Information Basis in Dynamic Robust Optimization

Ningji Wei
ningjiwei@cmu.edu

Peter Zhang
pyzhang@cmu.edu

Abstract

Dynamic robust optimization deals with sequential, multi-stage decisions in the face of uncertain, worst-case scenarios. To manage its complexity and the curse of dimensionality, decision rules simplify the search for an optimal policy. This paper explores a middle ground between two common decision rules: simple but imprecise constant policies, and accurate but less scalable affine policies. We introduce a method that achieves linear convergence to the performance of optimal affine policies by iteratively enriching the information basis with new components. This leads to near-optimal affine policies with a compact information basis. Additionally, we offer an efficient way to build this basis, with time complexity similar to random generation. This advancement unlocks potential benefits in applications such as approximating large-scale dynamic robust optimization problems or constructing nonlinear decision rules with a small number of parameters. Our proposed approach is also amenable to parallel computations on GPUs/TPUs.

Keywords: robust optimization, affine policy, linear decision rule, information basis

1 Introduction

Dynamic robust optimization focuses on addressing multi-stage problems under uncertainty where decisions and uncertainty parameters are sequentially revealed over time. Many decision problems, such as inventory management and manufacturing planning, exhibit such sequential adaptive decision-making nature, which can be generally described as follows,

$$\left(\min_{y_1\in\mathcal{Y}_1}\max_{\xi_1\in\Xi_1(\eta_1)}\min_{y_2\in\mathcal{Y}_2(h_2)}\max_{\xi_2\in\Xi_2(\eta_2)}\cdots\min_{y_\tau\in\mathcal{Y}_\tau(h_\tau)}\max_{\xi_\tau\in\Xi_\tau(\eta_\tau)}\right) f(\xi,y). \tag{1}$$

With τ as the final stage, $\eta_t := (\xi_i)_{i<t}$ and $h_t := (\xi_i, y_i)_{i<t}$ are the uncertainty revealing history and the joint decision-uncertainty history, respectively. In this setting, decisions will not influence uncertainty, while uncertainty realizations may affect future solution spaces.

This class of problems is generally intractable due to its multi-stage interactions, space dependencies, and the curse of dimensionality [4, 5]. Consequently, a widely adopted and studied approach to tackle this challenge is a general approximation scheme based on policy space parameterization [23]. Consider the following reformulation equivalent to (1),

$$\min_{y(\cdot)\in\mathcal{Y}_\Xi}\max_{\xi\in\Xi} f(\xi,y), \tag{2}$$

where Ξ is the joint uncertainty set and $\mathcal{Y}_\Xi := \prod_{\xi\in\Xi}\mathcal{Y}_\xi$ contains all the history-dependent feasible policies with $\mathcal{Y}_\xi$ as the joint decision space for a fixed ξ. Clearly, solving (1) is equivalent to searching for an optimal policy within the policy space $\mathcal{Y}_\Xi$. Indeed, the optimal solution of (1)

can be considered as a specific policy obtained by observing the sequential worst-case uncertainty realizations. From the perspective of (2), *policy families* (also known as the decision rules) can be introduced to establish various types of trade-offs between tractability and optimality. A policy family is defined as a function $\theta : \mathcal{P} \times \Xi \to \mathcal{Y}$ that parameterizes a subset of $\mathcal{Y}_\Xi$ using some Euclidean space $\mathcal{P}$. Among various policy families, the constant and affine policy families specified below have gained popularity in various problem settings [1, 2, 6, 8, 9, 10, 14, 15, 16, 19, 20].

$$\text{Constant:} \quad \dot{\theta}(p_0, \xi) = p_0, \qquad p_0 \in \dot{\mathcal{P}} \tag{3a}$$

$$\text{Affine:} \quad \bar{\theta}(\bar{P}, \xi) = p_0 + P^\intercal \xi, \qquad \bar{P} = (p_0, P) \in \bar{\mathcal{P}} = \dot{\mathcal{P}} \times \mathcal{P} \tag{3b}$$

The former is often efficient to solve, yet its optimality condition is quite restrictive [22]. As a result, it rarely achieves good approximation performance. On the other hand, the latter has proven to achieve optimal or near-optimal policies in many problem settings [7, 11]. However, it suffers from scalability issues when τ is reasonably large or when the decisions are associated with complex structures like networks [13]. For instance, in a multi-location newsvendor problem [20] with n inventory nodes and k (uncertain) demand nodes, we have $k\tau$ uncertain parameters and $n\tau$ decision variables. Thus, the dimension of $\mathcal{P}$ is of order $O(nk\tau^2)$. If we model a resilient network design problem with n nodes, $O(n^2)$ edges, and supply / transportation risks on nodes and edges, the number of adjustable decisions is $O(n^2\tau)$, the number of uncertain parameters is $O(n^2\tau)$, and the dimension of $\mathcal{P}$ is of order $O(n^4\tau^2)$. Therefore, evenly though polynomial in the input dimensions, these problems quickly become intractable in practice. In the authors' experience working with supply chain companies, the task of designing resilient supply chain networks is an important but open problem, because real network design problems often involve at least 10^3 to 10^4 nodes.

The apparent gap between these two policy families has recently motivated two research branches to identify some "sweet spot" policy families in between: (i) entry-wise sparsity that focuses on policies with a small number of nonzeros in P (either naturally due to the problem setup and time dependency, or imposed by a user to reduce computational burden); (ii) information basis that searches for a small set of matrices in $\mathcal{P}$ to span a policy subspace. To date, considerable attention has been dedicated to (i), yielding intriguing findings [4, 8, 11, 18]. In contrast, much less effort has been made to study the information basis. By our knowledge, we are among the first to explicitly discuss the design of (sparse) information basis, even though the concept of information basis was already discussed when affine decision rules are first applied to robust optimization problems [5].

While the study of entry-wise sparsity brings mostly computational benefits, the search for efficient information basis represents an orthogonal focus on reducing computational complexity and identifying the "principal components" of the policy space with respect to the problem data (cost and constraint coefficients, parameter uncertainty coefficients). This paper incorporates ex-ante entry-wise sparsity and ex-post information basis sparsity to examine general problem instances characterized by linear objectives, linear constraints, and polyhedral uncertainty sets. The key findings are summarized as follows:

> *With proper construction, a compact-sized basis can yield a near-optimal affine policy. With proper implementation, the complexity of constructing such a basis is equivalent to random generation.*

We will give rigorous statements about how "compact" this basis can be. For now, we want to mention that the first part of the findings leads to an algorithm that has *low iteration count*, since a basis with a small number of components only requires a small number of steps to be generated.

The second half of the findings show that we can also construct this algorithm to have a *low computational cost within each iteration*. Thus, these two findings together enable us to design an iterative solution algorithm that has low overall time complexity.

Given the close relationship between entry-wise sparsity and information basis, we will review these topics and their connections to policy sparsity in the following subsections.

1.1 Sparsity in Affine Policy Family

Applying affine policy family in the dynamic robust optimization (2) often results in a massive formulation even for moderate-sized problems [13]. The study of sparsity in affine policy family has been brought up to address this issue. The main idea is to significantly reduce the dimension of $\mathcal{P}$ while still preserving an adequate or even optimal affine policy performance. This leads to two highly related yet different approaches.

1.1.1 Entry-wise Sparsity

A constant policy is a special type of affine policy where all entries in P are set to zero. This perspective naturally leads to the concept of *entry-wise sparsity* where only a small portion of the parameters in matrix P in (3b) can be nonzero. Indeed, varying the number of nonzero entries in P exhibits a transition between constant and affine policies.

In specific problems, the design of entry-wise sparse policies can often be justified by the problem's inherent uncertainty dependency. For instance, in Problem (1), any valid matrix P must be history-dependent. Hence, for the nontrivial case $\tau > 1$, the transpose matrix $P^\intercal$ is a block-lower-triangular matrix (*e.g.*, the following structure when $\tau = 5$ where white area is all zeros). This shape can be considered as an *information filter* added on top of the parameter space $\mathcal{P}$. In this

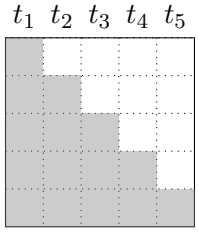


Figure 1: A Time-Dependent Information Filter

case, the optimal affine policy is guaranteed to be sparse in this block-lower-triangular shape. More intricate sparsity filters can be derived for specific problem structures with a certain optimality guarantee. El Housni and Goyal [11] showed that sparse affine policies can achieve an $O(\frac{\log n}{\log \log n})$-approximation in two-stage robust optimization problems with budgeted uncertainty sets. Recently, Lu and Sturt [18] proved that, for the class of multi-stage robust production-inventory problems, there exist optimal affine policies that are entry-wise sparse.

In general, sparse-and-optimal policies rarely exist or are difficult to prove for many practical problems. Even in these cases, many decision-makers would still prefer sparse policies if the trade-off on the optimality side is tolerable. This enables the design of various information filters based on the analysis of historical data, or the experience and knowledge of domain experts. For instance, the various filters in Figure 2 suggest different types of uncertainty dependencies.

Filters (a) and (b) both have column-wise sparsity, where the former indicates that only the uncertainty parameters in a few decision stages are important for the policy design, and the latter

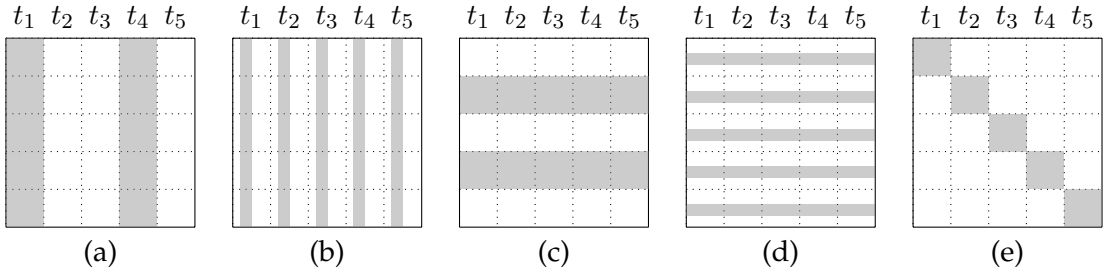


Figure 2: Various Information Filters

implies only certain uncertainty features in each decision stage are crucial. In contrast, filters (c) and (d) focus on limiting the adjustable decisions, where the former only allows the decisions in certain stages to be adjustable, and the latter limits the adjustability to a few number of decision variables in each stage. Finally, filter (e) suggests a Markovian-like dependency, *i.e.*, the future decisions only depend on the current uncertainty realization regardless of the past. All these information filters can be further combined by intersection. For instance, the time-dependent filter plus filter (d) will produce a time-dependent policy family with sparse adjustable decisions.

1.1.2 Information Basis Sparsity

The entry-wise sparsity, in fact, serves two purposes at once. On the one hand, the design of the information filters incorporates prior knowledge, which often makes the resulting policy user-friendly: it has simple interpretations due to the sparse dependencies, and the solutions can be easily implemented when only a few decisions are adjustable. On the other hand, these information filters also reduce the dimension of the parameter space $\mathcal{P}$, improving the efficiency of searching the optimal policy within the filtered policy space.

For the second purpose, however, the entry-wise sparsity can be further generalized. Take any set of linearly independent matrices $\{T_i\}_{i\in[d]} \subseteq \mathcal{P}$, we can use it to span the subspace

$$\mathcal{T} := \operatorname{span}\{T_i\}_{i\in[d]} = \left\{ \sum_{i\in[d]} p_i T_i \mid p_i \in \mathbb{R} \right\},$$

where the set $\{T_i\}$ is called the *information basis*. We say that an information basis is sparse if $d \ll \dim \mathcal{P}$. Clearly, every information filter (entry-wise sparsity) is associated with a specific information basis. Let E_{ij} be the matrix with one at the (i,j)th entry and zeros otherwise. Then, all such matrices $\{E_{ij}\}$ form the standard basis for $\mathcal{P}$ in (3b). Thus, every entry-wise sparse policy family chooses some subset of $\{E_{ij}\}$ to span its policy space.

