A Computational Toolbox for Linear Optimization with Joint Affine Chance Constraints

Yi Chu[1,*] Welington de Oliveira[1] Wim van Ackooij[2]

[1]CMA – Centre de Mathématiques Appliquées, MINES Paris–PSL, Sophia Antipolis, France
[2]OSIRIS, EDF Lab Paris-Saclay, 7 Boulevard Gaspard Monge, 91120 Palaiseau, France

[*]Corresponding author: yi.chu@minesparis.psl.eu

Abstract

We present a Julia computational toolbox for linear optimization problems with joint affine chance constraints under elliptically symmetric uncertainty. The toolbox combines a spherical–radial oracle for estimating the joint probability and its gradient with three structured optimization methods: Proximal, Feasible, and Penalty. The oracle supports several elliptically symmetric distributions and is integrated with these methods through a common computational interface. Interfaces to a level-bundle method and IPOPT are also provided for comparison. Under suitable regularity assumptions, the structured methods satisfy their respective convergence guarantees, and feasible critical points satisfy the Karush–Kuhn–Tucker (KKT) conditions under an appropriate constraint qualification. Numerical experiments on cash-matching, transportation, and realistic hydro-valley problems under Gaussian and Student-t uncertainty assess computational time, probability-oracle evaluations, and sensitivity to initialization. The results show that the structured methods are generally more efficient and robust than the benchmark approaches across the tested problem classes.

Keywords: Joint chance constraints; Linear programming; Spherical–radial decomposition; Proximal methods; Computational toolbox.

1 Introduction

In this paper, we consider linear optimization problems with a joint affine chance constraint of the form

$$\begin{aligned} \min_{x\in X} \quad & c^\top x \\ \text{s.t.} \quad & \mathbb{P}(W\xi \le Hx + h) \ge p, \end{aligned} \tag{P}$$

where $x \in \mathbb{R}^n$ is the decision vector, $c \in \mathbb{R}^n$ is a deterministic cost vector, and

$$X := \{x \in \mathbb{R}^n : Ax \le b,\ A_{\text{eq}}x = b_{\text{eq}},\ l \le x \le u\}$$

is a compact polyhedral set representing the deterministic constraints and variable bounds. The random vector $\xi \in \mathbb{R}^m$ represents the uncertain quantities affecting the system of inequalities $W\xi \le Hx + h$ and is assumed throughout the paper to follow a continuous, nondegenerate, elliptically symmetric distribution. The matrices and vectors $W \in \mathbb{R}^{d\times m}, H \in \mathbb{R}^{d\times n}, h \in \mathbb{R}^d$ are deterministic and of compatible dimensions. All inequalities inside the probability are interpreted componentwise, and $p \in (0, 1)$ denotes the prescribed reliability level.

The probabilistic constraint in problem (P) requires all d inequalities to hold simultaneously and thus controls the reliability of the system as a whole. This is fundamentally different from

imposing separate individual chance constraints, which, in general, do not guarantee control of the corresponding joint probability, particularly when the uncertain quantities are dependent; see, e.g., [14] and [38, Ch. 8].

Models of the form (P) arise in transportation, production and inventory planning, energy-system operation, network design, and supply-chain optimization; see, for example, [1, 18, 19, 23]. The principal application considered in this paper is hydro-valley reservoir management. In this setting, the decision variables describe water releases and hydropower production over a planning horizon, while the random vector represents uncertain future inflows. Reservoir dynamics transform these inflows into random storage trajectories, and a joint chance constraint can require all lower and upper storage bounds to be satisfied throughout the planning horizon with probability at least p. The Isère and Ain Valley instances used in our experiments arise from realistic energy-planning models previously studied in [30, 33].

Many engineering constraints can be written directly in the form of problem (P). For example, $H_\ell x + h_\ell \leq \widetilde{W}\xi \leq H_u x + h_u$, which corresponds to

$$W := \begin{pmatrix} \widetilde{W} \\ -\widetilde{W} \end{pmatrix}, \quad H := \begin{pmatrix} H_u \\ -H_\ell \end{pmatrix}, \quad h := \begin{pmatrix} h_u \\ -h_\ell \end{pmatrix}.$$

Although this construction produces a rank-deficient matrix W, the resulting box-probability function may still satisfy the joint and marginal $C^{1,1}$ regularity conditions required by the analysis; see Remark 3.

Define the probability function $\varphi(x) := \mathbb{P}(W\xi \leq Hx + h)$. The chance constraint can then be written equivalently as $p - \varphi(x) \leq 0$. Although the objective and deterministic constraints in problem (P) are linear, φ is the multivariate distribution function of $W\xi$ evaluated at the affine image $Hx + h$, and is therefore not, in general, an affine or convex function of x. Consequently, the resulting optimization problem may be nonconvex.

The main computational difficulty lies in repeatedly evaluating $\varphi(x)$. Each evaluation requires computing the probabilistic content of the polyhedron $\{z \in \mathbb{R}^m : Wz \leq Hx + h\}$, which can be expensive when either the dimension of the uncertainty or the number of probabilistic inequalities is large. Moreover, optimization algorithms require gradient information describing how the probability changes with the decision vector. Probability and gradient evaluations may therefore account for a substantial portion of the total computational cost, and their computation is often affected by numerical inaccuracies and statistical estimation errors.

Several numerical approaches have been developed for chance-constrained optimization. Exact deterministic reformulations are available in certain special cases, such as individual linear chance constraints under Gaussian uncertainty [15] (but also elliptical see e.g., [11]). When the probability function is log-concave, its superlevel sets are convex, which permits the use of convex optimization methods, including level-bundle methods [30]. For nonconvex settings, another line of work exploits difference-of-convex (DC) structure: suitably structured chance-constrained models or their scenario-based approximations can be reformulated as DC programs and addressed with dedicated DC or bundle methods; see, e.g., [26, 29] and [38, Chp. 19]. More generally, the literature includes conservative convex approximations [21], scenario-based methods [2], and sample-average approximation methods [18, 23]. Chance-constrained problems have also been treated directly as standard nonlinear programs when sufficiently accurate probability and derivative evaluations are available [17, 39].

These approaches address complementary problem settings. They nevertheless motivate a modular computational framework that can treat the original joint probability constraint directly, accommodate non-log-concave probability functions, and exploit probability-specific gradient information. The framework developed in this paper addresses these objectives by separating the deterministic optimization model, the probability calculation, and the optimization method.

Let

$$\eta := W\xi, \quad F_\eta(y) := \mathbb{P}(\eta \leq y), \quad y(x) := Hx + h.$$

Then $\varphi(x) = F_\eta(y(x))$. Since the class of elliptically symmetric distributions is closed under affine transformations, the transformed random vector $\eta = W\xi$ is also elliptically symmetric; see [16, Theorem 2.3.8]. Introducing η therefore absorbs the fixed transformation W into the distribution of the uncertainty and expresses the chance constraint directly in the d-dimensional space of probabilistic inequalities. Each component η_i corresponds to one random inequality, while the decision vector enters only through the threshold $y(x)$. This representation exposes the joint and marginal distributions used by the probability oracle and the local model, and remains valid for rank-deficient transformations, including stacked two-sided constraints.

This composition separates the deterministic and probabilistic components of the model. The deterministic component maps x to the threshold vector $y(x)$, while the probability oracle evaluates $F_\eta(y(x))$ and returns the corresponding gradient information. Under the regularity conditions introduced in Section 2, the probability function is $C^{1,1}$ and through the chain rule we have $\nabla\varphi(x) = H^\top \nabla F_\eta(y(x))$. The optimization method therefore interacts with the probability calculation only through oracle calls: it supplies a threshold vector and receives a probability value together with its gradient. The deterministic model, probability oracle, and optimization algorithm can consequently be implemented as separate modules.

For elliptically symmetric random vectors, the probability oracle is based on a spherical–radial decomposition; see, e.g., [9, 10]. The radial component is treated analytically, while the remaining expectation over the unit sphere is approximated numerically. The same directional calculations provide estimates of both the probability and its gradient; see [31, 32, 34, 35].

Certain structured optimization methods introduced below exploit a copula decomposition of the joint probability. Writing F_i for the marginal distribution function of η_i, Sklar's theorem gives, under the regularity conditions stated later, $\varphi(x) = C(F_1(y_1(x)), \ldots, F_d(y_d(x)))$, where C is the copula of η; see [8]. This representation motivates a structured local model that linearizes the copula while retaining the marginal distribution functions in their nonlinear form. The construction and its required regularity conditions are detailed in Section 3.2.

Based on this common local model, we develop two optimization methods in this paper: the *Feasible method*, which starts from a feasible point and accepts only trial points satisfying the original chance constraint, and the *Penalty method*, which incorporates chance-constraint violations into the objective. The toolbox also implements the *Proximal method* developed in [37], which balances objective improvement and chance-constraint satisfaction. For comparison, we include a level-bundle method for the log-concave setting and an interface to IPOPT as a general-purpose nonlinear-programming baseline. The Proximal, Feasible, Penalty, and IPOPT approaches require the probability function φ to be sufficiently smooth, whereas the level-bundle method instead relies on the convexity induced by log-concavity of φ. The precise regularity requirements of the different methods are stated in Section 3.

Applicability and toolbox interface. The toolbox is designed for problems of the form (P), with a linear objective, a compact polyhedral deterministic feasible set, affine probabilistic inequalities, and a fixed, decision-independent uncertainty distribution. The toolbox is implemented in Julia, whose combination of high-level mathematical programming tools and efficient numerical computation is well suited to integrating the probability oracle with the optimization methods considered here. Its interface can be summarized through the required model data, the available computational components, and the returned solution information.

The required inputs are:

- the deterministic optimization data $c, A, b, A_{\text{eq}}, b_{\text{eq}}, l, u$;
- the joint chance-constraint data W, H, h, p;
- the uncertainty model ξ, including its distribution family, location vector μ, and dispersion matrix Σ; and

- the numerical parameters of the probability oracle and the selected optimization method, together with an optional initial point.

Given these inputs, the toolbox provides:

- a spherical–radial oracle for estimating $\varphi(x)$ and its gradient;
- the Proximal method of [37], together with two new methods developed in this paper, namely the Feasible and Penalty methods; and
- interfaces to the level-bundle method and IPOPT.

For a selected solution method, the toolbox returns a candidate solution $\widehat{x}$, its objective value $c^\top \widehat{x}$, the estimated joint probability $\widehat{\varphi}_N(\widehat{x})$, an estimate of the numerical accuracy of the probability evaluation, the number of oracle calls, and the termination status. The convergence guarantees additionally require the compactness and $C^{1,1}$ regularity conditions stated in Section 2.

General nonlinear objectives or deterministic constraints, nonlinear transformations of x or ξ inside the probabilistic inequalities, and decision-dependent uncertainty distributions are outside the scope of the present paper and toolbox. We focus explicitly on the linear setting because of its prevalence in applications and simplicity; this restriction does not imply that nonlinear extensions cannot be treated efficiently in general.

The main contributions of this paper are as follows.

- We present a modular computational toolbox for linear optimization problems with joint affine chance constraints. The deterministic model, probability oracle, and optimization method communicate through a common computational interface. Our code, including several test problems, is available at `https://gitlab.com/stochasticprogramming/ProbLiM`.
- We integrate a spherical–radial probability oracle that estimates the joint probability and its gradient for several elliptically symmetric distributions.
- We construct a $C^{1,1}$ copula-based local model for problem (P). The model linearizes the copula while retaining the marginal distribution functions in their nonlinear form. Importantly, the required linearization can be constructed without explicit knowledge or evaluation of the underlying copula, using the marginal-assessment procedure of [36]; see also [37].
- We establish convergence results for the two proposed methods under the stated compactness and $C^{1,1}$ regularity assumptions.
- We evaluate the toolbox on cash-matching, transportation, and realistic hydro-valley problems and compare the three structured methods with a level-bundle method and IPOPT.

The remainder of the paper is organized as follows. Section 2 presents the probability and gradient oracle based on spherical–radial decomposition. Section 3 describes the five solution approaches and states the convergence results for the two proposed methods. Section 4 reports numerical experiments on the cash-matching, transportation, and hydro-valley benchmark problems. Section 5 summarizes the main findings. Formal criticality concepts and the convergence proofs for the Feasible and Penalty methods are collected in the appendices; the Proximal result follows from [37].

2 Probability and first-order oracle

A central component of the toolbox is an oracle for evaluating

$$\varphi(x) = \mathbb{P}(W\xi \leq y(x)), \; y(x) := Hx + h,$$

together with its gradient with respect to x. The following assumptions are used throughout the probability analysis and the convergence analysis of the optimization methods. In particular, full row rank of W is not required.

2.1 Regularity assumptions

Assumption 1. *The following conditions hold:*

A.1 The set

$$X = \{x \in \mathbb{R}^n : Ax \le b,\ A_{\mathrm{eq}}x = b_{\mathrm{eq}},\ l \le x \le u\}$$

is nonempty and compact.

A.2 The feasible set of problem (P) *is nonempty. Consequently, problem* (P) *has a finite optimal value and admits an optimal solution.*

A.3 The random vector $\xi \in \mathbb{R}^m$ is continuous, nondegenerate, and elliptically symmetric, with positive definite dispersion matrix Σ.

A.4 Let $\eta := W\xi \in \mathbb{R}^d$, and let F_η denote its joint distribution function. The function F_η is $C^{1,1}$ on an open neighbourhood of $y(X) := \{Hx + h : x \in X\}$; see also [36] for convergence results under $C^{1,1}$ regularity assumptions.

A.5 For each $i = 1, \dots, d$, the component η_i is nondegenerate, and its distribution function F_i is $C^{1,1}$ on an open neighbourhood U_i of $y_i(X) := \{(Hx + h)_i : x \in X\}$, with density $f_i := F_i' > 0$ on U_i.

Remark 1. The compactness of X in Assumption A.1 is not necessary for the convergence analysis of the proximal, feasible, or penalty methods considered in this paper; it is imposed only to simplify the presentation and proofs. See, for example, [37] for convergence results for the proximal method under more general settings without assuming compactness of X.

Remark 2 ($C^{1,1}$ regularity of the probability function). Under Assumption 1, $\varphi(x) = F_\eta(Hx+h)$ is $C^{1,1}$ on an open neighbourhood of X. Indeed, F_η is $C^{1,1}$ on a neighbourhood of $y(X)$ by Assumption A.4, while $x \mapsto Hx + h$ is affine. Therefore, their composition is $C^{1,1}$, with

$$\nabla\varphi(x) = H^\top \nabla F_\eta(Hx + h).$$

In particular, $\nabla\varphi$ is Lipschitz continuous on a neighbourhood of X.

Remark 3 (Stacked two-sided constraints). Assumptions A.4 and A.5 do not require W to have full row rank. They can therefore also accommodate the stacked formulation of the two-sided probabilistic constraints $\ell(x) \le \widetilde{W}\xi \le u(x)$, where $\ell(x) := H_\ell x + h_\ell$ and $u(x) := H_u x + h_u$.

Indeed, let $\widetilde{\eta} := \widetilde{W}\xi \in \mathbb{R}^q$, $\eta := W\xi = \begin{pmatrix} \widetilde{\eta} \\ -\widetilde{\eta} \end{pmatrix} \in \mathbb{R}^{2q}$. Then, for the stacked formulation,

$$F_\eta(Hx + h) = F_\eta\left(\begin{pmatrix} u(x) \\ -\ell(x) \end{pmatrix}\right) = \mathbb{P}\left(\eta \le \begin{pmatrix} u(x) \\ -\ell(x) \end{pmatrix}\right) = \mathbb{P}\big(\ell(x) \le \widetilde{\eta} \le u(x)\big),$$

where all inequalities are understood componentwise. Consequently, Assumption A.4 holds for the stacked formulation if $\ell_i(x) < u_i(x)$, $i = 1, \dots, q$, on a neighbourhood of X, and the box-probability $\mathbb{P}(r \le \widetilde{\eta} \le s)$ is $C^{1,1}$ on a neighbourhood of $\big\{\big(\ell(x), u(x)\big) : x \in X\big\}$. A sufficient condition is that $F_{\widetilde{\eta}}$ be $C^{1,1}$ on a neighbourhood of all corner vectors generated by the lower and upper thresholds. Similarly, Assumption A.5 holds if each $\widetilde{\eta}_i$ is nondegenerate, its marginal distribution function is $C^{1,1}$ near both the corresponding lower and upper thresholds, and its density is positive there. We refer to [12] and [28] for a full exploration of the Gaussian singular case.

