li. Chance constrained programming for optimal power flow under uncertainty. *IEEE Transactions on Power Systems*, 26(4):2417–2424, 2011.

[65] Siqi Zhang and Niao He. On the convergence rate of stochastic mirror descent for nonsmooth nonconvex optimization. *arXiv preprint arXiv:1806.04781*, 2018.

A Algorithm outlines

A.1 Implementation of the scenario approximation algorithm

Algorithm 7 details our implementation of the tuned scenario approximation algorithm. Instead of solving each scenario approximation problem instance using an off-the-shelf solver by imposing all of the sampled constraints at once, we adopt a cutting-plane approach that iteratively adds some of the violated scenario constraints at a candidate solution to the list of enforced constraints in a bid to reduce the overall computational effort (cf. Algorithm 2.6 in Bienstock et al. [9]). We use $M = 50$ iterations and $R = 20$ replicates for our computational experiments. Settings for sample sizes N_i, constraints added per iteration N_c, and number of samples N_{MC} are problem dependent and provided below. Line 19 in Algorithm 7, which estimates the risk level of a candidate solution, may be replaced, if possible, by one that computes the true analytical risk level or a numerical estimate of it [59].

- $N_i = \lceil 10^{a_i} \rceil$, $i = 1, \cdots, 50$, where: for Case study 1: $a_i = 1 + \frac{5}{49}(i - 1)$; for Case study 2: $a_i = 1 + \frac{4}{49}(i-1)$; for Case study 3: $a_i = 1 + \frac{\log_{10}(50000)-1}{49}(i-1)$; for Case study 4: $a_i = 1 + \frac{4}{49}(i-1)$; and for Case study 5: $a_i = 1 + \frac{4}{49}(i-1)$ (the upper bounds on N_i were determined based on memory requirements)

- N_c: Case study 1: 1000; Case study 2: 100000; Case study 3: 10; Case study 4: 5; and Case study 5: 10 (these values were tuned for good performance)

- N_{MC}: See Section 5.1 of the paper (we use the same Monte Carlo samples that were used by our proposed method to estimate risk levels).

A.2 Solving (CCP) for a fixed risk level

Algorithm 8 adapts Algorithm 2 to solve (CCP) for a given risk level $\hat{\alpha} \in (0, 1)$. An initial point $\bar{x}^0$ and an upper bound on the optimal objective value ν_{up} can be obtained in a manner similar to Algorithm 2, whereas an initial lower bound on the optimal objective value ν_{low} can be obtained either using lower bounding techniques (see [41, 43]), or by trial and error. Note that the sequence of approximations in line 9 of Algorithm 8 need not be solved until termination if Algorithm 3 determines that $\bar{\alpha}^i < \hat{\alpha}$ before its termination criteria have been satisfied.

A.3 Implementation of the sigmoidal approximation algorithm

We used the following 'tuned' settings to solve each iteration of the sigmoidal approximation problem (see Section 4 of Cao and Zavala [13]) using IPOPT: `tol` $= 10^{-4}$, `max_iter = 10000`, `hessian_approximation = limited_memory`, `jac_c_constant=yes`, and `max_cpu_time=3600` seconds. We terminated the loop of Algorithm `SigVar-Alg` of Cao and Zavala [13] when the objective improved by less than 0.01% relative to the previous iteration. In what follows, we use the notation of Algorithm `SigVar-Alg` of Cao and Zavala [13]. For our 'adaptive γ' setting, we use the proposal of Cao and Zavala [13] to specify γ when the solution of the CVaR problem corresponds to $t_c(\alpha) < 0$. When $t_c(\alpha) = 0$ at the CVaR solution returned by Gurobi, we try to estimate a good value of $t_c(\alpha)$ by looking at the true distribution of $g(x_c(\alpha), \xi)$.

A.4 Tailored projection step for Case study 2

Because Algorithm 2 requires projecting onto the sets $X_\nu := \{x \in \Delta_N : x^{\mathrm{T}}\Sigma x \leq \nu\}$ many times for Case study 2 with different values of ν, we develop a tailored projection routine that is computationally more efficient than solving quadratically-constrained quadratic programs to compute these projections.

Algorithm 7 Scenario approximation for approximating the efficient frontier of (CCP)

1: **Input**: Number of major iterations M, distinct sample sizes N_i, $i = 1, \cdots, M$, number of replicates per major iteration R, maximum number of scenario constraints added per iteration N_c, number of samples to estimate risk levels N_{MC}, and initial guess $\hat{x} \in X$.

2: **Output**: Pairs $(\bar{\nu}^{i,r}, \bar{\alpha}^{i,r})$, $i \in \{1, \cdots, M\}$ and $r \in \{1, \cdots, R\}$, of objective values and risk levels that can be used to approximate the efficient frontier, and corresponding sequence of solutions $\{\bar{x}^{i,r}\}$.

3: **Preparation**: Draw a (fixed) sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$ from $\mathcal{P}$ for estimating risk levels of candidate solutions.

4: **for** major iteration $i = 1$ to M **do**

5:     **for** replicate $r = 1$ to R **do**

6:         Sample: Draw i.i.d. sample $\{\xi^{l,r}\}_{l=1}^{N_i}$ from $\mathcal{P}$.

