Optimization Online
(Preprint)

Robust Optimization Under Sparse Uncertainty

Abbas Khademi · **Maryam Daryalal**

Received: date / Accepted: date

Abstract Classical robust optimization relies on convex and bounded uncertainty sets, an assumption that is inadequate for sparse uncertainty, where only a small, unknown subset of parameters deviates from its nominal value. This sparsity makes the uncertainty set nonconvex and turns separation into a combinatorial problem, so standard duality-based reformulations do not apply. We study two settings. For a pure sparsity constraint, we introduce an adaptive gradient projection method with backtracking for smooth sparse optimization, establish stationarity of its accumulation points, and use its output in a cutting-plane method that provides valid lower bounds. When sparsity is combined with a compact convex set, we derive exact mixed-integer lifted formulations and continuous relaxations. For adjustable robust optimization with fixed recourse, dualization gives a joint separation problem over uncertainty and recourse multipliers, in which every feasible pair yields a valid cut and the resulting master problems provide monotone lower bounds. We also show that continuous sparse box uncertainty and discrete budgeted uncertainty have the same worst-case value for linear dependence on uncertainty. Computational studies across six applications demonstrate the framework's scalability and reveal how coupling constraints shape relaxation quality.

Keywords Robust optimization · sparse uncertainty · lifting techniques · projection-based relaxations · cutting-plane methods

Mathematics Subject Classification (2020) 90C26 · 65K05 · 49M37

1 Introduction

Uncertainty in optimization data can make a nominal solution infeasible or suboptimal once the true parameters are realized. Two paradigms address this challenge. Stochastic programming models the parameters as random variables and optimizes an expected objective over an assumed distribution [60], whereas robust optimization protects the solution against every realization in a given uncertainty set [13]. Robust optimization provides worst-case guarantees without assuming a distribution, but its tractability depends on the structure of the uncertainty set. We study robust optimization under *sparse* uncertainty, in which deviations from nominal data are confined to a small subset of parameters. We develop formulations and algorithms for the resulting combinatorial separation problems and nonconvex robust counterparts.

Robust optimization has found applications in a broad range of settings, including network control [31], portfolio selection [63], location-transportation [55], inventory management [51], scheduling [27], operations management [54], electricity generation [64], and election security [30]. Robust optimization is widely used in part because many uncertainty sets of practical interest admit tractable robust counterparts. The resulting deterministic reformulations can be solved by standard optimization methods [9, 39, 67]. For instance, in the standard linear setting, box uncertainty leads to linear programming reformulations, ellipsoidal uncertainty to second-order cone programs, and polyhedral uncertainty to linear robust counterparts. Classical robust optimization therefore focuses on convex, compact uncertainty sets [11, 12, 61].

Corresponding author: Abbas Khademi

A. Khademi · M. Daryalal
Department of Decision Sciences, HEC Montréal, 3000 Côte-Sainte-Catherine Road, Montréal, Québec H3T 2A7, Canada
E-mail: abbas.khademi@hec.ca; maryam.daryalal@hec.ca

Despite this tractability, such models may not capture sparse uncertainty, where only a bounded number of parameters with unknown identities may deviate from their nominal values. This occurs in control systems with a few sensor failures, financial monitoring with a small fraction of fraudulent transactions, supply chains with localized disruptions, and healthcare systems with a few atypical patients creating disproportionate demand. With sparse uncertainty, the robust counterpart must identify the worst subset of deviating parameters, so the inner maximization includes discrete support choices, loses the standard convex structure, and becomes a mixed-integer optimization problem. A closely related model is budgeted uncertainty [19, 20], where a budget parameter limits the number of coefficients that may deviate simultaneously. The continuous variant is defined over a convex polyhedral uncertainty set and therefore preserves tractability, while the discrete model restricts deviations to specified extreme values. Budgeted uncertainty has become a standard modelling paradigm with various extensions (e.g., see [2, 3, 21, 25, 28, 32, 37, 38, 49]). In another direction, Lu and Sturt [53] study sparsity in optimal decision rules for adjustable robust optimization under box uncertainty and show that under certain conditions policies are sparse. Their focus is sparsity of decision rules, whereas ours is sparsity of the uncertainty set through cardinality constraints on active deviations. Each active parameter is allowed to vary continuously within its feasible region, potentially subject to additional compact constraints. Section 3.1 establishes a further connection, in which we show that discrete budgeted uncertainty and continuous sparse box uncertainty yield the same worst-case value when uncertainty enters linearly. This equivalence does not necessarily hold for nonlinear dependence.

When robust optimization problems do not admit tractable dual reformulations, cutting-plane methods provide a general solution framework. These methods alternate between a restricted master problem and adversarial separation, adding each violated scenario as a constraint until convergence [57]. Variants of this approach have been developed for adjustable robust optimization, where dual-based reformulations are often nonconvex [44, 48, 66]. Nonconvex robust optimization under convex uncertainty sets has also been studied and solved using neighbourhood search, perspectification, and cutting-plane methods [15, 16, 18].

Our approach builds on the cutting-plane framework without assuming tractable separation problems. As discussed, the separation problem in our setting requires optimizing over sparse deviation patterns and is therefore combinatorial. To address this, we develop a unified framework combining lifting techniques with scenario reduction and deterministic approximation schemes, embedded within a cutting-plane algorithm that iteratively tightens the formulation. We further relate our construction to *coupled uncertainty*, where intersections of uncertainty sets are used to reduce conservatism [17]. Building on this idea, we introduce sparsity-coupled uncertainty by intersecting sparse deviation constraints with a compact convex set. This models both the sparsity and magnitude of disruptions.

Contributions. We make the following advances:

- We formulate robust optimization with cardinality-constrained deviations over a pure sparse set and its intersection with a compact convex set.
- For smooth sparse optimization, we develop a backtracking adaptive gradient projection method requiring no global smoothness constant as input. We prove stationarity of its accumulation points and use its output to generate scenarios for robust optimization problems.
- For separable right-hand-side uncertainty, we characterize and use the sparse adversarial problem's basic scenarios to build finite approximations with valid lower bounds. For quadratic dependence on uncertainty, we also derive an upper bound through semidefinite relaxation.
- For compact sparsity-coupled sets, we derive an exact mixed-integer lift with binary support variables and linear support-magnitude links, plus a continuous convex relaxation. We embed exact and relaxed separation and scenario projection in a cutting-plane algorithm whose master values form a nondecreasing sequence of valid lower bounds.
- We extend the method to adjustable robust optimization with fixed recourse. After dualizing recourse, joint separation over uncertainty and recourse multipliers uses alternating maximization. Each feasible pair yields a valid convex cut, producing nondecreasing lower bounds.
- We evaluate six applications: sparse signal recovery, sparse logistic regression, robust portfolio optimization, linear optimization with implementation error, and static and adjustable wireless downlink planning. Against alternative sparse optimization and separation methods, we compare both solution quality and computation time.

Section 2 formulates the sparse-uncertainty robust problem and develops the sparsity-constrained algorithms. Section 3 develops the sparsity-coupled formulations, a cutting-plane algorithm, and an

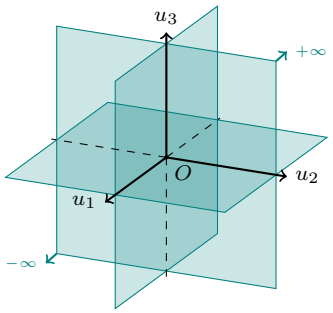

Fig. 1: Visualization of the sparsity set $\{u \in \mathbb{R}^3 : \|u\|_0 \le 2\}$. Each coloured plane represents a two-dimensional coordinate subspace S_I (e.g., the (u_1, u_2)-plane); the union of these planes together with all coordinate axes (and the origin) constitutes $\mathcal{U}$. The set is nonconvex: a midpoint of points on different planes can lie outside $\mathcal{U}$

adjustable model. Section 4 reports computational experiments, and Section 5 concludes. The appendices provide data and instance details and examples. Online Resource provides complete results.

Notation. Let $\mathbb{R}^n$ denote the n-dimensional Euclidean space. For a square matrix B, we use $\mathrm{tr}(B)$ for its trace and $\mathrm{vec}(B)$ for its vectorization. The notation $A \succeq B$ indicates that $A - B$ is positive semidefinite. We denote by I_n the $n \times n$ identity matrix, by $\mathbf{1}$ the vector of all ones, and by $\mathbf{e}_i$ the i-th standard basis vector (dimension inferred from context). The linear subspace spanned by the vectors $\mathbf{e}_i$ for indices $i \in I$ is written as $\mathrm{span}\{\mathbf{e}_i : \ i \in I\}$. For any positive integer n, we write $[n] := \{1, \dots, n\}$. For $u \in \mathbb{R}^n$, $\mathrm{supp}(u) := \{i \in [n] : \ u_i \neq 0\}$ is its support. For $u \in \mathbb{R}^n$ and $q \ge 1$, the ℓ_q-norm is defined as $\|u\|_q := \left(\sum_{i=1}^n |u_i|^q\right)^{1/q}$. The zero norm $\|\cdot\|_0$, which counts the number of nonzero elements in a vector, is defined as $\|u\|_0 := \#\{i : \ u_i \neq 0\}$. We use $\mathrm{sgn}(u)$ to denote the sign function. The set $\mathrm{conv}(\mathcal{C})$ denotes the convex hull of $\mathcal{C}$. A continuously differentiable function h is said to be L-smooth if its gradient ∇h is Lipschitz continuous with constant $L > 0$, i.e., for any $u, v \in \mathbb{R}^n$: $\|\nabla h(u) - \nabla h(v)\|_2 \le L\|u - v\|_2$. We define $\mathit{\Pi}_{\mathcal{C}}(v)$ as the orthogonal projection onto the set $\mathcal{C}$, i.e., $\mathit{\Pi}_{\mathcal{C}}(v) \in \mathrm{argmin}_{u \in \mathcal{C}} \|v - u\|_2^2$. For $\mathcal{C} \subseteq \mathbb{R}^n \times \mathbb{R}^p$, $\mathrm{Proj}_u(\mathcal{C}) := \{u \in \mathbb{R}^n : \ \exists\, w \in \mathbb{R}^p, \ (u, w) \in \mathcal{C}\}$ denotes its u-projection.

2 Sparsity-Constrained Uncertainty

We study robust optimization problems under sparse uncertainty:

$$\begin{aligned} &\min_{x \in \mathcal{X}} \ f_0(x) \\ &\text{s.t. } f_j(x, u) \le 0, \quad \forall u \in \mathcal{U}, \ j \in [m], \end{aligned} \tag{P}$$

where $f_0 : \mathbb{R}^{n_x} \to \mathbb{R}$ is the objective function, and $f_1, \dots, f_m : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ are continuous. For every fixed $x \in \mathcal{X}$ and $j \in [m]$, we assume that $\sup_{u \in \mathcal{U}} f_j(x, u)$ is finite. Whenever a result involves a worst-case realization, we additionally assume that this supremum is attained. The set $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ includes deterministic constraints on the decision variable x, and $u \in \mathcal{U}$ denotes the uncertain parameter vector. The uncertainty set $\mathcal{U}$ is

$$\mathcal{U} := \{u \in \mathbb{R}^{n_u} : \ \|u\|_0 \le s\},$$

where $s \in [n_u - 1]$ is the maximum number of nonzero components, typically $s \ll n_u$. Geometrically, $\mathcal{U}$ is a finite union of coordinate subspaces, that is, for each $I \subseteq [n_u]$ with $|I| \le s$, the subspace $S_I := \mathrm{span}\{\mathbf{e}_i : i \in I\}$ is contained in $\mathcal{U}$, and every $u \in \mathcal{U}$ lies in at least one such S_I. Consequently, $\mathcal{U}$ is closed, nonconvex, and unbounded, as sparsity limits how many entries can deviate, not how large their magnitudes can be. The set $\mathcal{U}$ is also star-shaped about the origin: if $u \in \mathcal{U}$ and $t \in [0, 1]$, then $tu \in \mathcal{U}$. For every $s \ge 1$, $\mathrm{conv}(\mathcal{U}) = \mathbb{R}^{n_u}$ because $\mathcal{U}$ contains every coordinate axis. Thus convexifying the pure sparse set alone gives no useful restriction. Although the zero norm is nonconvex, it is *star quasi-convex* at the origin because its sublevel sets are star-shaped [42, Section 5.3]. See Figure 1 for an illustration in $\mathbb{R}^3$. These geometric properties distinguish our formulation from classical robust optimization.

In many applications it is also natural to *couple* sparsity with range or size constraints. We therefore consider the variant $\mathcal{U}_s := \mathcal{U} \cap \mathcal{G}$, where $\mathcal{G} \subset \mathbb{R}^{n_u}$ is a compact convex set capturing the underlying structure of uncertainty patterns (e.g., an ℓ_∞ box or an ℓ_2 ball). Both $\mathcal{U}$ and $\mathcal{U}_s$ model the structural prior

that only a few parameters deviate from their nominal values; $\mathcal{U}_s$ can additionally ensure boundedness in magnitude, which can be useful for existence and stability of robust solutions. This perspective addresses a limitation of conventional robust optimization: standard models either (i) allow all parameters to deviate simultaneously within a convex, bounded set, or (ii) limit the aggregate magnitude of deviations via a norm budget (e.g., budgeted uncertainty sets). In contrast, we constrain the number of perturbed parameters via $\|u\|_0 \leq s$, which better reflects scenarios where only a few entries change. See Figure 2 for a comparison of a sparsity-coupled ℓ_∞ box and a budgeted uncertainty. This model can be less conservative when only a few uncertain parameters are expected to change at the same time.

(a) Sparsity: $\{u : \|u\|_\infty \leq \frac{2}{3},\ \|u\|_0 \leq 1\}$ (b) Budgeted: $\{u : \|u\|_\infty \leq \frac{2}{3},\ \|u\|_1 \leq 1\}$

Fig. 2: Bounded uncertainty sets in $\mathbb{R}^2$ with box radius $\frac{2}{3}$ and budget 1. (a) sparse support (at most one active component): "cross" or union of axes, constrains *which* coordinates may be nonzero. (b) budgeted deviation: octagon (box $\cap$ diamond) constrains *how much* total deviation is allowed

For instance, in adversarial machine learning, robustness is often studied under sparse feature perturbations. In image classification an attacker may alter at most s pixels to trigger misclassification. In [62], authors demonstrate that even modifications to just a single pixel can fool deep neural networks in up to 67.97% of cases in test datasets. Similarly, in sensor networks, malicious attacks may compromise a limited number of sensors, as demonstrated in [52], where adversarial attacks are constrained to compromising only specific meters or limited by the resources required to tamper with measurement devices. Even under these constraints, attackers could still systematically construct attack vectors to manipulate state estimation results.

The next proposition writes $\mathcal{U}$ as a finite union of convex sets.

Proposition 1 *Let $\mathcal{U} = \{u \in \mathbb{R}^{n_u} : \|u\|_0 \leq s\}$, where $s \in [n_u - 1]$. Then the set $\mathcal{U}$ can be expressed as the union of finitely many convex sets. Specifically, there exists a positive integer K and convex subsets $\{\mathcal{U}^i\}_{i=1}^K \subseteq \mathbb{R}^{n_u}$ such that $\mathcal{U} = \bigcup_{i=1}^K \mathcal{U}^i$. Consequently, the robust optimization problem* (P) *is equivalent to*

$$\begin{array}{ll} \min\limits_{x \in \mathcal{X}} & f_0(x) \\ \text{s.t.} & f_j(x,u) \leq 0, \quad \forall u \in \mathcal{U}^i,\ j \in [m],\ i \in [K]. \end{array} \tag{1}$$

Proof. Let $K := \binom{n_u}{s}$. For any index set $I \subseteq [n_u]$ with $|I| = s$, define the coordinate subspace $S_I := \text{span}\{\mathbf{e}_i : i \in I\} \subseteq \mathbb{R}^{n_u}$. Enumerate all such s-element subsets as $\{I_i\}_{i=1}^K$, and set $\mathcal{U}^i := S_{I_i}$. Each $\mathcal{U}^i$ is a convex, closed, unbounded linear subspace. We claim $\mathcal{U} = \bigcup_{i=1}^K \mathcal{U}^i$. If $u \in \mathcal{U}$ has support $\text{supp}(u)$ with $|\text{supp}(u)| \leq s$, then there exists an index set I of size s with $\text{supp}(u) \subseteq I$; thus $u \in S_I \subseteq \bigcup_{i=1}^K \mathcal{U}^i$. Conversely, if $u \in S_I$ for some $|I| = s$, then $\|u\|_0 \leq s$, so $u \in \mathcal{U}$. Finally, for fixed x and j, the requirement $f_j(x,u) \leq 0$ for all $u \in \mathcal{U}$ is equivalent to requiring it for all u in each $\mathcal{U}^i$, $i \in [K]$, i.e.,

$$f_j(x,u) \leq 0, \quad \forall u \in \mathcal{U} \iff \begin{cases} f_j(x,u) \leq 0, & \forall u \in \mathcal{U}^1 \\ \vdots & \vdots \\ f_j(x,u) \leq 0, & \forall u \in \mathcal{U}^K \end{cases}$$

This establishes the formulation in (1). □

While Proposition 1 establishes an exact reformulation of the robust problem, its practical use depends on the size of the uncertainty dimension. The number of convex sets grows combinatorially, with $K = \binom{n_u}{s} \sim \frac{n_u^s}{s!}$ when s is fixed and $n_u \to \infty$. This motivates alternatives to (1).

2.1 Right-Hand-Side Uncertainty

We next study the case where uncertainty enters additively on the right-hand-side, i.e., for $j \in [m]$,

$$f_j(x,u) = g_j(x) + h_j(u),$$

leading to the robust program

$$\begin{array}{l} \min\limits_{x\in\mathcal{X}} f_0(x) \\ \text{s.t. } g_j(x) + h_j(u) \le 0, \quad \forall u \in \mathcal{U},\ j \in [m]. \end{array} \tag{P-RHS}$$

By separability, the robust constraint is $g_j(x) + \sup_{u\in\mathcal{U}} h_j(u) \le 0$, where the supremum is independent of x. We assume each $h_j : \mathbb{R}^{n_u} \to \mathbb{R}$ is continuously differentiable, bounded above on $\mathcal{U}$, and attains its supremum there. For $s \ge 1$, an affine h_j that depends on u is unbounded above on $\mathcal{U}$, so we use the bounded sets of Section 3.

Definition 1 ([8]) A vector $\hat{u} \in \mathcal{U}$ is called a *basic scenario* for h if

1. $\|\hat{u}\|_0 < s \Rightarrow \nabla h(\hat{u}) = 0$,
2. $\|\hat{u}\|_0 = s \Rightarrow [\nabla h(\hat{u})]_i = 0, \quad \forall i \in \operatorname{supp}(\hat{u})$.

First-order optimality applies only along feasible infinitesimal directions. If $\|\hat{u}\|_0 < s$, every coordinate can be perturbed, so a local maximizer satisfies $\nabla h(\hat{u}) = 0$. If $\|\hat{u}\|_0 = s$, feasible small perturbations preserve the support, so stationarity is required only on that support. Every worst-case realization of (P-RHS) is a basic scenario, as shown next.

Lemma 2 *Assume each inner (adversarial) objective h_j in* (P-RHS) *is continuously differentiable. For any constraint j, let $u_j^* \in \operatorname{argmax}_{u\in\mathcal{U}} h_j(u)$ be a worst-case realization for* (P-RHS)*. Then u_j^* is a basic scenario.*

Proof. Because u_j^* maximizes h_j over $\mathcal{U}$, the standard sparsity optimality argument of [8] applies.

(i) If $\|u_j^*\|_0 < s$, there exists $\varepsilon_i > 0$ such that $u_j^* + t\mathbf{e}_i \in \mathcal{U}$ for all $t \in (-\varepsilon_i, \varepsilon_i)$. For sufficiently small t of either sign, optimality and differentiability give

$$\frac{d}{dt} h_j(u_j^* + t\mathbf{e}_i)\Big|_{t=0} = \langle \nabla h_j(u_j^*), \mathbf{e}_i \rangle = 0, \quad \forall i,$$

hence $\nabla h_j(u_j^*) = 0$.

(ii) For $\|u_j^*\|_0 = s$, let $d \in \mathbb{R}^{n_u}$ such that $d_i = 0$, for all $i \notin \operatorname{supp}(u_j^*)$. Choose

$$\varepsilon \in \left(0, \min\left\{ \tfrac{|(u_j^*)_i|}{|d_i|} :\ i \in \operatorname{supp}(u_j^*),\ d_i \ne 0 \right\}\right),$$

with the convention that the minimum over an empty set is $+\infty$. Then for all $t \in (-\varepsilon, \varepsilon)$ we have $u_j^* + td \in \mathcal{U}$. A nonzero inner product would contradict optimality for a sufficiently small signed t. Hence

$$\langle \nabla h_j(u_j^*), d \rangle = 0, \quad \forall d \in \left\{ d \in \mathbb{R}^{n_u} :\ \ d_i = 0,\ \forall i \notin \operatorname{supp}(u_j^*) \right\}.$$

Thus, in both cases u_j^* satisfies the defining conditions of a basic scenario. □

A partial converse of Lemma 2 holds under concavity. Dropping the index j, consider a generic adversarial objective h. If h is concave and $\|u^*\|_0 < s$, then for all $u \in \mathcal{U}$, the gradient inequality implies

$$h(u) \le h(u^*) + \nabla h(u^*)^\top (u - u^*).$$

By Definition 1, $\nabla h(u^*) = 0$, hence $h(u) \le h(u^*)$ for all $u \in \mathcal{U}$. Thus any basic scenario with $\|u^*\|_0 < s$ is a global maximizer when h is concave. The case $\|u^*\|_0 = s$ generally requires additional conditions to guarantee sufficiency. As a concrete counterexample for a strictly concave function, let $s = 1$, $h(u_1, u_2) = -(u_1 - 1)^2 - u_2^2 + 3u_2$. Then $u^* = (1,0)^\top$ has $|\operatorname{supp}(u^*)| = 1$ and $[\nabla h(u^*)]_1 = 0$, so it is a basic scenario. However, $h(0, 1.5) = 1.25 > h(1,0) = 0$, so u^* is not globally optimal on $\mathcal{U}$.