Compared to the entry-wise sparsity, the extra liberty in designing the basis matrices T_i's enables the possibility to obtain a near-optimal affine policy with a much fewer number of parameters. For instance, when $d = 1$, any basis matrix T_1 yields the following policy family.

$$\theta_{T_1}(p_0, p_1, \xi) = p_0 + p_1 T_1 \xi,$$

where p_1 is a scalar parameter. With just one more degree of freedom than the constant policy family, this new policy family may contain a much better policy with a proper design of T_1.

The concept of information basis was first introduced in the seminal book [5]. However, to the authors' best knowledge, this topic has yet to be explored further. One possibility is that there are no obvious universal rules for constructing the information basis properly.

1.2 Contributions

In this paper, we focus on designing an algorithm that integrates ex-ante entry-wise sparsity and ex-post information basis sparsity to produce a near-optimal affine policy. We restrain our scope to dynamic robust optimization problems with linear structures, where both the objective function and constraints are linear, and the uncertainty set is a polyhedron. Our contributions are as follows:

- We use pre-defined information filters to incorporate prior structural knowledge and reduce the affine policy space to the filtered space. Then, we identify a natural transition between the constant and filtered policy spaces using information bases.
- We design a procedure to construct an information basis iteratively (Part 1). We prove that this construction is optimal for random instances and show that it converges exponentially to the optimal affine policy. That is, a compact-sized basis can produce a near-optimal affine policy. This leads to an intermediate algorithm with low iteration count (but not necessarily low overall efficiency due to potentially high computational cost per iteration).
- By incorporating a key component, *stochastic coordinate descent* (SCD) [17], we improve the efficiency per iteration and therefore the overall efficiency (Part 2). Specifically, the runtime complexity of the overall method is equivalent to randomly generating an information basis, and its most time-consuming operations can be further accelerated by graphical processing units (GPUs) or tensor processing units (TPUs).

The rest of the paper is organized as follows: Section 2 introduces the preliminaries, including the notation set, problem setting, and a standard reformulation method. Section 3 presents our Part 1 algorithm for constructing an information basis. Section 4 justifies our construction and proves the convergence of policy performance. Section 5 describes Part 2 of the algorithm and an efficient and scalable implementation. Section 6 presents experiments to test the proposed algorithms. Finally, in Section 7, we conclude the paper with further discussions.

2 Preliminaries

2.1 Notation

Given a matrix A, we use $A^\dagger$ for the Moore-Penrose inverse of A, and $\mathrm{vec}(A)$ is the vectorization of A by vertically stacking the columns of A. The reverse operation $\mathrm{vec}_k^{-1}(v)$ for some vector $v \in V$ and $k \in \mathbb{Z}_+$ evenly cuts the vector v into k segments and horizontally stack them into a matrix, which is well-defined only if the size of v is divisible by k.

A 3-tensor T of shape $l \times m \times n$ can be considered a three-dimensional array. We use $T_{:jk}$, $T_{i:k}$, and T_{ij} to denote the row (mode-1), column (mode-2), and tube (mode-3) fibers, and use T_{i}, $T_{:j}$, and $T_{::k}$ to represent the horizontal, lateral, and frontal slices. We use $T_{(2)}$ to denote the mode-2 matrixization defined as the transpose of the horizontal stacking of lateral slices, or equivalently, the matrix each column of which is $\mathrm{vec}(T_i)$,

$$T_{(2)} = [T_{:1}, \ldots, T_{:m}]^\intercal = [\mathrm{vec}(T_i)]_{i\in[l]}.$$

For any $p \in \mathbb{R}^l$, we define $T(p) := \sum_{i\in[l]} p_i T_i$ as the linear combination of the horizontal slices of T with coefficients from p. Given two matrices A and B, $A \otimes B$ denotes the Kronecker product.

For a subspace $\mathcal{U}$ of some inner product space $\mathcal{V}$, we use $\mathcal{U}^\perp$ to denote the corresponding orthogonal complement. We also use $\|\cdot\|$ to denote the 2-norm for a vector and the Frobenius norm for a matrix.

2.2 Problem Setting

Under the linear setting, Formulation (2) can be specified as follows with input parameters (a, A, c, C, Ξ),

$$\Pi := \min_{y(\cdot)\in\mathcal{Y}_\Xi} \max_{\xi\in\Xi} \ \langle c, \xi\rangle + \langle a, y(\xi)\rangle \tag{5a}$$

$$\text{s.t. } C\xi - Ay(\xi) \leq 0, \quad \forall \xi \in \Xi. \tag{5b}$$

Given Euclidean spaces $\mathcal{X} := \mathbb{R}^n$ and $\mathcal{Y} := \mathbb{R}^m$, $\Xi := \{\xi \in \mathcal{X} \mid B\xi \leq b\} \subseteq \mathcal{X}$ is a polyhedral uncertainty set with $B \in \mathbb{R}^{s\times n}$, and $\mathcal{Y}_\Xi$ is the set containing all the history-dependent feasible policies that can be recursively defined as

$$\mathcal{Y}_\Xi := \prod_{\xi\in\Xi} \mathcal{Y}_\xi \text{ with } \mathcal{Y}_\xi := \left\{y = (y_i)_{i\in[\tau]} \mid y_i \in \mathcal{Y}(h_i)\right\}.$$

That is, $\mathcal{Y}_\xi$ contains all multi-stage decisions where the decision at each state y_i is feasible to its history $h_i = (\xi_j, y_j)_{j<i}$, and every element $y(\cdot) \in \prod_{\xi\in\Xi} \mathcal{Y}_\xi$ is a dependent function that produces a feasible multi-stage decision for every input $\xi \in \Xi$. Matrices $C \in \mathbb{R}^{k\times n}$ and $A \in \mathbb{R}^{k\times m}$ describe the dependency between decisions variables and uncertainty parameters, which also characterizes the feasible policy space. For simplicity, we define the notation for augmented matrices as $\bar{C} := [c^\intercal; C]$ and $\bar{A} := [a^\intercal; -A]$ where the semicolon represents the row-wise stacking. Throughout the paper, we assume that Π is feasible and bounded for both the decision-maker and the nature (uncertainty).

Given any policy family $\theta : \mathcal{P} \times \Xi \to \mathcal{Y}_\Xi$, the problem (5) can be approximated by the following parameterized policy problem.

$$\Pi_\theta : \quad \min_{p\in\mathcal{P}} \max_{\xi\in\Xi} \ \langle c, \xi\rangle + \langle a, \theta(p, \xi)\rangle \tag{6a}$$

$$\text{s.t. } C\xi - A\theta(p, \xi) \leq 0, \quad \forall \xi \in \Xi. \tag{6b}$$

In particular, replacing θ with the constant and affine policy families $\dot{\theta}$ and $\bar{\theta}$ defined in (3) gives the corresponding constant and affine policy approximations.

As discussed before, this paper focuses on studying the information basis of the affine policy family. This amounts to addressing the following four research inquiries: (i) How to incorporate both information filters and information basis to design a transition between $\dot{\theta}$ and the filtered space? (ii) How to construct an effective information basis? (iii) What is the tradeoff between the basis size and policy performance? (iv) How can such a basis be efficiently constructed? We will address the first question in the following subsection.

2.3 Information Filters and Information Bases

In this subsection, we will introduce a class of policy families using information filters and information bases. We will show that these families form a natural transition between the constant and the filtered policy families. We start with the following definition.

Definition 1 (Information Filter)**.** Let $\dot{\mathcal{P}}$ and $\mathcal{P}$ be the same as in (3b), an information filter F is a 0-1 matrix in $\mathcal{P}$ that acts as a projection map sending certain entries of $P \in \mathcal{P}$ to zeros. We define $\mathcal{F} := F(\mathcal{P})$ the filtered information space, and use $\bar{\mathcal{F}}$ to denote the augmented space $\dot{\mathcal{P}} \times \mathcal{F}$.

Note that $\bar{\mathcal{F}}$ is a subspace of the parameter space $\bar{\mathcal{P}}$ of the affine policy family. The following proposition further confirms the intuition that every filtered information space $\bar{\mathcal{F}}$ labels an isomorphic copy within the policy space under mild conditions.

Proposition 1. *Suppose Ξ is full-dimensional, then the affine policy family $\bar{\theta} : \bar{\mathcal{P}} \to \mathcal{Y}^{\Xi}$ is an injective linear map. In particular, for every filtered information space $\bar{\mathcal{F}}$, we have $\bar{\mathcal{F}} \cong \bar{\theta}(\bar{\mathcal{F}})$.*

Proof. It is obvious that $\bar{\theta}$ is a linear map. For injectivity, we show that $\bar{\theta}(\bar{P}) = 0$ implies $\bar{P} = 0$. By definition, $\bar{\theta}(\bar{P}) = 0$ means $p_0 + P^\intercal \xi = 0$ for all $\xi \in \Xi$. Since Ξ is full-dimensional, there is some interior point $\xi \in \Xi$ along with a scalar $\epsilon > 0$ such that $\xi + \epsilon e_i \in \Xi$ for every standard basis element e_i. We then have

$$0 = p_0 + P^\intercal(\xi + \epsilon e_i) = p_0 + P^\intercal \xi + \epsilon P^\intercal e_i = 0 + \epsilon P^\intercal e_i,$$

which implies P is zero. Thus, we also have $p_0 = 0$. The second statement is a trivial consequence. □

This observation allows us to focus on the filtered information space $\bar{\mathcal{F}}$ as any maneuver performed on it will be equivalently transported to the filtered policy subspace $\bar{\theta}(\bar{\mathcal{F}})$. For a given filtered information space $\bar{\mathcal{F}}$, the following definition provides a natural transition between the constant and filtered policy families.

Definition 2 (Information Basis and the Induced Policy Family)**.** Given a filtered information space $\mathcal{F}$, let T be a 3-tensor where all the horizontal slices T_i's are from $\mathcal{F}$, then the corresponding *information basis induced* (IBI) policy family with respect to T is defined as

$$\theta_T(\bar{p}, \xi) = p_0 + T(p)\xi, \tag{7}$$

where $\bar{p} = (p_0, p) \in \bar{\mathcal{P}}_T = \dot{\mathcal{P}} \times \mathcal{P}_T$ is the corresponding policy parameters and $T : \mathcal{P}_T \to \mathcal{F}$ is an injective linear transformation we call the *information basis*. Each horizontal slice T_i of T is named the ith basis matrix. When the number of basis matrices $d \ll \dim \mathcal{F}$, we call θ_T a *sparse information basis induced* (SIBI) policy family.

By this definition, every information basis T can be considered as a particular way to embed the parameter space $\mathcal{P}_T$ (with a compatible dimension) into the filtered space $\mathcal{F}$ with $\{T_i\}_{i\in[d]}$ as the basis. It can be considered as a d-dimensional linear slice of $\mathcal{F}$. According to Proposition 1, this also induces a linear slice in the policy space.

2.4 IBI Bidual Reformulation

The central question regarding the IBI policy family in (7) is how to design the information basis T. For certain types of problems, this design could be motivated by the specific problem structure. In this paper, however, we aim to automate the construction of T, which is enabled by the following reformulation of (6) under a given IBI policy family θ_T.

Definition 3. The IBI bidual formulation of the policy problem Π_{θ_T} is defined as,

$$\begin{aligned}
\Delta_{\theta_T} : \quad \max_{\xi\in\Xi, u\geq 0, V} \quad & \langle \bar{C}, \bar{V} \rangle && && \text{(8a)}\\
\text{s.t.} \quad & BV^\intercal \leq bu^\intercal, && && \text{(8b)}\\
& A^\intercal u = a, && p_0 && \text{(8c)}\\
& \langle \bar{A}^\intercal \bar{V}, T_j \rangle = 0, \quad \forall j \in [d], && p && \text{(8d)}
\end{aligned}$$

where $\bar{V} := [\xi^\intercal; V]$ is an augmented matrix, and p_0, p are the dual variables of the corresponding constraints. We also define $\mathcal{U} := \{u \geq 0 \mid A^\intercal u = a\}$ and call it the *dual polyhedron*.