The spherical–radial representation developed below gives exact integral expressions for $\varphi(x)$ and $\nabla\varphi(x)$.

2.2 Spherical–radial representation

The probability oracle exploits the spherical–radial representation of elliptically symmetric random vectors. A random vector $\xi \in \mathbb{R}^m$ is elliptically symmetric with location $\mu \in \mathbb{R}^m$ and positive definite dispersion matrix $\Sigma = LL^\top$ if

$$\xi \overset{d}{=} \mu + RL\zeta,$$

where L is nonsingular, ζ is uniformly distributed on the unit sphere $\mathbb{S}^{m-1} := \{z \in \mathbb{R}^m : \|z\| = 1\}$, and $R \geq 0$ follows a distribution that is independent of ζ; see, e.g., [16, Sec. 2.3]. Thus, ζ determines a direction, L accounts for the dispersion structure, and R determines the distance from the location μ. The distribution of R determines the particular elliptical family. Important examples include the multivariate Gaussian and Student-t distributions, the symmetric multivariate Laplace distribution, and the uniform distribution on an ellipsoid.

We denote the distribution function and density of R by F_R and f_R, respectively. Several prominent choices have been pre-implemented in the toolbox as summarized in Table 1, but all are available through an appropriate API for the user. Here, F_{χ_m} and f_{χ_m} denote the distribution function and density of a chi distribution with m degrees of freedom, while $F_{m,\nu}$ and $f_{m,\nu}$ denote those of an F-distribution (i.e., Fisher-Snedecor distribution) with m numerator and ν denominator degrees of freedom. For the Student-t distribution, Σ is the dispersion, or scale, matrix.

For the symmetric Laplace distribution, $E \sim \mathrm{Exp}(1)$ and $Q \sim \chi_m$ are independent, and K_α denotes the modified Bessel function of the second kind. For the uniform distribution, $U \sim \mathcal{U}(0,1)$ and $a > 0$ is the radial scale. The expressions for $F_R(t)$ and $f_R(t)$ below are given for $t \geq 0$; both are zero for $t < 0$.

Table 1: Radial distributions supported by the toolbox.

Distribution	Radial law	$F_R(t)$	$f_R(t)$
Gaussian	$R \sim \chi_m$	$F_{\chi_m}(t)$	$f_{\chi_m}(t)$
Student-t_ν	$R^2/m \sim F_{m,\nu}$	$F_{m,\nu}\left(\dfrac{t^2}{m}\right)$	$\dfrac{2t}{m} f_{m,\nu}\left(\dfrac{t^2}{m}\right)$
Symmetric Laplace	$R = \sqrt{E}\,Q$	$\displaystyle\int_0^\infty 2se^{-s^2} F_{\chi_m}\left(\frac{t}{s}\right) ds$	$\dfrac{2^{(6-m)/4}}{\Gamma(m/2)} t^{m/2} K_{m/2-1}\left(\sqrt{2}\,t\right)$
Uniform inside an ellipsoid	$R = aU^{1/m}$	$\min\left\{1, \left(\dfrac{t}{a}\right)^m\right\}$	$\dfrac{mt^{m-1}}{a^m} \mathbf{1}_{[0,a]}(t)$
User-defined elliptical	User-specified R	Associated $F_R(t)$	Associated $f_R(t)$

Substituting $\xi = \mu + RL\zeta$ into the chance constraint gives $W\mu + RWL\zeta \leq y(x)$. Once a direction ζ is fixed, the only remaining random quantity is the scalar radius R, and the d inequalities reduce to bounds on R. Let $w_j^\top$ denote the j-th row of W. The j-th inequality is $R\, w_j^\top L\zeta \leq y_j(x) - w_j^\top \mu$. Whenever $w_j^\top L\zeta \neq 0$, define the corresponding boundary radius by

$$\rho_j(x,\zeta) = \frac{y_j(x) - w_j^\top \mu}{w_j^\top L\zeta}.$$

For the first-order analysis, we exclude points at which the location lies on a constraint boundary, i.e., $w_j^\top \mu = y_j(x)$ for some j. Thus, $w_j^\top \mu \neq y_j(x)$ for every j, leaving two cases: strict feasible case $W\mu < y(x)$, or at least one constraint is strictly violated.

Suppose first that $W\mu < y(x)$. For a fixed direction ζ, constraints with $w_j^\top L\zeta > 0$ impose upper bounds $R \leq \rho_j(x,\zeta)$, whereas those with $w_j^\top L\zeta \leq 0$ are satisfied for every $R \geq 0$. The largest feasible radius is therefore

$$\rho(x,\zeta) = \min_{j:\, w_j^\top L\zeta > 0} \rho_j(x,\zeta),$$

where the minimum of the empty set is $+\infty$. Hence, the feasible radii are $0 \le R \le \rho(x,\zeta)$, and

$$\chi(x,\zeta) = F_R\big(\rho(x,\zeta)\big),$$

with $F_R(+\infty) = 1$.

To obtain the gradient, assume $\lim_{r\to+\infty} r^2 f_R(r) = 0$ and the Rank-2 Constraint Qualification of [34, Corollary 1]. In the present setting, the latter requires any two constraints that are simultaneously active at a feasible point to have linearly independent rows in W. Since the constraints are affine, the remaining smoothness, convexity, and constraint-growth requirements of [34, Theorem 1] are automatic; see also [34, Lemma 5]. Under these conditions, the constraint determining $\rho(x,\zeta)$ is unique for almost every direction. Denoting it by j^*,

$$e(x,\zeta) := \nabla_x \chi(x,\zeta) = f_R\big(\rho(x,\zeta)\big)\frac{H_{j^*}^\top}{w_{j^*}^\top L\zeta}.$$

If $\rho(x,\zeta) = +\infty$, we set $e(x,\zeta) := 0$. Moreover, [34, Corollary 1] permits differentiation under the spherical integral, yielding

$$\varphi(x) = \int_{\mathbb{S}^{m-1}} \chi(x,\zeta)\,\mathrm{d}\sigma(\zeta), \qquad \nabla\varphi(x) = \int_{\mathbb{S}^{m-1}} e(x,\zeta)\,\mathrm{d}\sigma(\zeta),$$

where σ is the uniform probability measure on $\mathbb{S}^{m-1}$.

The next subsection treats the remaining case, in which the location violates at least one probabilistic inequality.

2.3 Partitioning of the radial interval

Suppose now that $W\mu \not\le y(x)$. Since boundary cases are excluded, at least one inequality is strictly violated at μ. For a fixed direction ζ, the ray may enter the feasible region at a positive radius, leave it later, or fail to intersect it altogether.

The j-th inequality gives a lower bound $R \ge \rho_j(x,\zeta)$ when $w_j^\top L\zeta < 0$, and an upper bound $R \le \rho_j(x,\zeta)$ when $w_j^\top L\zeta > 0$. Combining these bounds with $R \ge 0$, define the entry and exit radii by

$$\rho_l(x,\zeta) = \max\Big\{0,\ \rho_j(x,\zeta) : w_j^\top L\zeta < 0\Big\}, \qquad \rho_u(x,\zeta) = \min\Big\{\rho_j(x,\zeta) : w_j^\top L\zeta > 0\Big\},$$

where $\rho_u(x,\zeta) = +\infty$ if no upper bound is present.

If $w_j^\top L\zeta = 0$, the j-th constraint does not depend on R: it is satisfied along the entire ray when $w_j^\top\mu < y_j(x)$, and otherwise no radius is feasible. Provided all such constraints are satisfied, the feasible radii form $[\rho_l(x,\zeta), \rho_u(x,\zeta)]$ when $\rho_l(x,\zeta) < \rho_u(x,\zeta)$. Hence,

$$\chi(x,\zeta) = \begin{cases} F_R\big(\rho_u(x,\zeta)\big) - F_R\big(\rho_l(x,\zeta)\big), & \rho_l(x,\zeta) < \rho_u(x,\zeta) < +\infty, \\ 1 - F_R\big(\rho_l(x,\zeta)\big), & \rho_l(x,\zeta) < \rho_u(x,\zeta) = +\infty, \\ 0, & \text{otherwise}, \end{cases}$$

where the first two cases additionally require $w_j^\top\mu < y_j(x)$ whenever $w_j^\top L\zeta = 0$. Thus, for each direction, the oracle determines the feasible radial interval and evaluates its probability under R.

For the gradient, assume in addition that, for every $i \ne j$, $\big(y_i(x) - w_i^\top\mu\big)w_j^\top \ne \big(y_j(x) - w_j^\top\mu\big)w_i^\top$. This is the qualification condition of [35, Theorem 4.2] specialized to the present linear system and ensures that the active entry and exit constraints are unique for almost every

direction. Denote these constraints by j_1^* and j_2^*. For each finite endpoint determined by a constraint,

$$\nabla_x \rho_k(x,\zeta) = \frac{H_{j_k^*}^\top}{w_{j_k^*}^\top L\zeta}, \qquad k = 1, 2.$$

When $\rho_l(x,\zeta) = 0$ is determined by $R \geq 0$, we set $\nabla_x \rho_l(x,\zeta) = 0$. The directional gradient is

$$e(x,\zeta) = \begin{cases} f_R(\rho_u)\nabla_x \rho_u - f_R(\rho_l)\nabla_x \rho_l, & \rho_l < \rho_u < +\infty, \\ -f_R(\rho_l)\nabla_x \rho_l, & \rho_l < \rho_u = +\infty, \\ 0, & \text{otherwise}, \end{cases}$$

where the arguments (x,ζ) are omitted for readability. Under the radial growth condition above and the qualification condition of [35, Theorem 4.2], the directional gradient is well defined for almost every direction. Moreover, [35, Theorems 4.1–4.2] justify interchange of differentiation and spherical integration, yielding

$$\varphi(x) = \int_{\mathbb{S}^{m-1}} \chi(x,\zeta)\,\mathrm{d}\sigma(\zeta), \qquad \nabla\varphi(x) = \int_{\mathbb{S}^{m-1}} e(x,\zeta)\,\mathrm{d}\sigma(\zeta).$$

The preceding formulas provide exact spherical integral representations of the probability and its gradient. To obtain a computational oracle, we now approximate these integrals using a finite set of spherical directions.

2.4 Probability and first-order estimates

Given directions $\zeta_1, \ldots, \zeta_N \in \mathbb{S}^{m-1}$, the spherical integrals are approximated by

$$\widehat{\varphi}_N(x) = \frac{1}{N}\sum_{s=1}^{N} \chi(x,\zeta_s), \qquad \widehat{\nabla\varphi}_N(x) = \frac{1}{N}\sum_{s=1}^{N} e(x,\zeta_s).$$

Figure 1 summarizes the resulting oracle: for a given x, the directional probability and gradient contributions are evaluated and averaged to return estimates of $\varphi(x)$ and $\nabla\varphi(x)$.

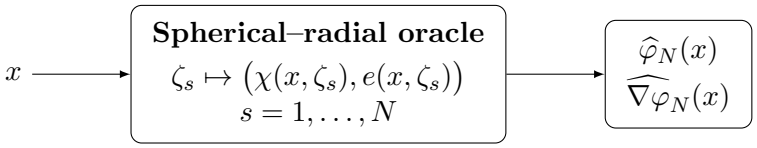

Figure 1: Spherical–radial probability oracle.

Under the qualification conditions above, exact ties between active boundaries occur only on a set of spherical measure zero. If a numerical tie occurs in the implementation, a consistent active-boundary selection or an average of the tied contributions may be used. For each direction, the radial variable is integrated analytically through F_R and f_R, while only the spherical integral is approximated numerically. Such spherical–radial estimators can provide substantial variance reduction compared with standard Monte Carlo sampling; this reduction can be particularly pronounced for probabilities close to 0 or 1; see, e.g., [13].

The toolbox provides the following procedures for generating spherical directions:

- **Monte Carlo (MC) sampling.** Independent vectors $v_s \sim \mathcal{N}(0, I_m)$ are normalized as $\zeta_s = v_s/\|v_s\|$, yielding directions uniformly distributed on $\mathbb{S}^{m-1}$. The antithetic variant (MCA) evaluates each direction ζ_s together with $-\zeta_s$.

Table 2: Gaussian sampling comparison with $N = 10^4$ sampling directions. Each entry reports the mean absolute probability error e_φ (top), the mean gradient error e_∇ (middle), and the mean elapsed time in seconds (bottom, in parentheses), averaged in $J = 10$ trials, where $e_\varphi = \frac{1}{J}\sum_{j=1}^{J} |\widehat{\varphi}_j - \varphi|$, $e_\nabla = \frac{1}{J}\sum_{j=1}^{J} \|\widehat{\nabla\varphi}_j - \nabla\varphi\|_2$.

m	MC	MCA	QMC	QMCA	Deák
1	6.37×10^{-5} 1.51×10^{-4} (0.009)	0 0 (0.006)	0 0 (0.047)	0 0 (0.014)	0 0 (0.008)
5	6.52×10^{-4} 3.68×10^{-3} (0.014)	9.75×10^{-4} 3.28×10^{-3} (0.011)	2.84×10^{-5} 4.60×10^{-4} (0.216)	7.74×10^{-5} 8.62×10^{-4} (0.092)	1.05×10^{-3} 5.38×10^{-4} (0.033)
10	1.56×10^{-3} 4.77×10^{-3} (0.020)	8.01×10^{-4} 5.59×10^{-3} (0.018)	3.78×10^{-4} 1.52×10^{-3} (0.297)	4.07×10^{-4} 3.38×10^{-3} (0.149)	3.29×10^{-3} 1.92×10^{-3} (0.042)
50	4.18×10^{-3} 1.12×10^{-2} (0.060)	1.88×10^{-4} 1.23×10^{-2} (0.049)	5.78×10^{-4} 8.80×10^{-3} (1.101)	1.68×10^{-4} 8.65×10^{-3} (0.606)	3.70×10^{-3} 1.12×10^{-2} (0.126)
100	1.07×10^{-3} 9.82×10^{-3} (0.101)	8.97×10^{-4} 1.01×10^{-2} (0.092)	2.10×10^{-3} 9.40×10^{-3} (1.929)	5.07×10^{-4} 1.05×10^{-2} (1.130)	4.29×10^{-4} 9.61×10^{-3} (0.378)

- **Randomized quasi–Monte Carlo (QMC) sampling.** Owen-scrambled Sobol points $u_s \in [0,1]^m$ are transformed componentwise as $v_{s,j} = \Phi^{-1}(u_{s,j})$ and $\zeta_s = v_s/\|v_s\|$. This construction provides a low-discrepancy coverage of the sphere [22, 25] and has also been used within spherical–radial probability calculations; see, e.g., [9]. It can also be combined with antithetic directions. The antithetic randomized quasi–Monte Carlo variant (QMCA) is the default sampling procedure in the toolbox.
- **Deák sampling.** Structured direction sets are constructed from randomly generated orthonormal systems $U = [u_1, \ldots, u_m]$, and the radial contributions are averaged over the resulting direction blocks; see [4, 5]. Orthonormal systems can be generated following the approach laid out in [6]. Following the suggestions by [27], we consider four variants: *plain*, using $\{u_i\}_{i=1}^m$; *ZD*, using $\{\pm u_i\}_{i=1}^m$; *DD*, using all pairwise directions $\{(\pm u_i \pm u_j)/\sqrt{2} : i < j\}$; and *AD*, using $\{\pm(u_i - u_j)/\sqrt{2} : i < j\}$. The variant *DD* corresponds to Deák's original scheme.

Table 2 compares these procedures for evaluating the probability function and its gradient on a test instance with $\xi \sim \mathcal{N}(0, I_m)$, $W = H = I_m$, $h = \mu = 0$, and $x = 2\mathbf{1}_m$. In this setting, the exact values are $\varphi(x) = \Phi(2)^m$, $\nabla\varphi(x) = \varphi(2)\Phi(2)^{m-1}\mathbf{1}_m$, where Φ and φ denote the standard normal distribution and density functions, respectively. Each procedure uses the total sampling directions $N = 10{,}000$. For the Deák method, we use the ZD variant: for each randomly generated orthonormal system $\{u_1, \ldots, u_m\}$, the corresponding direction block consists of the $2m$ signed directions $\{\pm u_i : i = 1, \ldots, m\}$.

Remark 4 (Sampling-budget convention). For $m = 1$, the sphere reduces to $\mathbb{S}^0 = \{-1, +1\}$. Antithetic and signed-direction schemes evaluate these two directions with equal weight and are therefore exact in this special case. Plain MC, in contrast, samples the two signs independently, so a finite sample need not contain exactly the same number of $+1$ and -1 directions; consequently, its probability and gradient estimates can have nonzero error even when $m = 1$.