The next result gives a condition ensuring finitely many basic scenarios.

Proposition 3 *Let $h : \mathbb{R}^{n_u} \to \mathbb{R}$ be continuously differentiable and strictly concave, and fix $s \in [n_u - 1]$. Define the set of basic scenarios for sparsity s by*

$$\mathcal{B}_s(h) := \big\{u \in \mathbb{R}^{n_u} : \ \|u\|_0 < s, \ \nabla h(u) = 0\big\} \cup \bigcup_{\substack{S \subseteq [n_u], \\ |S| = s}} \big\{u \in \mathbb{R}^{n_u} : \ \|u\|_0 = s, \ [\nabla h(u)]_i = 0, \ u_i \neq 0 \ \forall i \in S\big\}.$$

Then $\mathcal{B}_s(h)$ is finite. More specifically, there is at most one element with $\|u\|_0 < s$ and at most one element per support S with $|S| = s$, so

$$|\mathcal{B}_s(h)| \ \leq \ 1 + \binom{n_u}{s}.$$

Proof. Strict concavity of h is preserved under restriction to any linear subspace of $\mathbb{R}^{n_u}$. We consider the two types of basic scenarios.
Case I: $\big\{u \in \mathbb{R}^{n_u} : \ \|u\|_0 < s, \ \nabla h(u) = 0\big\}$. Over $\mathbb{R}^{n_u}$, a continuously differentiable and strictly concave function has at most one stationary point, i.e., with $\nabla h(u) = 0$. Thus, Case I contains at most one point.
Case II: $\big\{u \in \mathbb{R}^{n_u} : \ \|u\|_0 = s, \ [\nabla h(u)]_i = 0, \ u_i \neq 0 \ \forall i \in S\big\}$ for fixed S. Fix a support $S \subseteq [n_u]$ with $|S| = s$. Let $S^c := [n_u] \setminus S$ and define the subspace

$$V_S := \{u \in \mathbb{R}^{n_u} : u_{S^c} = 0\}.$$

Define the restriction of h over subspace V_S as $g_S := h|_{V_S}$ with $g_S(\bar{u}) = h(\bar{u}, 0_{S^c})$ for $\bar{u} \in \mathbb{R}^S$. Then, for $u = (\bar{u}, 0_{S^c}) \in V_S$ and each $i \in S$,

$$[\nabla g_S(\bar{u})]_i = \lim_{t \to 0} \frac{g_S(\bar{u} + t\mathbf{e}_i) - g_S(\bar{u})}{t} = \lim_{t \to 0} \frac{h(u + t\mathbf{e}_i) - h(u)}{t} = [\nabla h(u)]_i.$$

Therefore, the condition $[\nabla h(u)]_i = 0$ for all $i \in S$ is equivalent to the restricted function satisfying $\nabla g_S(\bar{u}) = 0$. Because h is strictly concave on $\mathbb{R}^{n_u}$, g_S is strictly concave on V_S. Consequently, g_S has at most one stationary point. Let us call this unique stationary point u_S^*, if it exists. For this point u_S^* to be a basic scenario belonging to the support S in Case II, it must have exact sparsity s (i.e., $[u_S^*]_i \neq 0$ for all $i \in S$). If all components of u_S^* in S are nonzero, it contributes exactly one basic scenario to Case II. If u_S^* has a zero component in S, then its support is strictly smaller than s. Such a point belongs to Case I only when its full gradient is zero. Otherwise, it is not a basic scenario and contributes no point for the support S in Case II. Consequently, for each support S of size s, there is at most one s-sparse basic scenario. As there are exactly $\binom{n_u}{s}$ possible supports S with $|S| = s$, the union over all such supports yields at most $\binom{n_u}{s}$ basic scenarios. Combining both cases, the total number of basic scenarios is bounded by

$$|\mathcal{B}_s(h)| \ \leq \ 1 + \binom{n_u}{s}. \qquad \square$$

A finite $\mathcal{W}^j \subseteq \mathcal{U}$ relaxes the j-th robust constraint and gives a lower bound. Adding scenarios tightens this bound, and exactness holds once $\mathcal{W}^j$ contains a maximizer of h_j over $\mathcal{U}$.

Theorem 4 *Consider the robust optimization problem* (P-RHS) *with sparsity level $s \in [n_u - 1]$. Assume that for each constraint $j \in [m]$, the inner (adversarial) objective $h_j : \mathbb{R}^{n_u} \to \mathbb{R}$ is continuously differentiable, strictly concave, and that the supremum of h_j over $\mathcal{U}$ is attained. Let $\mathcal{W}^j \subseteq \mathcal{U}$ denote the set of all basic scenarios of h_j at sparsity level s. Then, $\mathcal{W}^j$ is a finite set, and the original robust constraints are exactly equivalent to enforcing the constraints over $\mathcal{W}^j$. Equivalently,* (P-RHS) *has the finite deterministic counterpart below*

$$\begin{array}{ll} \min\limits_{x \in \mathcal{X}} & f_0(x) \\ \text{s.t.} & g_j(x) + h_j(u) \leq 0, \quad \forall u \in \mathcal{W}^j, \ j \in [m]. \end{array} \tag{2}$$

Proof. Under the assumptions of the theorem, for each $j \in [m]$, the supremum $\sup_{u \in \mathcal{U}} h_j(u)$ is attained at some worst-case realization $u_j^* \in \operatorname{argmax}_{u \in \mathcal{U}} h_j(u)$. By Lemma 2, any such maximizer u_j^* is a basic scenario. Therefore, $u_j^* \in \mathcal{W}^j$:

$$\max_{u \in \mathcal{U}} h_j(u) = h_j(u_j^*) \leq \max_{u \in \mathcal{W}^j} h_j(u).$$

Since $\mathcal{W}^j \subseteq \mathcal{U}$ by definition, $\max_{u\in\mathcal{W}^j} h_j(u) \le \max_{u\in\mathcal{U}} h_j(u)$. Thus,

$$\max_{u\in\mathcal{U}} h_j(u) = \max_{u\in\mathcal{W}^j} h_j(u), \qquad \forall j \in [m].$$

Since Proposition 3 ensures each set $\mathcal{W}^j$ is finite, this equality confirms that the robust problem (P-RHS) is exactly equivalent to the finite deterministic counterpart (2). □

Suppose $h(u) = u^\top Q u + 2b^\top u$ with $Q \in \mathbb{R}^{n_u\times n_u}$ symmetric negative definite, so h is strictly concave. For a sparsity level s and any support $S \subseteq [n_u]$ with $|S| = s$, stationarity on the subspace $\{u : u_{S^c} = 0\}$ gives

$$\big[\nabla h(u)\big]_S = 2Q_{SS}u_S + 2b_S = 0,$$

so the unique candidate on support S is

$$u^S = \big(u_S^S, 0_{S^c}\big), \qquad u_S^S = -Q_{SS}^{-1}b_S,$$

where Q_{SS} and b_S are the principal submatrix and subvector indexed by S. Since $Q \prec 0$, Q_{SS} is also negative definite and invertible. The corresponding objective value is available in closed form

$$h(u^S) = -b_S^\top Q_{SS}^{-1} b_S.$$

Basic scenarios with $\|u\|_0 = s$ are those u^S for which $(u_S^S)_i \neq 0$ for all $i \in S$. If the unique stationary point $u^\circ := -Q^{-1}b$ happens to satisfy $\|u^\circ\|_0 < s$, then u° is also a basic scenario and should be included. Example 1 in Appendix A illustrates this case.

2.2 An Adaptive Algorithm for Sparse Optimization

To avoid enumerating all $\binom{n_u}{s}$ supports, we approximately solve the sparse adversarial problem below, which arises from the right-hand-side model (P-RHS):

$$\max\{h(u) : u \in \mathbb{R}^{n_u},\ \|u\|_0 \le s\}, \tag{3}$$

with $h : \mathbb{R}^{n_u} \to \mathbb{R}$ continuously differentiable, L-smooth and bounded above on $\mathcal{U} = \{u : \|u\|_0 \le s\}$. Problem (3) is nonconvex. Projected gradient ascent provides a feasible sparse point at each iteration, but it does not certify a global maximizer. For any $z \in \mathbb{R}^{n_u}$, let $T_s(z)$ contain the indices of the s largest values of $|z_i|$, with smaller indices selected when magnitudes are equal. The orthogonal projection onto the sparsity set $\mathcal{U}$ is the operator $\Pi_{\mathcal{U}}(z) = H_s(z)$:

$$\big[H_s(z)\big]_i := \begin{cases} z_i, & i \in T_s(z) \\ 0, & i \notin T_s(z) \end{cases}.$$

Then $H_s(z) \in \mathcal{U}$ and $H_s(z) \in \operatorname{argmin}_{u\in\mathcal{U}} \|u - z\|_2^2$. This costs $O(n_u \log n_u)$ via sorting or $O(n_u)$ with a selection routine. The Adaptive Gradient Projection (AGP) scheme in Algorithm 1 applies this projection after each gradient step. Our algorithm adapts the projected gradient framework introduced in [8], the so-called iterative hard thresholding. However, inspired by the adaptive strategy in [45], we do not rely on a global smoothness constant L to fix the stepsize. Instead, we estimate the local curvature at each iteration via a regularized gradient-to-variable ratio and an adaptive backtracking search. The smoothness constant L is not an input; the numerical parameters used by the procedure are listed in Algorithm 1.

At each iteration of Algorithm 1, we take a gradient-ascent step to increase h, then project back onto the sparsity set by keeping the s largest-magnitude entries. A backtracking rule adaptively selects the stepsize as $1/L_k$, where L_k is the local smoothness constant. Step 1 checks that the denominator is positive before forming the ratio. The parameter $\delta > 0$ then regularizes the curvature estimate from below. In Step 2, the factor $\gamma_k \in [\underline{\gamma}, 1]$ reduces the local smoothness estimate in order to allow a potentially larger stepsize. In Step 3, the next scenario is computed and projected onto the sparsity set. If the stepsize is too aggressive, Step 4 applies backtracking by increasing L_k geometrically with factor $\beta > 1$ until a sufficient local ascent condition is satisfied. By applying Descent Lemma [6, Theorem 18.15] to $-h$, if ∇h is L-smooth, then for any $L_k \ge L$ the iterates u^k and u^{k+1} satisfy

$$h(u^{k+1}) \ge h(u^k) + \nabla h(u^k)^\top (u^{k+1} - u^k) - \tfrac{L_k}{2}\|u^{k+1} - u^k\|_2^2. \tag{5}$$

Starting from its scaled estimate, Step 4 increases L_k geometrically until (4) holds. Consequently, finite backtracking terminates with

$$\underline{\gamma}\delta \le L_k \le \beta(L + \delta).$$

Algorithm 1 Adaptive Gradient Projection Algorithm

Input: An initial point $u^0 \in \mathcal{U}$, an auxiliary point $u^{-1} \in \mathbb{R}^{n_u}$ with $u^{-1} \neq u^0$, constants $\beta > 1$, $\delta > 0$, $\epsilon > 0$, a lower scaling bound $\underline{\gamma} \in (0, 1)$, a sequence $\{\gamma_k\}_{k\geq 0} \subseteq [\underline{\gamma}, 1]$, and a positive integer iteration limit $K_{\max}$.
Initialization: Set iteration counter $k \leftarrow 0$.
General Steps: Execute the following steps $(k = 0, 1, \cdots)$

(Step 1) If $\|u^k - u^{k-1}\|_2 \leq \epsilon$, stop at $u^{\text{end}} = u^k$; otherwise estimate L_k by

$$L_k \leftarrow \frac{\left\|\nabla h(u^k) - \nabla h(u^{k-1})\right\|_2}{\left\|u^k - u^{k-1}\right\|_2} + \delta.$$

(Step 2) Scale the local smoothness

$$L_k \leftarrow \gamma_k L_k.$$

(Step 3) Project the estimated next scenario

$$v^k \leftarrow u^k + \tfrac{1}{L_k}\nabla h(u^k), \qquad u^{k+1} \in \Pi_{\mathcal{U}}(v^k).$$

(Step 4) While the sufficient increase condition is not satisfied, i.e.,

$$h(u^{k+1}) < h(u^k) + \nabla h(u^k)^\top (u^{k+1} - u^k) - \tfrac{L_k}{2}\|u^{k+1} - u^k\|_2^2. \tag{4}$$

Do:

$$L_k \leftarrow \beta L_k, \qquad v^k \leftarrow u^k + \tfrac{1}{L_k}\nabla h(u^k), \qquad u^{k+1} \in \Pi_{\mathcal{U}}(v^k).$$

(Step 5) Stop if $k \geq K_{\max}$ or $\|u^{k+1} - u^k\|_2 \leq \epsilon$; otherwise set $k \leftarrow k + 1$.

Output: The final scenario $u^{\text{end}} \leftarrow u^{k+1}$.

The accepted value may be smaller than L when the curvature along $[u^k, u^{k+1}]$ is below the global smoothness bound. The convergence analysis uses the sufficient-increase condition and the uniform bounds on L_k. It therefore permits a dynamically selected sequence γ_k provided that $\gamma_k \in [\underline{\gamma}, 1]$ for all k.

By the definition of the orthogonal projection operator $\Pi_{\mathcal{U}}(\cdot)$, we have

$$u^{k+1} \in \Pi_{\mathcal{U}}\left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right) = \operatorname*{argmin}_{u\in\mathcal{U}} \left\{ \left\| u - \left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right)\right\|_2^2 \right\}.$$

Define the quadratic model at u^k

$$H(u) := h(u^k) + \nabla h(u^k)^\top (u - u^k) - \tfrac{L_k}{2}\|u - u^k\|_2^2.$$

By completing the square, we can rewrite $H(u)$ as

$$H(u) = -\tfrac{L_k}{2}\left\| u - \left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right)\right\|_2^2 + \underbrace{h(u^k) + \tfrac{1}{2L_k}\left\|\nabla h(u^k)\right\|_2^2}_{\text{constant with respect to } u}.$$

In maximizing $H(u)$, we can disregard the u-independent term:

$$\begin{aligned}
\operatorname*{argmax}_{u\in\mathcal{U}} H(u) &= \operatorname*{argmax}_{u\in\mathcal{U}} \left\{ -\frac{L_k}{2}\left\| u - \left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right)\right\|_2^2 \right\} \\
&= \operatorname*{argmin}_{u\in\mathcal{U}} \left\{ \frac{L_k}{2}\left\| u - \left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right)\right\|_2^2 \right\} \\
&= \operatorname*{argmin}_{u\in\mathcal{U}} \left\{ \left\| u - \left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right)\right\|_2^2 \right\}.
\end{aligned}$$

Therefore

$$u^{k+1} \in \Pi_{\mathcal{U}}\left(u^k + \tfrac{1}{L_k}\nabla h(u^k)\right) \iff u^{k+1} \in \operatorname*{argmax}_{u\in\mathcal{U}} H(u). \tag{6}$$

We refer to Example 2 in the appendix to illustrate how Algorithm 1 executes on a numerical instance. The sufficient increase check (4) yields the following convergence result for Algorithm 1.

Proposition 5 *Suppose $h : \mathbb{R}^{n_u} \to \mathbb{R}$ is continuously differentiable, L-smooth and bounded above on $\mathcal{U}$. Let $\{u^k\}_{k\geq 0}$ be the sequence generated by Algorithm 1. Then the sequence $\{h(u^k)\}_{k\geq 0}$ is nondecreasing and convergent.*

Proof. From equivalence (6), u^{k+1} maximizes $H(u)$ over the set $\mathcal{U}$. Since $u^k \in \mathcal{U}$, we have $H(u^{k+1}) \geq H(u^k)$. The definition of H gives $H(u^k) = h(u^k)$. The backtracking check (4) gives exactly $h(u^{k+1}) \geq H(u^{k+1})$. Thus,

$$h(u^{k+1}) \geq H(u^{k+1}) \geq H(u^k) = h(u^k).$$

Since h is bounded above on $\mathcal{U}$ and $\{h(u^k)\}_{k\geq 0}$ is nondecreasing, the sequence converges to some $\bar{h} \leq \sup_{u\in\mathcal{U}} h(u)$. □

The next theorem shows that coincident successive iterates yield a basic scenario; otherwise, every accumulation point of Algorithm 1 is a basic scenario.

Theorem 6 *Let $h : \mathbb{R}^{n_u} \to \mathbb{R}$ be L-smooth. Suppose that the upper level set*

$$\mathcal{L} := \{u \in \mathcal{U} : h(u) \geq h(u^0)\}$$

is bounded. Let $\{u^k\}_{k\geq 0}$ be the sequence of iterates generated by Algorithm 1 with $\epsilon = 0$ and $K_{\max} = +\infty$. If two successive iterates coincide, the returned point satisfies the projection relation below for the corresponding accepted value and is a basic scenario. Otherwise, the sequence $\{u^k\}_{k\geq 0}$ is bounded and possesses at least one accumulation point. Every accumulation point u^* obeys the following projection relation:

$$u^* \in \Pi_{\mathcal{U}}\left(u^* + \frac{1}{L^*}\nabla h(u^*)\right)$$

It holds for some $L^* > 0$, and the point is a basic scenario for problem (3).

Proof. If $u^{k+1} = u^k$, the accepted projection at iteration k gives

$$u^k \in \Pi_{\mathcal{U}}\left(u^k + \frac{1}{L_k}\nabla h(u^k)\right),$$

which is the stated stationarity relation. Now suppose the run is infinite. By Proposition 5, the sequence of objective values $\{h(u^k)\}_{k\geq 0}$ is nondecreasing. Therefore, $h(u^k) \geq h(u^0)$ for all $k \geq 0$. Since the projection step in Algorithm 1 ensures that $u^k \in \mathcal{U}$ for all k, every iterate u^k belongs to the upper level set $\mathcal{L}$. Because $\mathcal{L}$ is bounded, the sequence $\{u^k\}_{k\geq 0}$ is bounded. By the Bolzano-Weierstrass theorem [24, Theorem 3.26], the bounded sequence $\{u^k\}_{k\geq 0}$ possesses at least one accumulation point, denoted by u^*. Consequently, there exists a subsequence of indices $\mathcal{K}^1$ such that $u^k \to u^*$ as $k \to \infty, k \in \mathcal{K}^1$. Because the sequence of successive iterates $\{u^{k+1}\}_{k\in\mathcal{K}^1}$ is also bounded, there exists a further subsequence $\mathcal{K}^2 \subseteq \mathcal{K}^1$ such that $u^{k+1} \to w^*$ as $k \to \infty, k \in \mathcal{K}^2$, for some $w^* \in \mathcal{U}$. Furthermore, the algorithm ensures that the sequence of adaptive local smoothness parameters $\{L_k\}_{k\geq 0}$ is uniformly bounded within the compact interval $[\underline{\gamma}\delta, \beta(L + \delta)]$.

Since the subsequence $\{L_k\}_{k\in\mathcal{K}^2}$ is bounded, there exists a further subsequence $\mathcal{K}^3 \subseteq \mathcal{K}^2$ such that $L_k \to L^*$ as $k \to \infty, k \in \mathcal{K}^3$, for some $L^* \in [\underline{\gamma}\delta, \beta(L + \delta)]$. For notational convenience, we relabel the indices in $\mathcal{K}^3$ as $\{k_j\}_{j\geq 0}$. Along this subsequence,

$$u^{k_j} \to u^*, \qquad u^{k_j+1} \to w^*, \qquad L_{k_j} \to L^* > 0.$$

Let $H(u) := h(u^k) + \nabla h(u^k)^\top(u - u^k) - \frac{L_k}{2}\|u - u^k\|_2^2$. By Step 3 and (6), $u^{k+1} \in \operatorname{argmax}_{u\in\mathcal{U}} H(u)$. Since $u^k \in \mathcal{U}$, we have $H(u^{k+1}) \geq H(u^k) = h(u^k)$:

$$\nabla h(u^k)^\top(u^{k+1} - u^k) - \frac{L_k}{2}\|u^{k+1} - u^k\|_2^2 \geq 0. \tag{7}$$

The backtracking condition in Step 4 guarantees

$$h(u^{k+1}) - h(u^k) \geq \nabla h(u^k)^\top(u^{k+1} - u^k) - \frac{L_k}{2}\|u^{k+1} - u^k\|_2^2. \tag{8}$$

Because $\{h(u^k)\}_{k\geq 0}$ is nondecreasing and bounded above, it converges. Hence,

$$h(u^{k_j+1}) - h(u^{k_j}) \to 0.$$

Taking limits in (7) and (8), and using the continuity of ∇h, gives

$$\nabla h(u^*)^\top(w^* - u^*) - \frac{L^*}{2}\|w^* - u^*\|_2^2 = 0. \tag{9}$$

For each j, let H_{k_j} denote the quadratic model H at iteration k_j, and define

$$H_*(u) := h(u^*) + \nabla h(u^*)^\top (u - u^*) - \frac{L^*}{2}\|u - u^*\|_2^2.$$

For all $u \in \mathcal{U}$, model optimality gives $H_{k_j}(u^{k_j+1}) \geq H_{k_j}(u)$, and by taking limits $H_*(w^*) \geq H_*(u)$. Equation (9) gives $H_*(w^*) = H_*(u^*)$. Hence, u^* maximizes H_* over $\mathcal{U}$. Completing the square gives

$$u^* \in \Pi_{\mathcal{U}}\left(u^* + \frac{1}{L^*}\nabla h(u^*)\right). \tag{10}$$

Finally, we verify that u^* satisfies the conditions of a basic scenario in Definition 1. Let $S = \mathrm{supp}(u^*)$. Because u^* minimizes the distance to $u^* + \frac{1}{L^*}\nabla h(u^*)$ over $\mathcal{U}$, its nonzero components must match the corresponding components of $u^* + \frac{1}{L^*}\nabla h(u^*)$. Thus,

$$u_i^* = u_i^* + \frac{1}{L^*}[\nabla h(u^*)]_i, \qquad i \in S,$$

which gives $[\nabla h(u^*)]_i = 0$ for every $i \in S$. If $\|u^*\|_0 = s$, this satisfies the second condition in Definition 1. If $\|u^*\|_0 < s$, then u^* is an s-sparse projection using fewer than s nonzero entries, so the sth largest absolute component of $u^* + \frac{1}{L^*}\nabla h(u^*)$ must be zero. Therefore,

$$u_j^* + \frac{1}{L^*}[\nabla h(u^*)]_j = 0, \qquad j \in S^c.$$

Since $u_j^* = 0$ for $j \in S^c$, we obtain $[\nabla h(u^*)]_j = 0$ for every $j \in S^c$. Combined with the result on S, this gives $\nabla h(u^*) = 0$, which completes the proof. □

The relation (10) is known in the literature as L-stationarity, a standard necessary optimality condition for sparse optimization [8, 46]. For a robust constraint j, Algorithm 1 approximately solves $\max\{h_j(u) : \|u\|_0 \leq s\}$ and returns a candidate scenario u^{end}. This scenario can generate cutting planes in a scenario-based outer loop or warm-start enumeration over promising supports. A separate valid bound, developed next, yields a global optimality gap.