This is obtained by performing the classic bidualization technique [3] for robust optimization on Π_{θ_T}. First, each constraint in (6b) can be rewritten as

$$\max_{\xi \in \Xi} \{C\xi - A\theta(p, \xi)\} \leq 0,$$

the left side of which is a maximization problem that can be dualized. Then, we swap the order of minimization and maximization in the objective function, invoking the minimax theorem. Finally, we dualize the inner minimization problem for a fixed $\xi \in \Xi$. Since the strong duality holds throughout the entire process under the feasibility and boundedness assumptions, we have the following proposition.

Proposition 2. *For every IBI policy family θ_T, we have $z(\Delta_{\theta_T}) = z(\Pi_{\theta_T})$.*

From this bidual formulation, we obtain a new interpretation regarding the effects of the 3-tensor T. Notice that we have $\langle \bar{A}^\intercal \bar{V}, T_j \rangle = \langle \bar{V}, \bar{A} T_j \rangle$. Hence, each basis matrix T_j is to restrain the feasible space of $\bar{V}$ to be within the hyperspace equipped with the normal vector $\bar{A}T_j$ (consider this matrix as a vector in the vector space of matrices). When $T = 0$, the optimal dual variables p_0 recover the optimal constant policy; when the basis matrices T_j's span the entire filtered space $\mathcal{F}$, matrix $\bar{V}$ has to lie inside the orthogonal complement of $\bar{A}(\mathcal{F})$, and the corresponding dual variables p_0 and p will retrieve the optimal affine policy in $\bar{\theta}(\mathcal{F})$. A transition between these two extreme cases naturally occurs when we keep adding constraints in (8d) with linearly independent basis matrices T_j's. Moreover, each properly selected basis matrix will reduce the dimension of the feasible space of $\bar{V}$ by at least one.

In the next section, we will utilize this new interpretation of T to design an algorithm to construct the information basis T iteratively.

3 Part 1 Algorithm: Designing Information Basis

According to the bidual formulation (8), adding linearly independent basis matrices T_j's in sequence will eventually recover the optimal filtered policy after $\dim \mathcal{F}$ iterations. The question is how to construct T_j to make this convergence efficient. In this section, we introduce our main algorithm and demonstrate its performance. To ease the notation, we denote

$$\bar{\mathcal{U}} = \{(1, u) \mid u \in \mathcal{U}\} \text{ and } \mathcal{V} = \{\bar{V} \mid B\bar{V}^\intercal \leq b\bar{u}^\intercal \text{ for some } \bar{u} \in \bar{\mathcal{U}}\}$$

so that (8) can be rewritten as

$$\max_{\bar{V}} \quad \langle \bar{C}, \bar{V} \rangle \tag{9a}$$

$$\text{s.t.} \quad \bar{V} \in \mathcal{V}, \tag{9b}$$

$$\langle \bar{A}T_j, \bar{V} \rangle = 0, \quad \forall j \in [d]. \tag{9c}$$

As mentioned before, Constraint (9c) serves as a transition between the constant and filtered policy family. This indicates a potential dynamic procedure to construct the information basis on-the-fly. Specifically, we dissect (9) into a sequence of problems defined below.

Definition 4. The *iterative IBI bidual* problem is defined as follows,

$$\phi(\bar{V}) := \langle \bar{C}, \bar{V} \rangle \tag{10a}$$

$$\bar{V}_j^\star := \arg \max_{\bar{V} \in \mathcal{V} \cap \mathcal{S}_j} \phi(\bar{V}) \tag{10b}$$

$$\mathcal{S}_0 := \mathbb{R}^{(k+1)\times n} \tag{10c}$$

$$\mathcal{S}_j := \mathcal{S}_{j-1} \cap \left\{ \bar{V} \mid \langle \bar{A} h(\bar{V}_{j-1}^\star), \bar{V} \rangle = 0 \right\}, \ \forall j \in [\dim \mathcal{F}] \tag{10d}$$

$$h(\cdot) := \mathrm{vec}_n^{-1} \left(E_{(2)} \left((I_n \otimes \bar{A}) E_{(2)} \right)^\dagger \mathrm{vec}(\cdot) \right) \tag{10e}$$

$$E := \text{any 3-tensor with horizontal slices forming a basis of } \mathcal{F}, \tag{10f}$$

where $E_{(2)}$ is the mode-2 matrixrization of E and each basis matrix T_j is designed as $h(\bar{V}_{j-1}^\star)$.

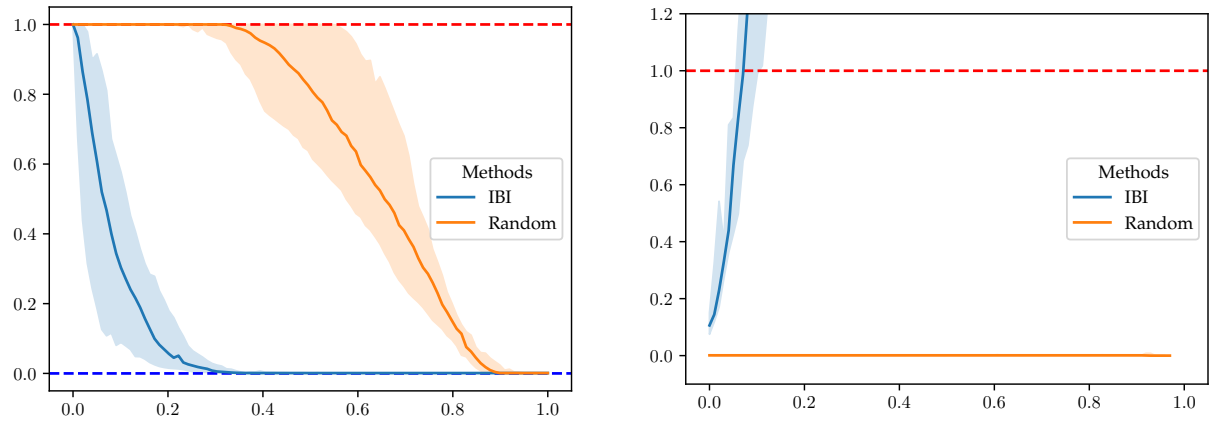
This can be considered as a constraint generation procedure where the equality added in each iteration j is determined by $h(\cdot)$ and the previous optimal solution $\bar{V}_{j-1}^\star$. The corresponding pseudocode can be found in Algorithm 1. By this design, our algorithm solves (10b) iteratively and obtains the corresponding $\bar{V}_j^\star$, then update $\mathcal{S}_{j+1}$ using (10d) for the next iteration. After the algorithm terminates, $h(\bar{V}_j^\star)$'s form the information basis T, and the optimal IBI policy can be recovered from the dual variables p by $\sum_{j\in[d]} p_j h(\bar{V}_j^\star)$ with the optimal objective value $z_d^\star$. Moreover, for every $d' < d$, we also obtained an information basis with the associated policy and objective value.

Algorithm 1: IBI Bidual Algorithm.

Data: Problem data (a, A, c, C, Ξ); iteration limit d
Initialization: $j \leftarrow 0$;
while $j \leq d$ **do**
  $z_j^\star \leftarrow \max_{\bar{V} \in \mathcal{V} \cap \mathcal{S}_j} \phi(\bar{V})$;
  $\bar{V}_j^\star \leftarrow \arg\max_{\bar{V} \in \mathcal{V} \cap \mathcal{S}_j} \phi(\bar{V})$;
  $\mathcal{S}_{j+1} \leftarrow \mathcal{S}_j$ with a new constraint $\langle \bar{A} h(\bar{V}_j^\star), \bar{V} \rangle = 0$;
  $j \leftarrow j + 1$;
return $\{z_j^\star\}_{j\in[d]}$, information basis $\{h(\bar{V}_j^\star)\}_{j\in[d]}$, dual variables p of constraints in $\mathcal{S}_d$;

The core design of this algorithm is the assignment of T_j's using the function $h(\cdot)$ as it determines several aspects simultaneously: (i) the information basis matrices; (ii) how to extend the parameter space to include more affine policies; (iii) the objective value convergence rate in each iteration; (iv) the corresponding optimal policy within the slice. We will analyze and justify our design in the next section.

Figure 3 illustrates the convergence and runtime performance of Algorithm 1 using the experiment results from 50 randomly generated instances. The detailed experiment environment and instance generation can be found in Section 6. In the two sub-figures, the blue curves represent the iterative objective values and execution times obtained by Algorithm 1, while the orange curves are the objective and time counterparts obtained using randomly generated information basis. Specifically, we first arbitrarily select a number of $\dim \bar{A}\mathcal{F}$ matrices from $\mathcal{F}$ and project them into $\bar{A}\mathcal{F}$ to form a basis, then evaluate its performance at each iteration j by using only the first j matrices to create constraints in (9c). We normalized both axes to the range of $[0, 1]$. That is, 1 in x-axis represents the maximum iteration number $\dim \bar{A}\mathcal{F}$ for each instance; 0 and 1 in the y-axis of Figure 3a represent the objective values of the optimal affine and constant policy, respectively; 0 and 1 in the y-axis of Figure 3b denote the starting time and the normalized runtime of solving for the optimal affine policy (exclude the formulation setup time). The blue and orange regions in both sub-figures are the corresponding 95% confidence intervals.



(a) Objective Values vs Iterations

(b) Runtimes vs Iterations

Figure 3: Convergence and runtime performances of algorithms based on IBI and randomly generated information basis. x-axis in both sub-figures represents normalized iterations; 0 and 1 in y-axis of 3a are the normalized objective values from optimal affine and constant policies, respectively; y-axis of 3b represents normalized runtimes (exclude the formulation setup time) with 1 as the runtime of solving for the optimal affine policy. The blue and orange regions are the corresponding 95% confidence intervals.

In terms of convergence, Figure 3a shows that both algorithms eventually converge to the optimal affine policy's performance within $\dim \bar{A}\mathcal{F}$ steps. However, the IBI algorithm converges much faster than the random counterpart. This also implies the information basis selected by Algorithm 1 has a better quality than the randomly generated ones. Regarding the runtime performance shown in Figure 3b, it is expected that the runtime for random generation is independent of the iterations and has a small variation, and it is much faster than solving for an optimal affine policy. In contrast, the IBI algorithm started relatively efficiently, yet followed by a steep increase in runtime. In each iteration, this algorithm solves a version of (9) with the number of constraints in (9c) increases from zero to $\dim \bar{A}\mathcal{F}$. Hence, it is somewhat surprising to see that its runtime soon surpasses solving the affine IBI formulation (the red dashed line). This behavior is likely attributed to the solver's re-optimization mechanism. It lacks any presolve steps and employs the previous solution, which becomes infeasible after adding a new constraint, as a warm-start.

In the next section, we will delve into an analysis of the convergence performance of the IBI algorithm, as depicted in Figure 3a. Following this, Section 5 will concentrate on improving the algorithm's runtime performance. The objective is to attain runtime complexity comparable to random generation, while preserving a similar level of convergence rate in objective values.

4 Algorithm Analysis

In this section, we analyze the design of T and $h(\cdot)$ in detail and provide the corresponding convergence analysis. In particular, we will prove the following: (i) the design of $h(\cdot)$ is optimal for random instances, (ii) Algorithm 1 converges exponentially to the optimal affine policy under mild assumptions, and (iii) this convergence rate can be further improved by adding a regularizer term in the objective function of (9). Though this paper only considers problems with linear structures, most analyses in this section can be easily adapted to problems of the form (9) with a convex $\mathcal{V}$ and a convex objective function.

We start with the observation that in both (9a) and (9c), matrix $\bar{A}$ serves as some transformer for

T_j. Specifically, suppose we have a full-size information basis of $\mathcal{F}$ (*i.e.*, (9) produces the optimal filtered policy), Constraint (9c) says $\bar{V}$ belongs to $(\bar{A}\mathcal{F})^{\perp}$. Hence, to converge to the optimal affine policy rapidly, we need to guide the solution $\bar{V}$ to enter $(\bar{A}\mathcal{F})^{\perp}$ as early as possible. Thus, the part of $\bar{V}_j^{\star}$ that is outside $(\bar{A}\mathcal{F})^{\perp}$ impedes its convergence. This motivates a design to remove the undesired part of $\bar{V}_j^{\star}$ using the following projection operation.