7:         Initialize: Starting point $x^0 := \hat{x}$, list of scenarios enforced for each chance constraint $\mathcal{I}_k^0 = \emptyset$, $k \in \{1, \cdots, m\}$,
            and counter $q = 0$.

8:         **repeat**

9:             Update $q \leftarrow q + 1$, number of scenario constraints violated $V \leftarrow 0$, $\mathcal{I}_k^q = \mathcal{I}_k^{q-1}$, $\forall k$.

10:             Solve the following scenario approximation problem (locally) using the initial guess x^{q-1} to obtain
        a solution x^q and corresponding optimal objective ν^q:

$$\min_{x \in X} \ f(x)$$
$$\text{s.t. } g_k(x, \xi^{l,r}) \leq 0, \quad \forall l \in \mathcal{I}_k^{q-1}.$$

11:         **for** $k = 1$ to m **do**

12:             Evaluate k^{th} random constraint g_k at x^q for each of the scenarios $l = 1, \cdots, N_i$.

13:             Sort the values $\{g_k(x^q, \xi^{l,r})\}_{l=1}^{N_i}$ in decreasing order.

14:             Increment V by the number of scenario constraints satisfying $g_k(x^q, \xi^{l,r}) > 0$.

15:             Add at most N_c of the most violated scenario indices l to $\mathcal{I}_k^q$.

16:         **end for**

17:         **until** $V = 0$

18:         Set $\hat{x} = x^{i,r} = x^q$ and $\bar{\nu}^{i,r} = \nu^q$ to their converged values.

19:         Estimate risk level $\bar{\alpha}^{i,r}$ of candidate solution $\hat{x}$ using the sample $\{\bar{\xi}^l\}_{l=1}^{N_{MC}}$.

20:     **end for**

21: **end for**

Algorithm 8 Solving (CCP) for a fixed risk level

1: **Input**: target risk level $\hat{\alpha} \in (0, 1)$, 'guaranteed' lower and upper bounds on the optimal objective value ν_{low} and ν_{up} with $\nu_{low} \leq \nu_{up}$, and initial point $\bar{x}^0 \in X$.

2: **Set algorithmic parameters**: in addition to line 2 of Algorithm 2, let $\nu_{tol} > 0$ denote an optimality tolerance.

3: **Output**: approximate optimal objective value $\hat{\nu}$ of (CCP) for a risk level of $\hat{\alpha}$.

4: **Preprocessing**: let $\bar{\nu}^0 = \frac{1}{2}(\nu_{low} + \nu_{up})$, and determine smoothing parameters $\{\tau_{k,j}\}_{k=1}^K$ scaled at $\text{proj}\left(\bar{x}^0, X_{\bar{\nu}^0}\right)$ using Algorithm 4 and an initial sequence of step lengths $\{\bar{\gamma}_k\}_{k=1}^K$ for the corresponding sequence of approximating problems (APP$_\nu$) with $\nu = \bar{\nu}^0$ using Algorithm 5.

5: **Optimization Phase**:

6: Initialize index $i = 0$.

7: **repeat**

8:     Update iteration count $i \leftarrow i + 1$ and set objective bound $\bar{\nu}^i = \frac{1}{2}(\nu_{low} + \nu_{up})$.

9:     Obtain $(\bar{\alpha}^i, \bar{x}^i)$ by solving sequence of approximations (APP$_\nu$) using Algorithm 3 with the above algorithmic
      parameter settings and $\text{proj}\left(\bar{x}^{i-1}, X_{\bar{\nu}^i}\right)$ as the initial guess.

10:     **if** $\bar{\alpha}^i \geq \hat{\alpha}$ **then** set $\nu_{low} \leftarrow \bar{\nu}^i$; **else** set $\nu_{up} \leftarrow \bar{\nu}^i$.

11: **until** $\nu_{low} \geq \nu_{up} - \nu_{tol}$

12: Set $\hat{\nu} = \nu_{up}$.

To project a point $y \in \mathbb{R}^N$ onto X_ν for some $\nu > 0$, note that the KKT conditions for the projection problem $\min\limits_{x \in X_\nu} 0.5\|x - y\|^2$ yield:

$$x_i^* = \frac{y_i - \mu + \pi_i}{1 + 2\sigma_i^2 \lambda}, \quad x_i^* \geq 0, \quad \pi_i \geq 0, \quad x_i^* \pi_i = 0, \quad \forall i \in \{1, \cdots, N\},$$

$$\lambda \geq 0, \quad \sum_{i=1}^{N} x_i^* = 1, \quad \sum_{i=1}^{N} \alpha_i^2 (x_i^*)^2 \leq \nu, \quad \lambda \left(\sum_{i=1}^{N} \alpha_i^2 (x_i^*)^2 - \nu \right) = 0,$$

where $x^* \in \mathbb{R}^N$ denotes the projection and μ, λ, and $(\pi_1, \cdots, \pi_N)$ denote KKT multipliers. Note that the inequalities in the first row can be simplified as $x_i^* = \max\left\{0, \frac{y_i - \mu}{1 + 2\sigma_i^2 \lambda}\right\}$, $\forall i \in \{1, \cdots, N\}$. We first check whether $\lambda = 0$ satisfies the above system of equations (i.e., if the quadratic constraint is inactive at the solution) by setting x^* to be the projection of y onto the unit simplex. If $\lambda = 0$ is infeasible, we solve the following system of equations by using binary search over λ:

$$x_i^* = \max\left\{0, \frac{y_i - \mu}{1 + 2\sigma_i^2 \lambda}\right\}, \quad \forall i \in \{1, \cdots, N\}, \quad \sum_{i=1}^{N} x_i = 1, \quad \sum_{i=1}^{N} \alpha_i^2 (x_i^*)^2 = \nu,$$

where for each fixed $\lambda > 0$, the first two sets of equations in x^* and μ are solved by adapting the algorithm of Condat [18] for projection onto the unit simplex.