2.3 Convexifying Sparse Uncertainty by Lifting the Space

We begin with two classical norm inequalities used to construct a convex lifted outer approximation of sparse uncertainty.

Lemma 7 *For any $u \in \mathbb{R}^{n_u}$, the following properties hold.*

1. $\|u\|_1 \leq \|u\|_\infty \|u\|_0$.
2. $\|u\|_1 \leq \|u\|_2 \sqrt{\|u\|_0}$.

Proof. Let $t := \|u\|_0$. Denote its nonzero entries by $u_1, \ldots, u_t$.

$$\|u\|_1 = \sum_{i=1}^{t} |u_i| \leq \sum_{i=1}^{t} \|u\|_\infty = \|u\|_\infty t = \|u\|_\infty \|u\|_0,$$

which proves inequality 1. For inequality 2, note that

$$\begin{aligned}\|u\|_1 &= \sum_{i=1}^{n_u} |u_i| = \sum_{i=1}^{n_u} \mathrm{sgn}(u_i) u_i = u^\top \mathrm{sgn}(u) \\ &\leq \|u\|_2 \|\mathrm{sgn}(u)\|_2 = \|u\|_2 \sqrt{t} = \|u\|_2 \sqrt{\|u\|_0},\end{aligned}$$

derived using the Cauchy-Schwarz inequality. □

We seek a convex lift containing $\mathcal{U}$. Lemma 7 gives

$$\|u\|_1^2 \leq s\|u\|_2^2, \qquad u \in \mathcal{U}. \tag{11}$$

This inequality is not convex in u. We introduce a matrix variable U intended to represent $uu^\top$. For rank-one lifting $U := uu^\top$, observe that

$$\|\,\mathrm{vec}(U)\|_1 = \sum_{i,j=1}^{n_u} |u_i u_j| = \Big(\sum_{i=1}^{n_u} |u_i|\Big)^2 = \|u\|_1^2 \quad \text{and} \quad \mathrm{tr}(U) = \|u\|_2^2.$$

Using inequality (11), for any u with $\|u\|_0 \le s$,

$$\|\operatorname{vec}(U)\|_1 = \|u\|_1^2 \le s\|u\|_2^2 = s\operatorname{tr}(U). \tag{12}$$

Therefore, inequality (12) is a convex necessary condition that every lifted s-sparse pair $(u, uu^\top)$ satisfies. Now consider the lifted space $\{(u,U) : U = uu^\top,\ \|\operatorname{vec}(U)\|_1 \le s\operatorname{tr}(U)\}$. The bilinear equality $U = uu^\top$ is nonconvex. We relax it by $U \succeq uu^\top$, which is equivalent to the linear matrix inequality $\left(\begin{smallmatrix} U & u \\ u^\top & 1 \end{smallmatrix}\right) \succeq 0$ using a Schur complement [26]. The resulting lifted relaxation is

$$\mathcal{U}_{\mathrm{SDP}} := \left\{(u,U) :\ \left(\begin{smallmatrix} U & u \\ u^\top & 1 \end{smallmatrix}\right) \succeq 0,\ \|\operatorname{vec}(U)\|_1 \le s\operatorname{tr}(U)\right\}. \tag{13}$$

Consequently, for every $u \in \mathcal{U}$, the pair $(u, uu^\top) \in \mathcal{U}_{\mathrm{SDP}}$, and the projection $\mathrm{Proj}_u(\mathcal{U}_{\mathrm{SDP}})$ contains $\mathcal{U}$. Replacing an uncertainty set by a larger convex set expands the feasible region of the inner adversarial problem. Because $\mathcal{U} \subseteq \mathrm{Proj}_u(\mathcal{U}_{\mathrm{SDP}})$, the relaxation gives an upper bound on the worst-case value.

The lift alone provides no new information when $s > 1$ unless the function is lifted as well. For any $u \in \mathbb{R}^{n_u}$ and $t \ge 0$, set $U = uu^\top + tI_{n_u}$. Then $U - uu^\top = tI_{n_u} \succeq 0$, while $\|\operatorname{vec}(U)\|_1 = \|u\|_1^2 + n_u t$ and $\operatorname{tr}(U) = \|u\|_2^2 + n_u t$. For sufficiently large t, the inequality $\|u\|_1^2 + n_u t \le s(\|u\|_2^2 + n_u t)$ holds. Thus $\mathrm{Proj}_u(\mathcal{U}_{\mathrm{SDP}}) = \mathbb{R}^{n_u}$, and maximizing an unlifted function $h(u)$ over the lifted set is the same as maximizing it over $\mathbb{R}^{n_u}$. A finite lifted bound must therefore use the algebraic structure of the function. For example, when the uncertain function is quadratic, its quadratic term can be represented linearly in the lifted matrix.

For the right-hand-side model (P-RHS), when the uncertain function is quadratic, $h_j(u) = u^\top Q_j u + 2b_j^\top u$, we exploit its structure to lift the function alongside the feasible set. Specifically, we lift the quadratic term by replacing $u^\top Q_j u$ with $\operatorname{tr}(Q_j U)$, which gives a linear objective in the lifted variables. Thus, we define the true worst-case value and its lifted SDP upper bound as

$$\begin{aligned} \phi_j &:= \sup_{u \in \mathcal{U}} \left\{u^\top Q_j u + 2b_j^\top u\right\}, \\ \Phi_j^{\mathrm{SDP}} &:= \sup_{(u,U) \in \mathcal{U}_{\mathrm{SDP}}} \left\{\operatorname{tr}(Q_j U) + 2b_j^\top u\right\} \ge \phi_j. \end{aligned}$$

Here Φ_j^{SDP} is interpreted in the extended-real sense and may equal $+\infty$; for example, $Q_j \prec 0$ is sufficient for finiteness. Then the robust feasible set $\{x \in \mathcal{X} :\ g_j(x) + \phi_j \le 0,\ j \in [m]\}$ contains $\{x \in \mathcal{X} : g_j(x) + \Phi_j^{\mathrm{SDP}} \le 0,\ j \in [m]\}$, and for a minimization objective, the lifted formulation results in a valid upper bound on the optimal value of the original robust program, which is nontrivial when the SDP values are finite; see Example 3 in the appendix.

In the next section we focus on bounded sparsity-coupled uncertainty sets, for which nontrivial convex outer approximations by lifting are possible.

3 Sparsity-Coupled Uncertainty

We now study problem (P) under uncertainty sets obtained by intersecting sparsity constraints with a compact convex set $\mathcal{G} \subseteq \mathbb{R}^{n_u}$ that captures magnitude or structural information about the admissible deviations:

$$\mathcal{U}_s := \{u \in \mathbb{R}^{n_u} :\ \|u\|_0 \le s,\ u \in \mathcal{G}\},$$

and assume $\mathcal{U}_s \neq \emptyset$. Under the standing assumptions of Section 2, each $f_j(x,\cdot)$ is continuous and, for fixed x, bounded above on $\mathcal{U}_s$. Because $\{u : \|u\|_0 \le s\}$ is closed and $\mathcal{G}$ is compact, their intersection is compact, so the worst-case value $\sup_{u \in \mathcal{U}_s} f_j(x,u)$ is attained by the Weierstrass theorem [7, Theorem 2.30]. Verifying that a given vector lies in $\mathcal{U}_s$ is straightforward whenever membership in $\mathcal{G}$ can be tested efficiently, as it suffices to check $u \in \mathcal{G}$ and count the nonzeros of u. The adversarial problem $\max\{f_j(x,u) : u \in \mathcal{U}_s\}$, by contrast, is generally intractable. This difficulty already arises in cardinality-constrained quadratic optimization, for which mixed-integer methods are generally required [23]. Exact separation methods are therefore unavailable for the model class in general.

The following proposition shows that, using n_u auxiliary binary variables, the set $\mathcal{U}_s$ admits an exact lifted description with linear coupling constraints.

Proposition 8 *There exist vectors $\alpha, \beta \in \mathbb{R}^{n_u}$ such that $\mathcal{U}_s = \mathrm{Proj}_u(\mathcal{Z}_s)$, where*

$$\mathcal{Z}_s := \left\{ \begin{pmatrix} u \\ z \end{pmatrix} :\ \begin{array}{l} u \in \mathcal{G},\ \mathbf{1}^\top z \le s,\ z \in \{0,1\}^{n_u}, \\ \alpha_i z_i \le u_i \le \beta_i z_i,\ i \in [n_u] \end{array} \right\}. \tag{14}$$

Proof. Because $\mathcal{G}$ is compact, the componentwise extreme values

$$\alpha_i := \min_{u\in\mathcal{G}} u_i \quad \text{and} \quad \beta_i := \max_{u\in\mathcal{G}} u_i, \quad i \in [n_u],$$

are finite and attained, and every $u \in \mathcal{G}$ satisfies $\alpha \le u \le \beta$. To lift the uncertainty set $\mathcal{U}_s$, for $u \in \mathcal{U}_s$ we define auxiliary binary variables $z \in \{0,1\}^{n_u}$ where $z_i = 1$ if $u_i \neq 0$ and $z_i = 0$ otherwise. Then $\mathbf{1}^\top z = \|u\|_0 \le s$. For each index with $z_i = 1$ we have $\alpha_i \le u_i \le \beta_i$ since $u \in \mathcal{G}$, while for $z_i = 0$ the bounds reduce to $u_i = 0$, which holds by construction. Together with $u \in \mathcal{G}$ this shows $(u, z) \in \mathcal{Z}_s$. Conversely, let $(u, z) \in \mathcal{Z}_s$. The bounds $\alpha_i z_i \le u_i \le \beta_i z_i$ force $u_i = 0$ if $z_i = 0$, so $\|u\|_0 \le \mathbf{1}^\top z \le s$. With $u \in \mathcal{G}$, $u \in \mathcal{U}_s$. □

The lifted set $\mathcal{Z}_s$ is exact but mixed-integer. Relaxing z gives the following convex outer approximation for scenario generation:

$$\widehat{\mathcal{Z}}_s := \left\{ \begin{pmatrix} u \\ z \end{pmatrix} : \begin{array}{l} u \in \mathcal{G},\ \mathbf{1}^\top z \le s,\ z \in [0,1]^{n_u}, \\ \alpha_i z_i \le u_i \le \beta_i z_i,\ i \in [n_u] \end{array} \right\}.$$

Since every point of $\mathcal{Z}_s$ is feasible for $\widehat{\mathcal{Z}}_s$, we have $\mathcal{Z}_s \subseteq \widehat{\mathcal{Z}}_s$ and hence $\mathrm{conv}(\mathcal{Z}_s) \subseteq \widehat{\mathcal{Z}}_s$. The inclusion is generally strict, so $\widehat{\mathcal{Z}}_s$ is an outer approximation rather than the convex hull. The binary support constraints use the standard indicator formulation of a cardinality restriction. Their role here is to give exact separation and a common continuous relaxation for the nonlinear robust constraints in Section 3.2.

Proposition 8 yields two cases. In both, u is a deviation from a nominal value, so we assume the admissible region contains the origin.

Corollary 9 *The following are exact lifted formulations for two common uncertainty sets.*

1. *For the box uncertainty set with $\alpha \le 0 \le \beta$,*

$$\mathcal{U}_{\text{Box}} := \{u \in \mathbb{R}^{n_u} :\ \|u\|_0 \le s,\ \alpha \le u \le \beta\},$$

the lifted set is

$$\mathcal{Z}_{\text{Box}} := \left\{ \begin{pmatrix} u \\ z \end{pmatrix} : \begin{array}{l} \mathbf{1}^\top z \le s,\ z \in \{0,1\}^{n_u}, \\ \alpha_i z_i \le u_i \le \beta_i z_i,\ i \in [n_u] \end{array} \right\}.$$

2. *For the simplex uncertainty set*

$$\mathcal{U}_{\text{Simplex}} := \{u \in \mathbb{R}^{n_u} :\ \mathbf{1}^\top u \le 1,\ u \ge 0,\ \|u\|_0 \le s\},$$

the lifted set is

$$\mathcal{Z}_{\text{Simplex}} := \left\{ \begin{pmatrix} u \\ z \end{pmatrix} : \begin{array}{l} \mathbf{1}^\top z \le s,\ \mathbf{1}^\top u \le 1, \\ 0 \le u \le z,\ z \in \{0,1\}^{n_u} \end{array} \right\}.$$

For the box, the assumption $\alpha \le 0 \le \beta$ guarantees that zeroing a coordinate keeps the point inside the box. For the simplex we take $\alpha = 0$ and $\beta = \mathbf{1}$.

We treat sparsity combined with an ℓ_2 ball, which does not fit the box template but admits a convex relaxation through the norm inequalities used in Section 2.3. Consider

$$\mathcal{U}_{\text{Ball}} := \{u :\ \|u\|_0 \le s,\ \|u\|_2 \le 1\}.$$

For any $u \in \mathcal{U}_{\text{Ball}}$, the second inequality of Lemma 7 gives $\|u\|_1 \le \|u\|_2\sqrt{\|u\|_0} \le \sqrt{s}$. Hence $\mathcal{U}_{\text{Ball}}$ is contained in the convex set

$$\widehat{\mathcal{U}}_{\text{Ball}} := \{u :\ \|u\|_1 \le \sqrt{s},\ \|u\|_2 \le 1\}.$$

The convex hull of s-sparse vectors in the unit ℓ_2 ball is the s-support-norm ball, strictly contained in $\{u :\ \|u\|_1 \le \sqrt{s},\ \|u\|_2 \le 1\}$ [4]. Therefore, $\widehat{\mathcal{U}}_{\text{Ball}}$ coincides with the convex hull only when $s = 1$ or $s = n_u$, and is otherwise a strict relaxation

$$\mathcal{U}_{\text{Ball}} \ \subseteq\ \mathrm{conv}(\mathcal{U}_{\text{Ball}}) \ \subseteq\ \widehat{\mathcal{U}}_{\text{Ball}}.$$

Example 4 in the appendix gives a numerical illustration of the resulting upper bound. When each $f_j(x,\cdot)$ is concave in u, this relaxation is a standard convex program. Using the lifting in Section 2.3, define $U = uu^\top$ and relax it to $U \succeq uu^\top$ to obtain the following alternative convex description of $\widehat{\mathcal{U}}_{\text{Ball}}$:

$$\mathcal{S}_{\text{Ball}} := \left\{ (u, U) :\ \begin{pmatrix} U & u \\ u^\top & 1 \end{pmatrix} \succeq 0,\ \|\mathrm{vec}(U)\|_1 \le s,\ \mathrm{tr}(U) \le 1 \right\}.$$

The next proposition identifies the u-projection of $\mathcal{S}_{\text{Ball}}$ with $\widehat{\mathcal{U}}_{\text{Ball}}$.

Proposition 10 $\mathrm{Proj}_u(\mathcal{S}_{\mathrm{Ball}}) = \widehat{\mathcal{U}}_{\mathrm{Ball}}$.

Proof. Let $(u, U) \in \mathcal{S}_{\mathrm{Ball}}$. By the Schur complement, the linear matrix inequality is equivalent to $U - uu^\top \succeq 0$. Taking traces,

$$0 \le \mathrm{tr}(U - uu^\top) = \mathrm{tr}(U) - \mathrm{tr}(uu^\top) \implies \|u\|_2^2 = \mathrm{tr}(uu^\top) \le \mathrm{tr}(U) \le 1,$$

so $\|u\|_2 \le 1$. Since $U - uu^\top \succeq 0$,

$$\begin{aligned} 0 &\le \mathrm{sgn}(u)^\top (U - uu^\top)\mathrm{sgn}(u) \\ &= \sum_{i,j=1}^{n_u} \mathrm{sgn}(u_i)\mathrm{sgn}(u_j)U_{ij} - \left(\mathrm{sgn}(u)^\top u\right)^2 \\ &\le \|\,\mathrm{vec}(U)\|_1 - \|u\|_1^2, \end{aligned}$$

hence $\|u\|_1^2 \le \|\,\mathrm{vec}(U)\|_1 \le s$, i.e., $\|u\|_1 \le \sqrt{s}$. Thus $u \in \widehat{\mathcal{U}}_{\mathrm{Ball}}$. Conversely, let $u \in \widehat{\mathcal{U}}_{\mathrm{Ball}}$ and set $U := uu^\top$. Then $U - uu^\top = 0 \succeq 0$, $\mathrm{tr}(U) = \|u\|_2^2 \le 1$, and $\|\,\mathrm{vec}(U)\|_1 = \|u\|_1^2 \le s$, so $(u, U) \in \mathcal{S}_{\mathrm{Ball}}$. Therefore $u \in \mathrm{Proj}_u(\mathcal{S}_{\mathrm{Ball}})$. □

3.1 Relation to Discrete Budgeted Uncertainty

The discrete budgeted uncertainty model in [19, 20] was designed for linear (combinatorial) problems with interval cost data, and it is among the first frameworks to control the number of deviating coefficients rather than only their aggregate magnitude. It is defined for an uncertain linear objective $u^\top x$ in which each coefficient u_i is confined to a known interval $[c_i, d_i]$ with $d \ge c$ componentwise, and at most Γ coefficients may reach their upper bounds at once, where Γ is a fixed positive integer:

$$\mathcal{B}_\Gamma = \left\{ u \in \mathbb{R}^{n_u} : \exists\, \delta \in \{0,1\}^{n_u} \text{ s.t. } u_i = c_i + (d_i - c_i)\delta_i,\ \sum_{i=1}^{n_u} \delta_i \le \Gamma \right\}. \tag{15}$$

The associated robust problem is

$$\min_{x \in \mathcal{X}} \left\{ \max_{u \in \mathcal{B}_\Gamma}\ u^\top x \right\}. \tag{16}$$

For a linear objective, the inner worst-case value in (16) is attained at a binary extreme point of $\mathcal{B}_\Gamma$ and can be found by sorting or linear programming duality. When $\mathcal{X} \subseteq \{0,1\}^{n_x}$, the robust combinatorial problem remains in the nominal complexity class [20, 59], which explains why (16) is usually studied for binary decisions. Our framework permits nonbinary $\mathcal{X}$ and nonlinear dependence on u.

We decompose the cost vector u into a nominal part c and a deviation ξ, with $u = c + \xi$. We introduce an epigraph variable τ. Then (16) becomes

$$\min_{x \in \mathcal{X},\, \tau} \{\tau : (c + \xi)^\top x \le \tau,\ \forall \xi \in \mathcal{U}_\Gamma\},$$

where the deviations ξ range over the sparsity-coupled set

$$\mathcal{U}_\Gamma = \{\xi \in \mathbb{R}^{n_u} : 0 \le \xi \le d - c,\ \|\xi\|_0 \le \Gamma\}. \tag{17}$$

This $\mathcal{U}_\Gamma$ is the box uncertainty set of Corollary 9 with $\alpha = 0$, $\beta = d - c$, and sparsity level Γ. Note that the sets $\mathcal{B}_\Gamma$ and $\mathcal{U}_\Gamma$ are not identical, as $\mathcal{B}_\Gamma$ restricts deviations to the endpoints of the intervals, whereas $\mathcal{U}_\Gamma$ permits continuous deviations. The next proposition shows that (15) and (17) have the same worst-case value for every linear objective.

Proposition 11 *For every $x \in \mathbb{R}^{n_u}$,*

$$\max_{u \in \mathcal{B}_\Gamma}\ u^\top x = \max_{\xi \in \mathcal{U}_\Gamma}\ (c + \xi)^\top x. \tag{18}$$

Proof. To show the first inequality, $\max_{u \in \mathcal{B}_\Gamma} u^\top x \le \max_{\xi \in \mathcal{U}_\Gamma} (c+\xi)^\top x$, take any $u \in \mathcal{B}_\Gamma$ with associated $\delta \in \{0,1\}^{n_u}$, $\sum_i \delta_i \le \Gamma$, and $u_i = c_i + (d_i - c_i)\delta_i$. Set $\xi_i := (d_i - c_i)\delta_i$. Since $\delta_i \in \{0,1\}$ and $d \ge c$, we have $\xi_i \in \{0, d_i - c_i\} \subseteq [0, d_i - c_i]$, so $0 \le \xi \le d - c$, and $\|\xi\|_0 \le \sum_i \delta_i \le \Gamma$, hence $\xi \in \mathcal{U}_\Gamma$. Therefore, every $u \in \mathcal{B}_\Gamma$ maps to a feasible $\xi \in \mathcal{U}_\Gamma$ with $u^\top x = (c + \xi)^\top x$ and the inequality follows.