Definition 5 (Matrix Projection)**.** Given a subspace $\mathcal{Q}$ of the matrix space $\mathcal{P}$, let E be a 3-tensor where the horizontal slices E_i's span $\mathcal{Q}$. Then, the projection operator $\text{proj}_{\mathcal{Q}}$ is defined as

$$\text{proj}_{\mathcal{Q}}(D) = \text{vec}_n^{-1}\left(E_{(2)}E_{(2)}^{\dagger}\,\text{vec}(D)\right),$$

where n is the number of columns in each matrix E_i and $E_{(2)}$ is the mode-2 matrixization of E.

The design of $h(\cdot)$ in (10e) is essentially based on the above definition according to the next proposition.

Proposition 3. *By the design in* (10e)*, we have* $\bar{A}h(\cdot) = \text{proj}_{\bar{A}\mathcal{F}}(\cdot)$.

Proof. By direct computation, we have

$$\begin{aligned}\bar{A}h(D) &= \bar{A}\,\text{vec}_n^{-1}\left(E_{(2)}\left((I_n\otimes\bar{A})E_{(2)}\right)^{\dagger}\text{vec}(D)\right)\\ &= \text{vec}_n^{-1}\left((I_n\otimes\bar{A})E_{(2)}\left((I_n\otimes\bar{A})E_{(2)}\right)^{\dagger}\text{vec}(D)\right)\\ &= \text{proj}_{\mathcal{Q}}(D),\end{aligned}$$

where $\mathcal{Q}$ is spanned by

$$\text{vec}_n^{-1}((I_n\otimes\bar{A})\,\text{vec}(E_i)) = \text{vec}_n^{-1}(\text{vec}(\bar{A}E_i)) = \bar{A}E_i.$$

Since E_i's span $\mathcal{F}$, we have $\mathcal{Q} = \bar{A}\mathcal{F}$. □

Intuitively, this design of $h(\cdot)$ eliminates the unwanted part of the current optimal solution $\bar{V}_j^{\star}$. According to this proposition, we also have the following optimality guarantee for the main algorithm.

Theorem 1. *In Algorithm 1, an optimal filtered policy can be computed for at most* $\dim\bar{A}\mathcal{F}$ *iterations. Moreover, suppose we obtain* $\bar{V}_j^{\star}\in\left(\bar{A}\mathcal{F}\right)^{\perp}$ *at some iteration* $j<\dim\bar{A}\mathcal{F}$*, the corresponding IBI policy is also an optimal filtered policy.*

Proof. By the design of $h(\cdot)$, the generated information basis matrices $h(\bar{V}_j^{\star})$'s blong to $\bar{A}\mathcal{F}$ and are mutually orthogonal. Thus, for at most $\dim\bar{A}\mathcal{F}$ iterations, Constraint (9c) will be equivalent to $\bar{V}\in(\bar{A}\mathcal{F})^{\perp}$, which proves the first claim. For the second statement, since each iteration is a relaxation of finding the optimal affine policy in the filtered space, $\bar{V}_j^{\star}$ corresponds to an upper bound of $z(\Delta_{\theta_T})$. On the other hand, $\bar{V}_j^{\star}\in(\bar{A}\mathcal{F})^{\perp}$ implies it is feasible to Δ_{θ_T}, *i.e.*, it provides a lower bound. Together, they imply that $\bar{V}_j^{\star}$ is optimal to Δ_{θ_T}. Hence, the corresponding IBI policy is also an optimal affine policy in the filtered policy space. □

In the rest of the section, we will focus on proving two things. First, this design of $h(\cdot)$ is optimal for random instances. Second, the iterative objective values of Algorithm 1 converges exponentially to the optimal filtered policy. All these analyses are enabled by the reformulation method introduced in the next subsection.

4.1 Quotient Reformulation

According to the above discussion, we aim to find a solution $\bar{V}^\star$ inside or near $(\bar{A}\mathcal{F})^\perp$. Moreover, for the purpose of analyzing the design of $h(\cdot)$, we are indifferent to the elements in $(\bar{A}\mathcal{F})^\perp$, since once $\bar{V}$ is inside $(\bar{A}\mathcal{F})^\perp$, the maximization will take care of finding the optimal filtered policy. This intuition motivates us to define a quotient space of $\mathcal{V}$ and reformulate (9) as follows.

Definition 6 (Quotient of $\mathcal{V}$ & Iterative Quotient Bidual)**.** We define the quotient space of $\mathcal{V}$ as $\mathcal{X} := \text{proj}_{\bar{A}\mathcal{F}}(\mathcal{V})$. For each $X \in \mathcal{X}$, the *preimage* over X is

$$\mathcal{V}_X := \{\bar{V} \in \mathcal{V} \mid \text{proj}_{\bar{A}\mathcal{F}}(\bar{V}) = X\}.$$

We also define a new function ψ over $\mathcal{X}$ as follows,

$$\psi(X) := \max_{\bar{V} \in \mathcal{V}_X} \langle \bar{C}, \bar{V} \rangle, \tag{11}$$

Then, we call

$$\max_{X \in \mathcal{X} \cap \mathcal{S}_j} \psi(X), \ \forall j \in [d] \tag{12}$$

the *iterative quotient bidual formulation*, where $\mathcal{S}_j$ is the same as defined in (10).

By this definition, we clearly have

$$z(\Pi_{\dot{\theta}}) = z(\Delta_{\dot{\theta}}) = \max_{X \in \mathcal{X}} \psi(X)$$
$$z(\Pi_{\bar{\theta}_\mathcal{F}}) = z(\Delta_{\bar{\theta}_\mathcal{F}}) = \psi(0),$$

where $\bar{\theta}_\mathcal{F}$ is the filtered affine policy family. Moreover, adding Constraint (9c) in each iteration is equivalent to reducing the dimension of $\mathcal{X}$ by the intersection operation $\mathcal{X} \cap \mathcal{S}_j$. Thus, we trivially have the following result.

Lemma 1. *For all iteration j, the following identity holds,*

$$\max_{\bar{V} \in \mathcal{V} \cap \mathcal{S}_j} \phi(\bar{V}) = \max_{X \in \mathcal{X} \cap \mathcal{S}_j} \psi(X).$$

This reformulation divides out the nonessential part $(\bar{A}\mathcal{F})^\perp$ (along with its translations) and has several good properties for further analysis.

Proposition 4. *Function ψ has the following properties:*

- *ψ is a concave function over $\mathcal{X}$;*
- *ψ is Lipschitz.*

Proof. For the first, note that every $\bar{V} \in \mathcal{V}$ can be decomposed into $\text{proj}_{\bar{A}\mathcal{F}}(\mathcal{V}) + \text{proj}_{(\bar{A}\mathcal{F})^\perp}(\mathcal{V})$. Thus, ψ defined in (11) can be considered as a supremum projection of a concave function. For the second, it is clear that ψ is continuous and defined over a compact set, thanks to the boundedness and feasibility assumptions of the original problem. □

Corollary 1. *For any $c \in \mathbb{R}$, the upper level set $\mathcal{X}_c := \{X \in \mathcal{X} \mid \psi(X) \geq c\}$ is convex. Moreover, in each iteration j of the iterative quotient bidual (12), the optimizer $X_j^\star$ belongs to the level set $\mathcal{X}_{\psi(0)}$.*

Proof. The first statement is a direct implication of Proposition 4. For the second, we have $\mathcal{X}_c \subseteq \mathcal{X}_{c'}$ whenever $c \geq c'$ by definition. Since $\psi(0)$ is the value corresponding to the optimal affine policy, we clearly have $\psi(X_j^\star) \geq \psi(0)$ for all j, which implies $X_j^\star \in \mathcal{X}_{\psi(0)}$. □

According to these results, we use the following assumptions throughout this section to simplify our analysis.

Assumption 1. Without loss of generality, we assume the following for the domain $\mathcal{X}$ and objective function ψ.

(a). $\mathcal{X}$ is a polytope within the unit n-ball $\mathcal{B}^n$ centered as 0.

(b). ψ is a piece-wise concave function over $\mathcal{X}$ with $\psi(\mathcal{X}) \subseteq [0, 1]$.

(c). 0 is a boundary point of $\mathcal{X}$.

(d). All the boundary points of $\mathcal{X}$ have the same value, *i.e.*, $\psi(\partial\mathcal{X}) = \{\psi(0)\}$.

These assumptions do not restrict the application scope of our analysis since the boundedness and feasibility assumptions ensure that the original polytope and input parameters can be scaled so that the $\mathcal{X}$ and ψ fit into the assumptions (a) and (b). For (c) and (d), note that the optimal affine policy is obtained at $X = 0$ with the optimal value $\psi(0)$. Moreover, by Corollary 1, the objective value obtained at each iteration is strictly greater than $\psi(0)$ before an optimal affine policy is obtained. Thus, for convergence analysis purposes, we can safely discard the redundant part $\mathcal{X} \setminus \mathcal{X}_{\psi(0)}$ of $\mathcal{X}$, which results in a convex set with 0 on the boundary and having $\psi(0)$ as the unique value for all boundary points. We will conduct our analysis using the quotient reformulation under these assumptions. For notation convenience, we will also use lowercase x instead of X to denote an element in $\mathcal{X}$ hereafter.

4.2 Design of $h(\cdot)$ is Optimal

In this section, we show that the design of $h(\cdot)$ in (10e) is optimal in a random setting. This randomness may occur from two sources. First, since we develop a method for general dynamic robust optimization problems that fall into (5), the potential problem structure to solve is random. Second, in each iteration j of solving the iterative IBI bidual, the feasible solution space $\mathcal{V} \cap \mathcal{S}_j$ is a lower-dimensional slice in $\mathcal{V} \cap \mathcal{S}_{j-1}$. Hence, the new problem at iteration j is the objective ϕ restricted to this new slice. It is unlikely to access the structural information of these slices and the corresponding restricted functions that are created on-the-fly. Thus, we can only assume they are random polytopes and concave functions. We define the family of potential polyhedrons and functions as follows.

Definition 7. The class of bounded polytopes is defined as

$$\mathfrak{X}^n = \{\mathcal{X} \subseteq \mathcal{B}^n \mid \mathcal{X} \text{ is a polytope and has } 0 \text{ as a boundary point}\}.$$

For every $\mathcal{X} \in \mathfrak{X}^n$, the class of bounded piece-wise concave functions over $\mathcal{X}$ is

$$\mathfrak{G}_{\mathcal{X}}^n = \{\psi : \mathcal{X} \to [0, 1] \mid \psi \text{ is a piece-wise concave function such that } \psi(\partial\mathcal{X}) = \{\psi(0)\}\},$$

where ∂X is the boundary of $\mathcal{X}$ under the Euclidean topology. When the dimension is obvious, we drop the superscript n in both sets.

Because elements in both sets $\mathfrak{X}$ and $\mathfrak{G}_{\mathcal{X}}$ can be represented by matrices that form compact sets in the corresponding space, they are Borel sets that can be assigned with probability measures (*e.g.*, uniform probability measures). With a slight abuse of notation, we also take $\mathfrak{X}$ and $\mathfrak{G}_{\mathcal{X}}$ to represent the corresponding probability spaces. We use *hyperspace* to refer to a subspace having one less dimension than $\mathcal{X}$. Under these definitions, we have the following observations. Lemma 2 can be directly verified from the definitions, so we omit the proof.

Lemma 2. *For every $\mathcal{X} \in \mathfrak{X}^n$, $\psi \in \mathfrak{G}^n_{\mathcal{X}}$, and some hyperspace S such that $\dim \mathcal{X} \cap S = m < n$, we have*

$$\mathcal{X} \cap S \in \mathfrak{X}^m \text{ and } \phi|_{\mathcal{X}\cap S} \in \mathfrak{G}^m_{\mathcal{X}},$$

where $\phi|_{\mathcal{X}\cap S}$ is function ϕ restricted to the set $\mathcal{X} \cap S$.