B Proof of Proposition 6

We first sketch an outline of the proof (note that our proof will follow a different order for reasons that will become evident). We will establish that (see Chapters 4 and 5 of Rockafellar and Wets [54] for definitions of technical terms)

$$\limsup_{\substack{x \to \bar{x} \\ k \to \infty}} \partial \hat{p}_k(x) + N_{X_\nu}(x) = \limsup_{\substack{x \to \bar{x} \\ k \to \infty}} \partial \hat{p}_k(x) + \limsup_{\substack{x \to \bar{x} \\ k \to \infty}} N_{X_\nu}(x). \tag{3}$$

Then, because of the outer semicontinuity of the normal cone mapping, it suffices to prove that the outer limit of $\partial \hat{p}_k(x)$ is a subset of $\{\nabla p(\bar{x})\}$. To demonstrate this, we will first show that $\partial \hat{p}_k(x) = -\partial \int_{-\infty}^{\infty} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$, where F is the cumulative distribution function of $\max[\bar{g}(x, \xi)]$. Then, we will split this integral into two parts - one accounting for tail contributions (which we will show vanishes when we take the outer limit), and the other accounting for the contributions of F near $(x, \eta) = (\bar{x}, 0)$ that we will show satisfies the desired outer semicontinuity property.

Since $\hat{p}_k(x)$ can be rewritten as $\hat{p}_k(x) = \mathbb{E}\left[\phi\left(\max[\bar{g}(x, \xi)]; \tau_c^{k-1}\right)\right]$ by Assumptions 4B and 6, we have

$$
\begin{aligned}
\hat{p}_k(x) = \mathbb{E}\left[\phi\left(\max[\bar{g}(x, \xi)]; \tau_c^{k-1}\right)\right] &= \int_{-\infty}^{+\infty} \phi\left(\max[\bar{g}(x, \xi)]; \tau_c^{k-1}\right) d\mathbb{P} \\
&= \int_{-\infty}^{+\infty} \phi\left(\eta; \tau_c^{k-1}\right) dF(x, \eta) \\
&= \lim_{\eta \to +\infty} \phi\left(\eta; \tau_c^{k-1}\right) - \int_{-\infty}^{+\infty} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta,
\end{aligned}
$$

where the second line is to be interpreted as a Lebesgue-Stieljes integral, and the final step follows by integrating by parts and Assumption 4. This yields (see Proposition 5 and Assumption 6 for the existence of these quantities)

$$\partial \hat{p}_k(x) \subset -\partial \int_{|\eta| \geq \varepsilon_k} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta - \partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta \tag{4}$$

by the properties of the Clarke generalized gradient, where $\{\varepsilon_k\}$ is defined in Assumption 6.

Let $\{x_k\}$ be any sequence in X_ν converging to $\bar{x} \in X_\nu$. Suppose $v_k \in -\partial \int_{|\eta| \geq \varepsilon_k} F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$ with $v_k \to v \in \mathbb{R}^n$. We would like to show that $v = 0$. Note that by an abuse of notation (where we actually take norms of the integrable selections that define v_k)

$$\lim_{k \to \infty} \|v_k\| = \lim_{k \to \infty} \left\| -\partial \int_{|\eta| \geq \varepsilon_k} F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta \right\| = \lim_{k \to \infty} \left\| \int_{|\eta| \geq \varepsilon_k} \partial_x F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta \right\|$$

$$\leq \lim_{k \to \infty} \int_{|\eta| \geq \varepsilon_k} \|\partial_x F(x_k, \eta)\| d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$$

$$\leq \lim_{k \to \infty} \int_{|\eta| \geq \varepsilon_k} L_F(\eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta = 0, \quad (5)$$

where the first step follows from Theorem 2.7.2 of Clarke [17] (whose assumptions are satisfied by virtue of Assumption 6B), the second inequality follows from Proposition 2.1.2 of Clarke [17], and the final equality follows from Assumption 6C.

The above arguments establish that $\limsup_{\substack{x \to \bar{x} \\ k \to \infty}} -\partial \int_{|\eta| \geq \varepsilon_k} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta = \{0\}$. Consequently, we have $\limsup_{\substack{x \to \bar{x} \\ k \to \infty}} \partial \hat{p}_k(x) \subset \limsup_{\substack{x \to \bar{x} \\ k \to \infty}} -\partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$ from Equation (4). We now consider the outer limit $\limsup_{\substack{x \to \bar{x} \\ k \to \infty}} -\partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$. Suppose $w_k \in -\partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$ with $w_k \to w \in \mathbb{R}^n$. We wish to show that $w = \nabla p(\bar{x})$. Invoking Theorem 2.7.2 of Clarke [17] once again, we have that

$$\limsup_{k \to \infty} -\partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta \subset \limsup_{k \to \infty} -\int_{-\varepsilon_k}^{\varepsilon_k} \nabla_x F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta$$