Overall, the results indicate that the QMC-based schemes generally provide competitive and stable accuracy, particularly for the gradient estimates. Deák sampling remains competitive across the tested dimensions and is substantially faster than the QMC-based schemes, especially

in higher dimensions. With the probability oracle and its first-order estimates in place, we now turn to the optimization methods that use this information to solve problem (P).

3 Solution methods

The toolbox provides five methods for solving problem (P). The proximal, feasible, and penalty methods share the structured local model introduced in Section 3.2, while the level-bundle method and IPOPT are included as alternative approaches. All five methods access the probability oracle through the same interface, but differ in their structural assumptions, initialization requirements, and treatment of feasibility.

3.1 Choosing a solution method

Table 3 summarizes the practical role of each method. The proximal, feasible, and penalty methods solve a $C^{1,1}$ nonlinear master problem at each iteration using IPOPT, with the required second-order information numerically approximated by IPOPT. The level-bundle method instead uses quadratic programming (QP) master problems and does not require such smoothness. All four methods are designed to tolerate inexact probability and gradient evaluations. IPOPT, when applied directly to the original $C^{1,1}$ nonlinear program, is more sensitive to oracle inaccuracy.

Table 3: Practical guide for selecting a solution method.

Method	Master problem	Best suited for/when	Main limitation
Proximal [37]	$C^{1,1}$ NLP with improvement function	General-purpose use, including difficult nonconvex instances.	The master problem can be costly.
Feasible	$C^{1,1}$ NLP with feasibility correction	A good feasible initial point is available.	Requires feasible initialization and may explore the feasible region conservatively.
Penalty	$C^{1,1}$ NLP with penalized objective	The chance constraint is not highly restrictive.	Penalty selection can be delicate, and fixed-penalty run does not guarantee feasibility.
Level-bundle	QP	Log-concave probability functions; smoothness is not required.	Requires log-concavity.
IPOPT	Direct $C^{1,1}$ NLP	A standard NLP baseline when accurate probability and gradient evaluations are available.	Sensitive to oracle inaccuracy and may require many oracle calls.

For the above methods, a useful sufficient condition for global optimality is that the distribution of ξ is log-concave. By Prékopa's Theorem [24], this implies that φ is log-concave and hence that $\{x : \varphi(x) \geq p\}$ is convex. Since X is convex and the objective is linear, problem (P) is then a convex optimization problem. Consequently, feasible first-order stationary points satisfying the appropriate constraint qualification are globally optimal. For the penalty method, this additionally requires a sufficiently large penalty parameter and recovery of feasibility. The level-bundle method exploits the stronger convex representation $\log p - \log \varphi(x) \leq 0$ directly.

Having summarized the roles and applicability of the five solution methods, we now develop the structured local model shared by the three structured methods.

3.2 Common approximation model for the structured methods

The Proximal, Feasible, and Penalty methods construct local models of the chance constraint using the marginal distributions of the transformed random vector $\eta := W\xi$ and its copula. Let F_i denote the distribution function of η_i, for $i = 1, \ldots, d$, and define $F(y(x)) := \big(F_1(y_1(x)), \ldots, F_d(y_d(x))\big)$. Since the marginal distributions are continuous, Sklar's theorem yields a unique copula C such that $\varphi(x) = C\big(F(y(x))\big)$.

The three methods maintain a *stability center*, defined as the current reference point at which the local model is constructed. The initial stability center is the starting point x^0, and it is replaced by a trial point only after a serious step. We denote the stability center after ν serious steps by $\hat{x}^\nu$.

We next show how the probability gradient and the marginal distributions are combined to construct the local model.

3.3 Local model construction

Assume that $\nabla\varphi(\hat{x}^\nu)$ is available. In the implementation, this gradient is supplied by the probability oracle described in Section 2. Under Assumption 1, the underlying (and potentially unknown) copula C is differentiable on a neighbourhood of $F(y(X))$, and the exact copula-gradient vector at $\hat{x}^\nu$ is $v^\nu := \nabla C\big(F(y(\hat{x}^\nu))\big) \in [0,1]^d$. The copula itself need not be known or evaluated explicitly. Indeed, the chain rule gives

$$\sum_{j=1}^{d} v_j^\nu f_j\big(y_j(\hat{x}^\nu)\big)\nabla y_j(\hat{x}^\nu) = \nabla\varphi(\hat{x}^\nu), \tag{1}$$

where $f_j := F_j'$ is the density of the j-th marginal distribution. Following the marginal-assessment approach of [36], the vector $v^\nu \in [0,1]^d$ is obtained by solving this linear system. The exact copula gradient $v^\nu = \nabla C\big(F(y(\hat{x}^\nu))\big)$ is therefore one of its solutions; see also [8, Theorem 1.6.1]. Thus, the coefficients required to construct the local model can be recovered from the probability gradient and the marginal densities without explicit knowledge of C.

The common local model of the chance-constraint function $p - \varphi(x)$ is

$$M_\nu(x) = p - \varphi(\hat{x}^\nu) - \sum_{j=1}^{d} v_j^\nu \left[F_j\big(y_j(x)\big) - F_j\big(y_j(\hat{x}^\nu)\big)\right]. \tag{2}$$

This model linearizes the dependence structure represented by the copula while retaining the marginal transformations $F_j \circ y_j$. It is constructed using the probability oracle described in Section 2. Since the transformed random vector η is also elliptically distributed, the same spherical–radial framework can be used to evaluate $\varphi(\hat{x}^\nu)$, its gradient, and the marginal distribution functions and densities required to determine v^ν and construct M_ν.

Each of the three structured methods computes a trial point by solving a regularized master problem based on M_ν. A rejected trial leaves the stability center and its first-order model unchanged, while the regularization is strengthened. Each trial requires a call to the probability oracle, which jointly evaluates φ and $\nabla\varphi$. If the trial is rejected, the returned gradient is not used to update the model; the marginal quantities required to construct a new M_ν are computed only after a serious step is accepted.

We now describe the three structured methods and how each incorporates this model.

3.4 Structured methods

3.4.1 Proximal method

The proximal method introduced in [37] serves as the toolbox's core general-purpose solver, seeking a trial point that decreases either the objective or the infeasibility. A quadratic proximal

term keeps the trial point in a region where the local probability model is informative. The exact joint probability at the trial point is then used to decide whether the predicted improvement is credible. An accepted trial becomes the new stability center; a rejected trial leaves the model unchanged and increases the proximal parameter.

At the current stability center, define $\tau(\hat{x}^\nu) := c^\top \hat{x}^\nu + \rho[p - \varphi(\hat{x}^\nu)]_+$, where $\rho \geq 0$, and introduce the improvement function

$$H(x; \hat{x}^\nu) := \max\left\{c^\top x - \tau(\hat{x}^\nu),\ p - \varphi(x)\right\}.$$

Its model is

$$H_\nu^M(x; \hat{x}^\nu) := \max\left\{c^\top x - \tau(\hat{x}^\nu),\ M_\nu(x)\right\}. \tag{3}$$

The trial point x^{k+1} is obtained by solving, with $\mu_k > 0$ a prox-parameter,

$$\min_{x \in X}\left\{H_\nu^M(x; \hat{x}^\nu) + \frac{\mu_k}{2}\|x - \hat{x}^\nu\|^2\right\}. \tag{MP-Prox}$$

The trial is accepted when $H(x^{k+1}; \hat{x}^\nu) \leq H(\hat{x}^\nu; \hat{x}^\nu) - \frac{\gamma}{2}\|x^{k+1} - \hat{x}^\nu\|^2$, where $\gamma \in (0, 1)$. Otherwise, the proximal parameter is increased and a new master problem is solved at the same stability center. The method is detailed in Algorithm 1.

Algorithm 1 Proximal method

Input: $x^0 \in X$, $\mu_0 > 0$, serious-step reset bound $\mu_{\max} > 0$, $\gamma \in (0, 1)$, $\beta > 1$, $\rho \geq 0$, and $\texttt{Tol} \geq 0$
Output: candidate solution $\hat{x}^\nu$, objective value $c^\top \hat{x}^\nu$, and probability value $\varphi(\hat{x}^\nu)$
Initialize $\hat{x}^0 = x^0$ and $\nu = 0$
for $k = 0, 1, \ldots$ **do**
    Construct M_ν and $H_\nu^M(\cdot; \hat{x}^\nu)$.
    Compute a stationary point x^{k+1} of (MP-Prox) satisfying

$$H_\nu^M(x^{k+1}; \hat{x}^\nu) + \frac{\mu_k}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \leq H_\nu^M(\hat{x}^\nu; \hat{x}^\nu). \tag{4}$$

    if $\|x^{k+1} - \hat{x}^\nu\| \leq \texttt{Tol}$ **then**
        return $\hat{x}^\nu$, $c^\top \hat{x}^\nu$, and $\varphi(\hat{x}^\nu)$
    end if
    Evaluate $\varphi(x^{k+1})$ for the acceptance test
    if $H(x^{k+1}; \hat{x}^\nu) \leq H(\hat{x}^\nu; \hat{x}^\nu) - \frac{\gamma}{2}\|x^{k+1} - \hat{x}^\nu\|^2$ **then** ▷ Serious step
        $\hat{x}^{\nu+1} \leftarrow x^{k+1}$ and $\nu \leftarrow \nu + 1$
        Choose $\mu_{k+1} \in (0, \mu_{\max}]$
    else ▷ Null step
        Keep $\hat{x}^\nu$ and M_ν unchanged and set $\mu_{k+1} \leftarrow \beta\mu_k$
    end if
end for

It is worth noting that condition (4) is not restrictive. Indeed, since $\hat{x}^\nu \in X$ is available, we can initialize the solver to handle (MP-Prox) with $\hat{x}^\nu$ when computing the next iterate. Any mathematically sound solver implementing a descent algorithm will return a stationary point that is at least as good as the initial point.

Remark 5 (Practical implementation). The master problem admits the epigraphical formulation

$$\begin{aligned}
\min_{x,s}\quad & s + \frac{\mu_k}{2}\|x - \hat{x}^\nu\|^2 \\
\text{s.t.}\quad & c^\top x - \tau(\hat{x}^\nu) \leq s, \\
& p - \varphi(\hat{x}^\nu) - \sum_{j=1}^{d} v_j^\nu \left[F_j(y_j(x)) - F_j(y_j(\hat{x}^\nu))\right] \leq s, \\
& x \in X, \qquad s \in \mathbb{R}.
\end{aligned}$$

This is a $C^{1,1}$ nonlinear program on the relevant region. The coefficients v^ν defining M_ν in (2) are obtained by solving the linear system (1). The exact criticality condition corresponds to $\texttt{Tol} = 0$, while $\texttt{Tol} > 0$ provides a practical approximate stopping criterion.

We first recall the criticality and KKT conditions used in the convergence statements below.

Definition 1 (Criticality)**.** A point $\bar{x} \in X$ is critical for problem (P) if there exists $\theta \in [0,1]$ such that

$$0 \in \theta c - (1-\theta)\nabla\varphi(\bar{x}) + N_X(\bar{x}), \quad \text{where} \quad \theta \in \begin{cases} \{1\}, & \varphi(\bar{x}) > p, \\ [0,1], & \varphi(\bar{x}) = p, \\ \{0\}, & \varphi(\bar{x}) < p. \end{cases}$$

Definition 2 (KKT conditions)**.** A point $\bar{x} \in X$ satisfies the KKT conditions for problem (P) if there exists $\lambda \geq 0$ such that

$$0 \in c - \lambda\nabla\varphi(\bar{x}) + N_X(\bar{x}), \quad \varphi(\bar{x}) \geq p, \quad \lambda\big(p - \varphi(\bar{x})\big) = 0.$$

The following lemma summarizes the optimality conditions implied by criticality.

Lemma 1. *Let $\bar{x} \in X$ be critical for problem* (P)*. Then:*

i) if $\varphi(\bar{x}) > p$, then $\bar{x}$ satisfies the KKT conditions with multiplier $\lambda = 0$;

ii) if $\varphi(\bar{x}) = p$ and $0 \notin -\nabla\varphi(\bar{x}) + N_X(\bar{x})$, then $\bar{x}$ satisfies the KKT conditions with $\lambda = (1-\theta)/\theta$;

iii) if $\varphi(\bar{x}) < p$, then $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$; we call such a point feasibility-stationary*.*

Proof. By criticality, $0 \in \theta c - (1-\theta)\nabla\varphi(\bar{x}) + N_X(\bar{x})$ for some admissible $\theta \in [0,1]$. If $\varphi(\bar{x}) > p$, then $\theta = 1$, giving the KKT conditions with $\lambda = 0$. If $\varphi(\bar{x}) = p$, the stated condition implies $\theta > 0$; dividing by θ and setting $\lambda = (1-\theta)/\theta \geq 0$ gives the KKT conditions. Finally, if $\varphi(\bar{x}) < p$, then $\theta = 0$, and hence $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$, which is the feasibility-stationary condition. □

The following convergence result is a specialization of a more general result. Assumption 1 is stronger than the assumptions required in that general setting and therefore constitutes a special case thereof.

Theorem 2. *(Convergence of the proximal method, [37, Theorem 8]) Consider Algorithm 1 with $\texttt{Tol} = 0$, applied to problem* (P)*. Suppose that Assumption 1 holds and $\mu_k > 0$ for every $k \in \mathbb{N}$. Then:*

i) The sequence of iterates $(x^k)_{k\in\mathbb{N}}$ is well defined.

ii) If $\varphi(\hat{x}^\iota) \geq p$ for some $\iota \in \mathbb{N}$, then $\varphi(\hat{x}^\nu) \geq p$ for every $\nu \geq \iota$.

iii) The algorithm either terminates at a critical point of problem (P)*, or every cluster point of the stability centers is critical.*

iv) Let $\hat{x}$ be either the final stability center upon termination or a cluster point of the stability centers. If $\varphi(\hat{x}) > p$, or if $\varphi(\hat{x}) = p$ and $0 \notin -\nabla\varphi(\hat{x}) + N_X(\hat{x})$, then $\hat{x}$ satisfies the KKT conditions for problem (P)*. If $\varphi(\hat{x}) < p$, then $0 \in -\nabla\varphi(\hat{x}) + N_X(\hat{x})$, i.e., $\hat{x}$ is feasibility-stationary.*

We next present the Feasible method proposed in this paper.

3.4.2 Feasible method

The feasible method starts from a feasible stability center and uses the local model to propose an objective-improving trial point. The true joint probability is then evaluated: only a feasible trial is accepted. If the trial is infeasible, the local model is retained and its quadratic correction is strengthened, forcing the next candidate closer to the current feasible center.

At the current stability center, a trial point is obtained from

$$\begin{aligned} \min_{x \in X} \quad & c^\top x + \frac{\delta}{2}\|x - \hat{x}^\nu\|^2, \\ \text{s.t.} \quad & M_\nu(x) + \frac{\ell_k}{2}\|x - \hat{x}^\nu\|^2 \le 0, \end{aligned} \tag{MP-Feas}$$

where $\delta > 0$ regularizes the objective and $\ell_k > 0$ controls the quadratic correction of the model constraint.

Algorithm 2 Feasible method

Input: $x^0 \in X$ with $\varphi(x^0) \ge p$, $\delta > 0$, $\ell_0 > 0$, serious-step reset bound $\ell_{\max} > 0$, $\beta > 1$, and $\texttt{Tol} \ge 0$
Output: candidate solution $\hat{x}^\nu$, objective value $c^\top \hat{x}^\nu$, and probability value $\varphi(\hat{x}^\nu)$
Initialize $\hat{x}^0 = x^0$ and $\nu = 0$
for $k = 0, 1, \ldots$ **do**
    Construct M_ν as in Algorithm 1.
    Compute a stationary point x^{k+1} of (MP-Feas) satisfying

$$c^\top x^{k+1} + \frac{\delta}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \le c^\top \hat{x}^\nu. \tag{5}$$

    if $\|x^{k+1} - \hat{x}^\nu\| \le \texttt{Tol}$ **then**
        return $\hat{x}^\nu$, $c^\top \hat{x}^\nu$, and $\varphi(\hat{x}^\nu)$
    end if
    Evaluate $\varphi(x^{k+1})$
    if $\varphi(x^{k+1}) \ge p$ **then** ▷ Serious step
        $\hat{x}^{\nu+1} \leftarrow x^{k+1}$ and $\nu \leftarrow \nu + 1$
        Choose $\ell_{k+1} \in (0, \ell_{\max}]$
    else ▷ Null step
        Keep $\hat{x}^\nu$ and M_ν unchanged and set $\ell_{k+1} \leftarrow \beta \ell_k$
    end if
end for

Remark 6 (Practical implementation). The current stability center satisfies the model constraint because $M_\nu(\hat{x}^\nu) = p - \varphi(\hat{x}^\nu) \le 0$, and therefore provides a feasible initialization for the master problem. Increasing ℓ_k strengthens the local feasibility correction and forces subsequent trial points toward the current stability center.