Define $M_{\mathcal{X}}(0) := \{a \mid \langle a, x' \rangle \geq 0 \, \forall x' \in \mathcal{X}\}$, *i.e.*, $M_{\mathcal{X}}(0)$ is the reversed (multiply by -1) normal cone of $\mathcal{X}$ at the origin, and let S_a for some $a \in M_{\mathcal{X}}(0)$ be the hyperspace with normal vector a, then we have,

Lemma 3. *For every $\mathcal{X} \in \mathfrak{X}$, $\psi \in \mathfrak{G}_{\mathcal{X}}$, and $a \in M_{\mathcal{X}}(0)$, we have*

$$\max_{x \in \mathcal{X} \cap S_a} \psi(x) = \psi(0).$$

Proof. Since $\mathcal{X}$ is a polytope (thus convex set) that contains 0 as a boundary point, $M_{\mathcal{X}}(0)$ is not empty. Then, the corresponding S_a intersects $\mathcal{X}$ only on the boundary $\partial\mathcal{X}$ by design. Moreover, since $\psi(\partial\mathcal{X}) = \{\psi(0)\}$ by definition, we have the desired result. □

Hence, any vector $a \in M_{\mathcal{X}}(0)$ can be considered as an ideal element for the information basis since setting $\bar{A}h(\bar{V}^\star_{j-1}) := a$ in the jth iteration in (10) will obtain the optimal filtered policy immediately in the next iteration. However, for a fixed $\mathcal{X} \in \mathfrak{X}$, the reversed normal cone $M_{\mathcal{X}}(0)$ might contains multiple directions. We use the following definition to restrict to a unique choice.

Definition 8 (Plummet Vector & Plummet Hyperspace)**.** Given $\mathcal{X} \in \mathfrak{X}$, $\psi \in \mathfrak{G}_{\mathcal{X}}$, and some $x^\star \in \arg\max_{x\in\mathcal{X}} \psi$. For any vector a, let

$$S_a = \{x \mid \langle a, x \rangle = 0\}$$

be the hyperspace with normal vector a. Then, any vector a that satisfies

$$\max_{a \in M_{\mathcal{X}}(0): \|a\|=1} \quad \frac{\langle x^\star, a \rangle}{\|x^\star\|} \tag{13}$$

is called the *plummet vector* with respect to $(\mathcal{X}, \psi, x^\star)$, and the corresponding hyperspace S_a is called the *plummet hyperspace* with respect to $(\mathcal{X}, \psi, x^\star)$.

Intuitively, among all the vectors in $M_{\mathcal{X}}(0)$, the plummet vector a is the one that has the smallest angle to the incumbent solution $x^\star$. Indeed, the following proposition shows that for any given $(\mathcal{X}, \psi, x^\star)$ with $\psi(x^\star) > \psi(0)$, the plummet vector and plummet hyperspace are both unique.

Proposition 5. *For any $\mathcal{X} \in \mathfrak{X}$, $\psi \in \mathfrak{G}_{\mathcal{X}}$, and $x^\star \in \arg\max_{x\in\mathcal{X}} \psi(x)$ such that $\psi(x^\star) > \psi(0)$, there exists a unique plummet vector and a unique plummet hyperspace.*

Proof. The existence has been shown in Lemma 3. For uniqueness, suppose $a = x^\star/\|x^\star\| \in M_{\mathcal{X}}(0)$, then the claim is trivially true. Otherwise, suppose there are two distinct plummet vectors a_1 and a_2. By the definition of plummet vector and $\psi(x^\star) > \psi(0)$, we have two vectors satisfy $\langle a_1, x^\star\rangle = \langle a_2, x^\star\rangle > 0$. Clearly, the vector $\hat{a} = (a_1+a_2)/\|a_1+a_2\|$ belongs to $M_{\mathcal{X}}(0)$ by the property of convex cones. We show that $\hat{a}$ has a strictly larger value in (13). Note that $a_1 \neq a_2$ implies $\|a_1 + a_2\| < \|a_1\| + \|a_2\| = 2$. Thus, we have

$$\frac{\langle x^\star, \hat{a}\rangle}{\|x^\star\|} = \frac{\langle x^\star, a_1 + a_2\rangle}{\|x^\star\|\|a_1 + a_2\|} > \frac{\langle x^\star, a_1\rangle}{\|x^\star\|} = \frac{\langle x^\star, a_2\rangle}{\|x^\star\|},$$

where the strictly greater is by the choice a_1 and a_2. This contradicts the optimality of a_1 and a_2. □

Proposition 5 provides a unique ideal selection of hyperspace to optimally decrease the objective value of ψ given a tuple $(\mathcal{X}, \psi, x^\star)$. The following theorem states that given $x^\star$ as the observed optimal solution, the "average" plummet vector is simply $x^\star$, which requires an assumption based on the following definition.

Definition 9 (Isometry-Invariant Measure)**.** The probability measure μ assigned on $\mathfrak{X}$ and probability measures $\mu_{\mathcal{X}}$ equipped on each $\mathfrak{G}_{\mathcal{X}}$ are say to be *isometry-invariant* if for every isometry $\phi \in \mathcal{L}(\mathcal{B}^n)$, $\mathcal{X} \in \mathfrak{X}$, and $\psi \in \mathfrak{G}_{\mathcal{X}}$, we have

$$\mu(\mathcal{X}) = \mu(\phi(\mathcal{X}))$$
$$\mu_{\mathcal{X}}(\psi) = \mu_{\phi(\mathcal{X})}(\psi \circ \phi^{-1}).$$

Intuitively speaking, an isometry-invariant measure implies that the probabilities are the same for problem structures (the polyhedron $\mathcal{X}$ and the concave function ψ over $\mathcal{X}$) that are equivalent under rotations and reflections.

Theorem 2. *Given isometry-invariant measure spaces $\mathfrak{X}$ and $\{\mathfrak{G}_{\mathcal{X}}\}_{\mathcal{X}\in\mathfrak{X}}$ and an optimal solution $x^\star \neq 0 \in \mathcal{B}^n$, define $x^\star$-conditional probability spaces as*

$$\mathfrak{X}_\star := \{\mathcal{X} \in \mathfrak{X} \mid x^\star \in \mathcal{X}\},$$
$$\mathfrak{G}^\star_{\mathcal{X}} := \{\psi \in \mathfrak{G}_{\mathcal{X}} \mid x^\star \in \arg\max \psi\}, \quad \forall \mathcal{X} \in \mathfrak{X}_\star,$$

with the induced conditional probability measures. Let $a_{\mathcal{X},\psi}$ be the plummet vector with respect to $\mathcal{X} \in \mathfrak{X}_\star$, $\psi \in \mathfrak{G}^\star_{\mathcal{X}}$, and $x^\star$. We have

$$\mathbb{E}[a_{\mathcal{X},\psi} \mid x^\star] := \int_{\mathcal{X}} \int_{\psi} a_{\mathcal{X},\psi} \; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star = \lambda x^\star$$

for some $\lambda > 0$.

Proof. To prove this, given $x^\star$, we define the following linear operator $\phi_\star$ on $\mathcal{B}^n$ (reflection over span$(x^\star)$) as

$$\phi_\star(x) := 2 \cdot \text{proj}_{x^\star}(x) - x = 2\frac{\langle x^\star, x\rangle}{\|x^\star\|^2}x^\star - x = \left(\frac{2}{\|x^\star\|^2}x^\star(x^\star)^\intercal - I\right)x.$$

The following claim lists some properties of this linear operator.

Claim 1. $\phi_\star$ has the following properties:

1. $\phi_\star$ is an isometry, *i.e.*, $\|\phi_\star(x)\| = \|x\|$;

2. $\phi_\star$ has the fixed point set as $\text{span}(x^\star)$;

3. $\phi_\star$ is idempotent, *i.e.*, $\phi_\star^2 = I$;

4. the induced operator $\Phi_\star : \mathfrak{X}_\star \to \mathfrak{X}_\star$ defined as

$$\Phi_\star(\mathcal{X}) := \{\phi_\star(x) \mid x \in \mathcal{X}\}$$

is an automorphism on the measure space $\mathfrak{X}_\star$;

5. the induced operator $\Psi_\star : \mathfrak{G}^\star_{\mathcal{X}} \to \mathfrak{G}^\star_{\Phi_\star(\mathcal{X})}$ defined as

$$\Psi_\star(\psi) := \psi \circ \phi_\star^{-1} = \psi \circ \phi_\star$$

is an isomorphism between the two measure spaces;

6. $a_{\Phi_\star(\mathcal{X}),\Psi_\star(\psi)} = \phi_\star(a_{\mathcal{X},\psi})$, and the mapping

$$a_{\mathcal{X},\psi} \mapsto a_{\Phi_\star(\mathcal{X}),\Psi_\star(\psi)}$$

is bijective on the set of all possible plummet vectors.

Proof. Property 1 can be verified directly. For 2, every fixed point satisfies $\phi_\star(x) = x$, which reduces to $\text{proj}_\star(x) = x$, which implies $\text{span}(x^\star)$ is the fixed point set. Property 3 is true by the property of reflection.

For 4, since the measure assigned on $\mathfrak{X}_\star$ is isometry-invariant, we only need to show that $\Phi_\star$ is a bijection on $\mathfrak{X}_\star$. First, we show that $\Phi_\star$ is closed on $\mathfrak{X}_\star$, *i.e.*, $\Phi_\star(\mathcal{X}) \in \mathfrak{X}_\star$ for all $\mathcal{X} \in \mathfrak{X}_\star$. By Property 1, $\Phi_\star(\mathcal{X}) \subseteq \mathcal{B}^n$. By Property 2, $x^\star \in \Phi_\star(\mathcal{X})$. Moreover, $\Phi_\star(\mathcal{X})$ is a polytope as $\phi_\star$ is a linear operator. Finally, $0 \in \text{span}(x^\star)$ is also a fixed point, thus is inside $\Phi_\star(\mathcal{X})$. These imply $\Phi_\star(\mathcal{X}) \in \mathfrak{X}_\star$. Then, the bijection is inherited from the bijection of $\phi_\star$.

For 5, again, because the measure assigned on both $\mathfrak{G}^\star_{\mathcal{X}}$ and $\mathfrak{G}^\star_{\Phi_\star(\mathcal{X})}$ are isometry-invariant, we only need to show $\Psi_\star$ is a bijection. First, $\Psi_\star(\psi)$ is a piece-wise concave function on $\Phi_\star(\mathcal{X})$ since $\phi_\star$ is a linear bijection. Moreover, $\Psi_\star(\psi)$ has values within $[0, 1]$ and satisfies $\Psi_\star(\psi)(\partial(\phi(\mathcal{X}))) = \{0\}$ directly by definition. To prove $\Psi_\star(\psi) \in \mathfrak{G}^\star_{\Phi_\star(\mathcal{X})}$, we left to show the following,

$$x^\star \in \arg\max_{x \in \mathcal{X}} \Psi_\star(\psi)(\phi_\star(x)).$$

By definition, we have $\Psi_\star(\psi)(\phi_\star(x)) = \psi \circ \phi_\star^{-1} \circ \phi_\star(x) = \psi(x)$, which proves the above claim trivially. Finally, the bijection of $\Psi_\star$ is due to the fact that $\Psi_\star^2(\psi) = \psi$, *i.e.*, it has a two-sided inverse.