$$= \limsup_{k \to \infty} -\left(\int_{-\varepsilon_k}^{\varepsilon_k} \nabla_x F(x_k, \eta) d\hat{\phi}(\eta) d\eta\right)\left(\int_{-\varepsilon_k}^{\varepsilon_k} d\phi\left(z; \tau_c^{k-1}\right) dz\right)$$

$$= \left(\limsup_{k \to \infty} -\int_{-\varepsilon_k}^{\varepsilon_k} \nabla_x F(x_k, \eta) d\hat{\phi}(\eta) d\eta\right)\left(\lim_{k \to \infty} \int_{-\varepsilon_k}^{\varepsilon_k} d\phi\left(z; \tau_c^{k-1}\right) dz\right)$$

$$= \limsup_{k \to \infty} -\int_{-\varepsilon_k}^{\varepsilon_k} \nabla_x F(x_k, \eta) d\hat{\phi}_k(\eta) d\eta, \quad (6)$$

where for each $k \in \mathbb{N}$ large enough, $\hat{\phi}_k : \mathbb{R} \to \mathbb{R}$ is defined as $\hat{\phi}_k(y) = \dfrac{\phi\left(y; \tau_c^{k-1}\right)}{\int_{-\varepsilon_k}^{\varepsilon_k} d\phi\left(z; \tau_c^{k-1}\right) dz}$, the first step follows (by an abuse of notation) from the fact that $\varepsilon_k < \theta$ for k large enough (see Assumption 6B), and the third and fourth steps follow from Assumption 6C. Noting that[4]

$$\int_{-\varepsilon_k}^{\varepsilon_k} \nabla_x F(x_k, \eta) d\hat{\phi}_k(\eta) d\eta = \left[\frac{\partial F}{\partial x_1}(x_k, \omega_{1,k}) \cdots \frac{\partial F}{\partial x_n}(x_k, \omega_{n,k})\right]^{\mathrm{T}} \int_{-\varepsilon_k}^{\varepsilon_k} d\hat{\phi}_k(\eta) d\eta = \begin{bmatrix} \dfrac{\partial F}{\partial x_1}(x_k, \omega_{1,k}) \\ \vdots \\ \dfrac{\partial F}{\partial x_n}(x_k, \omega_{n,k}) \end{bmatrix}$$

for some constants $\omega_{i,k} \in (-\varepsilon_k, \varepsilon_k)$, $i = 1, \cdots, n$, by virtue of Assumption 6B and the first mean value theorem for definite integrals, we have

$$\limsup_{k \to \infty} -\partial \int_{-\varepsilon_k}^{\varepsilon_k} F(x_k, \eta) d\phi\left(\eta; \tau_c^{k-1}\right) d\eta \subset \limsup_{k \to \infty} \left\{ -\left[\frac{\partial F}{\partial x_1}(x_k, \omega_{1,k}) \cdots \frac{\partial F}{\partial x_n}(x_k, \omega_{n,k})\right]^{\mathrm{T}} \right\}$$

$$= \{-\nabla_x F(\bar{x}, 0)\} = \{\nabla p(\bar{x})\},$$

---

[4]Lemma 4.2 of Shan et al. [57] reaches a stronger conclusion from Equation (6) than our result (they only require that the distribution function F is locally Lipschitz continuous). We do not adapt their arguments since we are unable to follow a key step (the first step in [57, pg. 191]) in their proof.

where the first equality above follows from Assumption 6B, and the second equality follows from the fact that $p(x) = 1 - F(x, 0)$ for each $x \in X_\nu$. Reconciling our progress with Equation (4), we obtain

$$\limsup_{\substack{x \to \bar{x} \\ k \to \infty}} \partial \hat{p}_k(x) \subset \{\nabla p(\bar{x})\}.$$

To establish the desirable equality in Equation (3), it suffices to show that $\partial \hat{p}_k(x_k) \subset C$ for k large enough, where $C \subset \mathbb{R}^n$ is independent of k. From Equation (5), we have that the first term in the right-hand side of Equation (4) is contained in a bounded set that is independent of $k \in \mathbb{N}$. From Equation (6), we have that any element of the second term in the right-hand side of Equation (4) is bounded above in norm by $\max_{\substack{x \in \text{cl}(B_\delta(\bar{x})) \\ \eta \in [-0.5\theta, 0.5\theta]}} \|\nabla_x F(x, \eta)\|$ for k large enough for any $\delta > 0$. The above arguments in conjunction with Equation (4) establish Equation (3). The desired result then follows from the outer semicontinuity of the normal cone mapping, see Proposition 6.6 of Rockafellar and Wets [54].