The method is particularly effective when a good feasible initial solution is available, or when feasibility is relatively easy to maintain. The price is the need for a feasible initial point and a potentially more conservative search, whereas the proximal method can explore infeasible points more freely and is generally preferable when finding or preserving feasibility is difficult. A feasible initial point may be obtained from a Bonferroni approximation or a feasibility-restoration procedure; see Section 4.1.

The proof of the following theorem can be found in Appendix C.

Theorem 3 (Convergence of the feasible method)**.** *Consider Algorithm 2 with* $\texttt{Tol} = 0$*, applied to problem* (P)*. Suppose that Assumption 1 holds,* $\delta > 0$*, and* $x^0 \in X$ *satisfies* $\varphi(x^0) \ge p$*. Then:*

i) The sequence of iterates $(x^k)_{k \in \mathbb{N}}$ *is well defined.*
ii) Every stability center $\hat{x}^\nu$ *satisfies* $\varphi(\hat{x}^\nu) \ge p$*.*
iii) The algorithm either terminates at a critical point of problem (P)*, or every cluster point of the stability centers is critical.*
iv) Let $\hat{x}$ *be either the final stability center upon termination or a cluster point of the stability centers. If* $\varphi(\hat{x}) > p$*, or if* $\varphi(\hat{x}) = p$ *and* $0 \notin -\nabla\varphi(\hat{x}) + N_X(\hat{x})$*, then* $\hat{x}$ *satisfies the KKT conditions for problem* (P)*.*

We next present the Penalty method proposed in this paper.

3.4.3 Penalty method

The penalty method replaces the chance constraint by a penalty for reliability violations. As the proximal method of Algorithm 1, it can therefore start from an infeasible point and combines objective reduction and feasibility in a single scalar merit function. Unlike the proximal method, where feasibility is embedded in the nonlinear improvement function, the penalty method transfers the chance constraint directly into the objective through the parameter r. This typically leads to a simpler subproblem and can be very effective when r is chosen well: the main difficulty. If r is too small, the method may converge to an infeasible point; if it is too large, the penalized problem may become poorly scaled and place excessive emphasis on feasibility. Consequently, the method is attractive when a suitable penalty level can be identified, but this tuning can be delicate. The proximal method avoids this explicit parameter choice, at the cost of a more complicated improvement-function model. Unlike the feasible method, a run with fixed r does not guarantee a feasible final solution.

For a fixed penalty parameter $r > 0$, define $\psi_r(x) := c^\top x + r[p - \varphi(x)]_+$, and its local model $\psi^M_{r,\nu}(x) := c^\top x + r[M_\nu(x)]_+$. The trial point is obtained from

$$\min_{x \in X} \left\{ \psi^M_{r,\nu}(x) + \frac{r\mu_k}{2} \|x - \hat{x}^\nu\|^2 \right\}. \tag{MP-Pen}$$

It is accepted as a stability center when $\psi_r(x^{k+1}) \le \psi_r(\hat{x}^\nu) - \frac{\gamma}{2}\|x^{k+1} - \hat{x}^\nu\|^2$.

Algorithm 3 Penalty method

Input: $x^0 \in X$, $\mu_0 > 0$, serious-step reset bound $\mu_{\max} > 0$, $r > 0$, $\gamma \in (0,1)$, $\beta > 1$, and $\texttt{Tol} \ge 0$
Output: candidate solution $\hat{x}^\nu$, objective value $c^\top \hat{x}^\nu$, and probability value $\varphi(\hat{x}^\nu)$
Initialize $\hat{x}^0 = x^0$ and $\nu = 0$
for $k = 0, 1, \ldots$ **do**
    Construct M_ν as in Algorithm 1 and $\psi^M_{r,\nu} := c^\top x + r[M_\nu(x)]_+$.
    Compute a stationary point x^{k+1} of (MP-Pen) satisfying

$$\psi^M_{r,\nu}(x^{k+1}) + \frac{r\mu_k}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \le \psi^M_{r,\nu}(\hat{x}^\nu). \tag{6}$$

    if $\|x^{k+1} - \hat{x}^\nu\| \le \texttt{Tol}$ **then**
        return $\hat{x}^\nu$, $c^\top \hat{x}^\nu$, and $\varphi(\hat{x}^\nu)$
    end if
    Evaluate $\varphi(x^{k+1})$ and hence $\psi_r(x^{k+1})$
    if $\psi_r(x^{k+1}) \le \psi_r(\hat{x}^\nu) - \frac{\gamma}{2}\|x^{k+1} - \hat{x}^\nu\|^2$ **then** ▷ Serious step
        $\hat{x}^{\nu+1} \leftarrow x^{k+1}$ and $\nu \leftarrow \nu + 1$
        Choose $\mu_{k+1} \in (0, \mu_{\max}]$
    else ▷ Null step
        Keep $\hat{x}^\nu$ and M_ν unchanged and set $\mu_{k+1} \leftarrow \beta\mu_k$
    end if
end for

Remark 7 (Practical implementation). The convergence analysis treats the penalty parameter r as fixed. Exact-penalty theory nevertheless provides some guidance for its choice. In particular, let $g(x) := p - \varphi(x)$ and $\mathcal{F} := \{x \in X : g(x) \le 0\}$. If the error bound $\operatorname{dist}(x, \mathcal{F}) \le \kappa[g(x)]_+$ holds, then the exact-penalty argument in [3, Proposition 9.1.1] implies that the penalized and constrained problems have the same global minimizers for any $r > \kappa\,\|c\|$. In practice, however, the error-bound modulus κ is generally unknown and may be difficult to estimate. We therefore initialize the penalty parameter as $r_0 = \|c\|$ and, if the returned point does not satisfy the prescribed feasibility tolerance, increase r and restart the method. Increasing r places greater emphasis on feasibility, but without an applicable exact-penalty condition it does not rule out convergence to an infeasible stationary point. Consequently, the final probability margin is always checked explicitly.

For the Penalty method, we use the following criticality notion associated with the penalized problem.

Definition 3 (Penalty criticality)**.** A point $\bar{x} \in X$ is critical for the penalized problem $\min_{x\in X} \psi_r(x)$ if

$$0 \in \alpha c - (1-\alpha)\nabla\varphi(\bar{x}) + N_X(\bar{x}), \quad \text{for some } \alpha \in \begin{cases} \{1\}, & \varphi(\bar{x}) > p, \\ \left[\dfrac{1}{1+r}, 1\right], & \varphi(\bar{x}) = p, \\ \left\{\dfrac{1}{1+r}\right\}, & \varphi(\bar{x}) < p. \end{cases} \tag{7}$$

This is the stationarity condition for ψ_r, normalized by $1 + r\theta$, where

$$\theta \in \begin{cases} \{0\}, & \varphi(\bar{x}) > p, \\ [0,1], & \varphi(\bar{x}) = p, \\ \{1\}, & \varphi(\bar{x}) < p. \end{cases}$$

If a penalty-critical point $\bar{x}$ satisfies $\varphi(\bar{x}) \geq p$, then it satisfies the KKT conditions for problem (P). Indeed, if $\varphi(\bar{x}) > p$, then $\alpha = 1$ and the KKT multiplier is zero. If $\varphi(\bar{x}) = p$, dividing (7) by $\alpha > 0$ gives $0 \in c - \lambda\nabla\varphi(\bar{x}) + N_X(\bar{x})$, $\lambda := \frac{1-\alpha}{\alpha} \in [0, r]$.

The proof of the following theorem and proposition are given in the Appendix D.

Theorem 4 (Convergence of the penalty method)**.** *Consider Algorithm 3 with* `Tol` $= 0$*, applied to problem* (P)*. Suppose that Assumption 1 holds and* $\mu_k > 0$ *for every* $k \in \mathbb{N}$*. Then:*

i) The sequence of iterates generated by Algorithm 3 is well defined.

ii) For every fixed $r > 0$*, the algorithm either terminates at a penalty-critical point, or every cluster point of the stability centers is penalty-critical.*

Proposition 5 (Increasing the penalty parameter)**.** *Under the assumptions of Theorem 4, let* $r_j \to +\infty$*, and for each* j*, let* $\bar{x}_j$ *be a penalty-critical point associated with* r_j*. Then every accumulation point* $\bar{x}$ *of* $(\bar{x}_j)$ *satisfies:*

i) if $\varphi(\bar{x}) > p$*, then* $\bar{x}$ *satisfies the KKT conditions with multiplier* $\lambda = 0$*;*

ii) if $\varphi(\bar{x}) = p$ *and* $0 \notin -\nabla\varphi(\bar{x}) + N_X(\bar{x})$*, then* $\bar{x}$ *satisfies the KKT conditions for problem* (P)*;*

iii) if $\varphi(\bar{x}) < p$*, then* $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$*, i.e.,* $\bar{x}$ *is feasibility-stationary.*

The preceding three methods exploit the common structured model M_ν. For comparison, the toolbox also includes two established baseline approaches that interact with the probability oracle without using this model.

3.5 Additional baseline methods

3.5.1 Level-bundle method

The level-bundle method is appropriate when the chance constraint admits a convex reformulation. If φ is log-concave, we define $L(x) := \log(p) - \log(\varphi(x))$, with oracle $(L(x), \nabla L(x)) = (\log(p) - \log(\varphi(x)), -\nabla\varphi(x)/\varphi(x))$ provided $\varphi(x) > 0$. Points in X with zero probability are excluded, for instance, when individual chance-constraints are incorporated into X (see remark R1 in the numerical section).

At iteration k, let B_k contain the previously evaluated points and let f_k^{low} be the lower bound obtained by solving the current cutting-plane model. Following the level-bundle mechanism of [38, Section 13.3], a target level is defined by $f_k^{\text{lev}} = f_k^{\text{low}} + \kappa \min_{j\leq k} \max\left\{c^\top x^j - f_k^{\text{low}},\, L(x^j)\right\}, \kappa \in (0,1)$. Thus, f_k^{lev} lies above the current lower bound and sets the objective level that the next trial point is required to attain. The parameter κ controls how aggressively this level is chosen.

Given a stability center $\hat{x}^k$, the next iterate is obtained from

$$\begin{aligned} x^{k+1} = \arg\min_{x \in X} \quad & \frac{1}{2}\|x - \hat{x}^k\|^2 \\ \text{s.t.} \quad & c^\top x \le f_k^{\text{lev}}, \\ & L(x^j) + \nabla L(x^j)^\top (x - x^j) \le 0, \qquad j \in B_k. \end{aligned}$$

Because L is convex, the bundle inequalities are valid supporting hyperplanes. Since X is polyhedral, the master problem is a strictly convex QP.

3.5.2 IPOPT

IPOPT treats problem (P) directly as the nonlinear program

$$\min_{x \in X} \quad c^\top x \quad \text{s.t.} \quad p - \varphi(x) \le 0.$$

Under Assumption 1, the exact constraint value and gradient are $p - \varphi(x)$ and $-\nabla\varphi(x)$; in the implementation, their numerical estimates are supplied to IPOPT [39]. This requires little method-specific development and provides a useful general-purpose baseline.

Unlike the three structured methods, however, IPOPT does not build and exploit a problem-specific local model of the probability function, and it may therefore require many nonlinear-programming iterations and oracle evaluations.

We now assess computational performances for above five methods when the probability and gradient information are supplied by the finite-sample spherical–radial oracle described in Section 2.

4 Numerical illustrations

4.1 Experimental setup

We consider several joint chance-constrained programming (JCCP) instances drawn from multiple sources. The selected problems, cash-matching [14], transportation [19], hydro reservoir [30], cover a range of structural and numerical characteristics (decision dimension, randomness dimension, and conditioning), providing a broad basis for evaluating the structured algorithms against established baselines.

All benchmark instances are expressed in the canonical JCCP form (P) with ξ following an elliptically symmetric distribution with mean μ and dispersion matrix Σ. In our implementation, the parameters μ and Σ are chosen according to specifications suggested in the literature whenever available (see [14, 19, 30] for the corresponding problem settings), in order to maintain consistency with previously reported numerical studies.

All benchmark experiments for solving problems of the form (P) are conducted using our *Probabilistic Linear Minimization (ProbLiM)* toolbox. The toolbox is implemented in the Julia programming language and integrated with the JuMP modeling environment, and is available as open-source software at `https://gitlab.com/stochasticprogramming/ProbLiM`.

For each benchmark instance, we compare: (i) our Proximal method (see Algorithm 1); (ii) our Feasible method (see Algorithm 2); (iii) our Penalty method (see Algorithm 3); (iv) level-bundle method; and (v) IPOPT.

Throughout the numerical section, all evaluations of the probability function and its gradient are performed numerically by the spherical-radial decomposition (SRD) oracle. More precisely, the algorithms use the estimates $\widehat{\varphi}_N(x)$ and $\widehat{\nabla\varphi}_N(x)$ introduced in Section 2, rather than the corresponding exact quantities. For readability, we continue to write $\varphi(x)$ and $\nabla\varphi(x)$ when describing the numerical procedures. The sampling budget and oracle tolerances are chosen so

that the resulting estimation errors are small relative to the feasibility and stopping tolerances used by the optimization methods. Consequently, these errors do not materially affect the reported objective values, feasibility conclusions, or comparisons among the methods. Sampling over the unit sphere in the SRD estimator is performed using antithetic randomized quasi–Monte Carlo (QMCA) sequences.

For demonstration purpose, we consider two families of distributions, namely, Gaussian and Student-t. For the Gaussian instances, we consider $p \in \{0.8, 0.85, 0.9, 0.95, 0.99\}$, whereas for the Student-t instances we omit $p = 0.99$, since no feasible point could be identified for some instances at this reliability level. For each instance, we report:

1. CPU time in seconds;
2. the number of probability-oracle evaluations.

All experiments were conducted on a Dell Precision 3571 workstation equipped with a 12th-generation Intel Core i9-12900H processor at 2.50 GHz and 32 GB of RAM, running a 64-bit operating system on an x64 architecture.

We also report data and performance profiles (see [7] and [20]) for the number of oracle calls $\#F$ and for CPU time (in seconds) across all benchmark instances for Gaussian distributions.

In the numerical experiments, the SRD oracle is run with $N = 32768 = 2^{15}$ for the cash-matching and transportation instances and $N = 131072 = 2^{17}$ for the hydro-valley instances. These powers-of-two sample sizes are chosen to better exploit the structure of QMCA while maintaining reasonable sampling errors for both φ and $\nabla\varphi$. Unless stated otherwise, whenever convergence is achieved, all methods return nearly identical objective values and satisfy the chance constraint within numerical tolerance.

Five remarks concerning the benchmark design and implementation are useful.

R.1 **Individual chance constraints.** For all five solution methods, we add the necessary marginal conditions $\mathbb{P}(\eta_i \le y_i(x)) \ge p,\ i = 1, \dots, d$, or equivalently,

$$y_i(x) \ge w_i^\top \mu + \kappa_\xi(p)\sqrt{w_i^\top \Sigma w_i},$$

where $\kappa_\xi(p)$ denotes the p-quantile of the standardized one-dimensional marginal distribution associated with the chosen elliptical family. Indeed, joint feasibility implies each marginal condition. Since these constraints are linear in x, they can be included in the initialization and algorithmic subproblems at negligible cost.

R.2 **Applicability of the level-bundle method.** The level-bundle method is applied to problem (P) with probability constraint replaced with

$$L(x) := \log(p) - \log(\varphi(x)) \le 0,\ \varphi(x) = \mathbb{P}(W\xi \le Hx + h)\,.$$

This formulation is convex when φ is log-concave, as in the Gaussian case. Since this property is not guaranteed for non-log-concave distributions, such as the Student-t distribution, level-bundle method is excluded from those experiments.

R.3 **Bonferroni initialization.** For all five solution methods, given $p_{\text{search}} \in [p, 1)$, we solve the conservative approximation

$$\mathbb{P}(\eta_i \le y_i(x)) \ge 1 - \frac{1 - p_{\text{search}}}{d},\ i = 1, \dots, d.$$

The value of p_{search} is adjusted by a coarse search and, when possible, bisection until a point x_0 satisfying $p \le \varphi(x_0) \le p + 10^{-3}$ is obtained. If this target interval is not reached within the prescribed budget, the closest feasible candidate is used.