Conversely, to establish the reverse inequality, $\max_{\xi \in \mathcal{U}_\Gamma} (c + \xi)^\top x \le \max_{u \in \mathcal{B}_\Gamma} u^\top x$, fix any $\bar{\xi} \in \mathcal{U}_\Gamma$ and let $\bar{S} = \mathrm{supp}(\bar{\xi})$, so $|\bar{S}| \le \Gamma$. Define $\xi_i^* := d_i - c_i$ if $i \in \bar{S}$ and $x_i > 0$, and $\xi_i^* := 0$ otherwise. Then $\xi_i^* \in \{0, d_i - c_i\}$ and $\mathrm{supp}(\xi^*) \subseteq \bar{S}$, so $\|\xi^*\|_0 \le |\bar{S}| \le \Gamma$, giving $u^* := c + \xi^* \in \mathcal{B}_\Gamma$. We check $\bar{\xi}_i x_i \le \xi_i^* x_i$:

if $i \in \bar{S}$ and $x_i > 0$, then $0 \le \bar{\xi}_i \le d_i - c_i = \xi_i^*$; if $i \in \bar{S}$ and $x_i \le 0$, then $\bar{\xi}_i \ge 0$ gives $\bar{\xi}_i x_i \le 0 = \xi_i^* x_i$; and if $i \notin \bar{S}$, then $\bar{\xi}_i = 0 = \xi_i^*$. Therefore

$$(c + \bar{\xi})^\top x \le (u^*)^\top x \le \max_{u \in \mathcal{B}_\Gamma} u^\top x.$$

Since $\bar{\xi}$ was selected arbitrarily, the inequality follows. □

Establishing the equivalence in Proposition 11 requires both the separable interval bounds and the linearity of the objective function in the uncertain parameters u. Thus the discrete endpoint set and the continuous sparse box have the same support function in every direction, although they are different sets. Beyond this linear interval setting, the sparsity-coupled construction permits a general compact convex $\mathcal{G}$ and nonlinear concave dependence on u, for which Proposition 11 generally does not apply.

Example 5 in the appendix numerically shows that, once the objective is nonlinear, the budgeted set and the sparsity-coupled set are no longer interchangeable. For reference, it uses the standard scaled budget set, $\{u : \|u\|_\infty \le r,\ \|u\|_1 \le \Gamma\}$, with budget Γ and component cap r [14, 47].

3.2 Reducing the Uncertainty Set

We now develop a scenario-based bounding scheme for the general model (P) under the sparsity-coupled uncertainty set with $\mathcal{G}$ compact and convex. We assume throughout that $\mathcal{G}$ contains the origin and is closed under coordinate zeroing, i.e., if $u \in \mathcal{G}$ and u' is obtained from u by setting some entries to zero, then $u' \in \mathcal{G}$ (see Remark 1 for three standard families that satisfy this). Given that evaluating the worst-case value $\max\{f_j(x,u) : u \in \mathcal{U}_s\}$ exactly is NP-hard, we instead generate a finite set of admissible scenarios and enforce the robust constraints only on that set.

Remark 1 The assumption that $\mathcal{G}$ contains the origin and is closed under coordinate zeroing is satisfied by several standard families of sets.

1. Downward-monotone sets. A set $\mathcal{G} \subseteq \mathbb{R}^{n_u}_+$ is downward monotone if $0 \le z \le u$ and $u \in \mathcal{G}$ imply $z \in \mathcal{G}$. Zeroing a coordinate of $u \ge 0$ produces a point z with $0 \le z \le u$, hence $z \in \mathcal{G}$, and taking $z = 0$ shows $0 \in \mathcal{G}$. Examples include the simplex $\{u \ge 0 : \mathbf{1}^\top u \le 1\}$, nonnegative boxes $\prod_i [0, \beta_i]$, and any sublevel set $\{u \ge 0 : \rho(u) \le c\}$ of a function ρ that is nondecreasing in the componentwise order.
2. Coordinate-symmetric sets. A set $\mathcal{G} \subseteq \mathbb{R}^{n_u}$ is coordinate-symmetric if $u \in \mathcal{G}$ and $|z_i| \le |u_i|$ for all i imply $z \in \mathcal{G}$. Zeroing a coordinate only decreases its absolute value, so the result stays in $\mathcal{G}$, and taking $z = 0$ shows $0 \in \mathcal{G}$. Examples include all ℓ_p-balls $\{u : \|u\|_p \le r\}$ for $p \in [1, \infty]$.
3. Boxes containing the origin. For a box $\mathcal{G} = \prod_i [\alpha_i, \beta_i]$ with $\alpha \le 0 \le \beta$, every coordinate interval contains 0, so setting any entry of $u \in \mathcal{G}$ to zero keeps it in $\mathcal{G}$, and $0 \in \mathcal{G}$ as well. Such a box need not be downward monotone or coordinate-symmetric, yet it still satisfies the closure property for this reason.

For each $j \in [m]$ at incumbent x, the first step solves

$$\tilde{u}^j \in \operatorname{argmax}\{f_j(x,u) : u \in \mathcal{G}\}, \tag{19}$$

which is tractable whenever $f_j(x,\cdot)$ is concave in u. The maximizer $\tilde{u}^j$ is not necessarily sparse, so we project it onto the sparsity set using the hard thresholding operator of Section 2.2, $u^j \in \Pi_{\mathcal{U}}(\tilde{u}^j) = H_s(\tilde{u}^j)$, which keeps the s entries of $\tilde{u}^j$ of largest absolute value and discards the rest. Since $\tilde{u}^j \in \mathcal{G}$ and $\mathcal{G}$ is closed under coordinate zeroing, the projected vector satisfies $u^j \in \mathcal{G}$ and $\|u^j\|_0 \le s$, hence $u^j \in \mathcal{U}_s$. Collecting these scenarios in a set $S \subseteq \mathcal{U}_s$ and imposing the constraints only on S gives the relaxation

$$\begin{aligned} &\min_{x \in \mathcal{X}} f_0(x) \\ &\text{s.t. } f_j(x, u^i) \le 0, \quad j \in [m],\ i \in [|S|], \end{aligned} \tag{20}$$

whose optimal value is a valid lower bound on the robust optimum because $|S|$ is finite and $S \subseteq \mathcal{U}_s$. We refer to this as the *projection-reduction* scheme. In the separable right-hand-side case $f_j(x,u) = g_j(x) + h_j(u)$ of (P-RHS), the construction simplifies, as the term $g_j(x)$ does not affect separation. So equation (19) reduces to $\tilde{u}^j \in \operatorname{argmax}\{h_j(u) : u \in \mathcal{G}\}$ and the generated scenarios are independent of x. The master (20) then takes the form $g_j(x) + h_j(u^i) \le 0$, and a single round of scenario generation already produces all the cuts needed. In this separable setting, Algorithm 1 can be applied independently to each h_j to identify a candidate support, yielding a terminal point $u^{j,\mathrm{end}}$. To generate the scenario,

one can then bypass Steps 1 and 2 of the general algorithm and instead solve the maximization problem restricted to that support

$$\hat{u}^j \in \operatorname{argmax}\{h_j(u) : u \in \mathcal{G},\ \operatorname{supp}(u) \subseteq \operatorname{supp}(u^{j,\mathrm{end}})\}.$$

Algorithm 2 embeds this scenario generation in a cutting-plane loop that alternates between identifying adversarial scenarios (Steps 1-2) and re-solving the scenario master problem (Step 4). Following the lifting idea of Section 2.3, scenarios can alternatively be drawn from the lifted description of $\mathcal{U}_s$. Algorithm 2 therefore solves its adversarial subproblem over a generic set $\mathcal{A}$, to be chosen as $\mathcal{G}$, as the lifted set $\mathcal{Z}_s$ of Proposition 8, or the relaxation $\widehat{\mathcal{Z}}_s$.

Algorithm 2 Cutting-Plane Algorithm

Input: $x^{(0)} \in \mathcal{X}$, $\epsilon > 0$; adversarial set $\mathcal{A} \in \{\mathcal{G},\ \mathcal{Z}_s,\ \widehat{\mathcal{Z}}_s\}$.
Initialization: Set iteration counter $\ell \leftarrow 0$, and initialize $S = \emptyset$.
Repeat: Execute the following steps:

(Step 1) Using the current incumbent $x^{(\ell)}$, compute for every $j \in [m]$

$$\tilde{u}^{j,(\ell)} \in \operatorname{argmax}\{f_j(x^{(\ell)}, u) : u \in \mathcal{A}\}.$$

(Step 2) For every $j \in [m]$, if $\mathcal{A} = \mathcal{Z}_s$, set $u^{j,(\ell)} = \tilde{u}^{j,(\ell)}$; otherwise set $u^{j,(\ell)} \in H_s\left(\tilde{u}^{j,(\ell)}\right)$.
(Step 3) Update the scenario set

$$S = S \cup \bigcup_{j \in [m]} \{u^{j,(\ell)}\}.$$

(Step 4) Solve the following optimization problem to find x^*

$$\begin{aligned} &\min_{x \in \mathcal{X}}\ f_0(x) \\ &\text{s.t.}\ \ f_j(x, u^k) \le 0,\ j \in [m],\ k \in [|S|]. \end{aligned}$$

Update iteration counter $\ell \leftarrow \ell + 1$, set $x^{(\ell)} \leftarrow x^*$ and $\tau^{(\ell)} \leftarrow f_0(x^*)$.

Until: A stopping criterion is satisfied, e.g., $|\tau^{(\ell)} - \tau^{(\ell-1)}| \le \epsilon$.
Output: $x^{(\mathrm{end})}$, $\tau^{(\mathrm{end})}$.

Step 1 admits three natural implementations, corresponding to the three choices of the adversarial set $\mathcal{A}$ and differing only in how the sparsity constraint is handled during scenario generation. The *Scenario Projection* (SP) variant takes $\mathcal{A} = \mathcal{G}$ and enforces sparsity through the hard thresholding projection H_s in Step 2. The *Exact Lifted* (EL) variant takes $\mathcal{A} = \mathcal{Z}_s$ with binary z, so the generated scenario is already s-sparse and the projection in Step 2 reduces to the identity. This is the most expensive option, as it requires a mixed-integer optimization at each iteration. The *Relaxed Lifted* (RL) variant takes $\mathcal{A} = \widehat{\mathcal{Z}}_s$, with z relaxed to $[0,1]^{n_u}$, keeping every subproblem convex and restoring sparsity by the projection in Step 2. We evaluate all three variants, and report their differences in solution quality and computation time in Section 4.2.

Potential stopping criteria for Algorithm 2 include a sufficiently small change in the objective value (e.g., $|\tau^{(\ell)} - \tau^{(\ell-1)}| \le \epsilon$), a small maximum constraint violation with respect to the generated scenarios (i.e., $\max_{j \in [m]} f_j(x^{(\ell)}, u^{j,(\ell)}) \le \epsilon$), and reaching a predefined maximum number of iterations. When the adversarial problems in Step 1 are solved globally, the resulting maximum constraint violation is exact, and a value at most ϵ certifies robust feasibility within ϵ. When they are solved approximately, this criterion applies only to the scenarios returned. The final solution is then optimized over the accumulated scenarios, while robust feasibility over all of $\mathcal{U}_s$ requires global separation.

Proposition 12 *At each iteration ℓ of Algorithm 2, the optimal value $\tau^{(\ell)}$ obtained in Step 4 is a lower bound on the optimal value of the robust problem* (P) *with uncertainty set $\mathcal{U}_s$.*

Proof. At iteration ℓ the scenario set S is a finite subset of $\mathcal{U}_s$, since every scenario produced in Steps 1-2 lies in $\mathcal{U}_s$ under the standing closure assumption on $\mathcal{G}$. Step 4 enforces the constraints only over S, which relaxes the semi-infinite constraint set $\{x \in \mathcal{X} :\ f_j(x,u) \le 0,\ \forall u \in \mathcal{U}_s\}$. The feasible region of Step 4 therefore contains that of the robust problem, so its optimal value $\tau^{(\ell)}$ cannot exceed the true robust optimum. □

Corollary 13 *The sequence $\{\tau^{(\ell)}\}_{\ell \ge 0}$ generated by Algorithm 2 is nondecreasing.*

3.3 Adjustable Robust Optimization under Sparse Uncertainty

Adjustable robust optimization (ARO) partitions decisions into first-stage choices and recourse that adapts after uncertainty is observed, reducing the conservatism of fully static decisions. Even linear adjustable models are generally NP-hard [10]; see [65] for a survey. We extend Algorithm 2 to the adjustable setting, yielding the following ARO model

$$\begin{aligned} \min_{x\in\mathcal{X}} \max_{u\in\mathcal{U}_s} \min_{y\in\mathbb{R}^{n_y}} \quad & f_0(x,u,y) \\ \text{s.t.} \quad & f_j(x,u,y) \le 0, \quad j\in[m], \end{aligned} \tag{21}$$

where $x \in \mathcal{X}$ is the first-stage decision vector, $y \in \mathbb{R}^{n_y}$ the recourse actions, and $u \in \mathcal{U}_s$ the uncertain parameters. Each f_j, for $j = 0, \ldots, m$, is assumed to be convex in both x and y, and concave in u.

Adapting Algorithm 2 to the ARO problem relies on dualizing the recourse. For an ARO problem with separable fixed recourse, let $f_j(x,u,y) = g_j(x,u) + p_j(y)$ for all $j = 0,1,\ldots,m$. Assuming g_j is convex in x and concave in u, and that p_j is convex in y, the authors in [44] derive an exact dual reformulation under a mild Slater condition. Applying Fenchel duality to the recourse variables y turns the inner minimization into a maximization over an augmented set that combines the original uncertainty u with the dual multipliers λ of the recourse constraints. For a fixed static decision x, the resulting separation problem has the form

$$\Psi(x) = \max_{u\in\mathcal{U}_s,\ \lambda\in\Lambda} \psi_x(u,\lambda), \tag{22}$$

where ψ_x is biconcave with a disjoint coupling between the uncertainty block u and the multiplier block λ. The multiplier set Λ is a convex set that collects the domain restrictions on the dual variables from the conjugates of the recourse functions p_j of the Fenchel dual. We refer to [44] for the explicit form of ψ_x and Λ. Such biconcave problems can be addressed by alternating maximization over the two blocks. Under the Slater condition, strong duality holds for the inner recourse, so for every static decision x the optimal value of (22) equals the worst-case recourse value, and problem (21) is equivalent to $\min_{x\in\mathcal{X}} \Psi(x)$.

Consequently, to solve problem (21) satisfying the stated assumptions, in Step 1 of Algorithm 2, instead of the per-constraint maximization $\max_{u\in\mathcal{A}} f_j(x^{(\ell)},u)$, we solve the separation problem (22) for the current incumbent $x^{(\ell)}$ by alternating maximization. With λ fixed, maximizing $\psi_{x^{(\ell)}}$ over $u \in \mathcal{A}$ is the sparse adversarial problem of Section 2.2, solved by projection or through the lifted sets $\mathcal{Z}_s$ and $\widehat{\mathcal{Z}}_s$. With u fixed, maximizing $\psi_{x^{(\ell)}}$ over $\lambda \in \Lambda$ is a convex program. Step 1 now returns a pair (u^k, λ^k), and Steps 2-3 accumulate the projected scenario u^k (together with its multiplier λ^k) in the scenario set S. Because $\psi_x(u^k,\lambda^k)$ is convex in x for each fixed pair (u^k,λ^k), each scenario adds a single convex *dual cut*, and Step 4 solves the following master problem, which has the dual form presented in [44]

$$\begin{aligned} \min_{x\in\mathcal{X},\,\tau} \quad & \tau \\ \text{s.t.} \quad & \psi_x(u^k,\lambda^k) \le \tau, \quad k\in[|S|], \end{aligned} \tag{23}$$

with the incumbent objective updated as $\tau^{(\ell)} \leftarrow \tau^*$.

Every cut in (23) is generated at a pair (u^k,λ^k) feasible for (22), so $\psi_x(u^k,\lambda^k) \le \Psi(x)$ for every x. Therefore (23) is a relaxation of $\min_{x\in\mathcal{X}} \Psi(x)$. Its optimum $\tau^{(\ell)}$ is a valid lower bound on the adjustable robust optimum of (21), and the bounds are nondecreasing in $|S|$, similar to Proposition 12 and Corollary 13. If recourse decisions are required for the sampled scenarios, separate variables y^k and the corresponding primal constraints may be added. This can strengthen the master without invalidating its lower bound.

Remark 2 The guarantee is one-sided: alternating maximization does not necessarily solve the joint nonconcave separation problem (22) globally, so robust feasibility and global optimality are not certified, even when EL solves the sparse u-block globally.

4 Computational Experiments

In this section, we evaluate the two algorithms developed for robust optimization under sparse uncertainty, in terms of both solution quality and computational efficiency. Section 4.1 examines the Adaptive Gradient Projection (AGP) algorithm, Algorithm 1, on two applications of sparsity-constrained problems: sparse signal recovery and sparse logistic regression. Section 4.2 examines the cutting-plane Algorithm 2 using four robust optimization problems with sparse uncertainty, including portfolio optimization, linear optimization with implementation error, and static and adjustable wireless downlink planning under jamming attacks.

Table 1: Mean solution gaps and computation times for sparse signal recovery.

	AGP		IHT		PSS	
Group (n, m, s)	Time (s)	SGap (%)	Time (s)	SGap (%)	Time (s)	SGap (%)
#1 (30000,100,5)	**0.28 [0.12]**	**13.67 [25.62]**	7.50 [0.22]	74.11 [15.41]	0.45 [0.08]	28.72 [35.71]
#2 (30000,100,10)	**0.32 [0.03]**	**3.22 [7.21]**	7.47 [0.16]	82.88 [6.51]	1.22 [0.29]	36.86 [22.26]
#3 (30000,100,20)	**0.35 [0.09]**	**0.00 [0.00]**	7.08 [0.16]	85.96 [3.40]	4.62 [0.44]	37.45 [16.60]
#4 (50000,100,5)	**0.53 [0.08]**	**22.11 [43.78]**	13.49 [0.34]	90.38 [12.74]	1.04 [0.17]	27.74 [41.71]
#5 (50000,100,10)	**0.62 [0.07]**	**8.27 [18.49]**	13.50 [0.13]	88.32 [2.91]	2.35 [0.58]	24.79 [22.66]
#6 (50000,100,20)	**0.86 [0.28]**	**4.55 [10.18]**	12.73 [0.36]	91.96 [5.73]	7.38 [0.75]	33.64 [21.94]
#7 (30000,200,5)	**0.33 [0.04]**	**0.00 [0.00]**	12.17 [0.38]	99.99 [0.01]	0.70 [0.16]	20.00 [44.72]
#8 (30000,200,10)	**0.47 [0.14]**	**8.74 [12.95]**	12.50 [0.45]	86.42 [17.21]	2.45 [1.12]	49.63 [50.00]
#9 (30000,200,20)	**0.66 [0.18]**	**0.00 [0.00]**	12.17 [0.19]	72.90 [6.32]	4.86 [1.10]	43.25 [8.42]
#10 (50000,200,5)	**0.89 [0.09]**	**20.00 [44.72]**	23.40 [0.17]	100.00 [0.00]	1.18 [0.09]	**20.00 [44.72]**
#11 (50000,200,10)	**1.29 [0.20]**	**0.00 [0.00]**	23.35 [0.11]	93.68 [9.87]	3.95 [0.79]	68.16 [43.60]
#12 (50000,200,20)	**1.47 [0.22]**	**0.00 [0.00]**	23.53 [0.30]	77.62 [13.66]	9.40 [2.92]	53.98 [20.64]

All methods were implemented in Julia [22] (version 1.12.7). Optimization models in the cutting-plane algorithm were built with JuMP [33] (version 1.31.2) and solved with MOSEK [56] (version 11.2). All experiments were run on a MacBook Pro with a 12-core Apple M4 Pro processor and 24 GB of memory. To compare solution quality across methods in the two sparse optimization experiments, we use the relative deviation from the best value found [43]:

$$\mathrm{SGap}(\%) = \left(\frac{|\mathrm{B} - \mathrm{B}^{\mathrm{Best}}|}{|\mathrm{B}| + 10^{-4}} \right) \times 100,$$

where B is the loss returned by a given method and $\mathrm{B}^{\mathrm{Best}}$ is the best loss across all compared methods. The regularization term 10^{-4} prevents division by a value close to zero. For replicated instances, the tables report results as "mean [standard deviation]," with the number of replications given in each subsection. Bold entries identify the smallest mean for each comparison. For all applications, the data source or instance construction is given in Appendix B, and the detailed per-instance results are reported in Online Resource.

4.1 Sparsity-Constrained Problems

We compare AGP with fixed-step IHT and coordinate-based PSS [8]. All methods share x^0, run for at most 3000 iterations, and stop when $\|x^{k+1} - x^k\|_2 \leq 10^{-5}$. Appendix B provides all remaining implementation settings.

4.1.1 Sparse Signal Recovery

Reconstructing a sparse high-dimensional signal from few linear measurements is a fundamental problem in compressed sensing and statistical learning. The problem arises naturally in MRI reconstruction, radar systems, and gene expression analysis, where data acquisition is costly or time-consuming but the underlying signal is inherently sparse. It can be modelled as

$$\min_{x \in \mathbb{R}^n} \ \|Ax - b\|_2^2 \quad \text{s.t. } \|x\|_0 \leq s,$$

with $A \in \mathbb{R}^{m \times n}$ as a measurement matrix with $m \ll n$, and $b \in \mathbb{R}^m$ the observed signal. Although x^{true} has zero residual, its support is hidden and must be recovered from ill-conditioned matrices with up to 50,000 columns.

Using the construction in Appendix B, we test 12 groups of five independently seeded instances; Table 1 lists (n, m, s). Table 1 shows that AGP is fastest and has the smallest mean SGap in every group, with a tie with PSS in Group 10. Its mean time ranges from 0.28 to 1.47 seconds, compared with 7.08 to 23.53 seconds for IHT. PSS obtains the smallest residual on some individual instances, but its time increases with s and its solution quality varies more across the groups. For $m = 200$, the mean IHT SGap remains above 72% in all six groups. Thus, on these ill-conditioned least squares instances, AGP gives a better balance of solution quality and time than fixed-step IHT and the coordinate updates of PSS.