C   Recourse formulation

As noted in Section 1 of the paper, Problem (CCP) can also be used to model chance-constrained programs with static recourse decisions, e.g., by defining g through the solution of the following auxiliary optimization problem for each $(x, \xi) \in X \times \Xi$:

$$
\begin{aligned}
g(x, \xi) := \min_{y, \eta} \ & \eta \\
\text{s.t. } & T(x, \xi) + W(\xi)y + \eta e \geq 0, \\
& y \in \mathbb{R}_+^{n_y}, \ \eta \geq -1,
\end{aligned}
$$

where $T : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^{m_r}$ is continuously differentiable, and $W : \mathbb{R}^d \to \mathbb{R}^{m_r \times n_y}$. Note that g can be recast in the form of a joint chance constraint by appealing to linear programming duality, viz.,

$$g(x, \xi) = \max_{(v, w) \in \text{EXT}(V(\xi))} -v^{\mathrm{T}} T(x, \xi) - w,$$

where $\text{EXT}(V(\xi))$ denotes the set of extreme points of the polytope

$$V(\xi) = \left\{ (v, w) \in \mathbb{R}_+^{m_r} \times \mathbb{R}_+ : v^{\mathrm{T}} W(\xi) \leq 0, \ v^{\mathrm{T}} e + w = 1 \right\}.$$

Explicitly reformulating the recourse constraints into the above explicit form is not generally practical since Algorithms 4 and 5 rely on stepping through each of the constraint functions, and the cardinality of the set of extreme points of $V(\xi)$ may be huge. Therefore, we consider the case when the approximations $\hat{p}_k$ are constructed using a single smoothing function, i.e., $\hat{p}_k(x) = \mathbb{E}[\phi_k(g(x, \xi))]$, where $\{\phi_k\}$ is a sequence of smooth scalar functions that approximate the step function.

Throughout this section, we assume that g that can be reformulated as $g(x, \xi) := \max_{j \in J} h_j(x, \xi)$, where J is a (potentially large) finite index set, and $h_j : \mathbb{R}^n \times R^d \to \mathbb{R}$ are continuously differentiable functions that are Lipschitz continuous and have Lipschitz continuous gradients in the sense of Assumption 3. We let $L_h'(\xi)$ denote the Lipschitz constant of $\nabla h(\cdot, \xi)$ on X_ν with $\mathbb{E}\left[\left(L_h'(\xi)\right)^2\right] < +\infty$. While the theory in this section is also applicable to the ordinary joint chance constrained setting, it is usually advantageous to consider individual smoothing functions for each chance constraint function in that case whenever feasible.

We first characterize the Clarke generalized gradient of the approximation $\hat{p}_k$ under this setup.

Proposition 11. Suppose Assumptions 4 and 5 and the above assumptions on $g(x, \xi) = \max[h(x, \xi)]$ and ϕ_k hold. Then $\partial \hat{p}_k(x) = \mathbb{E}\left[d\phi_k\left(\max[h(x, \xi)]\right) \times \partial_x \max[h(x, \xi)]\right]$.

Proof. Note that for any $x \in X$ and $\xi \in \Xi$, we have $\partial_x \phi_k(g(x, \xi)) = d\phi_k(g(x, \xi)) \partial_x \max[h(x, \xi)]$, see Theorem 2.3.9 of Clarke [17]. A justification for swapping the derivative and expectation operators then follows from Proposition 5 of the paper and the fact that $\hat{p}_k(x) = \mathbb{E}[\phi_k(g(x, \xi))] = \mathbb{E}[\max_j [\phi_k(g_j(x, \xi))]]$ since ϕ_k is monotonically nondecreasing on $\mathbb{R}$. $\qquad\square$

The following result establishes that the approximation $\hat{p}_k$ continues to enjoy the weak convexity property for the above setup under mild assumptions.

Proposition 12. Suppose Assumptions 1, 4, and 5 hold. Additionally, suppose $g(x,\xi) = \max[h(x,\xi)]$ satisfies the above conditions and ϕ_k is a scalar smoothing function. Then $\hat{p}_k(\cdot)$ is $\bar{L}_k$-weakly convex for some constant $\bar{L}_k$ that depends on $L_h'(\xi)$, $L_{\phi,k}'$, $M_{\phi,k}'$, and the diameter of X_ν.

Proof. First, note that $g(\cdot,\xi) := \max[h(\cdot,\xi)]$ is $L_h'(\xi)$-weakly convex on X_ν for each $\xi \in \Xi$, see Lemma 4.2 of Drusvyatskiy and Paquette [23]. In particular, this implies that for any $y, z \in X_\nu$,

$$g(z,\xi) = \max[h(z,\xi)] \geq g(y,\xi) + s_y^{\mathrm{T}}(\xi)(z-y) - \frac{L_h'(\xi)}{2}\|z-y\|^2$$

for any $s_y(\xi) \in \partial \max[h(y,\xi)]$. The Lipschitz continuity of $d\phi_k(\cdot)$ implies that for any scalars v and w:

$$\phi_k(w) \geq \phi_k(v) + d\phi_k(v)(w-v) - \frac{L_{\phi,k}'}{2}(w-v)^2.$$

From the monotonicity Assumption 4B and by recursively applying the above result, we have for any $\xi \in \Xi$ and $y, z \in X_\nu$:

$$\phi_k\left(g(z,\xi)\right) \geq \phi_k\left(g(y,\xi) + s_y^{\mathrm{T}}(\xi)(z-y) - \frac{L_h'(\xi)}{2}\|z-y\|^2\right)$$

$$\geq \phi_k\left(g(y,\xi) + s_y^{\mathrm{T}}(\xi)(z-y)\right) - \frac{L_h'(\xi)}{2}d\phi_k\left(g(y,\xi) + s_y^{\mathrm{T}}(z-y)\right)\|z-y\|^2 - \frac{L_{\phi,k}'\left(L_h'(\xi)\right)^2}{8}\|z-y\|^4$$