R.4 **Feasibility restoration.** When the initial point is infeasible, the Feasible method is pre-

ceded by a restoration phase. Given x^k, we compute

$$x^{k+1} \in \operatorname*{argmin}_{x \in X} \left\{ c^\top x : \varphi(x^k) + \sum_{j=1}^{d} \big(F_j(y_j) - F_j(y_j^k)\big) \geq p + \varepsilon \right\},$$

where $\varepsilon > 0$ is a safeguard parameter. The original probability function is then evaluated at x^{k+1}, and the procedure is repeated until a feasible point is found or a stopping criterion is reached.

R.5 **Benchmark metrics.** The Proximal, Feasible, and Penalty methods solve nonlinear master problems with IPOPT at each iteration, while the IPOPT baseline directly solves the nonlinear chance-constrained problem. Their numerical performance can therefore be sensitive to the scaling of the problem data, with the effect varying across reliability levels p and instances. Accordingly, the Isère and Ain instances are scaled using instance-dependent factors for these four methods. The level-bundle method, which solves convex QP subproblems, does not require this scaling. We also observed that different versions of IPOPT and Julia may lead to different numerical behavior. Therefore, to reproduce the numerical results reported in this section, the same versions of IPOPT and Julia specified on the GitLab page of the toolbox should be used.

4.2 Cash-matching problem

The first test instance is the cash-matching problem; see [14]. A pension fund invests an initial capital in n types of bonds to meet payment obligations over a planning horizon of m years. The objective is to maximize the terminal cash position while ensuring that all cumulative liabilities are covered with a prescribed probability.

Let α_{ik} denote the cash yield generated in year k by one unit of bond i, γ_i its purchase cost, β_k the payment obligation in year k, and K the initial capital. The decision variable $x_i \geq 0$ denotes the amount of bond i purchased. For $j = 1, \ldots, m$, define $a_{ij} := \sum_{k=1}^{j} \alpha_{ik} - \gamma_i$, and $b_j := \sum_{k=1}^{j} \beta_k - K$. Thus, $\sum_{i=1}^{n} a_{ij} x_i$ is the cumulative cash position up to year j, net of the initial investment.

Uncertainty is represented by the vector $\xi = (\xi_1, \ldots, \xi_m)^\top$ of cumulative liability levels, with mean $\mu = (b_1, \ldots, b_m)^\top$ and dispersion matrix $\Sigma \in \mathbb{R}^{m \times m}$. The resulting problem is

$$\begin{aligned} \max_{x \in \mathbb{R}^n_+} \quad & \sum_{i=1}^{n} a_{im} x_i \\ \text{s.t.} \quad & \mathbb{P}\left(\sum_{i=1}^{n} a_{ij} x_i \geq \xi_j, \quad j = 1, \ldots, m \right) \geq p. \end{aligned} \tag{8}$$

The benchmark contains $n = 3$ bond types and $m = 15$ planning periods, so that $x \in \mathbb{R}^3$ and $\xi \in \mathbb{R}^{15}$. We consider Gaussian and Student-t distributions, with $\nu = 5$ degrees of freedom in the latter case. Results with and without Bonferroni initialization are reported in Tables 4–7.

Penalty is the fastest method in most cash-matching instances, while Feasible and Proximal remain competitive. The level-bundle method is also effective on the Gaussian instances, particularly at $p = 0.99$. At this highest reliability level, some structured-method runs fall outside the relative objective tolerance of 5×10^{-4}. These stars do not indicate poor algorithmic behavior, but rather that the default stopping tolerances are not sufficiently tight; reducing the stopping tolerance would improve the objective accuracy. Bonferroni initialization does not provide a uniform computational benefit in this benchmark. Overall, IPOPT generally requires more computational effort, while the structured methods provide a favorable balance between efficiency and solution quality.

Table 4: Cash-matching benchmark under the Gaussian distribution, without Bonferroni initialization. Each entry reports CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	5.70 (12)	6.23 (12)	6.07 (13)	7.69 (16)	21.74 (49)*
Feasible	10.38 (20)	10.30 (21)	9.25 (20)	10.83 (23)	10.63 (24)
Penalty	4.78 (9)	3.76 (8)	3.29 (7)	6.88 (14)	10.37 (23)*
Level-bundle	18.30 (36)	9.63 (20)	14.22 (30)	12.80 (28)	6.99 (16)
IPOPT	19.07 (21)	18.89 (22)	18.87 (22)	12.57 (16)	12.40 (16)

Table 5: Cash-matching benchmark under the Gaussian distribution, with Bonferroni initialization. Each entry reports CPU time in seconds (total oracle calls). Constructing the Bonferroni initial point requires an average of 8.1 oracle calls for all methods. Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	8.40 (19)	8.13 (19)	8.57 (20)	10.43 (23)	22.35 (50)*
Feasible	9.29 (21)	10.93 (25)	11.29 (26)	13.71 (32)	9.98 (23)*
Penalty	8.09 (19)	7.33 (17)	9.05 (21)	8.46 (19)	10.76 (24)*
Level-bundle	19.29 (44)	19.79 (46)	14.29 (32)	10.81 (25)	6.87 (16)
IPOPT	27.91 (39)	19.66 (29)	18.23 (27)	18.44 (27)	12.75 (17)

Table 6: Cash-matching benchmark under the Student-t distribution, without Bonferroni initialization. Each entry reports CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	7.02 (14)	5.97 (14)	7.06 (17)	13.46 (32)
Feasible	7.43 (18)	9.31 (23)	9.78 (23)	10.16 (24)
Penalty	3.86 (9)	5.62 (13)	6.68 (16)	10.14 (24)
IPOPT	15.54 (21)	14.35 (19)	12.56 (17)	11.73 (16)

Table 7: Cash-matching benchmark under the Student-t distribution, with Bonferroni initialization. Each entry reports CPU time in seconds (total oracle calls). Constructing the Bonferroni initial point requires an average of 9.7 oracle calls for all methods. Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	8.00 (21)	8.21 (21)	9.60 (23)	15.65 (38)
Feasible	10.23 (26)	10.35 (26)	11.32 (26)	12.25 (29)
Penalty	7.87 (20)	10.08 (26)	11.54 (25)	12.80 (31)
IPOPT	20.64 (33)	17.45 (28)	15.57 (25)	12.04 (21)

4.3 Transportation problem

The second test instance is a probabilistic transportation problem adapted from [19]. We consider a classical transportation network with a set of suppliers and a set of customers. Shipment decisions are made before customer demands are observed. The objective is to minimize the total transportation cost while ensuring that all customer demands are satisfied simultaneously with a prescribed probability.

Let I denote the set of suppliers and D the set of customers, with $|I| = n$ and $|D| = m$. For each supplier–customer pair $(i, j) \in I \times D$, let $x_{ij} \geq 0$ denote the amount shipped from supplier i to customer j, c_{ij} the corresponding unit transportation cost, and M_i the capacity of supplier i. Let ξ_j denote the random demand of customer j. The resulting joint chance-constrained transportation problem reads as

$$\begin{aligned} \min_{x \geq 0} \quad & \sum_{i \in I} \sum_{j \in D} c_{ij} x_{ij} \\ \text{s.t.} \quad & \mathbb{P}\left(\sum_{i \in I} x_{ij} \geq \xi_j, \quad j \in D \right) \geq p, \\ & \sum_{j \in D} x_{ij} \leq M_i, \quad i \in I. \end{aligned} \tag{9}$$

Here, the decision variable belongs to $\mathbb{R}^{n \times m}$, while the random demand vector ξ belongs to $\mathbb{R}^m$.

In the benchmark instances, ξ has mean vector μ and diagonal dispersion matrix Σ, with parameters chosen following [19]. To investigate scalability with respect to both the decision dimension and the uncertainty dimension, we fix the number of suppliers at $n = 40$ and vary the number of customers as $m \in \{10, 20, 30, 40, 50\}$. The corresponding benchmark results are reported in Tables 8–11.

Across the transportation benchmarks, Feasible and Penalty are generally the fastest structured methods, while Proximal is typically somewhat slower but provides the most consistent solution quality. Bonferroni initialization substantially reduces the computational effort of the structured methods in many instances. At high reliability levels and larger problem sizes, several Feasible and Penalty runs, and only isolated Proximal runs, fall outside the prescribed relative objective tolerance. These stars indicate that the default stopping tolerances are not sufficiently tight and can be resolved by using a smaller stopping tolerance. Level-bundle is considerably more expensive but attains accurate solutions on the Gaussian instances, whereas IPOPT is generally much more computationally expensive and also exhibits an infeasible termination. Overall, Proximal provides the most consistent performance, while Feasible and Penalty offer the best computational efficiency.

4.4 Hydro reservoir management problem

The benchmark instances are hydro-valley reservoir management problems, namely the Isère Valley and Ain Valley instances, arising from realistic energy planning applications and widely studied in the context of joint chance-constrained programming; see, e.g., [30].

The model describes a cascaded multi-reservoir system operated over a finite planning horizon. At each time stage, release decisions must simultaneously satisfy electricity production requirements and maintain reservoir storage levels within admissible bounds. Uncertainty enters through stochastic inflows to the reservoirs, which accumulate over time and propagate through the cascading structure.

Let the decision vector x collects release decisions and other operational variables across all time stages. Let $\xi \in \mathbb{R}^{48}$ denote the vector of cumulative stochastic inflows over the horizon. In the benchmark experiments, ξ is modeled by a nondegenerate continuous elliptically symmetric distribution with location vector $\mu \in \mathbb{R}^{48}$ and positive definite dispersion matrix

Table 8: Transportation benchmark under the Gaussian distribution, without Bonferroni initialization. Each entry reports CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

p	Algorithm	$m = 10$	$m = 20$	$m = 30$	$m = 40$	$m = 50$
0.8	Proximal	12.90 (20)	34.19 (22)	127.10 (68)	92.75 (34)	93.22 (29)
	Feasible	7.72 (13)	18.50 (14)	36.72 (17)	56.88 (21)	70.48 (22)
	Penalty	10.36 (16)	34.03 (23)	18.50 (8)	29.80 (5)	69.91 (15)
	Level-bundle	30.29 (60)	92.78 (104)	223.58 (173)	421.29 (253)	926.64 (440)
	IPOPT	81.39 (90)	295.20 (167)	1018.94 (405)	1517.52 (487)	1629.22 (428)
0.85	Proximal	13.41 (21)	31.89 (25)	113.60 (60)	101.48 (40)	111.84 (30)
	Feasible	6.67 (12)	18.07 (15)	40.20 (16)	50.50 (17)	67.82 (18)
	Penalty	10.15 (15)	25.57 (16)	97.39 (36)	56.71 (19)	36.80 (5)
	Level-bundle	29.46 (67)	81.70 (104)	211.97 (174)	461.27 (283)	649.39 (301)
	IPOPT	109.94 (115)	319.73 (191)	943.49 (379)	1180.15 (383)	1807.74 (476)
0.9	Proximal	14.29 (23)	40.04 (29)	90.22 (43)	166.45 (61)	188.56 (56)
	Feasible	6.29 (11)	16.39 (12)	26.62 (14)	47.87 (16)	94.00 (19)
	Penalty	8.93 (12)	40.07 (29)	22.50 (8)	87.81 (17)	253.93 (57)
	Level-bundle	25.27 (61)	87.94 (115)	221.56 (183)	437.46 (263)	1152.84 (502)
	IPOPT	121.23 (138)	423.78 (256)	921.77 (396)	872.25 (285)	1802.28 (472)*
0.95	Proximal	18.09 (26)	73.08 (43)	79.17 (39)	161.71 (55)	221.45 (60)
	Feasible	7.16 (11)	19.86 (15)	28.87 (15)	58.74 (21)	102.38 (19)
	Penalty	7.28 (8)	64.52 (39)	104.56 (45)	115.96 (36)	181.96 (46)
	Level-bundle	24.50 (55)	115.51 (150)	257.98 (216)	848.48 (502)	1075.67 (502)
	IPOPT	108.67 (115)	415.07 (251)	1023.77 (406)	1172.41 (376)	1245.40 (329)
0.99	Proximal	15.42 (26)	34.25 (24)	161.75 (75)	243.35 (92)	426.30 (137)*
	Feasible	13.19 (27)	40.46 (41)*	46.24 (30)*	58.11 (26)*	147.46 (27)*
	Penalty	15.11 (24)	42.58 (34)*	56.97 (34)*	231.33 (116)*	287.48 (110)*
	Level-bundle	22.88 (59)	167.01 (225)	582.59 (502)	820.70 (502)	1011.95 (502)
	IPOPT	118.42 (126)	525.64 (320)	879.58 (374)	1346.15 (417)	1651.40 (436)

Table 9: Transportation benchmark under the Gaussian distribution, with Bonferroni initialization. Each entry reports CPU time in seconds (total oracle calls). Constructing the Bonferroni initial point requires an average of 8.3 oracle calls for all methods. Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

p	Algorithm	$m = 10$	$m = 20$	$m = 30$	$m = 40$	$m = 50$
0.8	Proximal	9.78 (20)	18.27 (19)	25.53 (18)	38.08 (20)	46.41 (19)
	Feasible	8.50 (18)	20.55 (20)	20.61 (15)	31.02 (16)	44.30 (17)
	Penalty	9.19 (17)	18.77 (18)	30.88 (16)	36.66 (16)	37.73 (16)
	Level-bundle	15.29 (40)	55.56 (75)	89.60 (80)	299.30 (195)	442.67 (220)
	IPOPT	47.38 (55)	254.62 (158)	944.91 (389)	644.70 (209)	540.03 (148)
0.85	Proximal	9.20 (20)	20.16 (21)	28.11 (20)	35.26 (19)	42.11 (16)
	Feasible	9.68 (18)	19.78 (20)	30.87 (17)	27.83 (15)	32.68 (13)
	Penalty	7.65 (15)	16.26 (16)	26.50 (16)	31.57 (16)	38.17 (14)
	Level-bundle	19.33 (50)	54.88 (74)	104.84 (86)	294.57 (188)	590.40 (287)
	IPOPT	42.83 (50)	224.59 (140)	1035.83 (435)	458.14 (163)	512.44 (149)
0.9	Proximal	10.19 (21)	26.74 (24)	29.41 (21)	38.19 (20)	51.69 (20)
	Feasible	9.69 (19)	15.18 (16)	23.62 (17)	27.51 (15)	42.63 (17)
	Penalty	7.58 (14)	14.44 (15)	29.16 (16)	31.26 (15)	65.70 (16)
	Level-bundle	20.16 (49)	68.31 (94)	164.62 (146)	396.63 (258)	742.55 (362)
	IPOPT	80.78 (95)	154.43 (105)	249.86 (112)	744.89 (264)	766.44 (224)
0.95	Proximal	8.25 (17)	29.43 (26)	28.27 (17)	38.69 (16)	55.75 (17)
	Feasible	5.26 (9)	12.60 (15)	13.26 (9)	18.62 (9)	30.21 (11)
	Penalty	4.03 (7)	17.11 (15)	17.89 (9)	21.67 (8)	78.86 (20)
	Level-bundle	16.88 (45)	82.38 (112)	193.99 (171)	786.34 (505)	884.85 (426)
	IPOPT	40.43 (47)	261.04 (174)	317.70 (142)	250.56 (89)	456.57 (132)
0.99	Proximal	9.11 (19)	28.36 (23)	48.01 (29)	43.38 (19)	91.62 (31)*
	Feasible	45.14 (80)	48.68 (50)*	37.49 (25)*	59.99 (29)*	107.59 (27)*
	Penalty	8.86 (17)	24.01 (20)	48.39 (28)	65.53 (26)	96.50 (31)*
	Level-bundle	17.46 (44)	112.28 (155)	570.11 (503)	801.06 (503)	1044.84 (503)
	IPOPT	38.97 (45)	236.51 (154)	445.32 (195)	757.67 (264)	1403.85 (403)*