Table 2: Mean gaps and computation times for sparse logistic regression.

	AGP		IHT		PSS	
# (data, m,n,s)	Time (s)	SGap (%)	Time (s)	SGap (%)	Time (s)	SGap (%)
#1 (a5a, 6414,123,5)	**0.12 [0.05]**	1.56 [1.32]	0.88 [0.04]	14.50 [12.02]	0.47 [0.18]	**0.69 [1.54]**
#2 (a5a, 6414,123,10)	1.10 [0.45]	**0.32 [0.47]**	**0.88 [0.05]**	4.82 [2.25]	1.18 [0.26]	1.23 [1.29]
#3 (a5a, 6414,123,20)	1.12 [0.53]	**0.00 [0.00]**	**0.87 [0.05]**	4.74 [1.56]	3.12 [0.40]	1.27 [0.73]
#4 (a6a, 11220,123,5)	**0.21 [0.05]**	1.50 [1.31]	1.65 [0.04]	14.13 [11.81]	0.94 [0.31]	**0.74 [1.64]**
#5 (a6a, 11220,123,10)	**0.85 [0.43]**	**0.21 [0.43]**	1.71 [0.03]	6.07 [2.19]	2.15 [0.30]	1.74 [1.91]
#6 (a6a, 11220,123,20)	1.98 [0.61]	**0.07 [0.17]**	**1.72 [0.04]**	4.72 [1.74]	14.02 [16.54]	1.32 [1.08]
#7 (a7a, 16100,123,5)	**0.42 [0.10]**	2.39 [1.41]	2.50 [0.08]	17.57 [13.21]	1.49 [0.31]	**0.00 [0.00]**
#8 (a7a, 16100,123,10)	**1.87 [0.66]**	**0.00 [0.00]**	2.63 [0.07]	6.66 [1.29]	3.37 [0.64]	2.03 [1.60]
#9 (a7a, 16100,123,20)	**2.28 [0.67]**	**0.00 [0.00]**	2.64 [0.07]	5.21 [1.47]	15.40 [10.88]	1.29 [0.66]
#10 (a8a, 22696,123,5)	**0.55 [0.14]**	2.91 [1.82]	3.56 [0.07]	19.40 [10.29]	2.12 [0.39]	**0.00 [0.00]**
#11 (a8a, 22696,123,10)	**3.42 [1.83]**	**0.32 [0.71]**	3.79 [0.17]	6.69 [1.74]	4.70 [1.00]	1.71 [1.88]
#12 (a8a, 22696,123,20)	4.98 [2.07]	**0.00 [0.00]**	**4.03 [0.03]**	5.23 [2.21]	28.28 [33.66]	1.89 [0.47]

4.1.2 Sparse Logistic Regression

Sparse logistic regression is a fundamental tool for high-dimensional classification, where the goal is to learn a linear classifier dependent on only a small subset of features. By constraining the number of nonzero coefficients, this approach facilitates feature selection, improves model interpretability, and reduces overfitting. The resulting sparsity-constrained problem is

$$\min_{x\in\mathbb{R}^n} \ \frac{1}{m}\sum_{i=1}^{m}\log\Big(1+e^{-y_i x^\top a_i}\Big) \quad \text{s.t. } \|x\|_0 \le s, \tag{24}$$

where $a_i^\top$ is the i-th row of the data matrix, and $y \in \{-1,1\}^m$ is the vector of binary class labels. We test 12 configurations from four LIBSVM datasets [29, 50], using the same five random initial points for every method; Table 2 lists the dimensions and sparsity levels, and Appendix B gives the data details.

Table 2 shows a different pattern across sparsity levels. PSS has the smallest mean SGap for all four configurations with $s = 5$, indicating that its coordinate updates are effective when very few variables are selected. For $s = 10$ and $s = 20$, AGP has the smallest mean SGap in all eight configurations. PSS becomes slower as s increases. On a8a, for example, its mean time rises from 2.12 seconds at $s = 5$ to 28.28 seconds at $s = 20$. IHT is faster than AGP in four configurations but has a larger mean SGap in all twelve. Overall, AGP provides the strongest balance of time and solution quality as the allowed support grows.

4.2 Robust Optimization with Sparse Uncertainty

We evaluate Algorithm 2 using AGP and IHT for pure sparsity constraints and SP, EL, and RL for compact sparsity-coupled sets. In the experiments, after an approximate method selects a support, we solve the continuous separation problem on that support so that the resulting scenario has the worst feasible magnitudes for the selected coordinates. In the static experiments, EL provides a global feasibility certificate, whereas SP and RL do not. Values with absolute magnitude below 10^{-5} are reported as 0.

4.2.1 Portfolio Optimization under Sparse Historical Stress Profiles

We consider an investor who chooses portfolio weights $x \in \mathbb{R}^{n_x}$ to balance expected return and variance against losses associated with a small number of adverse historical months. Each row of $B \in \mathbb{R}^{n_u\times n_x}$ records the losses of the assets in one such month relative to their sample means, so $(Bx)_r$ is the exposure of portfolio x to profile r. The adversary chooses a coefficient vector u with at most s nonzero entries to combine these profiles:

$$h(x,u) = 2u^\top Bx - u^\top Qu,$$

where the positive definite matrix Q penalizes large coefficients, making the inner maximum finite, and its dense structure allows the contribution of one profile to depend on the others. The robust portfolio problem is

$$\max_{\substack{x\ge 0\\ \mathbf{1}^\top x=1}} \left\{\mu^\top x - \lambda x^\top \Sigma x - \max_{\|u\|_0\le s}\left(2u^\top Bx - u^\top Qu\right)\right\}.$$

Table 3: Portfolio results using the 30-year Fama-French sample.

	Exact		AGP			IHT			AGP
q	Obj.	Time (s)	Obj.	Gap (%)	Time (s)	Obj.	Gap (%)	Time (s)	Gain (%)
24	-0.60860	13.35	-0.68005	11.74	0.44	-0.86772	42.58	0.12	21.63
36	-0.68261	65.76	-0.74169	8.66	0.30	-0.81718	19.71	0.21	9.24
54	-0.73061	527.87	-0.80826	10.63	0.30	-1.28992	76.55	0.22	37.34
90	TL	1000.01	-0.95363	n/a	0.64	-0.98233	n/a	0.57	2.92
108	TL	1000.00	-1.07842	n/a	0.93	-1.38089	n/a	0.26	21.90

Here μ and Σ are the estimated mean return vector and covariance matrix, and $\lambda > 0$ controls the variance penalty. The constraints require full investment and exclude short selling. Consider the epigraph reformulation

$$\begin{aligned} \max_{x,t}\ & \mu^\top x - \lambda x^\top \Sigma x - t \\ \text{s.t.}\ & \mathbf{1}^\top x = 1, \\ & x \geq 0, \\ & 2u^\top B x - u^\top Q u \leq t, \quad \forall u \in \mathcal{U}, \end{aligned} \tag{25}$$

where $\mathcal{U} = \{u : \|u\|_0 \leq s\}$.

We use 30 years of monthly returns [36] to form five nested stress libraries, with $s = 3$; Appendix B gives the construction of Q and the remaining settings. We set $\lambda = 0.5$ and compare the exact cutting-plane method with AGP and IHT. The two approximations use the same 15 starts, and every method has a limit of 1000 seconds. An active mixed-integer call may return slightly after this limit. The Exact column reports the full robust portfolio optimization with binary separation. After timing AGP and IHT, we evaluate each returned portfolio exactly by enumerating all supports containing three profiles and solving the concave quadratic problem on each support. The reported gap is relative to the exact robust optimum when available, gain is the percentage improvement of AGP over IHT, and TL denotes the time limit. Table 3 shows that exact optimization finishes for $q \leq 54$ and reaches the time limit for the two larger libraries. On the three solved cases, the AGP gap is between 8.66% and 11.74%, compared with 19.71% to 76.55% for IHT. For $q = 54$, AGP takes 0.30 seconds, while exact optimization takes 527.87 seconds. AGP gives the better exactly evaluated objective in all five cases and takes at most 0.93 seconds. Its improvement over IHT exceeds 15% in three of five cases, including two of the three with $q \geq 54$.

4.2.2 Linear Optimization with Implementation Error

We consider a linear optimization problem in which implementation errors affect the constraint coefficients. Let a^j denote an inequality row. The coefficient errors are not necessarily independent. We represent them through L shared sources of coefficient error, each of which can affect several variables. The vector $u \in \mathbb{R}^L$ gives the signed magnitudes of these sources, while $D_{i\ell}$ is the weight assigned to source ℓ for variable i. Thus, $(Du)_i = \sum_{\ell=1}^{L} D_{i\ell} u_\ell$ is their combined signed effect on variable i. With $\eta > 0$ denoting the maximum relative coefficient change, the contribution of variable i to row j is $a_i^j x_i\{1 + \eta(Du)_i\}$, where $D \in \mathbb{R}_+^{n \times L}$ satisfies $\sum_{\ell=1}^{L} D_{i\ell} = 1$ for every $i \in [n]$. So, $|(Du)_i| \leq 1$ whenever $|u_\ell| \leq 1$ for every $\ell \in [L]$. We use

$$\mathcal{W}_s = \left\{ u \in \mathbb{R}^L : \begin{array}{lll} \exists\, v, z, & -v \leq u \leq v, & 0 \leq v \leq z, \\ & Fv \leq d, & \mathbf{1}^\top z \leq s, \quad z \in \{0,1\}^L \end{array} \right\}.$$

Let c be the nominal objective vector, and let $\mathcal{X}$ collect the nominal equalities and variable bounds. The full robust model is

$$\begin{aligned} \min_{x \in \mathcal{X}}\ & c^\top x \\ \text{s.t.}\ & \sum_i a_i^j x_i\{1 + \eta(Du)_i\} \leq b_j, \quad \forall u \in \mathcal{W}_s,\ j \in [m]. \end{aligned}$$

Here v_ℓ bounds the magnitude of u_ℓ, z_ℓ indicates whether source ℓ is active, and each row of $Fv \leq d$ limits a specified combination of source magnitudes.

We test five NETLIB instances [58] with $\eta = 0.01$ and $s \in \{10, 20\}$; Table 4 gives their dimensions. EL solves binary separation globally, whereas RL relaxes the indicators and provides a valid lower bound. Appendix B gives the complete construction. Table 4 reports the percentage difference of the exact EL optimum and RL lower bound from the nominal objective, together with computation time. The last column gives the percentage gap between the RL lower bound and the exact EL optimum.

Table 4: Implementation error results on NETLIB test instances.

Instance	(n, m, L)	s	Exact (EL)		RL lower bound		
			Nominal diff. (%)	Time (s)	Nominal diff. (%)	Time (s)	Gap to EL (%)
SCFXM1	$(457, 143, 240)$	10	0.951	5.00	0.938	0.86	0.01346
		20	0.979	2.94	0.979	1.36	0.00
SCFXM3	$(1371, 429, 480)$	10	1.206	13.17	1.188	4.20	0.01785
		20	1.380	34.92	1.357	6.68	0.02189
25FV47	$(1571, 305, 600)$	10	1.812	26.86	1.802	8.65	0.01022
		20	2.064	58.62	2.038	13.47	0.02551
WOODW	$(8405, 13, 1200)$	10	0.792	2.17	0.785	0.86	0.00684
		20	1.108	2.72	1.102	1.19	0.00633
FIT2D	$(10500, 24, 1500)$	10	0.022	9.69	0.022	7.27	0.00031
		20	0.038	17.88	0.037	9.36	0.00082

Across all ten cases, this gap is at most 0.0256%, while RL uses 25% to 83% less computation time than EL. For WOODW and FIT2D, which have 8,405 and 10,500 variables, respectively, the RL gaps are at most 0.00684% and 0.00082%. Thus, the continuous relaxation remains very tight as the number of variables and uncertainty sources increases. Compared with the nominal linear programs, the exact robust objectives increase by 0.022% to 2.064%.

4.2.3 Wireless Downlink Planning under Sparse Jamming

We consider a wireless network in which n_{TX} coordinated transmitting antenna elements, located at one or more base stations (BSs), jointly serve m_{UE} user equipment (UE) devices under threat from n_{J} potential reactive jammers. We focus on transmission from the base stations to the users, which is known as the *downlink*. The transmitting elements form a coordinated cluster, so their received powers contribute to the useful signal. Robust transmit power minimization subject to worst-case *signal-to-interference-plus-noise ratio* (SINR) requirements is a standard wireless design objective [68]. The objective is to minimize the total transmit power while ensuring that every user attains a target SINR θ under every feasible sparse jamming pattern.

Let $p_i \in [0, \bar{p}_i]$ be the transmit power of element i, h_{ki} the *channel gain* from element i to UE k, which is the fraction of transmitted power surviving path loss and shadowing, and σ_k^2 the noise power at UE k. The parameter g_{kij} is the two-hop gain from element i through jammer j to UE k. For p and u, the SINR at user k is

$$\mathrm{SINR}_k(p, u) := \frac{\sum_i h_{ki} p_i}{\sigma_k^2 + \sum_j u_j \sum_i g_{kij} p_i}.$$

The numerator is the useful signal received from the transmitting elements, while the denominator is the sum of the noise power and the interference produced by the active jammers. We require $\mathrm{SINR}_k(p, u) \geq \theta$ for every user k and every $u \in \mathcal{U}_s$. Rearranging this inequality yields the robust model below:

$$\begin{aligned} \min_{p \in [0, \bar{p}]} \ & \sum_{i=1}^{n_{\mathrm{TX}}} p_i \\ \text{s.t.} \ & -\sum_{i=1}^{n_{\mathrm{TX}}} p_i h_{ki} + \theta \sigma_k^2 + \theta \sum_{j=1}^{n_{\mathrm{J}}} u_j \sum_{i=1}^{n_{\mathrm{TX}}} p_i g_{kij} \leq 0, \quad k \in [m_{\mathrm{UE}}], \ \forall u \in \mathcal{U}_s, \end{aligned} \tag{RO-RJ}$$

where u_j is the amplification factor of jammer j. The constraint ensures that for each user k the received signal power exceeds θ times the sum of noise and worst-case interference. Given a power allocation p, the worst-case interference for user k depends on the scenario generation subproblem $\max_{u \in \mathcal{U}_s} \sum_j u_j c_{kj}(p)$, where $c_{kj}(p) := \sum_i p_i g_{kij}$ is the cascade-weighted power reaching user k through jammer j. The uncertainty set is

$$\mathcal{U}_s = \left\{ u \in \mathbb{R}_+^{n_{\mathrm{J}}} : 0 \leq u \leq \bar{u}, \ Fu \leq d, \ \|u\|_0 \leq s \right\}, \qquad F \in \mathbb{R}_+^{q \times n_{\mathrm{J}}}. \tag{26}$$

Here q counts shared resource constraints, $\bar{u}_j$ caps jammer j, and d_r is the limit for row r. Each row of F models a shared controller, battery, link, or frequency resource. Since $F \geq 0$, feasibility is preserved under coordinate zeroing.

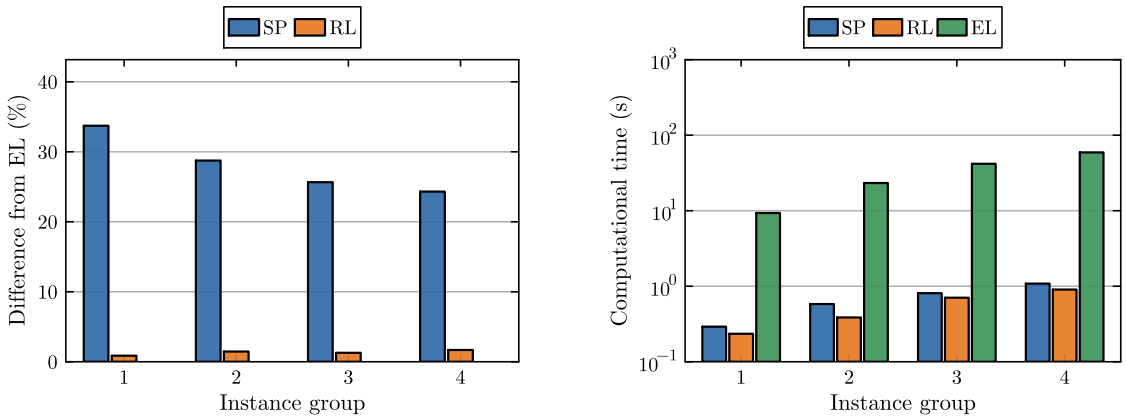

Fig. 3: Static wireless results from the DICHASUS measurements. Left: mean percentage difference from EL. Right: mean computation time on a log scale

In the following we compare SP and RL with EL as the number of potential jammers, user constraints, shared resource constraints, and active jammers increases. We also evaluate every returned power allocation by exact separation, so that objective differences and satisfaction of the robust SINR constraints are assessed separately. For a returned power vector $\hat{p}$, define the exact robust residual for user k by

$$R_k(\hat{p}) := \max_{u \in \mathcal{U}_s} \left\{ -\sum_i h_{ki}\hat{p}_i + \theta\sigma_k^2 + \theta \sum_j u_j \sum_i g_{kij}\hat{p}_i \right\}.$$

The binary lifted formulation computes $R_k(\hat{p})$. The reported exact robust violation is $V(\hat{p}) := \max\{0, \max_k R_k(\hat{p})\}$.

We use four groups of three selections from the public DICHASUS outdoor channel measurements [34, 35]. Appendix B describes the mapping to transmitter, user, and jammer gains, the instance dimensions, and the computational settings. Figure 3 summarizes the objective and time comparisons, and Table S5 in Online Resource reports every individual run. All three methods finish all 12 instances. Across the four groups, SP is 24.31% to 33.72% below EL, whereas RL is 0.86% to 1.68% below EL. RL takes 0.23 to 0.90 seconds and is faster than SP in every group mean. EL takes 9.32 to 59.15 seconds, which is 40 to 65 times the RL time. Exact evaluation of the returned power vectors gives mean violations between 0.00457 and 0.00631 for SP and between 0.00017 and 0.00040 for RL. Thus, compared with SP, RL gives a much tighter lower bound and reduces the mean exact violation by a factor of 11 to 38.

4.2.4 Adjustable Wireless Planning with Shared Recovery

We next consider an adjustable version of the wireless problem. The transmitting element power vector p is selected before the jammer pattern is observed. After the pattern is observed, the network may allocate protected recovery resources. The recovery structure represents resource sharing through Coordinated Multi-Point (CoMP) joint transmission. CoMP is part of the LTE physical layer coordination framework [1] and allows neighboring base stations to coordinate scheduling and transmission resources [5, 40]. Protected recovery resources may serve multiple users, so users share the recourse.

Let $r \in [R]$ index the protected recovery resources and let $y_r \geq 0$ be the amount allocated to resource r. The coefficient $B_{kr} \geq 0$ is the recovery supplied to user k per unit of y_r, and $c_r > 0$ is the corresponding unit cost. Define $\bar{h}_k := \max_i h_{ki}$ and the normalized service deficit

$$\Delta_k(p, u) := \frac{\theta\sigma_k^2 - \sum_i h_{ki}p_i + \theta \sum_j u_j \sum_i g_{kij}p_i}{\bar{h}_k}. \tag{27}$$

For fixed (p, u), the recovery problem is

$$Q(p, u) = \min_{y \geq 0} \{c^\top y : By \geq \Delta(p, u)\}. \tag{28}$$

Because B and c do not depend on u and y is selected afterward, the model (28) has fixed recourse; $By \geq \Delta(p, u)$ requires shared recovery to cover each user's service deficit. The nominal SINR constraints

ensure service before jamming, so the recovery layer cannot replace normal service when $u = 0$. The adjustable robust problem is

$$\begin{aligned} \min_{0\le p\le \bar{p}} \quad & \mathbf{1}^\top p + \max_{u\in\mathcal{U}_s} Q(p,u) \\ \text{s.t.} \quad & \sum_i h_{ki} p_i \ge \theta\sigma_k^2, \qquad k \in [m_{\mathrm{UE}}], \end{aligned} \tag{ARO-RJ}$$

with the uncertainty set in (26). The dual of (28) is

$$Q(p,u) = \max_{\lambda\ge 0}\{\lambda^\top \Delta(p,u) : B^\top\lambda \le c\}. \tag{29}$$

Consequently, separation for a fixed p has the form

$$\max_{u\in\mathcal{U}_s,\,\lambda\in\Lambda}\{\mathbf{1}^\top p + \lambda^\top\Delta(p,u)\}, \qquad \Lambda := \{\lambda\ge 0 : B^\top\lambda\le c\}. \tag{30}$$

We use multistart alternating maximization. For fixed u, the λ problem is linear. For fixed λ, the u problem is a sparse linear maximization over (26), solved by SP, RL, or EL. Each feasible pair (u^t,λ^t) generates

$$\mathbf{1}^\top p + (\lambda^t)^\top\Delta(p,u^t) \le \tau. \tag{31}$$

The left-hand side of (31) does not exceed the full separation value for any p. Consequently, each cut preserves a relaxation of (ARO-RJ), and the master objective is a valid lower bound. We call the lower bound from the dual cutting-plane master the dual bound (DB). The labels DB-SP, DB-RL, and DB-EL indicate the method used for the u step. Since the joint problem in (u,λ) is nonconcave, even DB-EL, which solves the u step exactly, remains a lower bound. On small instances with only five users, Exact-Full is the fully adjustable optimum obtained by enumeration, and AADR-Exact is the best affinely adjustable decision rule (AADR) obtained from the same enumeration. For the larger instances, AADR-RL uses the affine rule $y(u) = y^0 + Yu$ over the RL polyhedron, while RO-EL denotes the static robust policy computed with EL separation. The DB methods are compared through their lower bounds, while the AADR and RO-EL policies measure the value of adjustment.