$$\geq \phi_k\left(g(y,\xi)\right) + \left(d\phi_k\left(g(y,\xi)\right)s_y(\xi)\right)^{\mathrm{T}}(z-y) - \frac{L_{\phi,k}'}{2}\left(s_y^{\mathrm{T}}(\xi)(z-y)\right)^2 - \frac{L_h'(\xi)M_{\phi,k}'}{2}\|z-y\|^2 -$$
$$\frac{L_{\phi,k}'\left(L_h'(\xi)\right)^2}{8}\|z-y\|^4$$

$$\geq \phi_k\left(g(y,\xi)\right) + \left(d\phi_k\left(g(y,\xi)\right)s_y(\xi)\right)^{\mathrm{T}}(z-y) - \frac{\|z-y\|^2}{2}\left[L_{\phi,k}'\|s_y(\xi)\|^2 + L_h'(\xi)M_{\phi,k}' +\right.$$
$$\left. \frac{L_{\phi,k}'\left(L_h'(\xi)\right)^2 \operatorname{diam}(X_\nu)^2}{4}\right].$$

Taking expectation on both sides and noting that $\|s_y(\xi)\| \leq L_h'(\xi)$, we get

$$\mathbb{E}\left[\phi_k\left(g(z,\xi)\right)\right] \geq \mathbb{E}\left[\phi_k\left(g(y,\xi)\right)\right] + \mathbb{E}\left[\left(d\phi_k\left(g(y,\xi)\right)s_y\right)^{\mathrm{T}}(z-y)\right] -$$
$$\frac{\|z-y\|^2}{2}\left[L_{\phi,k}'\mathbb{E}\left[\left(L_h'(\xi)\right)^2\right] + \mathbb{E}\left[L_h'(\xi)\right]M_{\phi,k}' + \frac{L_{\phi,k}'\mathbb{E}\left[\left(L_h'(\xi)\right)^2\right]\operatorname{diam}(X_\nu)^2}{4}\right].$$

Therefore, $\hat{p}_k$ is $\bar{L}_k$-weakly convex on X_ν with

$$\bar{L}_k := L_{\phi,k}'\mathbb{E}\left[\left(L_h'(\xi)\right)^2\right] + \mathbb{E}\left[L_h'(\xi)\right]M_{\phi,k}' + \frac{1}{4}L_{\phi,k}'\mathbb{E}\left[\left(L_h'(\xi)\right)^2\right]\operatorname{diam}(X_\nu)^2.$$

$\square$

We now outline our proposal for estimating the weak convexity parameter $\bar{L}_k$ of $\hat{p}_k$ for the above setting. First, we note that estimate of the constants $L_{\phi,k}'$ and $M_{\phi,k}'$ can be obtained from Proposition 7. Next, we propose to replace the unknown constant $\operatorname{diam}(X_\nu)$ with the diameter $2r$ of the sampling ball in Algorithm 5. Finally, we note that estimating the Lipschitz constant $L_h'(\xi)$ for any $\xi \in \Xi$ is tricky since it would involve looking at all of the $|J|$ constraints in general, just the thing we wanted to avoid! To circumvent this, we propose to replace $L_h'(\xi)$ in $\bar{L}_k$ with an estimate of the local Lipschitz constant of our choice of the B-subdifferential element of $g(\cdot,\xi)$ through sampling (similar to the proposal in Algorithm 5). When g is defined through the solution of the recourse formulation considered, we can estimate a B-subdifferential element by appealing to linear programming duality and use this as a proxy for the 'gradient' of g. Note that a crude estimate of the step length does not affect the theoretical convergence guarantee of the stochastic subgradient method of Davis and Drusvyatskiy [20].

D   Additional computational results

We present results of our replication experiments for the four case studies in the paper, and also consider a variant of Case study 3 that has a known analytical solution to benchmark our proposed approach.

Figure 7 presents the enclosure of the trajectories of the EF generated by ten replicates of the proposed approach when applied to Case study 1. For each value of the objective bound ν, we plot the smallest and largest risk level determined by Algorithm 3 at that bound over the different replicates. We find that the risk levels returned by the proposed algorithm do not vary significantly across the different replicates, with the maximum difference in the risk levels across the 26 points on the EF being a factor of 1.48. Figure 8 presents the corresponding plot for Case study 2. We again find that the risk levels returned by the proposed algorithm do not vary significantly across the different replicates, with the maximum difference in the risk levels across the 20 points on the EF being a factor of 1.006. Figure 9 presents the corresponding plot for Case study 3. The maximum difference in the risk levels at the 31 points on the EF for this case is a factor of 1.22 across the ten replicates. Figure 10 presents the corresponding plot for Case study 4. The maximum difference in the risk levels at the 26 points on the EF for this case is a factor of 1.25 across the ten replicates.