Table 10: Transportation benchmark under the Student-t distribution, without Bonferroni initialization. Each entry reports CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

p	Algorithm	$m = 10$	$m = 20$	$m = 30$	$m = 40$	$m = 50$
0.8	Proximal	11.56 (26)	43.38 (45)	89.77 (61)	92.73 (45)	118.11 (45)
	Feasible	7.50 (15)	24.47 (22)	41.28 (24)	72.24 (27)	157.98 (31)
	Penalty	7.83 (15)	33.86 (27)	63.40 (35)	74.73 (35)	69.89 (18)
	IPOPT	66.92 (99)	389.83 (269)	754.94 (376)	1210.22 (439)	1805.61 (580)
0.85	Proximal	13.20 (28)	54.57 (48)	54.23 (36)	94.88 (45)	134.17 (49)
	Feasible	6.25 (14)	23.43 (21)	37.30 (21)	62.03 (23)	184.46 (28)
	Penalty	4.82 (9)	93.44 (75)	35.40 (18)	36.22 (15)	54.54 (18)
	IPOPT	61.99 (93)	432.91 (294)	506.02 (248)	898.86 (355)	1499.29 (477)
0.9	Proximal	15.61 (32)	43.89 (40)	100.01 (60)	110.87 (45)	131.17 (47)
	Feasible	7.92 (18)	21.64 (18)	41.03 (21)	62.35 (22)	147.48 (25)
	Penalty	4.04 (7)	25.58 (20)	26.47 (14)	82.61 (36)	17.68 (2)*
	IPOPT	52.65 (79)	290.00 (200)	718.75 (346)	1198.08 (466)	1805.44 (576)
0.95	Proximal	14.28 (26)	37.92 (33)	65.87 (37)	158.10 (65)	347.60 (94)
	Feasible	8.92 (20)	24.39 (26)	48.05 (29)	130.81 (44)*	312.05 (85)*
	Penalty	4.92 (8)	23.08 (16)	30.68 (17)	48.95 (18)	68.31 (20)*
	IPOPT	63.53 (96)	386.45 (267)	660.36 (309)	1248.54 (483)	1085.66 (346)

Table 11: Transportation benchmark under the Student-t distribution, with Bonferroni initialization. Each entry reports CPU time in seconds (total oracle calls). Constructing the Bonferroni initial point requires an average of 8.9 oracle calls for all methods. Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

p	Algorithm	$m = 10$	$m = 20$	$m = 30$	$m = 40$	$m = 50$
0.8	Proximal	10.93 (24)	23.10 (25)	33.38 (24)	43.06 (23)	46.12 (21)
	Feasible	8.93 (20)	14.60 (17)	28.09 (21)	31.15 (17)	37.67 (17)
	Penalty	6.42 (15)	23.39 (25)	42.60 (29)	53.63 (29)	81.66 (19)
	IPOPT	20.12 (35)	169.51 (122)	498.62 (248)	547.43 (213)	1023.79 (336)
0.85	Proximal	10.32 (24)	21.77 (21)	39.81 (28)	45.79 (24)	70.39 (27)
	Feasible	8.35 (19)	9.42 (9)	24.69 (19)	27.80 (17)	44.98 (20)
	Penalty	5.88 (14)	10.30 (10)	27.62 (18)	50.15 (25)	48.50 (21)
	IPOPT	40.18 (64)	125.11 (89)	175.25 (88)	441.57 (175)	1182.36 (387)
0.9	Proximal	12.09 (30)	29.25 (32)	40.50 (30)	46.72 (29)	67.56 (29)
	Feasible	9.07 (22)	14.20 (18)	21.65 (20)	34.80 (22)	44.79 (22)
	Penalty	6.79 (16)	12.34 (15)	24.05 (19)	37.75 (23)	97.33 (34)
	IPOPT	37.35 (53)	155.78 (108)	146.44 (74)	897.91 (358)	1164.97 (375)
0.95	Proximal	18.30 (37)	45.61 (41)	128.25 (68)	257.76 (87)	1431.74 (265)
	Feasible	8.01 (20)	16.50 (21)	56.51 (38)	105.62 (40)*	125.96 (33)
	Penalty	6.63 (16)	19.66 (19)	38.74 (19)	64.27 (25)*	146.09 (44)
	IPOPT	40.78 (67)	209.68 (151)	301.51 (151)	990.60 (381)	864.95 (282)

$\Sigma \in \mathbb{R}^{48\times 48}$. The radial law is chosen to give either the multivariate Gaussian case or the multivariate Student-t case. In the Gaussian case, Σ is the dispersion matrix, whereas in the Student-t case, Σ denotes the scale matrix (the dispersion matrix up to a multiplicative constant), together with the prescribed degrees of freedom. As shown in [30], the reservoir level constraints can be expressed as a two-sided probabilistic requirement of the form

$$\mathbb{P}\left(\tilde{A}x + a \le \xi \le \tilde{A}x + b\right) \ge p,$$

where the matrix $\tilde{A}$ captures the cumulative impact of release decisions on reservoir levels, and a, b incorporate deterministic components such as initial storage levels, deterministic inflows, and admissible storage bounds, with $a_i < b_i$ for all i. For algorithmic convenience, the two-sided constraint is rewritten in the canonical one-sided form by defining

$$W := \begin{bmatrix} I \\ -I \end{bmatrix}, \; H := \begin{bmatrix} \tilde{A} \\ -\tilde{A} \end{bmatrix}, \; h := \begin{bmatrix} b \\ -a \end{bmatrix}.$$

Then $\tilde{A}x + a \le \xi \le \tilde{A}x + b \iff W\xi \le Hx + h$, and the chance constraint becomes $\mathbb{P}(W\xi \le Hx + h) \ge p$.

The primitive inflow vector $\xi \in \mathbb{R}^{48}$ remains continuous, nondegenerate, and elliptically symmetric with positive definite dispersion matrix Σ. The transformed vector $W\xi \in \mathbb{R}^{96}$ is elliptically symmetric with location vector $W\mu$ and dispersion matrix $W\Sigma W^\top = \begin{bmatrix} \Sigma & -\Sigma \\ -\Sigma & \Sigma \end{bmatrix}$, which is rank deficient because W does not have full row rank. This rank deficiency is allowed under the assumptions of the paper, as discussed in Remark 3.

With this representation, the hydro-valley problem has the form (P), with $x \in \mathbb{R}^{672}$, $\xi \in \mathbb{R}^{48}$, and $W\xi \in \mathbb{R}^{96}$.

Here, the deterministic constraints $Ax \le b$ encode operational and capacity limits, while the joint chance constraint ensures that reservoir levels remain within admissible bounds at all time–reservoir combinations with probability at least p.

The hydro benchmark is reasonably large and structurally coupled. In particular, the dimension of the random vector equals the number of time–reservoir combinations, and evaluating

the joint probability involves high-dimensional probability estimation. This makes the problem especially suitable for assessing the scalability and robustness of algorithms for joint chance-constrained optimization.

4.4.1 Hydro benchmark results

We report benchmark results for the Isère hydro-valley instance under both Gaussian and Student-t distributions in Tables 12–15.

Table 12: Isère hydro benchmark (Gaussian distribution), without applying the Bonferroni procedure to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	387.10 (34)	442.55 (35)	347.81 (27)	213.37 (17)	304.60 (24)
Feasible	776.16 (63)	684.68 (60)	522.70 (49)	192.23 (17)	180.95 (17)
Penalty	371.97 (33)	482.58 (45)	223.08 (21)	203.38 (19)	215.36 (20)
Level-bundle	415.14 (42)	318.41 (38)	295.57 (35)	279.20 (33)	254.67 (29)
IPOPT	1810.47 (107)*	1035.52 (62)	1816.30 (109)*	821.93 (50)	1192.78 (70)

Table 13: Isère hydro benchmark (Gaussian distribution), with the Bonferroni procedure applied to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}. The average number of oracle calls required to construct the Bonferroni initial point is 11 for all methods, and is included in the table.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	354.07 (34)	198.96 (20)	299.61 (28)	257.78 (24)	326.95 (25)
Feasible	415.25 (40)	511.26 (48)	313.44 (30)	213.90 (21)	425.88 (39)
Penalty	262.02 (26)	229.24 (23)	282.94 (27)	207.02 (20)	234.30 (21)
Level-bundle	435.65 (50)	398.22 (46)	375.53 (43)	362.50 (41)	257.17 (29)
IPOPT	1889.05 (116)*	714.73 (48)	925.45 (59)	1894.94 (116)*	1170.63 (71)

Table 14: Isère hydro benchmark (Student-t distribution), without applying the Bonferroni procedure to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	372.49 (33)	327.88 (29)	293.08 (24)	226.20 (17)
Feasible	734.88 (65)	636.95 (56)	737.72 (62)	589.22 (52)
Penalty	369.08 (34)	364.81 (33)	216.00 (19)	192.91 (17)
IPOPT	1295.38 (81)	1944.18 (119)*	769.48 (46)	641.21 (41)

We next report benchmark results for the Ain hydro-valley instance under both Gaussian and Student-t distributions in Tables 16–19.

Across the two hydro instances, Penalty generally achieves the lowest CPU times, while Feasible becomes particularly competitive when Bonferroni initialization is used. Proximal also remains consistently competitive and exhibits stable performance across reliability levels and distributions. The level-bundle method is typically more computationally expensive, although

Table 15: Isère hydro benchmark (Student-t distribution), with the Bonferroni procedure applied to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}. The average number of oracle calls required to construct the Bonferroni initial point is 9.3 for all methods.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	290.83 (27)	300.72 (27)	376.15 (32)	409.38 (32)
Feasible	377.26 (34)	304.34 (28)	337.04 (30)	272.54 (25)
Penalty	447.30 (41)	336.76 (31)	316.77 (29)	282.72 (26)
IPOPT	543.03 (40)	516.56 (38)	1897.84 (124)*	1214.09 (82)

Table 16: Ain hydro benchmark (Gaussian distribution), without applying the Bonferroni procedure to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	424.68 (28)	399.51 (29)	314.57 (22)	376.90 (20)	396.16 (27)
Feasible	460.81 (45)	684.48 (66)	656.41 (65)	603.29 (56)	200.16 (20)
Penalty	238.77 (24)	149.63 (15)	186.61 (18)	231.47 (22)	297.74 (27)
Level-bundle	444.73 (53)	450.30 (18)	624.07 (36)	503.49 (49)	384.77 (46)
IPOPT	1882.45 (110)*	1892.91 (117)*	664.74 (41)*	1885.24 (111)*	959.33 (57)

Table 17: Ain hydro benchmark (Gaussian distribution), with the Bonferroni procedure applied to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}. The average number of oracle calls required to construct the Bonferroni initial point is 9.4 for all methods.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$	$p = 0.99$
Proximal	332.48 (28)	258.15 (23)	257.56 (23)	312.91 (25)	410.05 (28)
Feasible	285.33 (30)	230.82 (25)	222.09 (24)	242.12 (25)	138.94 (14)
Penalty	247.63 (27)	209.36 (23)	214.46 (23)	228.45 (24)	282.73 (28)
Level-bundle	492.15 (61)	510.86 (26)	683.48 (44)	547.68 (57)	386.01 (46)
IPOPT	1383.57 (86)	1977.32 (78)*	1520.24 (63)	1247.34 (53)	1485.63 (58)

Table 18: Ain hydro benchmark (Student-t distribution), without applying the Bonferroni procedure to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	264.42 (24)	328.82 (22)	448.74 (30)	538.36 (30)
Feasible	359.12 (35)	579.44 (53)	673.07 (58)	391.76 (37)
Penalty	200.71 (20)	267.65 (26)	167.03 (14)	179.02 (17)
IPOPT	1899.88 (76)*	1833.63 (76)*	1828.58 (74)*	1870.30 (77)*

Table 19: Ain hydro benchmark (Student-t distribution), with the Bonferroni procedure applied to obtain a good starting point. Each entry is reported as CPU time in seconds (oracle calls). Entries marked with $*$ indicate that the corresponding run did not attain an objective value within relative tolerance 5×10^{-4} of the best feasible objective value for that instance, or returned an infeasible solution under a feasibility tolerance of 10^{-3}. The average number of oracle calls required to construct the Bonferroni initial point is 9.3 for all methods.

Algorithm	$p = 0.8$	$p = 0.85$	$p = 0.9$	$p = 0.95$
Proximal	405.74 (31)	336.01 (25)	430.79 (29)	433.69 (28)
Feasible	381.34 (38)	387.35 (37)	275.25 (27)	628.28 (60)
Penalty	292.53 (30)	302.12 (29)	246.90 (25)	198.23 (20)
IPOPT	1923.73 (84)*	1507.10 (66)	897.66 (48)	1322.50 (80)

it can be competitive on some Gaussian instances. IPOPT is generally the most expensive method and, on several instances, either returns an infeasible solution or an objective value outside the prescribed comparison tolerance. Overall, the three structured methods provide the most favorable balance between computational efficiency, reliability, and solution quality.

Figures 2 and 3 report the full and zoomed performance profiles, together with the corresponding data profiles, for the Gaussian instances. The profiles are presented with respect to the number of oracle calls and CPU time, respectively. The Student-t profiles are omitted because level-bundle method is not applicable in that setting.

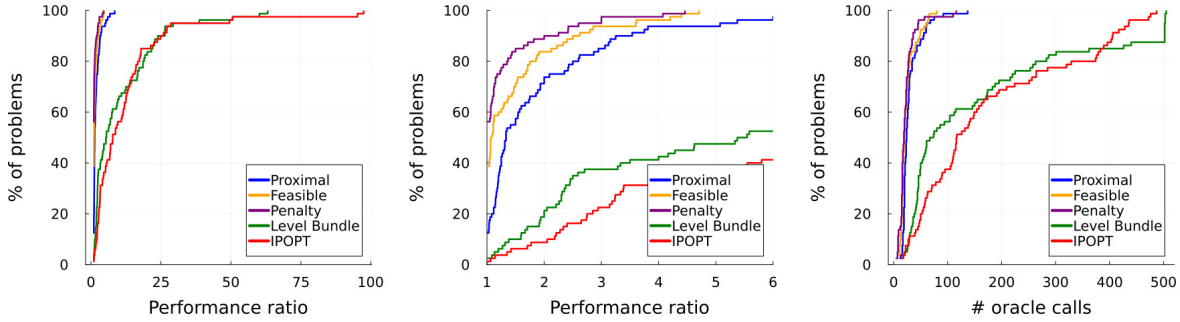

Figure 2: Gaussian instances: full performance profile, zoomed performance profile, and data profile with respect to the number of oracle calls ($\#F$).

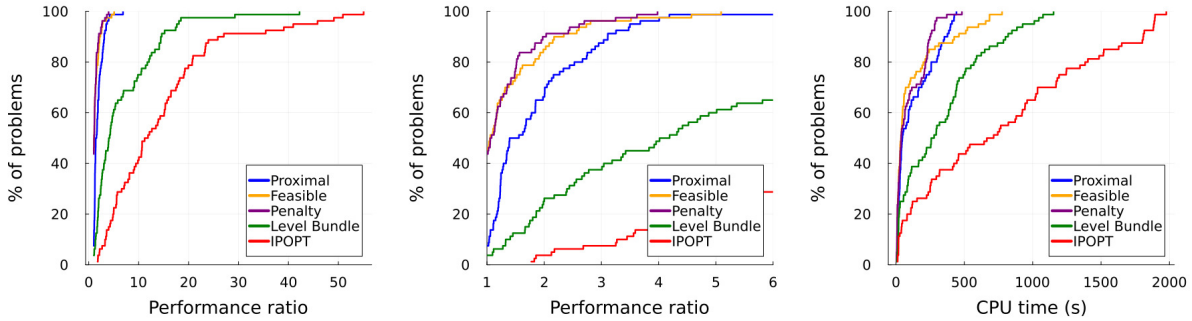

Figure 3: Gaussian instances: full performance profile, zoomed performance profile, and data profile with respect to CPU time in seconds.

Taken together, the benchmark tables and performance profiles provide a consistent comparison of the methods across problem classes, reliability levels, and computational budgets.

5 Conclusion

This paper presented a modular computational toolbox for linear optimization problems with joint affine chance constraints under elliptically symmetric uncertainty. The framework separates the deterministic model, probability evaluation, and optimization method through a common oracle interface. The oracle combines spherical–radial decomposition with numerical integration on the unit sphere to estimate both the joint probability and its gradient, and also accommodates rank-deficient transformations such as those induced by stacked two-sided constraints.

The Proximal, Feasible, and Penalty methods exploit a common structured local model M_ν that linearizes the dependence structure while retaining the nonlinear marginal distributions. The Proximal method provides a general-purpose scheme, while the Feasible and Penalty variants respectively maintain feasible stability centers and allow direct progress from infeasible points. Under the stated regularity assumptions, their cluster points satisfy the corresponding criticality conditions, and feasible critical points satisfy the KKT conditions under an appropriate constraint qualification.