We use three larger groups of three DICHASUS selections; Appendix B gives their dimensions, recovery-resource construction, and computational settings. We first consider the instances with five users, for which the fully adjustable optimum is available. The DB-RL lower bound equals this optimum at the reported precision. The exact affine policy costs 0.84% more than the optimum on average, whereas the static policy costs 29.37% more. Thus, the affine rule captures most of the benefit of full adjustment on these instances, and the much larger static policy difference shows the value of allowing recovery after the jammer pattern is observed.

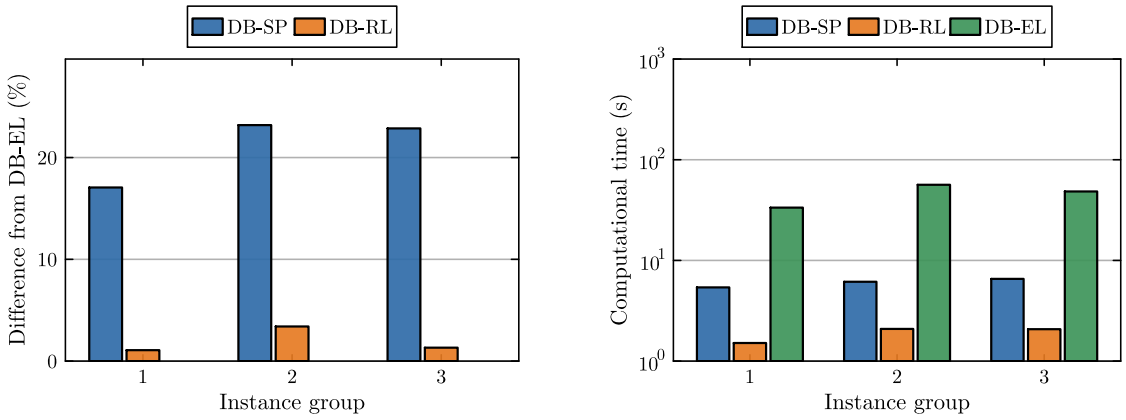

Fig. 4: DICHASUS adjustable-wireless results: mean percentage difference from DB-EL (left) and mean computation time on a log scale (right)

We next turn to the larger instances. Figure 4 shows a clear difference between the two approximate lower bounds. DB-SP is 17.06% to 23.19% below DB-EL, whereas DB-RL is 1.07% to 3.40% below

DB-EL. These values compare lower bounds and are not global optimality gaps, because the alternating maximization is local for all three methods. DB-RL takes 1.51 to 2.09 seconds and is faster than DB-SP for every group mean. DB-EL takes 33.44 to 56.39 seconds, or 22 to 27 times the DB-RL time. Hence, the continuous relaxation gives a lower bound close to DB-EL while avoiding the binary sparse optimization used in its u step. The affine policy improves on the static policy in all nine larger instances, by 2.13% on average, so shared recovery continues to reduce the policy cost as the number of users and potential jammers increases. Online Resource reports every run and the small exact comparison.

5 Conclusion

We developed a framework for robust optimization under sparse uncertainty, where the unknown identities of the deviating parameters make separation combinatorial. For a pure sparse set, we introduced an adaptive gradient projection method for smooth sparse optimization and obtained a semidefinite bound by lifting the uncertain objective together with the set. When sparsity is combined with a compact convex set, we derived an exact mixed-integer representation and continuous relaxations. These formulations, together with scenario projection, are incorporated into a cutting-plane method that produces nondecreasing valid lower bounds. After dualizing fixed recourse, the same design applies to adjustable robust optimization. We also proved that continuous sparse box uncertainty and discrete budgeted uncertainty have the same worst-case value when uncertainty enters linearly. Our numerical experiments show that the proposed methods are effective for both sparse optimization and robust optimization problems when exact solution becomes computationally costly.

Several directions remain open. First, local sparse separation and alternating maximization yield valid lower bounds but do not certify global feasibility or optimality. As such, global separation, tighter relaxations, and adversarial upper bounds would strengthen these guarantees. Second, the continuous lifted relaxation is exact only for certain coupling structures. Characterizing the convex hull of sets that combine sparsity with convex restrictions, and identifying when $\widehat{\mathcal{Z}}_s$ is tight, would improve the formulations. Finally, future work could develop distributionally robust and data-driven models that estimate s or $\mathcal{G}$ from data, as well as multistage adjustable extensions.

Statements and Declarations

Funding. Funding was provided by IVADO and the Canada First Research Excellence Fund, by the Natural Sciences and Engineering Research Council of Canada through grant RGPIN-2024-05908, and by grant CRC-2024-00178 from the Canada Research Chairs Program.

Competing interests. The authors declare no competing interests.

Data and code availability. The Fama-French, LIBSVM, NETLIB, and DICHASUS data used in the experiments are publicly available from the sources cited in the paper. The implementation code and processed data are available in the separate repository at `https://github.com/abbaskhademi/ROS`. The complete numerical results are provided in Online Resource.

References

1. 3GPP: Coordinated Multi-Point Operation for LTE Physical Layer Aspects. Technical Report TR 36.819, 3rd Generation Partnership Project (2013). Release 11
2. Alves Pessoa, A., Di Puglia Pugliese, L., Guerriero, F., Poss, M.: Robust constrained shortest path problems under budgeted uncertainty. Networks **66**(2), 98–111 (2015). https://doi.org/10.1002/net.21615
3. Ardestani-Jaafari, A., Delage, E.: Robust optimization of sums of piecewise linear functions with application to inventory problems. Operations Research **64**(2), 474–494 (2016). https://doi.org/10.1287/opre.2016.1483
4. Argyriou, A., Foygel, R., Srebro, N.: Sparse prediction with the k-support norm. Advances in Neural Information Processing Systems **25** (2012)
5. Bassoy, S., Imran, M.A., Yang, S., Tafazolli, R.: A load-aware clustering model for coordinated transmission in future wireless networks. IEEE Access **7**, 92693–92708 (2019). https://doi.org/10.1109/ACCESS.2019.2927093
6. Bauschke, H., Combettes, P.L.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces, 2nd edn. Springer, Cham, Switzerland (2017). https://doi.org/10.1007/978-3-319-48311-5
7. Beck, A.: Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with Python and MATLAB. SIAM, Philadelphia (2023). https://doi.org/10.1137/1.9781611977622
8. Beck, A., Eldar, Y.C.: Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. SIAM Journal on Optimization **23**(3), 1480–1509 (2013). https://doi.org/10.1137/120869778
9. Ben-Tal, A., den Hertog, D., Vial, J.P.: Deriving robust counterparts of nonlinear uncertain inequalities. Mathematical Programming **149**(1-2), 265–299 (2015). https://doi.org/10.1007/s10107-014-0750-8
10. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Mathematical Programming **99**(2), 351–376 (2004). https://doi.org/10.1007/s10107-003-0454-y
11. Ben-Tal, A., Nemirovski, A.: Robust convex optimization. Mathematics of Operations Research **23**(4), 769–805 (1998). https://doi.org/10.1287/moor.23.4.769
12. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. SIAM Review **53**(3), 464–501 (2011). https://doi.org/10.1137/080734510
13. Bertsimas, D., den Hertog, D.: Robust and adaptive optimization. Dynamic Ideas LLC, Belmont, MA (2022)
14. Bertsimas, D., den Hertog, D.: Fifty years of robust linear and convex optimization. European Journal of Operational Research (2026). https://doi.org/10.1016/j.ejor.2026.08.042
15. Bertsimas, D., Hertog, D.d., Pauphilet, J., Zhen, J.: Robust convex optimization: A new perspective that unifies and extends. Mathematical Programming **200**(2), 877–918 (2023). https://doi.org/10.1007/s10107-022-01881-w
16. Bertsimas, D., de Moor, D., den Hertog, D., Koukouvinos, T., Zhen, J.: An exact method for a class of robust nonlinear optimization problems. Optimization Online preprint (2024). URL https://optimization-online.org/?p=27202
17. Bertsimas, D., Na, L., Stellato, B., Wang, I.: The benefit of uncertainty coupling in robust and adaptive robust optimization. INFORMS Journal on Optimization **7**(2), 105–141 (2025). https://doi.org/10.1287/ijoo.2023.0007
18. Bertsimas, D., Nohadani, O., Teo, K.M.: Nonconvex robust optimization for problems with constraints. INFORMS Journal on Computing **22**(1), 44–58 (2010). https://doi.org/10.1287/ijoc.1090.0319
19. Bertsimas, D., Sim, M.: Robust discrete optimization and network flows. Mathematical Programming **98**(1), 49–71 (2003). https://doi.org/10.1007/s10107-003-0396-4
20. Bertsimas, D., Sim, M.: The price of robustness. Operations Research **52**(1), 35–53 (2004). https://doi.org/10.1287/opre.1030.0065
21. Bertsimas, D., Thiele, A.: A robust optimization approach to inventory theory. Operations Research **54**(1), 150–168 (2006). https://doi.org/10.1287/opre.1050.0238
22. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A fresh approach to numerical computing. SIAM Review **59**(1), 65–98 (2017). https://doi.org/10.1137/141000671
23. Bienstock, D.: Computational study of a family of mixed-integer quadratic programming problems. Mathematical Programming **74**(2), 121–140 (1996). https://doi.org/10.1007/BF02592208
24. Borthwick, D.: A Primer for Mathematical Analysis. Springer, Cham, Switzerland (2025). https://doi.org/10.1007/978-3-031-91713-4
25. Bougeret, M., Pessoa, A.A., Poss, M.: Robust scheduling with budgeted uncertainty. Discrete Applied Mathematics **261**, 93–107 (2019). https://doi.org/10.1016/j.dam.2018.07.001

26. Boyd, S.P., Vandenberghe, L.: Convex optimization. Cambridge University Press, Cambridge, UK (2004). https://doi.org/10.1017/CBO9780511804441
27. Breuer, D.J., Lahrichi, N., Clark, D.E., Benneyan, J.C.: Robust combined operating room planning and personnel scheduling under uncertainty. Operations Research for Health Care **27**, 100276 (2020). https://doi.org/10.1016/j.orhc.2020.100276
28. Büsing, C., Gersing, T., Koster, A.M.: A branch and bound algorithm for robust binary optimization with budget uncertainty. Mathematical Programming Computation **15**(2), 269–326 (2023). https://doi.org/10.1007/s12532-022-00232-2
29. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(3), 1–27 (2011). https://doi.org/10.1145/1961189.1961199
30. Crimmins, B.L., Halderman, J.A., Sturt, B.: Improving the security of United States elections with robust optimization. Operations Research **73**(1), 61–85 (2025). https://doi.org/10.1287/opre.2023.0422
31. Darivianakis, G., Georghiou, A., Shafiee, S., Lygeros, J.: A robust optimization approach to network control using local information exchange. Operations Research **73**(5), 2849–2866 (2025). https://doi.org/10.1287/opre.2020.0217
32. Dehghani Filabadi, M., Mahmoudzadeh, H.: Effective budget of uncertainty for classes of robust optimization. INFORMS Journal on Optimization **4**(3), 249–277 (2022). https://doi.org/10.1287/ijoo.2021.0069
33. Dunning, I., Huchette, J., Lubin, M.: JuMP: A modeling language for mathematical optimization. SIAM Review **59**(2), 295–320 (2017). https://doi.org/10.1137/15M1020575
34. Euchner, F., Gauger, M.: CSI dataset dichasus-dxxx-reduced: Outdoor, three arrays distributed along the facade (one antenna removed). DaRUS, University of Stuttgart (2022). URL https://dichasus.inue.uni-stuttgart.de/datasets/data/dichasus-dxxx-reduced/. Version 1; file dichasus-d036, accessed 27 August 2026
35. Euchner, F., Gauger, M., Dörner, S., ten Brink, S.: A distributed massive MIMO channel sounder for "big CSI data"-driven machine learning. In: WSA 2021; 25th International ITG Workshop on Smart Antennas, pp. 289–294. VDE, Berlin, Germany (2021)
36. French, K.R.: Data library: 49 industry portfolios. Tuck School of Business at Dartmouth. URL https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html. Accessed 21 August 2026
37. Goerigk, M., Khosravi, M.: Robust combinatorial optimization problems under budgeted interdiction uncertainty. OR Spectrum **47**(1), 255–285 (2025). https://doi.org/10.1007/s00291-024-00772-0
38. Goerigk, M., Lendl, S.: Robust combinatorial optimization with locally budgeted uncertainty. Open Journal of Mathematical Optimization **2**, 1–18 (2021). https://doi.org/10.5802/ojmo.5
39. Gorissen, B.L., Yanıkoğlu, İ., den Hertog, D.: A practical guide to robust optimization. Omega **53**, 124–137 (2015). https://doi.org/10.1016/j.omega.2014.12.006
40. Grebla, G., Birand, B., van de Ven, P., Zussman, G.: Joint Transmission in Cellular Networks with CoMP—Stability and Scheduling Algorithms. Performance Evaluation **91**, 38–55 (2015). https://doi.org/10.1016/j.peva.2015.06.004
41. ITU-R: Propagation data and prediction methods for the planning of short-range outdoor radiocommunication systems and radio local area networks in the frequency range 300 mhz to 300 ghz. Recommendation P.1411-13, International Telecommunication Union (2025). URL https://www.itu.int/rec/R-REC-P.1411-13-202509-I/en
42. Khademi, A.: The convexity zoo: a taxonomy of function classes in optimization. Optimization pp. 1–52 (2026). https://doi.org/10.1080/02331934.2026.2672525
43. Khademi, A., Marandi, A.: Quadratic optimization through the lens of adjustable robust optimization. INFORMS Journal on Computing (2025). https://doi.org/10.1287/ijoc.2024.0577
44. Khademi, A., Marandi, A., Soleimani-Damaneh, M.: A new dual-based cutting plane algorithm for nonlinear adjustable robust optimization. Journal of Global Optimization **89**(3), 559–595 (2024). https://doi.org/10.1007/s10898-023-01360-2
45. Khademi, A., Silveti-Falls, A.: Adaptive conditional gradient descent. arXiv preprint arXiv:2510.11440 (2025). URL https://arxiv.org/abs/2510.11440
46. Khademi, A., Soleimani-damaneh, M.: On necessity of L-stationarity in nonlinear optimization with a sparsity constraint. Mathematical Researches **6**(3), 387–392 (2020). https://doi.org/10.52547/mmr.6.3.387
47. Koster, A.M., Segschneider, J., Ventsch, N.: Γ-robust optimization of project scheduling problems. Computers & Operations Research **161**, 106453 (2024). https://doi.org/10.1016/j.cor.2023.106453
48. Lefebvre, H., Malaguti, E., Monaci, M.: Exact approaches for convex adjustable robust optimization. Mathematical Programming **218**, 439–476 (2026). https://doi.org/10.1007/s10107-025-02311-3
49. Ley, E., Marx, R., Merkert, M., Niemann, T., Stiller, S.: Decision-dependent robust optimization with elementwise controllable uncertainty sets. OR Spectrum (2026). https://doi.org/10.1007/s00291-026-00856-z
50. LIBSVM: Binary classification data sets. LIBSVM Data Repository. URL https://www.csie.ntu.edu.tw/%7Ecjlin/libsvmtools/datasets/binary.html. Accessed 27 August 2026
51. Lim, Y.F., Wang, C.: Inventory management based on target-oriented robust optimization. Management Science **63**(12), 4409–4427 (2017). https://doi.org/10.1287/mnsc.2016.2565
52. Liu, Y., Ning, P., Reiter, M.K.: False data injection attacks against state estimation in electric power grids. ACM Transactions on Information and System Security (TISSEC) **14**(1), 1–33 (2011). https://doi.org/10.1145/1952982.1952995
53. Lu, H., Sturt, B.: On the sparsity of optimal linear decision rules for a class of robust optimization problems with box uncertainty sets. Operations Research **74**(1), 500–516 (2026). https://doi.org/10.1287/opre.2023.0603
54. Lu, M., Shen, Z.J.M.: A review of robust operations management under model uncertainty. Production and Operations Management **30**(6), 1927–1943 (2021). https://doi.org/10.1111/poms.13239
55. Marandi, A., van Houtum, G.J., Khademi, A.: Multi-stage adjustable robust location-transportation problems with integer-valued demand. Optimization Online preprint (2025). URL https://optimization-online.org/?p=16185

56. MOSEK ApS: MOSEK Optimizer API for Julia, Version 11.2 (2026). URL https://docs.mosek.com/11.2/juliaapi/index.html
57. Mutapcic, A., Boyd, S.: Cutting-set methods for robust convex optimization with pessimizing oracles. Optimization Methods & Software **24**(3), 381–406 (2009). https://doi.org/10.1080/10556780802712889
58. NETLIB: Linear programming test problems. NETLIB Repository. URL https://www.netlib.org/lp/data/. Accessed 21 August 2026
59. Poss, M.: Robust combinatorial optimization with variable cost uncertainty. European Journal of Operational Research **237**(3), 836–845 (2014). https://doi.org/10.1016/j.ejor.2014.02.060
60. Shapiro, A., Dentcheva, D., Ruszczynski, A.: Lectures on stochastic programming: modeling and theory. SIAM, Philadelphia (2021). https://doi.org/10.1137/1.9781611976595
61. Soyster, A.L.: Convex programming with set-inclusive constraints and applications to inexact linear programming. Operations Research **21**(5), 1154–1157 (1973). https://doi.org/10.1287/opre.21.5.1154
62. Su, J., Vargas, D.V., Sakurai, K.: One pixel attack for fooling deep neural networks. IEEE Transactions on Evolutionary Computation **23**(5), 828–841 (2019). https://doi.org/10.1109/TEVC.2019.2890858
63. Xidonas, P., Steuer, R., Hassapis, C.: Robust portfolio optimization: A categorized bibliographic review. Annals of Operations Research **292**(1), 533–552 (2020). https://doi.org/10.1007/s10479-020-03630-8
64. Yang, H., Morton, D.P., Bandi, C., Dvijotham, K.: Robust optimization for electricity generation. INFORMS Journal on Computing **33**(1), 336–351 (2021). https://doi.org/10.1287/ijoc.2020.0956
65. Yanıkoğlu, İ., Gorissen, B.L., den Hertog, D.: A survey of adjustable robust optimization. European Journal of Operational Research **277**(3), 799–813 (2019). https://doi.org/10.1016/j.ejor.2018.08.031
66. Zeng, B., Zhao, L.: Solving two-stage robust optimization problems using a column-and-constraint generation method. Operations Research Letters **41**(5), 457–461 (2013). https://doi.org/10.1016/j.orl.2013.05.003
67. Zhen, J., Kuhn, D., Wiesemann, W.: A unified theory of robust and distributionally robust optimization via the primal-worst-equals-dual-best principle. Operations Research **73**(2), 862–878 (2025). https://doi.org/10.1287/opre.2021.0268
68. Zheng, G., Wong, K.K., Ng, T.S.: Robust Linear MIMO in the Downlink: A Worst-Case Optimization with Ellipsoidal Uncertainty Regions. EURASIP Journal on Advances in Signal Processing **2008**, 609028 (2008). https://doi.org/10.1155/2008/609028

Appendices

Appendix A contains the examples, Appendix B describes the data generation, and Online Resource reports the numerical results.

Appendix A Examples

Example 1 Consider

$$\begin{aligned} \min_{x_1,x_2}\ & -2x_1 + x_2 \\ \text{s.t.}\ & 5x_1 + u^\top Q u + 2b^\top u \le 0, \quad \forall u \in \mathcal{U} := \{u \in \mathbb{R}^5 :\ \|u\|_0 \le 2\}, \\ & -100 \le x_1, x_2 \le 100, \end{aligned}$$

with the following coefficient data:

$$Q = \begin{pmatrix} -2 & -1 & -1 & -1 & -1 \\ -1 & -2 & -1 & -1 & -1 \\ -1 & -1 & -2 & -1 & -1 \\ -1 & -1 & -1 & -2 & -1 \\ -1 & -1 & -1 & -1 & -2 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 3 \\ 2 \\ 3 \\ 12 \\ 5 \end{pmatrix}.$$

Let $h(u) = u^\top Q u + 2b^\top u$. A worst-case $\hat{u}$ with $\|\hat{u}\|_0 < 2$ can occur if and only if the unconstrained maximizer $u^\sharp := \operatorname{argmax} h(u)$ satisfies $\|u^\sharp\|_0 < 2$, which gives $u^\sharp = \hat{u}$. In this example, $u^\sharp = \frac{1}{6}\left(-7, -13, -7, 47, 5\right)^\top$, i.e., $\|u^\sharp\|_0 = 5 > 2$, so it suffices to check only the 2-sparse supports. Here $Q = -I_5 - \mathbf{1}\mathbf{1}^\top \prec 0$, so for each pair $S = \{i, j\}$ the principal matrix is $Q_{SS} = \left(\begin{smallmatrix} -2 & -1 \\ -1 & -2 \end{smallmatrix}\right)$ with inverse

$Q_{SS}^{-1} = \frac{1}{3}\left(\begin{smallmatrix} -2 & 1 \\ 1 & -2 \end{smallmatrix}\right)$. Thus, the $\binom{5}{2} = 10$ basic scenarios are

$$\mathcal{W} = \left\{ \begin{pmatrix} \frac{4}{3} \\ \frac{1}{3} \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -2 \\ 0 \\ 0 \\ 7 \\ 0 \end{pmatrix}, \begin{pmatrix} \frac{1}{3} \\ 0 \\ 0 \\ 0 \\ \frac{7}{3} \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{1}{3} \\ \frac{4}{3} \\ 0 \\ 0 \end{pmatrix}, \right.$$
$$\left. \begin{pmatrix} 0 \\ -\frac{8}{3} \\ 0 \\ \frac{22}{3} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{1}{3} \\ 0 \\ 0 \\ \frac{8}{3} \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -2 \\ 7 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ \frac{1}{3} \\ 0 \\ \frac{7}{3} \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{19}{3} \\ -\frac{2}{3} \end{pmatrix} \right\}.$$

Using $h(u^S) = -b_S^\top Q_{SS}^{-1} b_S$, the worst-case value is attained at $S = \{2, 4\}$ and equals

$$h\big((0, -\tfrac{8}{3}, 0, \tfrac{22}{3}, 0)^\top\big) = -\begin{pmatrix} 2 & 12 \end{pmatrix}\left(\frac{1}{3}\begin{pmatrix} -2 & 1 \\ 1 & -2 \end{pmatrix}\right)\begin{pmatrix} 2 \\ 12 \end{pmatrix} = \tfrac{248}{3}.$$

Therefore the robust constraint reduces to $5x_1 + \frac{248}{3} \le 0$, i.e., $x_1 \le -\frac{248}{15}$:

$$\begin{aligned} \min_{x_1, x_2} \; & -2x_1 + x_2 \\ \text{s.t.} \; & 5x_1 \le -\tfrac{248}{3}, \\ & -100 \le x_1, x_2 \le 100. \end{aligned}$$

The optimal solution is $x^\star = \big(-\frac{248}{15}, -100\big)^\top$ with optimal value $-2(-\frac{248}{15}) - 100 = -\frac{1004}{15}$.