Figure 7: Enclosure of the trajectories of the efficient frontier for Case study 1 generated by ten replicates of the proposed approach.
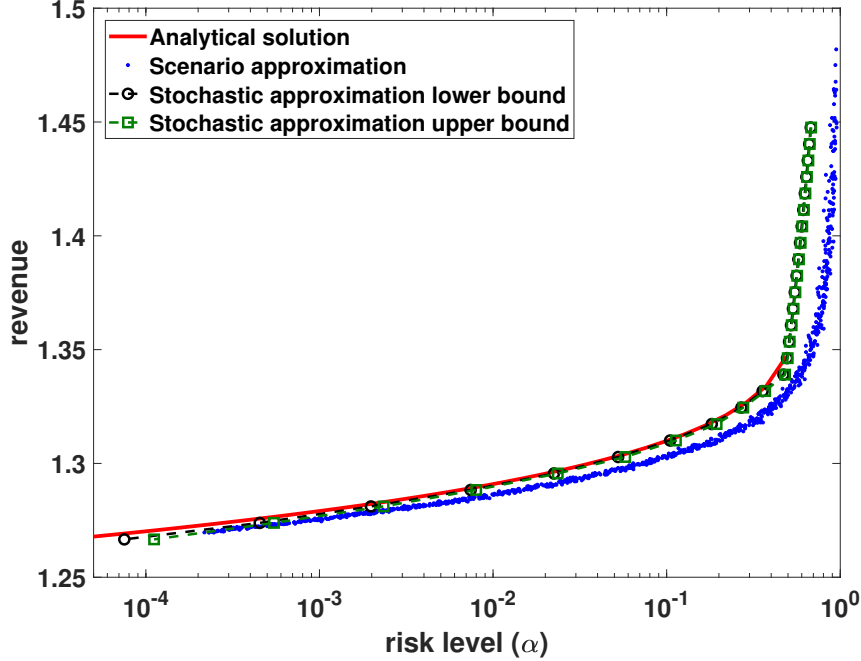
Case study 5. We consider Case study 3 when the random variables $\xi_{ij} \sim \mathcal{P} := \mathcal{N}(0,1)$ are i.i.d., the number of variables $n = 100$, the number of constraints $m = 100$, and bound $U = 100$. The EF can be computed analytically in this case, see Section 5.1.1 of Hong et al. [34]. Figure 11 compares a typical EF obtained using our approach against the analytical EF and the solutions generated by the tuned scenario approximation algorithm. Our proposed approach is able to converge to the analytical EF, whereas the scenario approximation method is only able to determine a suboptimal EF. Our proposal takes 2332 seconds on average (and a maximum of 2379 seconds) to approximate the EF using 32 points, whereas it took the scenario approximation a total of 16007 seconds to generate its 1000 points in Figure 11. We note that about 60% of the reported times for our method is spent in generating random numbers because the random variable ξ is high-dimensional.

Figure 12 presents an enclosure of the trajectories of the EF generated by the proposed approach over ten replicates for this case. We find that the risk levels returned by the proposed algorithm do not

Figure 8: Enclosure of the trajectories of the efficient frontier for Case study 2 generated by ten replicates of the proposed approach.
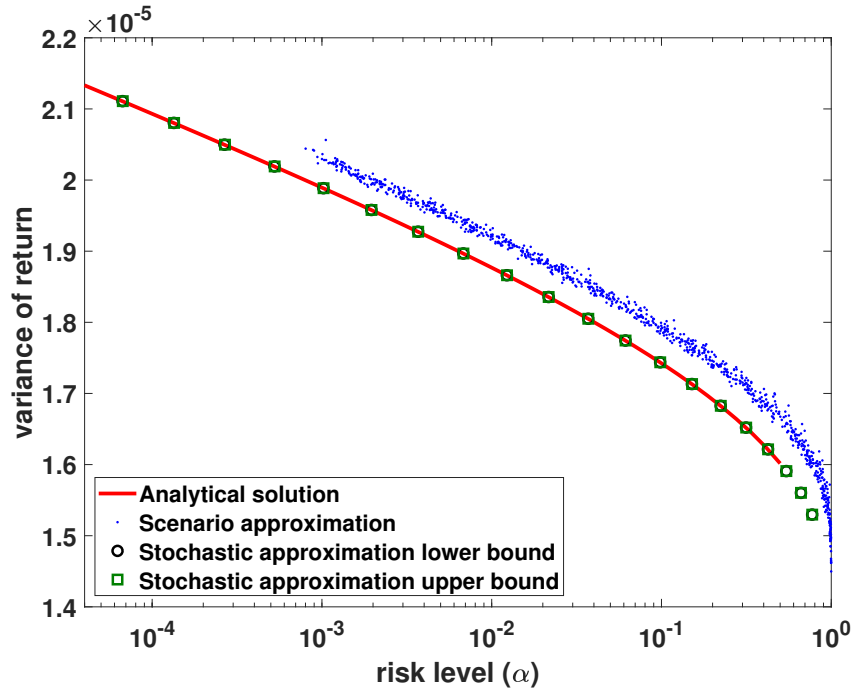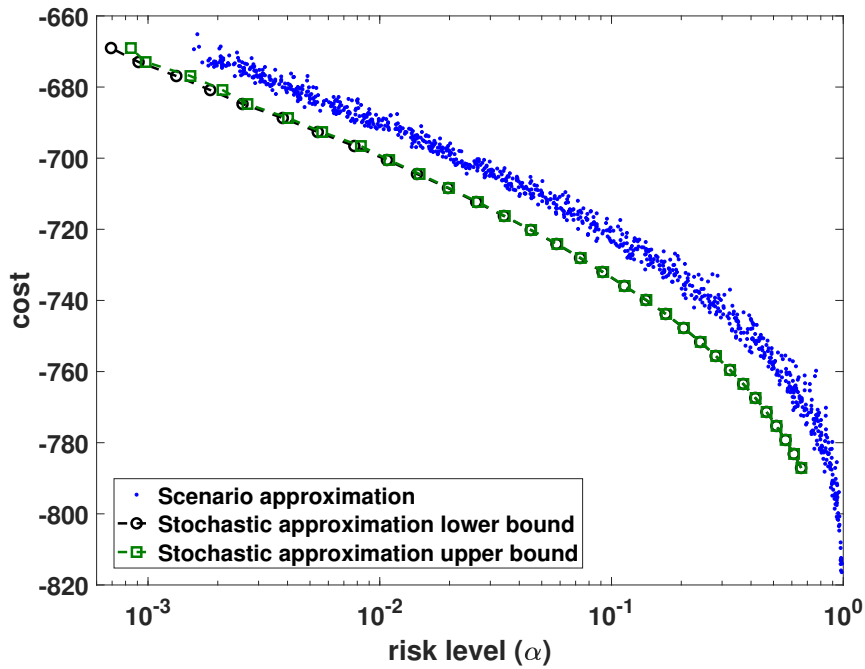