The numerical experiments on cash-matching, transportation, and hydro-valley instances show that the structured methods are competitive with, and often faster than, the level-bundle method and IPOPT. Although each structured iteration generally requires solving a more demanding $C^{1,1}$ master problem, the accurate local approximation provided by M_ν typically leads to substantially fewer iterations and probability-oracle evaluations, resulting in lower CPU times in most tested instances. The Penalty method is often the fastest, the Feasible method is particularly effective when a good feasible starting point is available, and the Proximal method remains a robust general-purpose alternative. Bonferroni initialization can further benefit the Feasible method. Overall, the results show that explicitly exploiting the structure of the probability constraint can provide both computational efficiency and robust performance with finite-sample probability and gradient estimates, while retaining applicability beyond the log-concave setting required by the level-bundle approach.

Declarations

Funding. Y. Chu and W. de Oliveira acknowledge financial support from PGMO (Programme Gaspard Monge pour l'Optimisation) through the project "Linear Chance-Constrained Programming: A Computational Tool".

Competing Interests. W. van Ackooij is an Associate Editor of *Mathematical Programming Computation*. The authors declare no other relevant financial or non-financial interests.

Data Availability. The benchmark data used in the numerical experiments are available with the reproducibility material at `https://gitlab.com/stochasticprogramming/ProbLiM`.

Code Availability. The ProbLiM toolbox, test problems, and reproduction material are available at `https://gitlab.com/stochasticprogramming/ProbLiM`.

Appendix A Preliminary results

We recall only the notation and results needed for the convergence analysis.

Notation

For $a \in \mathbb{R}$, let $[a]_+ := \max\{a, 0\}$. Since X is a closed convex polyhedron, its normal cone is

$$N_X(x) = \{z \in \mathbb{R}^n : \langle z, y - x\rangle \leq 0 \text{ for all } y \in X\}.$$

Moreover, although not needed, we recall that polyhedral have an explicit convex normal cone, see e.g., [38, Proposition 2.16]. We next record the approximation properties of the local model that are used repeatedly in the method-specific convergence analyses.

Appendix B Local model and uniform estimate

Recall from Section 3.2 that M_ν denotes the local model of $p-\varphi$ constructed at the stability center $\hat{x}^\nu$, using $v^\nu \in [0,1]^d$ satisfying the linear system defining the model. By construction,

$$M_\nu(\hat{x}^\nu) = p - \varphi(\hat{x}^\nu),\ \nabla M_\nu(\hat{x}^\nu) = -\nabla\varphi(\hat{x}^\nu).$$

Lemma 6 (Uniform model estimates)**.** *Suppose that Assumption 1 holds. Then there exists a constant $\ell > 0$, independent of ν, such that*

$$|p - \varphi(x) - M_\nu(x)| \le \frac{\ell}{2}\|x - \hat{x}^\nu\|^2 \text{ for all } x \in X, \tag{10}$$

and

$$\|-\nabla\varphi(x) - \nabla M_\nu(x)\| \le \ell\|x - \hat{x}^\nu\| \text{ for all } x \in X. \tag{11}$$

Proof. Let $D_\nu := p - \varphi - M_\nu$. By Assumption A.4, $\nabla\varphi$ is Lipschitz on X. Moreover, by Assumption A.5, each $f_j = F_j'$ is Lipschitz on $y_j(X)$. Since y_j is affine and $v^\nu \in [0,1]^d$, the gradients ∇M_ν are Lipschitz on X with a constant independent of ν. Hence ∇D_ν is uniformly Lipschitz, say with constant ℓ.

Since $D_\nu(\hat{x}^\nu) = 0$ and $\nabla D_\nu(\hat{x}^\nu) = -\nabla\varphi(\hat{x}^\nu) - \nabla M_\nu(\hat{x}^\nu) = 0$,

$$\|-\nabla\varphi(x) - \nabla M_\nu(x)\| = \|\nabla D_\nu(x) - \nabla D_\nu(\hat{x}^\nu)\| \le \ell\|x - \hat{x}^\nu\|,$$

which proves (11). Finally, since X is convex,

$$|D_\nu(x)| = \left|\int_0^1 \left\langle \nabla D_\nu\big(\hat{x}^\nu + t(x-\hat{x}^\nu)\big), x - \hat{x}^\nu\right\rangle dt\right| \le \int_0^1 \ell t\|x-\hat{x}^\nu\|^2\, dt = \frac{\ell}{2}\|x-\hat{x}^\nu\|^2,$$

which proves (10) with the same constant ℓ. □

Remark 8 (Quadratic upper approximation). Lemma 6 gives

$$p - \varphi(x) \le M_\nu(x) + \frac{\ell}{2}\|x - \hat{x}^\nu\|^2 \text{ for all } x \in X. \tag{12}$$

Consequently, if $M_\nu(x) + \frac{\ell_k}{2}\|x-\hat{x}^\nu\|^2 \le 0$ and $\ell_k \ge \ell$, then $\varphi(x) \ge p$.

With the common criticality framework and uniform model estimates in place, we now specialize the analysis to Algorithm 2.

Appendix C Feasible method: convergence analysis

Throughout this section, we consider Algorithm 2 with $\texttt{Tol} = 0$, applied to problem (P) under Assumption 1. We assume that $\delta > 0$ and that a feasible initial point is available: $x^0 \in X,\ \varphi(x^0) \ge p$.

Let $k(\nu)$ denote the iteration at which the next stability center $\hat{x}^{\nu+1}$ is generated from $\hat{x}^\nu$. Thus, $\hat{x}^{\nu+1}$ is a stationary point of

$$\begin{aligned} \min_{x\in X}\quad & c^\top x + \frac{\delta}{2}\|x - \hat{x}^\nu\|^2, \\ \text{s.t.}\quad & M_\nu(x) + \frac{\ell_{k(\nu)}}{2}\|x-\hat{x}^\nu\|^2 \le 0, \end{aligned}$$

and satisfies $c^\top \hat{x}^{\nu+1} + \frac{\delta}{2}\|\hat{x}^{\nu+1} - \hat{x}^\nu\|^2 \le c^\top \hat{x}^\nu$. Such a stationary point is well-defined, as X is a bounded polyhedron and the underlying functions are continuously differentiable. The following results provide the main argument used in the convergence analysis.

Lemma 7. *Let $\mathcal{N} \subset \mathbb{N}$ be infinite and suppose that $\lim_{\mathcal{N}\ni\nu\to\infty} \hat{x}^\nu = \lim_{\mathcal{N}\ni\nu\to\infty} \hat{x}^{\nu+1} = \hat{x}$. If $\big(\ell_{k(\nu)}\big)_{\nu\in\mathcal{N}}$ is bounded, then $\hat{x}$ is critical for problem* (P)*. The same conclusion holds if $x^{k+1} = \hat{x}^\nu$ at some iteration.*

Proof. Since every stability center is feasible, continuity of φ gives $\varphi(\hat{x}) \ge p$. Set $s_\nu := \hat{x}^{\nu+1} - \hat{x}^\nu$. The stationarity conditions for the feasible master problem yield $a_\nu, b_\nu \ge 0$, with $a_\nu + b_\nu = 1$, and $n_\nu \in N_X(\hat{x}^{\nu+1})$ such that

$$0 = a_\nu\big(c + \delta s_\nu\big) + b_\nu\big[\nabla M_\nu(\hat{x}^{\nu+1}) + \ell_{k(\nu)} s_\nu\big] + n_\nu, \tag{13}$$

and

$$b_\nu \left[M_\nu(\hat{x}^{\nu+1}) + \frac{\ell_{k(\nu)}}{2}\|s_\nu\|^2\right] = 0. \tag{14}$$

After passing to a (possible) further subsequence, $a_\nu \to a \in [0,1],\ b_\nu \to 1-a$. Since $s_\nu \to 0$ and $\ell_{k(\nu)}$ is bounded, $\delta s_\nu \to 0,\ \ell_{k(\nu)} s_\nu \to 0$. Moreover, Lemma 6 gives

$$\big\|\nabla M_\nu(\hat{x}^{\nu+1}) + \nabla\varphi(\hat{x}^{\nu+1})\big\| \le \ell\|s_\nu\| \to 0,$$

and hence $\nabla M_\nu(\hat{x}^{\nu+1}) \to -\nabla\varphi(\hat{x})$. Equation (13) implies that (n_ν) is bounded. Passing to a further subsequence and using the closed graph of N_X yields

$$0 \in ac - (1-a)\nabla\varphi(\hat{x}) + N_X(\hat{x}). \tag{15}$$

If $\varphi(\hat{x}) = p$, this is precisely the criticality condition. If $\varphi(\hat{x}) > p$, then Lemma 6, boundedness of $\ell_{k(\nu)}$, and $s_\nu \to 0$ give

$$M_\nu(\hat{x}^{\nu+1}) + \frac{\ell_{k(\nu)}}{2}\|s_\nu\|^2 \to p - \varphi(\hat{x}) < 0.$$

Thus the master constraint is eventually inactive, so (14) gives $b_\nu = 0$ eventually. Hence $a = 1$, and (15) again gives the required criticality.

If $x^{k+1} = \hat{x}^\nu$, the same argument applies with $s_\nu = 0$, using $M_\nu(\hat{x}^\nu) = p - \varphi(\hat{x}^\nu),\ \nabla M_\nu(\hat{x}^\nu) = -\nabla\varphi(\hat{x}^\nu)$. □

Proposition 8 (Null steps)**.** *Under Assumption 1, Algorithm 2 generates only finitely many consecutive null steps. Moreover, the parameters used to generate serious steps satisfy*

$$\ell_{k(\nu)} \le \bar{\ell} := \max\{\ell_0, \ell_{\max}, \beta\ell\}, \tag{16}$$

where ℓ is the constant in Lemma 6.

Proof. If x^{k+1} is feasible for the master problem, then $M_\nu(x^{k+1}) + \frac{\ell_k}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \le 0$. By (12), $p - \varphi(x^{k+1}) \le -\frac{\ell_k - \ell}{2}\|x^{k+1} - \hat{x}^\nu\|^2$. Therefore, if $\ell_k \ge \ell$, then $\varphi(x^{k+1}) \ge p$, so the trial point is accepted. Since a null step replaces ℓ_k by $\beta\ell_k$, only finitely many consecutive null steps are possible.

After a serious step, the parameter is reset below $\ell_{\max}$. If a serious step is preceded by a null step, then the parameter before that null step must be smaller than ℓ; hence the serious step is generated with a parameter smaller than $\beta\ell$. This proves (16). □

Proposition 9 (Serious steps)**.** *Suppose that Algorithm 2 does not terminate. Then $\nu \to \infty$, and every cluster point of $(\hat{x}^\nu)_{\nu\in\mathbb{N}}$ is critical for problem* (P)*.*

Proof. By Proposition 8, an infinite run cannot contain an infinite block of consecutive null steps. It therefore generates infinitely many serious steps, and hence $\nu \to \infty$.

At every serious step, $c^\top \hat{x}^{\nu+1} + \frac{\delta}{2}\|\hat{x}^{\nu+1} - \hat{x}^\nu\|^2 \leq c^\top \hat{x}^\nu$. Summing over the serious steps gives

$$\frac{\delta}{2} \sum_{\iota=0}^{\nu} \|\hat{x}^{\iota+1} - \hat{x}^\iota\|^2 \leq c^\top \hat{x}^0 - c^\top \hat{x}^{\nu+1}.$$

Since X is compact, $c^\top x$ is bounded below on X. Therefore, $\sum_{\iota=0}^{\infty} \|\hat{x}^{\iota+1} - \hat{x}^\iota\|^2 < +\infty$, and consequently $\|\hat{x}^{\nu+1} - \hat{x}^\nu\| \to 0$.

Let $\hat{x}$ be a cluster point of $(\hat{x}^\nu)_{\nu\in\mathbb{N}}$. By compactness of X, such cluster points exist. Along some infinite $\mathcal{N} \subset \mathbb{N}$, $\hat{x}^\nu \to \hat{x}$. Since $\|\hat{x}^{\nu+1} - \hat{x}^\nu\| \to 0$, we also have $\hat{x}^{\nu+1} \to \hat{x}$ along $\mathcal{N}$. Finally, Proposition 8 gives $\ell_{k(\nu)} \leq \bar{\ell}$ for all ν. Thus Lemma 7 applies and shows that $\hat{x}$ is critical for problem (P). □

We are now ready to prove the main convergence theorem for the Feasible Method.

C.1 Proof of Theorem 3

Proof. Item *ii)* follows by induction. The initial stability center is feasible by assumption, and the stability center is updated only when a trial point satisfying the true chance constraint is accepted.

Let $\hat{x}^\nu$ be the current stability center. By item *ii)* and the exactness of the model at the current stability center, $M_\nu(\hat{x}^\nu) = p - \varphi(\hat{x}^\nu) \leq 0$. Thus, $\hat{x}^\nu$ is feasible for the master problem (MP-Feas). Its feasible set is a nonempty closed subset of the compact set X, and its objective is continuous. Hence the master problem admits a global minimizer x^{k+1}. Since $\hat{x}^\nu$ is master-feasible, optimality gives $c^\top x^{k+1} + \frac{\delta}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \leq c^\top \hat{x}^\nu$. Moreover, a global minimizer satisfies the Fritz–John stationarity conditions used in Lemma 7. Therefore, the trial point required by Algorithm 2 exists, proving item *i)*.

Suppose that the algorithm terminates. Since $\texttt{Tol} = 0$, the termination test implies $x^{k+1} = \hat{x}^\nu$. The second statement of Lemma 7 then shows that the final stability center is critical. If the algorithm does not terminate, Proposition 9 shows that every cluster point of the stability centers is critical. Since X is compact, such a cluster point exists. This proves item *iii)*.

Finally, let $\hat{x}$ satisfy the assumptions of item *iv)*. By item *iii)*, $\hat{x}$ is critical. Hence Lemma 1 implies that $\hat{x}$ satisfies the KKT conditions for problem (P). □

We next turn to the convergence analysis of the Penalty method, which requires a criticality notion associated with the penalized objective.

Appendix D Penalty method: convergence analysis

For $r > 0$, recall $\psi_r(x) := c^\top x + r[p - \varphi(x)]_+$, and $\psi^M_{r,\nu}(x) := c^\top x + r[M_\nu(x)]_+$. The following results are crucial for convergence analysis of Algorithm 3.

D.1 Convergence analysis

Throughout this subsection, $r > 0$ is fixed. Recall that $k(\nu)$ denotes the iteration at which the stability center $\hat{x}^{\nu+1}$ is generated.

Lemma 10. *Let $\mathcal{N} \subset \mathbb{N}$ be infinite and suppose that $\lim_{\mathcal{N}\ni\nu\to\infty} \hat{x}^\nu = \lim_{\mathcal{N}\ni\nu\to\infty} \hat{x}^{\nu+1} = \hat{x}$. If $\big(\mu_{k(\nu)}\big)_{\nu\in\mathcal{N}}$ is bounded, then $\hat{x}$ is penalty-critical. The same conclusion holds if $x^{k+1} = \hat{x}^\nu$ at some iteration.*

Proof. Set $s_\nu := \hat{x}^{\nu+1} - \hat{x}^\nu$. Stationarity of $\hat{x}^{\nu+1}$ for the penalty master problem gives

$$0 \in c + r\theta_\nu \nabla M_\nu(\hat{x}^{\nu+1}) + r\mu_{k(\nu)} s_\nu + N_X(\hat{x}^{\nu+1}),$$

where

$$\theta_\nu \in \begin{cases} \{0\}, & M_\nu(\hat{x}^{\nu+1}) < 0, \\ [0,1], & M_\nu(\hat{x}^{\nu+1}) = 0, \\ \{1\}, & M_\nu(\hat{x}^{\nu+1}) > 0. \end{cases}$$

Define $\alpha_\nu := \frac{1}{1+r\theta_\nu}$. Since $N_X(\hat{x}^{\nu+1})$ is a cone,

$$0 \in \alpha_\nu c + (1-\alpha_\nu)\nabla M_\nu(\hat{x}^{\nu+1}) + \alpha_\nu r\mu_{k(\nu)} s_\nu + N_X(\hat{x}^{\nu+1}). \tag{17}$$

After passing to a subsequence, $\alpha_\nu \to \alpha \in \left[\frac{1}{1+r}, 1\right]$. Since $s_\nu \to 0$ and $\mu_{k(\nu)}$ is bounded, $\alpha_\nu r\mu_{k(\nu)} s_\nu \to 0$. Moreover, Lemma 6 gives $\nabla M_\nu(\hat{x}^{\nu+1}) \to -\nabla\varphi(\hat{x})$ and $M_\nu(\hat{x}^{\nu+1}) \to p - \varphi(\hat{x})$. Passing to the limit in (17) yields $0 \in \alpha c - (1-\alpha)\nabla\varphi(\hat{x}) + N_X(\hat{x})$.