Example 2 We revisit Example 1 and compute a worst-case scenario by solving

$$\max_{u \in \mathbb{R}^5} \big\{ h(u) = u^\top Q u + 2b^\top u \;:\; \|u\|_0 \le 2 \big\}.$$

We run Algorithm 1 with $\beta = 2$, $\delta = 10^{-10}$, $\varepsilon = 10^{-2}$, and $\gamma_k = 1$. We choose $u^{-1} = \mathbf{e}_1$ and a nontrivial start $u^0 = \mathbf{e}_2$. The iterates and objective values are given in Table A1. After about 9 iterations, Algorithm 1 identifies $u^* \approx \big(0, -\frac{8}{3}, 0, \frac{22}{3}, 0\big)^\top$, with $h(u^*) \approx \frac{248}{3}$, matching the worst-case basic scenario $(0, -\frac{8}{3}, 0, \frac{22}{3}, 0)^\top$ obtained by direct enumeration of all $\binom{5}{2} = 10$ supports. The resulting robust constraint is $5x_1 + \frac{248}{3} \le 0$, with optimal solution $x^* = (-\frac{248}{15}, -100)^\top$.

Table A1: Convergence of Algorithm 1.

Iteration (k)	Solution u^k	Objective $h(u^k)$
1	$(0.0000,\ 1.0000,\ 0.0000,\ 2.7500,\ 0.0000)^\top$	47.3750
2	$(0.0000,\ 0.0000,\ 0.0000,\ 4.6945,\ 0.4419)^\top$	68.4709
3	$(0.0000,\ -1.6297,\ 0.0000,\ 5.8215,\ 0.0000)^\top$	79.0804
4	$(0.0000,\ -1.9841,\ 0.0000,\ 7.0741,\ 0.0000)^\top$	81.9544
5	$(0.0000,\ -2.2498,\ 0.0000,\ 7.0347,\ 0.0000)^\top$	82.3897
6	$(0.0000,\ -2.4187,\ 0.0000,\ 7.0916,\ 0.0000)^\top$	82.5467
7	$(0.0000,\ -2.5498,\ 0.0000,\ 7.2130,\ 0.0000)^\top$	82.6385
8	$(0.0000,\ -2.6621,\ 0.0000,\ 7.3355,\ 0.0000)^\top$	82.6666
9	$(0.0000,\ -2.6649,\ 0.0000,\ 7.3333,\ 0.0000)^\top$	82.6667

Example 3 In Example 1, the true worst-case value over $\mathcal{U} = \{u : \|u\|_0 \le 2\}$ is $z^{\mathcal{U}} = \max_{u \in \mathcal{U}}\{u^\top Q u + 2b^\top u\} = 82.6667$, with the true 2-sparse maximizer $u^\star = (0, -2.667, 0, 7.333, 0)^\top$. If we naively maximize the unlifted objective over the lifted set $\mathcal{U}_{\mathrm{SDP}}$ of (13), we obtain

$$z = \max_{(u,U) \in \mathcal{U}_{\mathrm{SDP}}} \{u^\top Q u + 2b^\top u\} = \max_{u \in \mathbb{R}^5}\{u^\top Q u + 2b^\top u\} = 86.8333.$$

Substituting $z^{\mathcal{U}}$ in the robust constraint gives an original optimal value of -66.93, whereas using z returns an upper bound of -65.27. The associated worst-case scenario is dense:

$$\bar{u} = (-1.1667, -2.1667, -1.1667, 7.8333, 0.8333)^\top.$$

If we project it onto the 2-sparse set by keeping the two largest absolute components, we obtain

$$\tilde{u} = (0, -2.1667, 0, 7.8333, 0)^\top.$$

Using $\tilde{u}$ in the robust problem provides a lower bound of -67.53. In this example, coincidentally, the support of the projected scenario $\tilde{u}$ matches the true worst-case scenario $u^\star$.

Using the lifted objective $\mathrm{tr}(QU) + 2b^\top u$ over $\mathcal{U}_{\mathrm{SDP}}$ penalizes increases in U because $Q \prec 0$. Solving this lifted SDP yields the tighter upper bound

$$z^{\mathcal{U}_{\mathrm{SDP}}} = \max_{u,U}\{\mathrm{tr}(Q^\top U) + 2b^\top u : \ (u, U) \in \mathcal{U}_{SDP}\} = 85.9919.$$

The corresponding worst-case scenario is

$$u_{\mathrm{SDP}} = (-0.8388, -1.9504, -0.8388, 7.8721, 0.0920)^\top.$$

Substituting this valid lifted bound $z^{\mathcal{U}_{\mathrm{SDP}}}$ into the robust constraint provides a rigorous, improved upper bound on the optimal value of -65.60.

Example 4 Consider the problem of Example 1 with $\mathcal{U} = \mathcal{U}_{\mathrm{Ball}}$. The optimal value of the original problem is -90.67, with worst-case 2-sparse scenario $(0, 0, 0, 0.9446, 0.3284)^\top$. Using the convex outer approximation $\mathcal{U} = \widehat{\mathcal{U}}_{\mathrm{Ball}}$ gives the upper bound -90.51, with non-sparse scenario $(0.0907, 0, 0.0907, 0.9508, 0.2819)^\top$.

Example 5 Consider the uncertain problem

$$\begin{aligned} \max\ & 8x_1 + 12x_2 \\ \text{s.t.}\ & 10x_1 + 20x_2 + h(u)\,(x_1 + x_2) \le 140, \\ & 6x_1 + 8x_2 \le 72, \\ & x_1, x_2 \ge 0, \end{aligned} \tag{32}$$

under the scaled budgeted set $\mathcal{B} = \{u \in \mathbb{R}^2 : \|u\|_\infty \le \frac{2}{3},\ \|u\|_1 \le 1\}$ and the sparsity-coupled set $\mathcal{U} = \{u \in \mathbb{R}^2 : \|u\|_\infty \le \frac{2}{3},\ \|u\|_0 \le 1\}$, where $h(u) = u^\top A u + 2c^\top u$ with $A = \left(\begin{smallmatrix} -2 & -1 \\ -1 & -2 \end{smallmatrix}\right)$ and $c^\top = (1, 0)$. Because $x_1 + x_2 \ge 0$, the worst-case constraint uses $\max_u h(u)$ over each set, and a direct computation gives

$$\max_{\mathcal{B}} h(u) = \tfrac{2}{3} \quad \text{and} \quad \max_{\mathcal{U}} h(u) = \tfrac{1}{2}.$$

Substituting these constants yields two deterministic linear programs whose optimal values are (to two decimals) 98.48 for $\mathcal{B}$ and 98.87 for $\mathcal{U}$. Here $\mathcal{U} \subseteq \mathcal{B}$, since any ℓ_∞-feasible vector with a single nonzero entry satisfies $\|u\|_1 \le \frac{2}{3} \le 1$. The unconstrained maximizer of the concave h is $u^\star = (\frac{2}{3}, -\frac{1}{3})^\top$, which is feasible for $\mathcal{B}$ but has two nonzero entries and is therefore excluded by $\|u\|_0 \le 1$. The worst-case over $\mathcal{U}$ is thus strictly smaller, giving a less conservative robust solution. Thus these budgeted and sparse sets yield different worst-case values.

Appendix B Data Sources and Instance Design

This appendix records the source or generator for each experiment. Signal recovery uses five fixed seeds, logistic regression uses five common initial points, and the portfolio study uses five nested profile libraries from a fixed public data window ending in 2025. The implementation error study uses the named public instances. Public DICHASUS data support the wireless experiments below.

Sparse Signal Recovery. We form $A = U\Sigma V^\top$, where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times m}$ have orthonormal columns obtained from Gaussian matrices. Geometrically spaced singular values control the condition number, after which every column of A is normalized. The support of x^{true} is sampled uniformly, its s nonzero values are standard Gaussian, and $b = Ax^{\mathrm{true}}$. The construction gives a known zero global residual for evaluation, but the algorithms receive only A, b, and s; they do not receive the support. We test $m \in \{100, 200\}$, $n \in \{30000, 50000\}$, and $s \in \{5, 10, 20\}$. Before column normalization, the singular values range geometrically from 1 to $1/50$.

Sparse Logistic Regression. The four datasets `a5a`, `a6a`, `a7a`, and `a8a` are taken from the LIBSVM repository [29, 50]. They share the feature dimension $n = 123$ and differ in sample size, ranging from roughly 6000 to 23000 observations. For each dataset we use sparsity levels $s \in \{5, 10, 20\}$, and each algorithm uses the same five random initial points x^0.

For both sparse optimization experiments, AGP uses $\epsilon = 10^{-5}$, $\delta = 10^{-10}$, $\beta = 2$, $\gamma_0 = 1/4$, and $\underline{\gamma} = 10^{-4}$. We set $u^{-1} = u^0 + 10^{-3}d$, where d is generated from a standard normal distribution with seed 23. PSS computes exact coordinate updates and considers a support exchange when the support is full. The coordinate problems have analytic solutions for least squares and are solved by bisection for logistic regression. The common initial point has independent entries sampled uniformly from $[0, 1)$. IHT uses stepsize $1/L$, with $L = 2\|A\|_2^2$ for signal recovery and $L = \|A\|_2^2/(4m)$ for logistic regression. The supplied files record the seed for every run.

Portfolio Optimization under Sparse Stress Profiles. We use the monthly value-weighted returns of the 49 Fama-French industry portfolios from January 1996 through December 2025. The annualized sample mean and covariance give μ and Σ. We rank the 360 months by their equal-weighted industry return and form nested libraries from the lowest $q = 24, 36, 54, 90$, and 108 months. Their rows are

$$B_{ri} = \max\{\bar{r}_i - r_{ri}, 0\}.$$

We set $Q = BB^\top/49 + 0.01\,\mathrm{Diag}(\mathrm{diag}(BB^\top/49))$, with every added diagonal entry bounded below by 10^{-10} to make Q positive definite, and use $\lambda = 0.5$ and $s = 3$. The smaller diagonal term retains the dense dependence among the historical profiles. The exact cutting-plane method uses the binary quadratic separation formulation with relative gap 10^{-6}, and each complete run has a limit of 1000 seconds. AGP and IHT use the same 15 starts: zero and 14 points obtained by optimizing over supports drawn with a fixed seed. AGP uses $\gamma_0 = 0.05$, $\underline{\gamma} = 10^{-4}$, $\delta = 10^{-10}$, and $\beta = 2$. Each sparse subproblem has at most 3000 iterations and tolerance 10^{-5}. After recording the method time, we evaluate every returned approximate portfolio exactly. Since $s = 3$, this evaluation enumerates the $\binom{q}{3}$ supports of size three and solves the concave quadratic problem on each support. In Table S3 of Online Resource, Tail is the percentage $100q/360$ of months included in the profile library, and Shortfall is the percentage by which the stress found during the method is below the exact stress for its returned portfolio. Evaluation time is excluded from method time. Each row is deterministic, so brackets are not used.

Linear Optimization with Multiplicative Implementation Error. SCFXM1, SCFXM3, 25FV47, WOODW, and FIT2D are read from the NETLIB test collection. All upper and lower inequality rows are robustified; lower rows are multiplied by -1. Equalities and variable bounds use their nominal values. For $L = 240, 480, 600, 1200$, and 1500, respectively, the deterministic loading matrix $D \in \mathbb{R}_+^{n\times L}$ assigns each variable to six sources of coefficient error with positive weights summing to one. Their seed is $20000 + 13n + L$. Variable i contributes $a_i^j x_i\{1 + \eta(Du)_i\}$ to row j, with $\eta = 0.01$.

There are $\lceil L/8 \rceil$ resource pools. Each source belongs to four pools, with deterministic coefficients in $[0.5, 1.5]$ generated using seed $10000 + L$, and every pool has capacity 2. We use $s \in \{10, 20\}$. EL keeps binary support indicators. RL relaxes the indicators, selects the strongest s components, and solves the continuous problem on that fixed support. In Table S4 of Online Resource, Bound is the lower bound returned by RL, and Gap is its percentage difference from the exact EL optimum. Exact violation is the largest excess over a robust constraint, computed by binary separation at the returned decision. Each method has at most 100 cutting-plane iterations, and every case terminates before this limit.

Wireless Downlink Planning. The wireless experiments use file `dichasus-d036` from the public DICHASUS outdoor data collection [34, 35]. It contains 532 position-tagged observations from a moving transmitter to 47 antenna elements distributed among three arrays over 1024 OFDM subcarriers at 1.272 GHz. The supplied conversion script divides the subcarriers into 32 consecutive blocks and stores the mean squared channel magnitude. For observation r, antenna element i, and frequency block b, let P_{rib} denote the corresponding mean squared magnitude of the channel.

We use all 47 measured antenna elements as separately controlled transmitters in one coordinated cluster (the 47 elements are not 47 physical base stations) and apply channel reciprocity to obtain the downlink gains. User observations are spread over the measured path by farthest point selection. For user k, h_{ki} is the average of P_{rib} over the 32 blocks and the row is divided by its largest entry. Potential jammers are selected without replacement from the remaining measured positions, and each is assigned

one frequency block. Thus, the measured transmitting element to jammer gain is $a_{ji} = P_{r_j i b_j}$. If d_{kj} is the distance between the recorded positions of user k and jammer j, we set

$$r_{kj} = 10^{-(32.4+20\log_{10}(\max\{d_{kj},1\})+20\log_{10}(1.272))/10}.$$

This is the free-space basic transmission-loss expression given in ITU-R P.1411-13 [41], evaluated at the measurement carrier frequency. The one-metre floor supplies a common reference distance for nearly coincident recorded positions. The unscaled two-hop cascade is $\widetilde{g}_{kij} = r_{kj} a_{ji}$.

We set $\bar{u}_j = 1$ and apply one common cascade scale. Under equal transmitter powers, let T_k contain the s jammers with the largest values of $\sum_i \widetilde{g}_{kij}$. We use the common scale

$$\eta = 0.75 \min_k \frac{\sum_i h_{ki}}{\theta \sum_{j\in T_k} \sum_i \widetilde{g}_{kij}}, \qquad g_{kij} = \eta \widetilde{g}_{kij}.$$

Jamming scale is fixed before optimization and independent of support.

The measured jammer coordinates and assigned frequency blocks are normalized and used to form q resource pools. Each jammer belongs to its four nearest pools. For a selected pool r, its positive coefficient is between 0.75 and 1.25 and increases with the normalized distance to the pool centre. Memberships are added only when necessary to give every pool at least six potential jammers, and $d_r = 3.20$ for every pool. We set $\sigma_k^2 = 0.003\,\mathrm{median}_\ell\{\sum_i h_{\ell i}\}$, $\theta = 0.35$, $\bar{p}_i = 80$, a cut batch of three, an 80-iteration limit, and a 10^{-6} violation tolerance. Mixed-integer separation uses relative gap 10^{-6}. Every method has a limit of 1000 seconds, with at most 180 seconds for each mixed-integer separation. The tuples $(n_{\mathrm{TX}}, n_{\mathrm{J}}, m_{\mathrm{UE}}, s, q)$ are

$$(47, 200, 20, 8, 67), \quad (47, 250, 30, 12, 84),$$
$$(47, 300, 40, 14, 100), \quad (47, 300, 50, 14, 100).$$

Each tuple contains three data selections. All transmitting elements belong to one coordinated cluster, and reactive-jammer interference remains proportional to the received downlink energy.

Adjustable Wireless Planning with Shared Recovery. The adjustable experiment constructs separate instances from the public DICHASUS file used in the static experiment. It uses the same procedure to form the measured channel gains, jammer cascade gains, shared jammer resource constraints, and jamming scale. For $K = m_{\mathrm{UE}}$ users, we define $2K$ protected recovery resources and arrange the users in groups of five. For each user, one shared resource supplies one unit of recovery to that user and to the next user in the same group, with the last user linked back to the first. The unit cost of this resource is 1.4. One individual resource supplies one unit of recovery only to that user and has a unit cost of 1. These coefficients define B and c. Both are fixed, while the allocation y is selected after u is observed. The same recovery construction is used for every value of K.

The small comparison uses $(n_{\mathrm{TX}}, m_{\mathrm{UE}}, n_{\mathrm{J}}, s, q) = (5, 5, 5, 3, 5)$ and subsets of the same DICHASUS measurements, with $\bar{p}_i = 20$. We enumerate every vertex of every polytope restricted to a support to solve the fully adjustable model and the exact affine policy. The larger groups use $\bar{p}_i = 80$ and are

$$(47, 25, 250, 12, 84), \ (47, 30, 300, 14, 100), \ (47, 35, 300, 14, 100).$$

Each group contains three selections from the measured data. The cutting-plane methods have at most 80 iterations and 1000 seconds. Each mixed-integer call has at most 180 seconds and relative gap 10^{-6}. Alternating maximization uses 10 starts and at most 30 iterations for the larger groups, and 30 starts and at most 50 iterations for the small group. DB-SP, DB-RL, and DB-EL give lower bounds. AADR-RL is the affine-policy solution over the RL polyhedron, with RO-EL as the static benchmark.

Online Resources

This supplement reports each run behind the summary displays; objective directions and gap definitions follow the corresponding experiments.

Table S1: Detailed sparse signal recovery results.

	AGP		IHT		PSS	
Instance (n, m, s)	Loss	Time (s)	Loss	Time (s)	Loss	Time (s)
#1 (30000,100,5)	0.5948	0.47	1.5639	7.88	0.2444	0.60
#2 (30000,100,5)	1.9787	0.17	4.0152	7.40	1.7916	0.42
#3 (30000,100,5)	0.2293	0.22	4.2776	7.33	1.8002	0.39
#4 (30000,100,5)	0.9656	0.33	3.0069	7.40	1.2840	0.43
#5 (30000,100,5)	1.5095	0.20	4.7572	7.50	2.2049	0.41
#6 (30000,100,10)	0.9182	0.33	4.9576	7.37	1.7959	1.14
#7 (30000,100,10)	1.1254	0.32	5.2040	7.57	1.7475	0.92
#8 (30000,100,10)	0.6663	0.36	10.0229	7.64	1.5928	1.00
#9 (30000,100,10)	0.7860	0.30	4.1987	7.53	0.6592	1.62
#10 (30000,100,10)	1.0733	0.29	4.6472	7.25	1.8403	1.42
#11 (30000,100,20)	1.4281	0.45	8.1492	7.34	2.2593	5.37
#12 (30000,100,20)	0.9696	0.24	8.1493	7.05	1.3207	4.52
#13 (30000,100,20)	0.7585	0.39	5.3076	6.90	0.9456	4.59
#14 (30000,100,20)	0.3322	0.28	3.5001	7.06	0.5627	4.32
#15 (30000,100,20)	0.6809	0.41	4.0033	7.04	1.8472	4.27
#16 (50000,100,5)	0	0.59	2.9128	12.93	0	1.20
#17 (50000,100,5)	3.1723	0.60	9.7422	13.49	0	0.93
#18 (50000,100,5)	0.6141	0.44	3.3429	13.78	1.1163	1.16
#19 (50000,100,5)	0.0514	0.59	2.5970	13.73	0.8185	1.11
#20 (50000,100,5)	1.2275	0.44	3.9558	13.50	1.0982	0.79
#21 (50000,100,10)	0.3876	0.70	5.4457	13.71	0.9353	2.94
#22 (50000,100,10)	0.6729	0.66	5.0327	13.39	0.7710	2.08
#23 (50000,100,10)	0.8712	0.65	8.0828	13.47	1.3397	1.49
#24 (50000,100,10)	1.1382	0.54	9.1042	13.39	1.3831	2.74
#25 (50000,100,10)	2.9287	0.58	11.7412	13.54	1.7181	2.51
#26 (50000,100,20)	0.8364	0.80	21.2350	13.29	1.7404	6.96
#27 (50000,100,20)	0.3878	0.68	8.5209	12.59	0.5509	6.54
#28 (50000,100,20)	0.8358	1.30	7.1836	12.43	1.2302	7.54
#29 (50000,100,20)	0.4033	0.59	11.0566	12.88	0.8892	8.53
#30 (50000,100,20)	1.6134	0.94	7.5934	12.44	1.2460	7.33
#31 (30000,200,5)	0	0.34	0.5219	11.88	0	0.67
#32 (30000,200,5)	0	0.29	1.2144	12.55	0	0.75
#33 (30000,200,5)	0	0.38	2.8769	12.04	0	0.73
#34 (30000,200,5)	0	0.29	2.6324	12.60	0	0.88
#35 (30000,200,5)	0	0.33	2.3946	11.78	2.1587	0.45
#36 (30000,200,10)	0.8802	0.67	6.4024	12.84	1.6985	2.03
#37 (30000,200,10)	0.2872	0.44	1.7014	13.08	0.2043	4.25
#38 (30000,200,10)	1.4350	0.34	2.9027	12.06	1.2225	1.64
#39 (30000,200,10)	0	0.54	1.8687	12.45	0.4080	2.81
#40 (30000,200,10)	0	0.36	2.9824	12.08	1.5494	1.55
#41 (30000,200,20)	2.0881	0.72	6.6531	12.19	3.1459	4.77
#42 (30000,200,20)	1.3094	0.59	5.4618	11.94	2.6801	3.83
#43 (30000,200,20)	1.0016	0.56	4.3891	12.08	1.7395	6.65
#44 (30000,200,20)	1.1459	0.50	3.1816	12.45	1.8090	4.13
#45 (30000,200,20)	0.7907	0.96	3.7111	12.20	1.6622	4.94
#46 (50000,200,5)	0	0.93	5.6571	23.13	0	1.12
#47 (50000,200,5)	0	0.84	1.3331	23.52	0	1.19
#48 (50000,200,5)	0.9349	0.81	1.8573	23.43	0	1.26
#49 (50000,200,5)	0	0.84	6.9092	23.37	0	1.06
#50 (50000,200,5)	0	1.03	3.6455	23.57	1.3511	1.25
#51 (50000,200,10)	1.4390	1.29	6.3916	23.43	1.7590	2.77
#52 (50000,200,10)	0.4809	1.59	5.3121	23.31	0.6220	4.57
#53 (50000,200,10)	0	1.30	11.6907	23.22	1.8477	3.47
#54 (50000,200,10)	0	1.05	4.5230	23.49	0.3142	4.42
#55 (50000,200,10)	0	1.22	2.5388	23.29	0.3548	4.50
#56 (50000,200,20)	1.9559	1.15	4.2751	23.14	2.6981	11.01
#57 (50000,200,20)	1.6358	1.68	7.8231	23.29	3.6864	5.96
#58 (50000,200,20)	2.0973	1.67	11.9856	23.75	3.9059	7.94
#59 (50000,200,20)	1.1412	1.37	6.4544	23.81	2.5841	8.56
#60 (50000,200,20)	0.2475	1.48	2.4640	23.67	1.6092	13.51

Table S2: Detailed sparse logistic regression results on LIBSVM data sets.