Figure 9: Enclosure of the trajectories of the efficient frontier for Case study 3 generated by ten replicates of the proposed approach.

vary significantly across the different replicates, with the maximum difference in the risk levels across the 32 points on the EF being a factor of 1.001.

Figure 10: Enclosure of the trajectories of the efficient frontier for Case study 4 generated by ten replicates of the proposed approach.
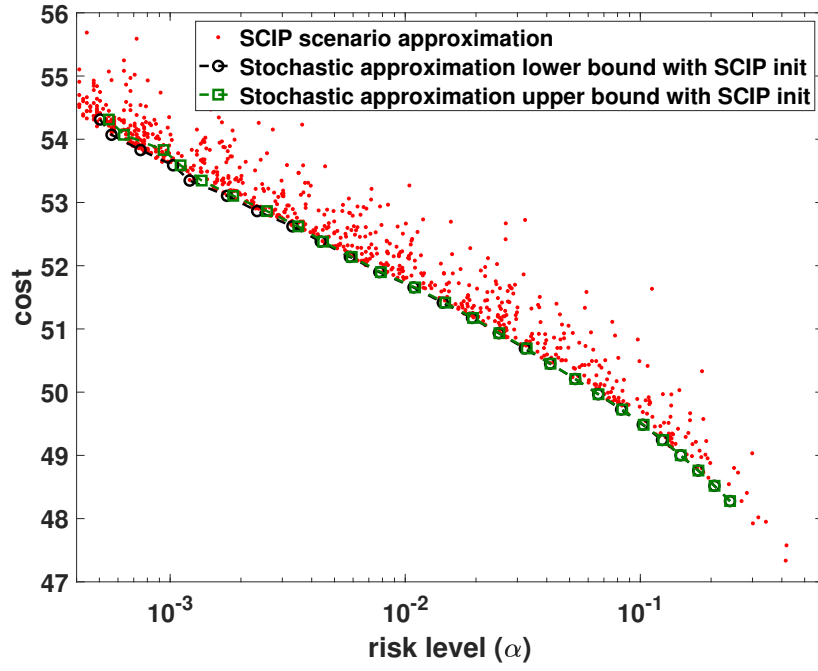

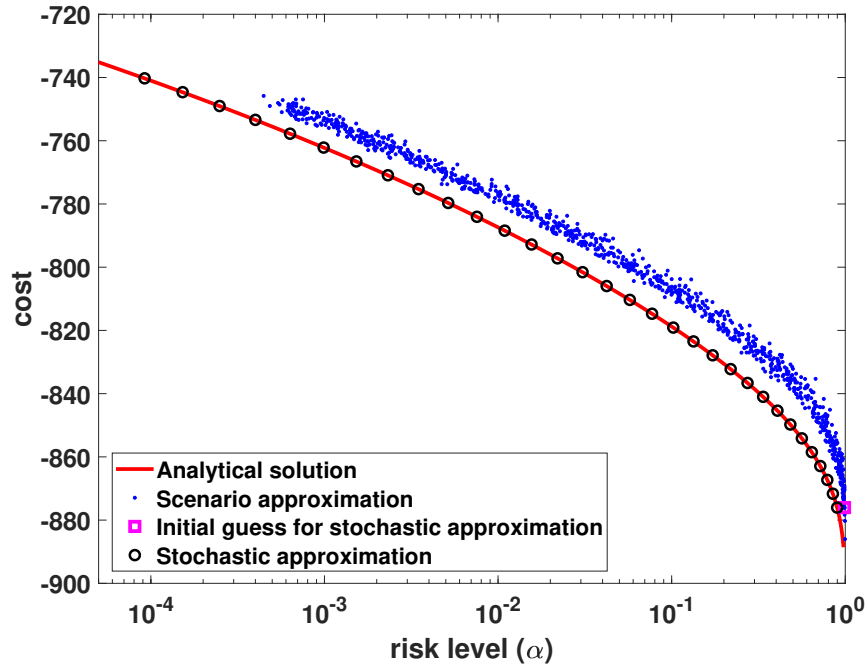Figure 11: Comparison of the efficient frontiers for Case study 5.

Figure 12: Enclosure of the trajectories of the efficient frontier for Case study 5 generated by ten replicates of the proposed approach.