If $\varphi(\hat{x}) > p$, then $M_\nu(\hat{x}^{\nu+1}) < 0$ eventually, so $\alpha = 1$. If $\varphi(\hat{x}) < p$, then $M_\nu(\hat{x}^{\nu+1}) > 0$ eventually, so $\alpha = 1/(1+r)$. If $\varphi(\hat{x}) = p$, then $\alpha \in [1/(1+r), 1]$. Hence $\hat{x}$ is penalty-critical.

If $x^{k+1} = \hat{x}^\nu$, the same argument applies directly, using $M_\nu(\hat{x}^\nu) = p - \varphi(\hat{x}^\nu)$, $\nabla M_\nu(\hat{x}^\nu) = -\nabla\varphi(\hat{x}^\nu)$. □

Proposition 11 (Null steps)**.** *Under Assumption 1, Algorithm 3 generates only finitely many consecutive null steps. Moreover, the parameters used to generate serious steps satisfy*

$$\mu_{k(\nu)} \le \bar{\mu} := \max\left\{\mu_0,\, \mu_{\max},\, \beta\left(\ell + \frac{\gamma}{r}\right)\right\}, \tag{18}$$

where ℓ is the constant in Lemma 6.

Proof. For every $x \in X$, Lemma 6 gives $p - \varphi(x) \le M_\nu(x) + \frac{\ell}{2}\|x - \hat{x}^\nu\|^2$. Therefore, $[p-\varphi(x)]_+ \le [M_\nu(x)]_+ + \frac{\ell}{2}\|x - \hat{x}^\nu\|^2$, and hence

$$\psi_r(x) \le \psi^M_{r,\nu}(x) + \frac{r\ell}{2}\|x - \hat{x}^\nu\|^2. \tag{19}$$

For a trial point x^{k+1}, the master decrease condition gives

$$\psi^M_{r,\nu}(x^{k+1}) + \frac{r\mu_k}{2}\|x^{k+1} - \hat{x}^\nu\|^2 \le \psi^M_{r,\nu}(\hat{x}^\nu) = \psi_r(\hat{x}^\nu).$$

Combining this inequality with (19) yields

$$\psi_r(x^{k+1}) \le \psi_r(\hat{x}^\nu) - \frac{r(\mu_k - \ell)}{2}\|x^{k+1} - \hat{x}^\nu\|^2.$$

Thus the trial is accepted whenever $r(\mu_k - \ell) \ge \gamma$. Since each null step replaces μ_k by $\beta\mu_k$, only finitely many consecutive null steps are possible.

After a serious step, $\mu_{k+1} \le \mu_{\max}$. If a serious step is preceded by a null step, then the parameter before that null step is smaller than $\ell + \gamma/r$; hence the serious step is generated with a parameter smaller than $\beta\left(\ell + \frac{\gamma}{r}\right)$. Including the initial parameter gives (18). □

Proposition 12 (Serious steps)**.** *Suppose that Algorithm 3 does not terminate. Then $\nu \to \infty$, and every cluster point of the sequence of stability centers is penalty-critical.*

Proof. By Proposition 11, an infinite run contains infinitely many serious steps, so $\nu \to \infty$. At every serious step,

$$\psi_r(\hat{x}^{\nu+1}) \le \psi_r(\hat{x}^{\nu}) - \frac{\gamma}{2}\|\hat{x}^{\nu+1} - \hat{x}^{\nu}\|^2.$$

Summing gives

$$\frac{\gamma}{2}\sum_{\iota=0}^{\nu} \|\hat{x}^{\iota+1} - \hat{x}^{\iota}\|^2 \le \psi_r(\hat{x}^0) - \psi_r(\hat{x}^{\nu+1}).$$

Since X is compact, ψ_r is bounded below on X. Therefore, $\sum_{\iota=0}^{\infty} \|\hat{x}^{\iota+1} - \hat{x}^{\iota}\|^2 < +\infty,\ \|\hat{x}^{\nu+1} - \hat{x}^{\nu}\| \to 0$.

Let $\hat{x}$ be a cluster point, and choose an infinite $\mathcal{N} \subset \mathbb{N}$ such that $\hat{x}^{\nu} \to \hat{x}$ along $\mathcal{N}$. Then also $\hat{x}^{\nu+1} \to \hat{x}$. By (18), $\big(\mu_{k(\nu)}\big)_{\nu\in\mathcal{N}}$ is bounded. Hence Lemma 10 implies that $\hat{x}$ is penalty-critical. □

We next characterize the limiting behavior of penalty-critical points as the penalty parameter tends to infinity. Depending on the feasibility of the limit point, three cases arise.

D.2 Proof of Proposition 5

Proof. Let $\bar{x}$ be an accumulation point of $(\bar{x}_j)$. Choose an infinite index set $\mathcal{J} \subset \mathbb{N}$ such that $\bar{x}_j \to \bar{x}$ as $j \to \infty,\ j \in \mathcal{J}$. Since $r_j \to +\infty$, the same holds along $\mathcal{J}$. For every $j \in \mathcal{J}$, penalty criticality gives

$$0 \in \alpha_j c - (1-\alpha_j)\nabla\varphi(\bar{x}_j) + N_X(\bar{x}_j), \tag{20}$$

with α_j satisfying Definition 3.

Suppose first that $\varphi(\bar{x}) > p$. By continuity of φ, $\varphi(\bar{x}_j) > p$ eventually along $\mathcal{J}$, and hence $\alpha_j = 1$. Thus $0 \in c + N_X(\bar{x}_j)$. Since the graph of N_X is closed, $0 \in c + N_X(\bar{x})$, which is the KKT condition with multiplier $\lambda = 0$.

Suppose next that $\varphi(\bar{x}) = p$ and $0 \notin -\nabla\varphi(\bar{x}) + N_X(\bar{x})$. There cannot exist an infinite $\mathcal{J}_1 \subset \mathcal{J}$ such that $\varphi(\bar{x}_j) < p$ for $j \in \mathcal{J}_1$. Indeed, in that case $\alpha_j = 1/(1+r_j) \to 0$. From (20), there exists $z_j \in N_X(\bar{x}_j)$ such that

$$z_j = -\alpha_j c + (1-\alpha_j)\nabla\varphi(\bar{x}_j) \to \nabla\varphi(\bar{x}).$$

Closedness of the graph of N_X would then imply $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$, a contradiction. Hence $\varphi(\bar{x}_j) \ge p$ eventually along $\mathcal{J}$.

For these indices, set $\lambda_j := \frac{1-\alpha_j}{\alpha_j} \ge 0$. Since $N_X(\bar{x}_j)$ is a cone, dividing (20) by $\alpha_j > 0$ gives

$$0 \in c - \lambda_j\nabla\varphi(\bar{x}_j) + N_X(\bar{x}_j), \quad \lambda_j\big(p - \varphi(\bar{x}_j)\big) = 0. \tag{21}$$

The sequence $(\lambda_j)_{j\in\mathcal{J}}$ is bounded. Otherwise, along some infinite $\mathcal{J}_2 \subset \mathcal{J}$, $\lambda_j \to +\infty$. Dividing (21) by λ_j and using again that $N_X(\bar{x}_j)$ is a cone and has closed graph yields $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$, again a contradiction. Therefore, along a further subsequence, $\lambda_j \to \bar{\lambda} \ge 0$. Passing to the limit in (21) gives

$$0 \in c - \bar{\lambda}\nabla\varphi(\bar{x}) + N_X(\bar{x}), \quad \bar{\lambda}\big(p - \varphi(\bar{x})\big) = 0.$$

Since $\varphi(\bar{x}) = p$, these are precisely the KKT conditions.

Finally, suppose that $\varphi(\bar{x}) < p$. By continuity, $\varphi(\bar{x}_j) < p$ eventually along $\mathcal{J}$, so $\alpha_j = 1/(1+r_j) \to 0$. As above, writing

$$z_j = -\alpha_j c + (1-\alpha_j)\nabla\varphi(\bar{x}_j) \in N_X(\bar{x}_j)$$

and passing to the limit gives $0 \in -\nabla\varphi(\bar{x}) + N_X(\bar{x})$. Thus $\bar{x}$ is feasibility-stationary. □

We are now ready to prove the main convergence theorem for the Penalty Method.

D.3 Proof of Theorem 4

Proof. The penalty master objective is continuous on the compact set X, so problem (MP-Pen) admits a global minimizer x^{k+1}. Since $\psi^M_{r,\nu}(\hat{x}^\nu) = \psi_r(\hat{x}^\nu)$, comparison with $\hat{x}^\nu$ shows that this minimizer satisfies (6). Moreover, a global minimizer is stationary for the master problem. Thus, the required trial point exists, proving item *i)*.

If the algorithm terminates, then $\texttt{Tol} = 0$ implies $x^{k+1} = \hat{x}^\nu$. The second statement of Lemma 10 shows that the final stability center is penalty-critical. If the algorithm does not terminate, Proposition 12 shows that every cluster point of the stability centers is penalty-critical. Existence of a cluster point follows from compactness of X. This proves item *ii)*.

Proposition 5. □

References

[1] Laurent Andrieu, René Henrion, and Werner Römisch. A model for dynamic chance constraints in hydro power reservoir management. *European Journal of Operational Research*, 207(2):579–589, 2010. doi: 10.1016/j.ejor.2010.05.013.

[2] Giuseppe C. Calafiore and Marco C. Campi. The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753, 2006. doi: 10.1109/TAC.2006.875041.

[3] Ying Cui and Jong-Shi Pang. *Modern nonconvex nondifferentiable optimization*, volume 29 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021. ISBN 978-1-61197-673-1.

[4] I Deák. Computing probabilites of rectangles in case of multinormal distribution. *Journal of Statistical Computation and Simulation*, 26(1-2):101–114, 1986.

[5] István Deák. Subroutines for computing normal probabilities of sets—computer experiences. *Annals of Operations Research*, 100(1):103–122, 2000.

[6] P. Diaconis and M. Shahshahani. The subgroup algorithm for generating uniform random variables. *Prob. in Engineering and Info. Sci*, 1:15–32, 1987.

[7] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002. doi: 10.1007/s101070100263.

[8] Fabrizio Durante and Carlo Sempi. *Principles of copula theory*. CRC Press, Boca Raton, FL, 2016. doi: 10.1201/b18674.

[9] 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. doi: 10.1007/s00186-016-0564-y.

[10] Holger Heitsch. On probabilistic capacity maximization in a stationary gas network. *Optimization*, 69(3):575–604, 2020. doi: 10.1080/02331934.2019.1625353.

[11] R. Henrion. Structural properties of linear probabilistic constraints. *Optimization: A Journal of Mathematical Programming and Operations Research*, 56(4):425–440, August 2007.

[12] R. Henrion and A. Möller. A gradient formula for linear chance constraints under Gaussian distribution. *Mathematics of Operations Research*, 37:475–488, 2012. doi: 10.1287/moor.1120.0544.

[13] R. Henrion, G. Stadler, and F. Wechsung. Optimal control under uncertainty with joint chance state constraints: Almost-everywhere bounds, variance reduction, and application to (bi)linear elliptic PDEs. *SIAM Journal on Uncertainty Quantification*, 13(3):1028–1053, 2025. doi: 10.1137/24M171557X.

[14] René Henrion. Introduction to chance constraint programming. Tutorial paper for the Stochastic Programming Community, 2004. URL `https://www.wias-berlin.de/people/henrion/publikat.html`.

[15] Shinji Kataoka. A stochastic programming model. *Econometrica*, 31(1/2):181–196, 1963. doi: 10.2307/1910956.

[16] Tõnu Kollo and Dietrich von Rosen. *Advanced multivariate statistics with matrices*, volume 579 of *Mathematics and Its Applications*. Springer, Dordrecht, 1 edition, 2005. ISBN 978-1-4020-3418-3. doi: 10.1007/1-4020-3419-9.

[17] 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. doi: 10.1016/j.compchemeng.2007.05.009.

[18] James Luedtke and Shabbir Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008. doi: 10.1137/070702928.

[19] James Luedtke, Shabbir Ahmed, and George L. Nemhauser. An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming*, 122(2):247–272, 2010. doi: 10.1007/s10107-008-0247-4.

[20] Jorge J. Moré and Stefan M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1):172–191, 2009. doi: 10.1137/080724083.

[21] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2006. doi: 10.1137/050622328.

[22] Art B. Owen. Randomly permuted (t, m, s)-nets and (t, s)-sequences. In Harald Niederreiter and Peter Jau-Shyong Shiue, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 106 of *Lecture Notes in Statistics*, pages 299–317. Springer-Verlag, New York, 1995. doi: 10.1007/978-1-4612-2552-2_19.

[23] Bernardo K. Pagnoncelli, Shabbir Ahmed, and Alexander Shapiro. Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, 142(2):399–416, 2009. doi: 10.1007/s10957-009-9523-6.

[24] András Prékopa. On logarithmic concave measures and functions. *Acta Scientiarum Mathematicarum*, 34:335–343, 1973.

[25] I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. doi: 10.1016/0041-5553(67)90144-9.

[26] Ksenia Syrtseva, Welington de Oliveira, Sophie Demassey, Hugo Morais, Paul Javal, and Bhargav Swaminathan. Difference-of-convex approach to chance-constrained optimal power flow modelling the DSO power modulation lever for distribution networks. *Sustainable Energy, Grids and Networks*, 36:101168, 2023. doi: 10.1016/j.segan.2023.101168.

[27] H. W. Teng, M. H. Kang, and C. D. Fuh. On spherical Monte Carlo simulations for multivariate normal probabilities. *Applied Probability*, 47(3):817–836, 2015.

[28] W. van Ackooij and M. Minoux. A characterization of the subdifferential of singular Gaussian distribution functions. *Set Valued and Variational Analysis*, 23(3):465–483, 2015. doi: 10.1007/s11228-015-0317-8.

[29] W. van Ackooij, S. Demassey, P. Javal, H. Morais, W. de Oliveira, and B. Swaminathan. A bundle method for nonsmooth DC programming with application to chance-constrained problems. *Computational Optimization and Application*, 78(1):451–490, 2021. doi: 10.1007/s10589-020-00241-8.

[30] Wim van Ackooij and Welington de Oliveira. Level bundle methods for constrained convex optimization with various oracles. *Computational Optimization and Applications*, 57(3):555–597, 2014. doi: 10.1007/s10589-013-9610-3.

[31] 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. doi: 10.1137/130922689.

[32] 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. doi: 10.1137/16M1061308.

[33] Wim van Ackooij, René Henrion, Andris Möller, and Riadh Zorgati. Joint chance constrained programming for hydro reservoir management. *Optimization and Engineering*, 15(2):509–531, 2014. doi: 10.1007/s11081-013-9236-4.

[34] Wim van Ackooij, Ivana Aleksovska, and Miguel Muñoz-Zúñiga. (Sub-)differentiability of probability functions with elliptical distributions. *Set-Valued and Variational Analysis*, 26(4):887–910, 2018. doi: 10.1007/s11228-017-0454-3.

[35] Wim van Ackooij, Paul Javal, and Pedro Pérez-Aros. Derivatives of probability functions: Unions of polyhedra and elliptical distributions. *Set-Valued and Variational Analysis*, 30(2):487–519, 2022. doi: 10.1007/s11228-021-00598-w. First published online in 2021.

[36] Wim van Ackooij, Carolina Chiu, Welington de Oliveira, and Pedro Pérez-Aros. Lipschitz gradient guarantees for probability functions and a new algorithm for probability maximization. Preprint available at Optimization Online, 2026. URL `https://optimization-online.org/?p=34983`.

[37] Wim van Ackooij, Yi Chu, Welington de Oliveira, and Pedro Pérez-Aros. A proximal approach for nonsmooth composite-constrained optimization, 2026. URL `https://optimization-online.org/?p=36714`. Preprint available at Optimization Online.

[38] Wim Stefanus van Ackooij and Welington Luis de Oliveira. *Methods of nonsmooth optimization in stochastic programming: From conceptual algorithms to real-world applications*, volume 363 of *International Series in Operations Research & Management Science*. Springer, Cham, 2025. doi: 10.1007/978-3-031-84837-7.

[39] 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. doi: 10.1007/s10107-004-0559-y.