	AGP		IHT		PSS	
Instance (data, m, n, s)	Loss	Time (s)	Loss	Time (s)	Loss	Time (s)
#1 (a5a, 6414, 123, 5)	0.3758	0.20	0.3841	0.95	0.3892	0.23
#2 (a5a, 6414, 123, 5)	0.3758	0.07	0.3841	0.88	0.3719	0.43
#3 (a5a, 6414, 123, 5)	0.3941	0.13	0.5038	0.87	0.3806	0.60
#4 (a5a, 6414, 123, 5)	0.3758	0.07	0.4333	0.86	0.3719	0.40
#5 (a5a, 6414, 123, 5)	0.3806	0.12	0.5206	0.86	0.3719	0.68
#6 (a5a, 6414, 123, 10)	0.3510	1.50	0.3758	0.78	0.3490	0.80
#7 (a5a, 6414, 123, 10)	0.3559	0.66	0.3673	0.89	0.3607	1.18
#8 (a5a, 6414, 123, 10)	0.3522	1.39	0.3587	0.89	0.3632	1.44
#9 (a5a, 6414, 123, 10)	0.3489	1.41	0.3725	0.91	0.3552	1.39
#10 (a5a, 6414, 123, 10)	0.3550	0.56	0.3725	0.90	0.3513	1.07
#11 (a5a, 6414, 123, 20)	0.3358	1.51	0.3547	0.83	0.3409	2.65
#12 (a5a, 6414, 123, 20)	0.3366	0.43	0.3439	0.93	0.3445	3.55
#13 (a5a, 6414, 123, 20)	0.3375	1.52	0.3545	0.84	0.3414	2.74
#14 (a5a, 6414, 123, 20)	0.3361	1.47	0.3584	0.91	0.3400	3.39
#15 (a5a, 6414, 123, 20)	0.3397	0.65	0.3585	0.83	0.3406	3.26
#16 (a6a, 11220, 123, 5)	0.3753	0.19	0.3826	1.64	0.3897	0.50
#17 (a6a, 11220, 123, 5)	0.3753	0.20	0.3826	1.71	0.3714	0.83
#18 (a6a, 11220, 123, 5)	0.3924	0.30	0.5002	1.63	0.3786	1.15
#19 (a6a, 11220, 123, 5)	0.3753	0.17	0.5124	1.62	0.3714	0.88
#20 (a6a, 11220, 123, 5)	0.3786	0.18	0.4321	1.64	0.3714	1.31
#21 (a6a, 11220, 123, 10)	0.3499	1.46	0.3826	1.67	0.3497	1.96
#22 (a6a, 11220, 123, 10)	0.3495	0.53	0.3650	1.69	0.3666	1.89
#23 (a6a, 11220, 123, 10)	0.3485	1.00	0.3607	1.71	0.3558	2.18
#24 (a6a, 11220, 123, 10)	0.3515	0.35	0.3786	1.74	0.3587	2.65
#25 (a6a, 11220, 123, 10)	0.3555	0.92	0.3786	1.72	0.3521	2.07
#26 (a6a, 11220, 123, 20)	0.3371	1.89	0.3524	1.74	0.3418	5.01
#27 (a6a, 11220, 123, 20)	0.3371	1.41	0.3445	1.65	0.3475	9.83
#28 (a6a, 11220, 123, 20)	0.3364	1.81	0.3534	1.74	0.3399	5.91
#29 (a6a, 11220, 123, 20)	0.3366	3.01	0.3617	1.75	0.3407	5.93
#30 (a6a, 11220, 123, 20)	0.3419	1.78	0.3600	1.74	0.3406	43.42
#31 (a7a, 16100, 123, 5)	0.3787	0.26	0.3839	2.50	0.3726	1.79
#32 (a7a, 16100, 123, 5)	0.3787	0.41	0.3853	2.64	0.3726	1.25
#33 (a7a, 16100, 123, 5)	0.3918	0.49	0.5043	2.45	0.3726	1.36
#34 (a7a, 16100, 123, 5)	0.3799	0.48	0.5140	2.47	0.3726	1.19
#35 (a7a, 16100, 123, 5)	0.3799	0.46	0.5176	2.46	0.3726	1.85
#36 (a7a, 16100, 123, 10)	0.3547	2.03	0.3839	2.53	0.3565	2.30
#37 (a7a, 16100, 123, 10)	0.3493	1.07	0.3729	2.63	0.3666	3.58
#38 (a7a, 16100, 123, 10)	0.3582	2.55	0.3761	2.63	0.3655	3.75
#39 (a7a, 16100, 123, 10)	0.3547	2.41	0.3796	2.70	0.3606	3.92
#40 (a7a, 16100, 123, 10)	0.3491	1.30	0.3797	2.68	0.3536	3.29
#41 (a7a, 16100, 123, 20)	0.3396	2.89	0.3598	2.66	0.3441	6.34
#42 (a7a, 16100, 123, 20)	0.3385	1.80	0.3492	2.75	0.3431	28.21
#43 (a7a, 16100, 123, 20)	0.3374	2.72	0.3605	2.58	0.3447	7.55
#44 (a7a, 16100, 123, 20)	0.3371	2.66	0.3608	2.63	0.3419	8.59
#45 (a7a, 16100, 123, 20)	0.3408	1.35	0.3566	2.59	0.3417	26.33
#46 (a8a, 22696, 123, 5)	0.3782	0.46	0.4057	3.55	0.3722	2.52
#47 (a8a, 22696, 123, 5)	0.3782	0.52	0.4048	3.69	0.3722	1.81
#48 (a8a, 22696, 123, 5)	0.3909	0.71	0.5028	3.54	0.3722	1.92
#49 (a8a, 22696, 123, 5)	0.3782	0.68	0.5101	3.50	0.3722	1.79
#50 (a8a, 22696, 123, 5)	0.3918	0.37	0.5148	3.54	0.3722	2.55
#51 (a8a, 22696, 123, 10)	0.3539	3.98	0.3912	3.54	0.3561	3.10
#52 (a8a, 22696, 123, 10)	0.3484	0.57	0.3727	3.72	0.3662	4.80
#53 (a8a, 22696, 123, 10)	0.3554	3.31	0.3755	3.77	0.3601	5.49
#54 (a8a, 22696, 123, 10)	0.3591	5.65	0.3791	3.96	0.3655	5.58
#55 (a8a, 22696, 123, 10)	0.3592	3.56	0.3791	3.94	0.3535	4.55
#56 (a8a, 22696, 123, 20)	0.3405	6.81	0.3590	4.02	0.3461	11.37
#57 (a8a, 22696, 123, 20)	0.3368	2.64	0.3460	4.03	0.3455	11.17
#58 (a8a, 22696, 123, 20)	0.3373	6.75	0.3697	4.02	0.3437	12.90
#59 (a8a, 22696, 123, 20)	0.3373	5.82	0.3535	4.00	0.3447	17.65
#60 (a8a, 22696, 123, 20)	0.3387	2.86	0.3566	4.08	0.3432	88.31

Table S3: Detailed portfolio results for the 49 Fama-French industry portfolios.

Tail	(q, s)	Quantity	Exact	AGP	IHT
6.7%	$(24, 3)$	Status	Optimal		
		Objective	-0.608601	-0.680045	-0.867723
		Time (s)	13.347	0.444	0.117
		Iterations	125	17	27
		Cuts	124	148	193
		Shortfall (%)		13.33	33.13
		Evaluation time (s)		0.0004	0.0004
10%	$(36, 3)$	Status	Optimal		
		Objective	-0.682606	-0.741690	-0.817179
		Time (s)	65.761	0.297	0.209
		Iterations	172	35	35
		Cuts	171	251	227
		Shortfall (%)		10.32	20.52
		Evaluation time (s)		0.0014	0.0014
15%	$(54, 3)$	Status	Optimal		
		Objective	-0.730607	-0.808257	-1.289916
		Time (s)	527.869	0.303	0.218
		Iterations	237	25	32
		Cuts	236	190	210
		Shortfall (%)		15.22	52.58
		Evaluation time (s)		0.0045	0.0047
25%	$(90, 3)$	Status	Time limit		
		Objective	n/a	-0.953634	-0.982332
		Time (s)	1000.007	0.635	0.575
		Iterations	56	34	35
		Cuts	55	300	333
		Shortfall (%)		21.03	29.12
		Evaluation time (s)		0.0307	0.0307
30%	$(108, 3)$	Status	Time limit		
		Objective	n/a	-1.078418	-1.380892
		Time (s)	1000.000	0.930	0.261
		Iterations	n/a	28	22
		Cuts	n/a	235	183
		Shortfall (%)		31.39	55.70
		Evaluation time (s)		0.0437	0.0450

Table S4: Detailed implementation error results on NETLIB test instances.

Instance	(n, m, L)	s	Quantity	Exact (EL)	RL lower bound
SCFXM1	$(457, 143, 240)$	10	Value	18591.9499	18589.4484
			Time (s)	4.997	0.856
			Iterations	4	5
			Cuts	85	87
			Exact violation	0.00	6.16228
			Gap (%)		0.01346
		20	Value	18597.1073	18597.1073
			Time (s)	2.935	1.363
			Iterations	7	8
			Cuts	112	115

Continued on next page

Table S4: Detailed implementation error results (continued).

Instance	(n, m, L)	s	Quantity	Exact (EL)	RL lower bound
			Exact violation	0.00	0.00
			Gap (%)		0.00
SCFXM3	$(1371, 429, 480)$	10	Value	55563.4905	55553.5742
			Time (s)	13.171	4.201
			Iterations	6	5
			Cuts	280	271
			Exact violation	0.00	4.96634
			Gap (%)		0.01785
		20	Value	55658.6671	55646.4813
			Time (s)	34.924	6.677
			Iterations	12	8
			Cuts	361	307
			Exact violation	0.00	10.68175
			Gap (%)		0.02189
25FV47	$(1571, 305, 600)$	10	Value	5601.5376	5600.9651
			Time (s)	26.857	8.651
			Iterations	13	11
			Cuts	235	217
			Exact violation	0.00	0.69740
			Gap (%)		0.01022
		20	Value	5615.4199	5613.9875
			Time (s)	58.621	13.468
			Iterations	14	16
			Cuts	317	279
			Exact violation	0.00	1.00643
			Gap (%)		0.02551

Continued on next page

Table S4: Detailed implementation error results (continued).

Instance	(n, m, L)	s	Quantity	Exact (EL)	RL lower bound
WOODW	$(8405, 13, 1200)$	10	Value	1.3148	1.3147
			Time (s)	2.166	0.861
			Iterations	11	9
			Cuts	12	10
			Exact violation	0.00	0.00008
			Gap (%)		0.00684
		20	Value	1.3189	1.3188
			Time (s)	2.719	1.186
			Iterations	11	11
			Cuts	12	12
			Exact violation	0.00	0.00007
			Gap (%)		0.00633
FIT2D	$(10500, 24, 1500)$	10	Value	-68449.1238	-68449.3366
			Time (s)	9.693	7.268
			Iterations	2	2
			Cuts	19	19
			Exact violation	0.00	0.88072
			Gap (%)		0.00031
		20	Value	-68438.4040	-68438.9662
			Time (s)	17.885	9.356
			Iterations	3	3
			Cuts	24	22
			Exact violation	0.00	3.71566
			Gap (%)		0.00082

Table S5: Detailed static wireless results from the DICHASUS measurements.

Instance	Tuple	Quantity	SP	RL	EL
1.1	$(47, 200, 20, 8, 67)$	Objective	0.02491	0.03710	0.03776
		Time (s)	0.40	0.27	8.91
		Iterations	5	5	4
		Cuts	8	9	9
		Exact violation	0.00667	0.00034	0.00
1.2	$(47, 200, 20, 8, 67)$	Objective	0.02506	0.03803	0.03803
		Time (s)	0.24	0.21	10.27
		Iterations	4	4	5
		Cuts	7	9	11
		Exact violation	0.00589	0.00	0.00
1.3	$(47, 200, 20, 8, 67)$	Objective	0.02448	0.03624	0.03655
		Time (s)	0.23	0.22	8.77
		Iterations	4	4	4
		Cuts	8	8	7
		Exact violation	0.00636	0.00015	0.00
2.1	$(47, 250, 30, 12, 84)$	Objective	0.02387	0.03102	0.03201
		Time (s)	0.46	0.48	28.75
		Iterations	4	5	6
		Cuts	7	12	13
		Exact violation	0.00445	0.00084	0.00

Continued on next page

Table S5: Detailed static wireless results (continued).

Instance	Tuple	Quantity	SP	RL	EL
2.2	$(47, 250, 30, 12, 84)$	Objective	0.02443	0.03472	0.03511
		Time (s)	0.72	0.39	19.97
		Iterations	5	4	4
		Cuts	9	9	9
		Exact violation	0.00577	0.00026	0.00
2.3	$(47, 250, 30, 12, 84)$	Objective	0.02558	0.03667	0.03674
		Time (s)	0.56	0.29	21.03
		Iterations	5	3	4
		Cuts	9	6	8
		Exact violation	0.00554	0.00004	0.00

Continued on next page

Table S5: Detailed static wireless results (continued).

Instance	Tuple	Quantity	SP	RL	EL
3.1	$(47, 300, 40, 14, 100)$	Objective	0.02490	0.03374	0.03439
		Time (s)	0.72	0.78	55.68
		Iterations	4	5	7
		Cuts	7	11	16
		Exact violation	0.00524	0.00043	0.00
3.2	$(47, 300, 40, 14, 100)$	Objective	0.02705	0.03468	0.03493
		Time (s)	0.96	0.62	35.05
		Iterations	4	4	4
		Cuts	7	8	8
		Exact violation	0.00378	0.00021	0.00
3.3	$(47, 300, 40, 14, 100)$	Objective	0.02440	0.03292	0.03333
		Time (s)	0.75	0.72	34.72
		Iterations	4	4	4
		Cuts	7	9	8
		Exact violation	0.00544	0.00042	0.00
4.1	$(47, 300, 50, 14, 100)$	Objective	0.02521	0.03312	0.03355
		Time (s)	1.63	1.17	64.53
		Iterations	7	6	6
		Cuts	12	14	15
		Exact violation	0.00448	0.00030	0.00
4.2	$(47, 300, 50, 14, 100)$	Objective	0.03023	0.03865	0.03943
		Time (s)	0.46	0.77	57.42
		Iterations	2	4	5
		Cuts	3	9	10
		Exact violation	0.00392	0.00052	0.00
4.3	$(47, 300, 50, 14, 100)$	Objective	0.02547	0.03326	0.03386
		Time (s)	1.16	0.77	55.51
		Iterations	5	4	5
		Cuts	10	9	10
		Exact violation	0.00531	0.00038	0.00

Table S6: Detailed adjustable wireless lower bounds from DICHASUS data.

Instance	Tuple	Quantity	DB-SP	DB-RL	DB-EL
1.1	$(47, 25, 250, 12, 84)$	Bound	0.02355	0.02887	0.02943
		Time (s)	6.28	1.24	31.25
		Iterations	14	8	10
		Cuts	14	8	10
		Final violation	0.00	0.00	0.00
1.2	$(47, 25, 250, 12, 84)$	Bound	0.02408	0.02808	0.02815
		Time (s)	5.56	1.76	39.12
		Iterations	6	12	10
		Cuts	6	12	10
		Final violation	0.00	0.00	0.00
1.3	$(47, 25, 250, 12, 84)$	Bound	0.02445	0.02904	0.02936
		Time (s)	4.34	1.54	29.97
		Iterations	7	10	9
		Cuts	7	10	9
		Final violation	0.00	0.00	0.00
2.1	$(47, 30, 300, 14, 100)$	Bound	0.02386	0.02904	0.02912
		Time (s)	8.17	2.97	65.17
		Iterations	13	14	15
		Cuts	13	14	15
		Final violation	0.00	0.00	0.00
2.2	$(47, 30, 300, 14, 100)$	Bound	0.02420	0.02985	0.03221
		Time (s)	3.06	1.29	49.26
		Iterations	7	7	10
		Cuts	7	7	10
		Final violation	0.00	0.00	0.00

Continued on next page

Table S6: Detailed adjustable wireless lower bounds (continued).

Instance	Tuple	Quantity	DB-SP	DB-RL	DB-EL
2.3	$(47, 30, 300, 14, 100)$	Bound	0.02421	0.03214	0.03299
		Time (s)	7.18	2.01	54.75
		Iterations	13	10	13
		Cuts	13	10	13
		Final violation	0.00	0.00	0.00
3.1	$(47, 35, 300, 14, 100)$	Bound	0.02397	0.03083	0.03103
		Time (s)	13.23	2.75	56.42
		Iterations	16	13	13
		Cuts	16	13	13
		Final violation	0.00	0.00	0.00
3.2	$(47, 35, 300, 14, 100)$	Bound	0.02492	0.03131	0.03205
		Time (s)	2.93	1.80	46.85
		Iterations	7	9	13
		Cuts	7	9	13
		Final violation	0.00	0.00	0.00
3.3	$(47, 35, 300, 14, 100)$	Bound	0.02465	0.03193	0.03226
		Time (s)	3.52	1.67	41.93
		Iterations	8	8	8
		Cuts	8	8	8
		Final violation	0.00	0.00	0.00

Table S7: Small-instance adjustable wireless results with an exact reference.

Instance	Quantity	Exact Full	DB-SP	DB-RL	DB-EL	AADR Exact	RO-EL
1	Objective	0.00927	0.00927	0.00927	0.00927	0.00927	0.01208
	Time (s)	0.33	0.91	0.62	2.03	0.03	0.00
	Iterations	1	4	4	4	1	1
	Cuts	0	4	4	4	0	0
2	Objective	0.00755	0.00755	0.00755	0.00755	0.00755	0.00949
	Time (s)	0.01	0.66	0.47	1.39	0.01	0.00
	Iterations	1	4	4	4	1	1
	Cuts	0	4	4	4	0	0
3	Objective	0.00861	0.00861	0.00861	0.00861	0.00883	0.01137
	Time (s)	0.01	1.14	1.21	2.63	0.01	0.00
	Iterations	1	9	9	9	1	1
	Cuts	0	9	9	9	0	0

Table S8: Affine and static policies for the adjustable DICHASUS instances.

Instance	Tuple	Quantity	AADR-RL	RO-EL
1.1	$(47, 25, 250, 12, 84)$	Value	0.03050	0.03072
		Time (s)	5.41	13.64
		Iterations		2
		Cuts		25
1.2	$(47, 25, 250, 12, 84)$	Value	0.02953	0.02989
		Time (s)	4.96	12.72
		Iterations		2
		Cuts		21
1.3	$(47, 25, 250, 12, 84)$	Value	0.02997	0.03075
		Time (s)	7.32	12.57

Continued on next page

Table S8: Affine and static policy results (continued).

Instance	Tuple	Quantity	AADR-RL	RO-EL
		Iterations		2
		Cuts		27
2.1	$(47, 30, 300, 14, 100)$	Value	0.03050	0.03105
		Time (s)	12.75	20.23
		Iterations		2
		Cuts		29
2.2	$(47, 30, 300, 14, 100)$	Value	0.03340	0.03407
		Time (s)	11.21	18.98
		Iterations		2
		Cuts		30
2.3	$(47, 30, 300, 14, 100)$	Value	0.03392	0.03542
		Time (s)	10.76	18.42
		Iterations		2
		Cuts		28
3.1	$(47, 35, 300, 14, 100)$	Value	0.03192	0.03247
		Time (s)	11.14	21.15
		Iterations		2
		Cuts		35
3.2	$(47, 35, 300, 14, 100)$	Value	0.03457	0.03529
		Time (s)	12.12	23.48
		Iterations		2
		Cuts		37
3.3	$(47, 35, 300, 14, 100)$	Value	0.03312	0.03414
		Time (s)	13.18	23.19
		Iterations		2
		Cuts		32