For the last property, by definition,

$$a_{\Phi_\star(\mathcal{X}),\Psi_\star(\psi)} \in \arg\max_{a \in M_{\Phi_\star(\mathcal{X})} : \|a\|=1} \frac{\langle x^\star, a\rangle}{\|x^\star\|}.$$

By the property of $\phi_\star$, we have $\phi_\star(a_{\mathcal{X},\psi}) \in \Phi_\star(\mathcal{X})$. In addition, Property 1 implies $\phi_\star(a_{\mathcal{X},\psi})$ is of norm one. Suppose $\phi_\star(a_{\mathcal{X},\psi})$ is not a maximizer, then using $\phi_\star$ to transport the maximizer back to the case with space $\mathcal{X}$ and function ψ will induce a contradiction. Moreover, by Proposition (5), the maximizer of the above formulation is unique, which proves $a_{\Phi_\star(\mathcal{X}),\Psi_\star(\psi)} = \phi_\star(a_{\mathcal{X},\psi})$. Finally, the bijection is inherited from Properties 4–5. □

Using these properties, we prove the main theorem as follows,

$$E[a_{\mathcal{X},\psi} \mid x^\star] = \int_{\mathcal{X}} \int_{\psi} a_{\mathcal{X},\psi} \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14a}$$

$$= \int_{\mathcal{X}} \int_{\psi} a_{\Phi_\star(\mathcal{X}),\Psi_\star(\psi)} \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14b}$$

$$= \int_{\mathcal{X}} \int_{\psi} \phi_\star(a_{\mathcal{X},\psi}) \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14c}$$

$$= \frac{1}{2} \int_{\mathcal{X}} \int_{\psi} a_{\mathcal{X},\psi} + \phi_\star(a_{\mathcal{X},\psi}) \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14d}$$

$$= \frac{1}{2} \int_{\mathcal{X}} \int_{\psi} 2\lambda_{\mathcal{X},\psi} x^\star \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14e}$$

$$= x^\star \int_{\mathcal{X}} \int_{\psi} \lambda_{\mathcal{X},\psi} \;\; d\mathfrak{G}^\star_{\mathcal{X}} d\mathfrak{X}_\star \tag{14f}$$

$$= \lambda x^\star. \tag{14g}$$

The second identity is due to Property 6 along with $\Phi_\star$ and $\Psi_\star$ are isomorphisms on the measures, which means the measures of any open neighborhood of $\mathcal{X}$ and ψ are equivalently carried over to $\Phi_\star(\mathcal{X})$ and $\Psi_\star(\psi)$; the third is due to Property 6; the fourth is averaging the right-hand side of (14a) and (14c); the fifth is by definition of $\phi_\star$ with $\lambda_{\mathcal{X},\psi} = \langle x^\star, a_{\mathcal{X},\psi}\rangle / \|x^\star\|^2$. By definition 8, $\lambda_{\mathcal{X},\psi} \in (0, 1]$, which implies $0 < \lambda < \infty$. □

This theorem says that, suppose the polytope $\mathcal{X}$ and the piece-wise concave function ψ over it is randomly chosen, and the corresponding probabilities are invariant under rotations and reflections on $\mathcal{X}$. Then, for the given optimal solution $x^\star$, the "average" plummet hyperspace is simply $S_{x^\star}$. In particular, in our algorithm, after observing optimal solution $\bar{V}^\star_{j-1}$ at the $j-1$ iteration, the corresponding $x^\star \in \mathcal{X}$ is $\text{proj}_{\bar{A}\mathcal{F}}(\bar{V}^\star_{j-1})$. Thus, the design of $h(\cdot)$ in (10e) is "averagely" optimal.

4.3 Convergence Rate

In this subsection, we study the convergence rate of the introduced iterative method. We will show that, under mild assumptions, the expected convergence rate is at least linear, *i.e.*, the objective value approaching the value of optimal affine policy exponentially.

Since we study the convergence, the exact value of ψ is nonessential for analysis. For simplicity, we further assume the following in the rest of this section.

Assumption 2. Without loss of generality, we have the following assumptions: (a) $\psi(0) = 0$; (b) $\max_{x\in\mathcal{X}} \psi(x) > 0$.

Conventionally, the convergence rate of an iterative method is defined as

$$\rho = \lim_{t\to\infty} \frac{|z_{t+1} - z^\star|}{|z_t - z^\star|},$$

where $z^\star$ is the value of convergence, and the rate ρ is said to be linear and superlinear if $\rho \in (0, 1)$ and $\rho = 0$, respectively. This definition does not suit our situation for two reasons: (i) we cannot exactly compute or effectively bound $z^\star_t$ (there may have instances where our method converges in one step or never improves until the last iteration), thus we can only study the convergence rate in an "average" sense; (ii) our method is guaranteed to converge to the optimal value $\psi(0) = 0$ within

finite steps, thus the above rate ρ becomes 0 at certain iteration t and will be undefined thereafter. Hence, we define the following.

Definition 10 (Iterative Convergence Rate)**.** Given $\mathcal{X} \in \mathfrak{X}$ and $\psi \in \mathfrak{G}_{\mathcal{X}}$ with $x^\star \in \arg\max_{x\in\mathcal{X}} \psi(x)$ and $z_0^\star := \max_{x\in\mathcal{X}} \psi(x)$ as the corresponding optimal solution and objective value. Let $z_1^\star$ be the deterministic optimal value obtained in the next iteration under Algorithm 1. Then, the *iterative convergence rate* relative to $\mathcal{X}$ and ψ at is defined as

$$\rho(\mathcal{X}, \psi) := \begin{cases} z_1^\star / z_0^\star, & \text{if } z_0^\star > 0, \\ 0, & \text{otherwise.} \end{cases}$$

The associated *expected iterative convergence rate* is

$$\mathbb{E}[\rho] := \int_{\mathcal{X}} \int_{\psi} \rho(\mathcal{X}, \psi)\, d\mathfrak{G}_{\mathcal{X}} d\mathfrak{X}.$$

We say this rate is (iteratively) linear if $\mathbb{E}[\rho] < 1$.

By this definition, when the expected iterative convergence rate is linear, the objective value of the main algorithm decreases exponentially in an "average" sense. To prove this convergence rate for the main algorithm, we need the following definitions.

Definition 11 (Support of A Measure Space)**.** Given a probability space $(\mathfrak{X}, \mathcal{A}, \mu)$, the support is defined as

$$\text{supp}(\mathfrak{X}) = \{x \in \mathfrak{X} \mid \mu(V_x) > 0 \text{ for every open } x\text{-neighborhood } V_x\}.$$

We say the measure space is *fully supported* if $\text{supp}(\mathfrak{X}) = \mathfrak{X}$.

Definition 12 (Optimal Cone)**.** Given $\mathcal{X} \in \mathfrak{X}$ and $\psi \in \mathfrak{G}_{\mathcal{X}}$, let $\mathcal{X}^\star := \arg\max_{x\in\mathcal{X}} \psi(x)$, the optimal cone is defined as $\mathcal{C}_{\mathcal{X},\psi} := \text{cone}(\mathcal{X}^\star)$. We say the optimal cone is acute-angled if for every $v_1, v_2 \in \mathcal{C}_{\mathcal{X},\psi}$ we have $\langle v_1, v_2 \rangle > 0$.

Theorem 3. *$\mathbb{E}[\rho] < 1$ if there exists some $\mathcal{X} \in Int(supp(\mathfrak{X}))$ and some $\psi \in Int(supp(\mathfrak{G}_{\mathcal{X}}))$ such that $\mathcal{C}_{\mathcal{X},\psi}$ is acute-angled, where $Int(\cdot)$ denotes the interior of the input set.*

Proof. By definition, all optimal solutions are contained in $\mathcal{X}^\star$. Thus, the hyperspace must be $S_{x^\star}$ for some $x^\star \in \mathcal{X}^\star$, and $\mathcal{C}_{\mathcal{X},\psi}$ being acute-angled implies any two elements $x_1^\star, x_2^\star \in \mathcal{X}^\star$ satisfy $\langle x_1^\star, x_2^\star \rangle > 0$. On the other hand, every element $x \in \mathcal{X} \cap S_{x^\star}$ satisfies $\langle x^\star, x \rangle = 0$ by our design of $h(\cdot)$. This implies $S_{x^\star} \cap \mathcal{X}^\star = \varnothing$. Thus, $z_1^\star < z_0^\star$. Moreover, since $\mathcal{X}$ and ψ are from the interior of the supports, there are sufficiently small open neighborhoods $\mathcal{X}$ and ψ, denoted by $V_{\mathcal{X}}$ and V_ψ, such that $\mathcal{C}_{\mathcal{X}',\psi'}$ is still acute-angled for every $\mathcal{X}' \in V_{\mathcal{X}}$ and $\psi' \in V_\psi$, which again leads to the same result $z_1^\star < z_0^\star$. Then, we have

$$\begin{aligned} \mathbb{E}[\rho] = {} & \mathbb{E}[\rho(\mathcal{X}', \psi') \mid \mathcal{X}' \in V_{\mathcal{X}}, \psi' \in V_\psi] \mu(V_{\mathcal{X}}, V_\psi) \\ & + \mathbb{E}[\rho(\mathcal{X}', \psi') \mid \mathcal{X}' \in \mathfrak{X} \setminus V_{\mathcal{X}}, \psi' \in \mathfrak{G}_{\mathcal{X}} \setminus V_\psi](1 - \mu(V_{\mathcal{X}}, V_\psi)). \end{aligned}$$

Since $\mathcal{X}$ and ψ are from the supports, we have $\mu(V_{\mathcal{X}}, V_\psi) > 0$ and the corresponding conditional expectation strictly less than one by the previous argument. Hence, we have $\mathbb{E}[\rho(\mathcal{X}', \psi') \mid \mathcal{X}' \in V_{\mathcal{X}}, \psi' \in V_\psi] < 1$, which proves the claim. □

This theorem says that as long as there exists a pair $(\mathcal{X}, \psi)$ in the interior of the support such that the optimal solution space $\mathcal{X}^\star$ is not too "wide", then the expected iterative convergence rate is strictly less than one. This result applies to all the iterations of the main algorithm. In particular, we have the following corollary.

Corollary 2. $\mathbb{E}[\rho] < 1$ *if one of the following conditions is satisfied,*

1. *There exists some* $\mathcal{X} \in$ *supp*$(\mathfrak{X})$ *and* $\psi \in$ *supp*$(\mathfrak{G}_{\mathcal{X}})$ *such that* $|\mathcal{X}^\star| = 1$.
2. $\mathfrak{X}$ *and* $\mathfrak{G}_{\mathcal{X}}$*'s are fully supported.*
3. $\mathfrak{X}$ *and* $\mathfrak{G}_{\mathcal{X}}$*'s are uniform probability measures.*

Proof. For 1, when $|\mathcal{X}^\star| = 1$ along with the assumption that $x^\star \neq 0$, we trivially have $\mathcal{C}_{\mathcal{X},\psi}$ as acute-angled. For 2, we can easily construct a $\mathcal{X}$ and ψ that satisfies Condition 1. The last condition is a special case of Condition 2. □

4.4 An Improved Convergence Rate

Though Theorem 3 and Corollary 2 show that the iterative objective values of the main algorithm are expected to decrease exponentially, these results are not particularly strong because (i) they depend on the underlying probability measures, which are unlikely to characterize, (ii) $\mathbb{E}[\rho] < 1$ does not exclude the possibility that multiple iterations end up without reducing the objective value at all. The following variant of the main algorithm mitigates these by adding a L_2 regularizer into the objective function. We will show that this small change will lead to a linear convergence rate regardless of the underlying probability measures.

Definition 13 (Regularized Iterative IBI Bidual)**.** The *regularized iterative IBI Bidual* is defined as

$$\begin{aligned} &\hat{\phi}(\bar{V}) := \langle \bar{C}, \bar{V} \rangle - \frac{\delta}{2} \|\bar{A}h(\bar{V})\|^2, \\ &\text{with (10b) — (10d)}, \end{aligned}$$

where $\delta > 0$ is a sufficiently small number and $\|\cdot\|$ is the L_2 norm.

According to [12], for any feasible and bounded linear program $\min_{x \in \mathcal{X}} \langle c, x \rangle$, there always exists some $\delta > 0$ such that the optimal solution of the regularized problem $\min_{x \in j\mathcal{X}} \{\langle c, x \rangle + \delta\phi(x)\}$ is also optimal to the original linear program. Hence, with a sufficiently small δ, the regularized algorithm above still returns an optimal solution for the unregularized version in each iteration. Moreover, by the design of $h(\cdot)$, the regularizer term $\|\bar{A}h(\bar{V})\|^2$ is equivalent to

$$\operatorname{vec}(\bar{V})^\intercal \left((I_n \otimes \bar{A})E_{(2)}\right) \left((I_n \otimes \bar{A})E_{(2)}\right)^\dagger \operatorname{vec}(\bar{V}),$$

which makes the problem a semidefinite program. Then, the following theorem provides the main convergence result of this variant.

Theorem 4. *In the regularized main algorithm, for every* $\mathcal{X} \in \mathfrak{X}$ *and* $\psi \in \mathfrak{G}_{\mathcal{X}}$*, the quotient counterpart of* $\hat{\phi}$ *is* $\hat{\psi}(x) := \psi(x) - (\delta/2)\|x\|^2$*. In particular, the set* $\arg\max_{x \in \mathcal{X}} \hat{\psi}(x)$ *is a singleton* $\{x^\star\}$*. Suppose* $x^\star \neq 0$*, we have*

$$\rho(\mathcal{X}, \hat{\psi}) \leq 1 - \frac{1}{\frac{2L}{\delta\|x^\star\|} - 1},$$

where L *is the Lipschitz constant of* ψ *over* $\mathcal{X}$*.*

Proof. In the definition of $\hat{\phi}(\bar{V})$, $\bar{A}h(\cdot)$ is the projection operator to define the quotient reformulation. Hence, we have $\hat{\phi}(\bar{V}) = \hat{\psi}(x)$. Note that $\hat{\psi}(x)$ is a δ-strongly concave function over a compact

domain $\mathcal{X}$. Thus, the maximizer is unique. Because $\hat{\psi}$ is δ-strongly concave and continuous, and $x^\star$ is a maximizer, we have

$$\hat{\psi}(x^\star) - \hat{\psi}(x) \geq \frac{\delta}{2}\|x^\star - x\|^2, \ \forall x \in \mathcal{X}.$$

In the next iteration of the IBI algorithm, we will have the new domain $\mathcal{X}' := \mathcal{X} \cap S_{x^\star}$ with the new problem $\max_{x\in\mathcal{X}'} \hat{\psi}(x)$, which again leads to a unique maximizer, say x'. Since $x' \in S_{x^\star}$, we have

$$\|x^\star - x'\| \geq \|x^\star\|$$

by the construction of $S_{x^\star}$. This gives,

$$z_0^\star - z_1^\star = \hat{\psi}(x^\star) - \hat{\psi}(x') \geq \frac{\delta}{2}\|x^\star\|^2,$$

which is equivalent to

$$\rho(\mathcal{X}, \hat{\psi}) = z_1^\star / z_0^\star \leq 1 - \frac{\delta\|x^\star\|^2}{2z_0^\star} = 1 - \frac{1}{\frac{2z_0^\star}{\delta\|x^\star\|^2}}.$$

Finally, we have $z_0^\star = \psi(x^\star) - (\delta/2)\|x^\star\|^2$ and

$$\frac{\phi(x^\star)}{\|x^\star\|} = \frac{|\phi(x^\star) - \phi(0)|}{\|x^\star - 0\|} \leq L,$$

which concludes the proof. □

Remark 1. This theorem demonstrates that by adding an L_2 regularizer with a sufficiently small δ, we can ensure an iteratively linear convergence rate regardless of the potential shapes of $\mathcal{X}$ and ψ and their probability measures. Moreover, when the violation $\|x^\star\|$ is large, the upper bound becomes tighter. Of course, δ must satisfy $\delta \leq L/\|x^\star\|$ throughout the process so that the bound is valid (*i.e.*, non-negative).

5 Part 2: An Accelerated IBI with Scalable Implementation

The convergence performance of Algorithm 1 proved in Section 4 and illustrated in Figure 3a demonstrate the potential to obtain a near-optimal affine policy with a compact-sized information basis. However, Algorithm 1 requires solving for the exact solution $\bar{V}_j^\star$ iteratively using linear programming. This often tends to be expensive after a certain number of iterations as shown in Figure 3b.

In this section, we will introduce a scalable implementation to construct an information basis that combines the perks of the two methods: it has a convergence performance close to the IBI algorithm and, similar to the random generation method, its iterative runtime is efficient and is independent of the iterations. To achieve this, we notice that the two main computational bottlenecks in Algorithm 1 are the optimization step for getting $\bar{V}_j^\star$ and the subsequent projection operator $\text{proj}_{\bar{A}\mathcal{F}}(\cdot)$. We will address the efficiency issue for both steps. The resulting convergence and runtime performances are illustrated in Figure 4 for the same set of instances.

5.1 An Improved Projection Operator

Algorithm 1 needs to perform the projection operator $\bar{A}h(\cdot)$ defined in (10e) in each iteration, where the corresponding projection matrix is $n(k+1)$ by $n(k+1)$. Thus, the time complexity for projection is $O(n^2k^2)$. According to the following proposition, with some initialization steps, this can be reduced to $O(k \dim \bar{A}\mathcal{F})$.

Proposition 6. *Let F be a given information filter with nonzero entries indexed by $S = \{(i,j) \mid F_{ij} \neq 0\}$, we define*

$$J = \{j \mid (i,j) \in S \text{ for some } i\},\ I_j = \{i \mid (i,j) \in S\},\ \hat{A}^j = \mathit{OrthoSet}(\bar{A}_{:I_j}),$$

where OrthoSet returns a matrix whose columns form an orthonormal basis of the range of the input matrix. Then, we have

$$\mathit{proj}_{\bar{A}\mathcal{F}}(D) = \left[\hat{A}^j(\hat{A}^j)^\intercal D_{:j}\right]_{j\in[n]}$$

where the corresponding time complexity for projection is $O(k \dim \bar{A}\mathcal{F})$.

Proof. By definition, the jth column of the projected matrix $\bar{A}h(D)$ is obtained by multiplying $\mathrm{vec}(D)$ with the submatrix consisting of the jth $k+1$ rows from the following matrix

$$(I_n \otimes \bar{A})E_{(2)}\left((I_n \otimes \bar{A})E_{(2)}\right)^\dagger.$$

Moreover, each column vector in $E_{(2)}$ is either a zero vector or equal to $\mathrm{vec}(E_{ij})$ for some $(i,j) \in S$ (E_{ij} is a matrix with one at the (i,j)th entry and zero otherwise). Then, a direct computation shows that

$$\left[\mathrm{proj}_{\bar{A}\mathcal{F}}(D)\right]_{:j} = \bar{A}_{:I_j}(\bar{A}_{:I_j})^\dagger D_{:j} = \hat{A}^j(\hat{A}^j)^\intercal D_{:j}.$$

For the time complexity, we further have

$$\hat{A}^j(\hat{A}^j)^\intercal D_{:j} = \sum_{i\in I'_j} \langle \hat{A}^j_{:i}, D_{:j}\rangle \hat{A}^j_{:i}$$

where I'_j is the column index set of $\hat{A}^j$. Note the time complexity for computing each summand is $O(k)$, and the terms to be summed are equal to $\dim \bar{A}\mathcal{F}$, which gives the claimed time complexity. □

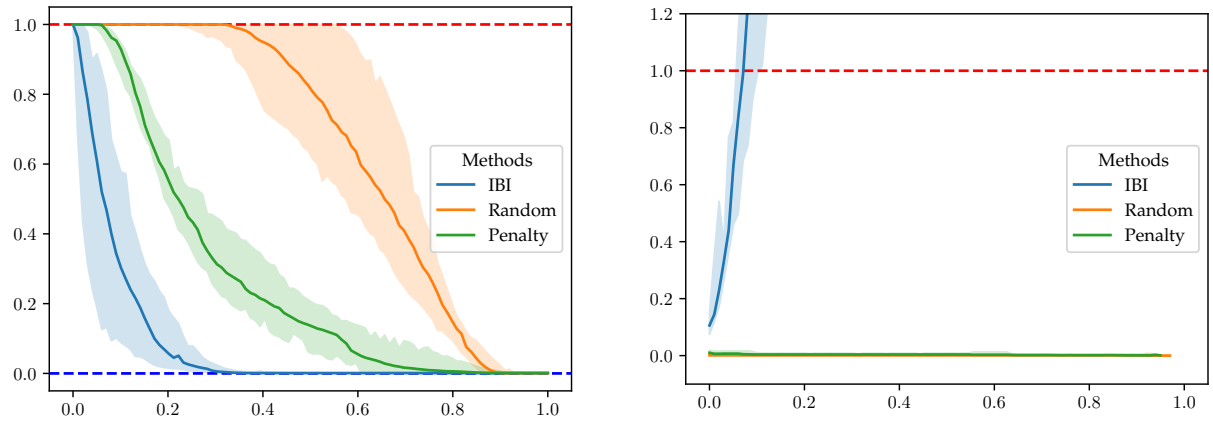
This implementation for the projection operator is much more efficient than the original definition and can benefit all basis generation algorithms that use the projection operator $\bar{A}h(\cdot)$.

5.2 Penalty Variant with SCD

To improve the efficiency of getting $\bar{V}_j^\star$, we note that Algorithm 1 always returns an optimal affine policy as long as a sufficient number of independent basis matrices are generated for (9c). Thus, we have no feasibility nor optimality requirements on $\bar{V}$ in each iteration. All we need is a solution $\bar{V}$ to indicate an approximated direction of $\bar{V}_j^\star$ so that the analysis in Section 4 is still credible.

This allows us to approximate the constrained problem $\max_{\bar{V}\in\mathcal{V}\cap\mathcal{S}_j} \phi(\bar{V})$ using the penalty method formulated as follows,

$$\min_{u\geq 0}\ -\langle \bar{C}, \bar{V}\rangle + \lambda\left(\left\|(B\bar{V}^\intercal - b\bar{u}^\intercal)_+\right\|^2 + \left\|\bar{A}^\intercal\bar{u}\right\|^2 + \sum_{j'<j}\langle \bar{A}h(\bar{V}^*_{j'}), \bar{V}\rangle^2\right) + \frac{1}{4\lambda}\left(\|u\|^2 + \|\bar{V}\|^2\right) \quad (16)$$



(a) Objective Values vs Iterations

(b) Runtimes vs Iterations

Figure 4: Convergence and runtime performances of the three algorithms.

where $\lambda > 0$ is the penalty parameter, $\bar{u}$ is the augmented vector $(1, u)$, function $(\cdot)_+$ denotes the nonnegativity projection $\max(\cdot, 0)$, matrices $\bar{V}_{j'}^*$'s are the solutions obtained in previous iterations using the same penalty method, and the last term is a regularizer.

In each iteration j, we solve this penalty variant using the SCD algorithm with a fixed number of steps to obtain a normalized approximated optimal solution $\bar{V}_j^*$, then add one more term to the summation in (16) and proceed to the next iteration. The details of the SCD algorithm and its accuracy and convergence in LP approximation have been extensively studied in [21].

In our implementation, the complexity of obtaining a solution $\bar{V}_j^*$ has the same order as random generation. Randomly producing a basis matrix $\bar{V}$ is of order $O(nk)$ as there are $n(k+1)$ entries in $\bar{V}$. On the other hand, since we fix the number of gradient steps, the complexity is dominated by a single gradient update step of the SCD algorithm. Furthermore, because only one entry of the solution will be updated in each step, it is easy to check that the number of affected gradient entries in (16) is below $O(nk)$ for every possible entry in u and $\bar{V}$. Therefore, the time complexity is dominated by the solution initialization step with order $O(nk)$. Due to its computational significance, we record this result in the following proposition.

Proposition 7. *The time complexity of obtaining an approximated solution of* (16) *using the SCD algorithm is* $O(nk)$*, which is equivalent to random generation.*

5.3 Performance and Scalability

Figure 4 illustrates this penalty variant's convergence and runtime performances. It still exhibits an exponentially decreasing trend in the objective value convergence, while the corresponding runtime performance is almost on par with the random generation and has a small variation. This also supports our complexity analysis derived in the previous subsection.

Finally, the scalability of this penalty variant stems primarily from its most computationally demanding steps: the projection operation and the gradient updates. Both of these processes are implemented through matrix-vector multiplication operations. Hence, by leveraging GPU and TPU acceleration, this method is quite scalable to handle large-sized input instances.

6 Experiment

In this section, we will provide details for previous computational results, conduct further experiments to study the effects of adding the regularizer and demonstrate the performance of the proposed method under different information filters.

We conducted all experiments on a computer equipped with a 12th Gen Intel(R) Core(TM) i5-12600 3.30 GHz processor and 16GB of RAM and running Ubuntu 20.04. All the main algorithms were implemented in Python 3.10.10 and solved by the commercial optimizer Gurobi 10.0.1. To optimize the SGD subroutine used for solving the penalty variant, we converted the corresponding Python codes into C language using the package Cython 0.29.35.

6.1 Test Instances

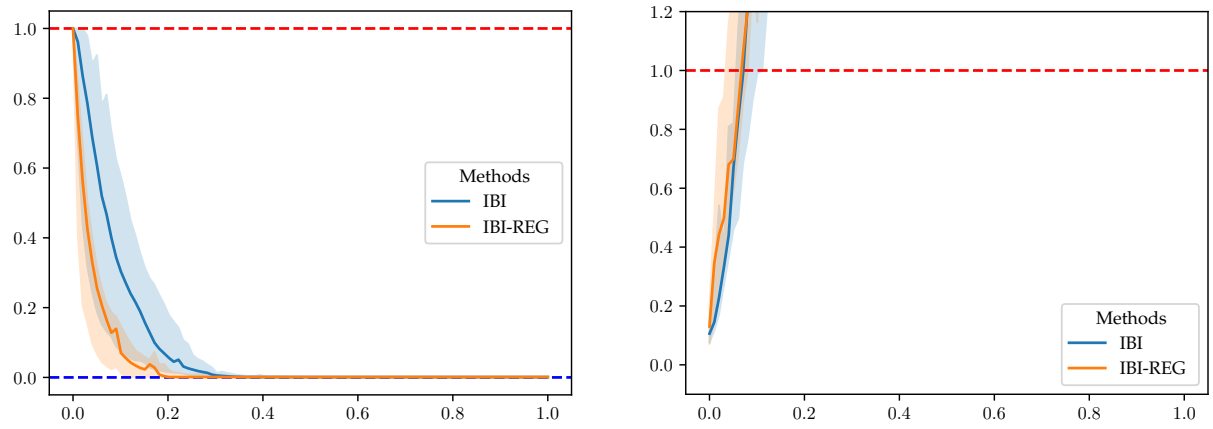
We generate 50 instances with 5 decision stages, each of which has 6 decision variables and 6 uncertain parameters. The number of constraints describing each stage's uncertainty set ranges from $\{3, 4, \ldots, 11\}$. For each instance, we design input parameters B and b so that the uncertainty set Ξ is a multi-stage budget set. We also randomly generate the rest of the input parameters (A, a, C, c) and reject the instance if the objective value is non-positive or the problem is infeasible or unbounded. This produces LP formulations for the affine bidual (9) with $1,800$ to $4,400$ variables and $3,500$ to $7,600$ constraints. We choose problems of this size because, for every algorithm, we need to solve each of these formulations with $\dim \bar{A}\mathcal{F}$ iterations to obtain the iterative performances such as in Figure 4, and the value $\dim \bar{A}\mathcal{F}$ of our instances ranges from 270 to $2,600$.

The iteration count and runtime required to generate the complete information basis differ across various algorithms and instances. The IBI algorithm operates within a range of 410 to $1,281$ iterations, with corresponding runtimes from 13.39 to 703.38 seconds. In contrast, the penalty variant requires 825 to $2,414$ iterations, with runtimes between 0.16 and 0.50 seconds, while the random counterpart operates within 953 to $2,469$ iterations, with runtimes ranging from 0.04 to 0.14 seconds. This illustrates the efficiency of both the penalty variant and random counterpart, as their time complexity for generating the entire information basis is of order $O(nk \dim \bar{A}\mathcal{F})$ according to Proposition 7. We note that the current experiment does not utilize any GPU speedup for solving these instances.

To facilitate a meaningful comparison among different instances and algorithms, we normalized the number of iterations, objective values, and runtimes to a $[0, 1]$ scale. Specifically, zero and one in iterations correspond to the initial and final iterations (*i.e.*, $\dim \bar{A}\mathcal{F}$), zero and one in objective values represent the optimal values obtained by affine and constant policies, respectively, and zero and one in runtime signify the start time and the time taken to solve the affine bidual formulation using Gurobi with default settings.

6.2 Convergence and Runtime Comparison

Figure 4 compares the three algorithms in the paper: the IBI algorithm (Algorithm 1), the penalty variant with the SCD algorithm, and the random counterpart. Figure 4a shows the iterative convergence of the three algorithms, while Figure 4b compares their basis generation runtime. Note that Algorithm 1 requires solving the associated optimization problem (9) in each iteration to generate a new basis matrix. Thus, it can also evaluate the corresponding objective value simultaneously. For the other two counterparts, the basis generation subroutine can be executed first, and then their objective convergence performance can be evaluated subsequently. From Figure 4, the benefits of such implementation are clear: the basis generation part becomes much more efficient, and its



(a) Objective Values vs Iterations

(b) Runtimes vs Iterations

Figure 5: Convergence and runtime comparison between Algorithm 1 with and without the regularizer.

runtime is uncorrelated to the number of constraints in (9c). For convergence, sacrificing some accuracy, it is expected that the penalty counterpart has a less impressive performance compared to Algorithm 1, but it still retains the momentum of a rapid decrease in the objective value.

6.3 Effects of Regularizer

Theorem 4 derived a much tighter convergence bound than Theorem 3 by adding a regularizer to Formulation (9). This conclusion is also supported by the experiment results shown in Figure 5.

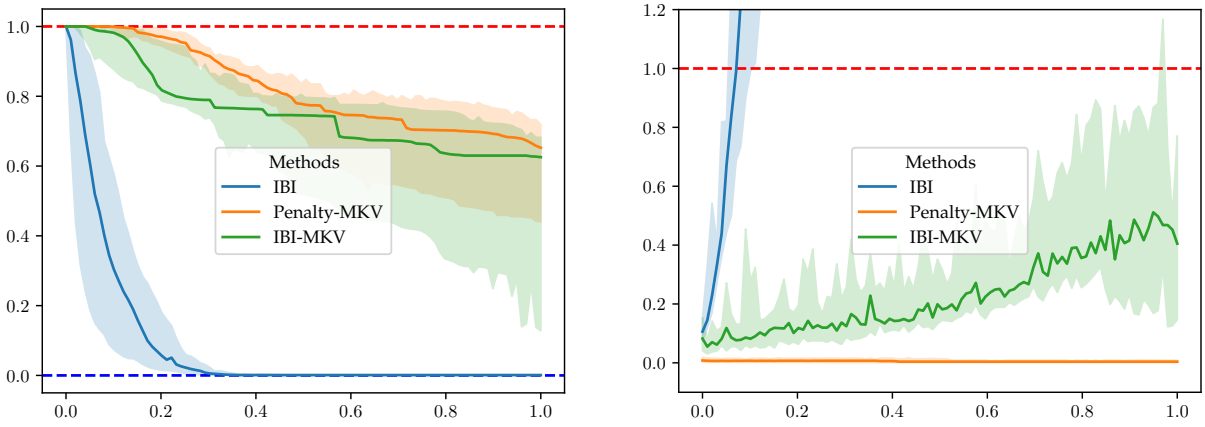
The orange and blue lines represent the average convergence and runtime performances of Algorithm 1 with and without the regularizer, respectively. Clearly, adding the regularizer has a significant impact on the convergence rate. On average, with around 10% of the number of basis matrices, the corresponding optimal IBI policy can close more than 90% of the gap between the optimal constant and affine policies, whereas a similar performance requires around 20% of the number of basis matrices if without the regularizer. Moreover, the convergence variation is also much smaller in the case with the regularizer.

In terms of runtime, both algorithms are quite inefficient, and the one with regularizer is slightly slower. This is unsurprising as Formulation (9) with a regularizer needs to be solved as a semidefinite program.

6.4 Markovian Information Filter

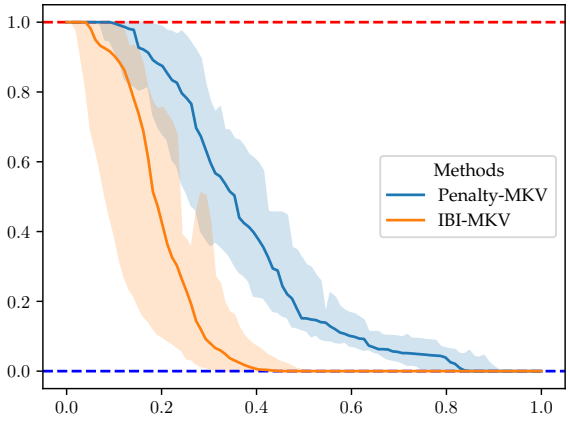
By design, the proposed algorithm in Section 3 and the corresponding analysis in Section 4 are valid for every given information filter F. In this subsection, we demonstrate this using the Markovian filter (see (e) in Figure 2).

Figure 6 shows the convergence and runtime performances for Algorithm 1 and the penalty variant under the Markovian filter, which are denoted by IBI-MKV and Penalty-MKV, respectively. We still use the standard IBI algorithm as the baseline algorithm for comparison. In Figure 6a, the values 0 and 1 in the y-axis still represent the normalized objective values obtained by the optimal affine and constant policies, respectively. This comparison result shows that the Markovian filter is not optimal for most instances since they did not converge to the value of the optimal affine policy even after a sufficient number of iterations. We can also observe that both IBI-MKV and



(a) Objective Values vs Iterations

(b) Runtimes vs Iterations



(c) Normalized Objective Values vs Iterations

Figure 6: Convergence and runtime performance comparison under the Markovian information filter.

Penalty-MKV algorithms have the same convergence trend. But, on average, IBI-MKV converges faster than Penalty-MKV.

In terms of runtime, IBI-MKV has a considerable improvement in efficiency compared to IBI thanks to the sparse Markovian filter, which significantly reduces the number of parameters. However, its runtime still increases with the number of iterations and has quite a large variation. In comparison, the runtime performance of Penalty-MKV is consistent with our previous conclusion: it is fast, independent of the iterations, and has a low variance.

Finally, in Figure 6c, we remove the IBI algorithm and re-normalized the objective values where 0 now represents the value of the optimal affine policy under the Markovian filter. From this figure, it becomes clear that both algorithms converge to the same value eventually, *i.e.*, the value of the optimal affine policy in the filtered space. Similar to the unfiltered case, both algorithms exhibit an exponential decrease in the objective values, yet IBI-MKV converges faster in general. This indeed shows that the proposed method and analysis are compatible with various predefined information filters. Moreover, the penalty variant with the SCD implementation still performs well to efficiently generate an effective information basis.

7 Conclusion

This paper focused on finding a sweet spot between the fast-but-imprecise constant policies and the accurate-yet-unscalable affine policies for dynamic robust optimization. We adopted the concept of information filters to incorporate prior knowledge and requirements about the policy space to reduce the problem size, then developed the IBI algorithm to construct a small-and-effective information basis for approximating the optimal policy in the filtered policy space.

The key finding is that, for general dynamic robust optimization problems, a near-optimal affine policy can be achieved with a compact information basis, and such a basis can be efficiently constructed with a proper implementation. Specifically, we proved that our proposed method returns the best information basis matrix in each iteration and can exponentially converge to the value of the optimal affine policy. We also developed a penalty counterpart with the SCD algorithm so that the basis generation efficiency is on par with random generation.

These results unlock many potential applications for information bases. For instance, large-sized multistage robust optimization can be approximated by efficiently generating a small number of basis matrices. Or, we can construct nonlinear decision rules using these basis matrices in order to obtain a better policy performance with a small number of parameters.

For future directions, it would be important to further enhance the penalty variant's convergence performance while preserving its computational efficiency. Additionally, it is also interesting to explore the generalization of our analysis to nonlinear settings.

References

[1] Amir Ardestani-Jaafari and Erick Delage. Robust optimization of sums of piecewise linear functions with application to inventory problems. *Operations Research*, 64(2):474–494, 2016.

[2] Pranjal Awasthi, Vineet Goyal, and Brian Y Lu. On the adaptivity gap in two-stage robust linear optimization under uncertain constraints. *Math. Program*, 2015.

[3] Amir Beck and Aharon Ben-Tal. Duality in robust optimization: primal worst equals dual best. *Operations Research Letters*, 37(1):1–6, 2009.

[4] Aharon Ben-Tal, Alexander Goryashko, Elana Guslitzer, and Arkadi Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical programming*, 99(2):351–376, 2004.

[5] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton university press, 2009.

[6] Dimitris Bertsimas and Vineet Goyal. On the power of robust solutions in two-stage stochastic and adaptive optimization problems. *Mathematics of Operations Research*, 35(2):284–305, 2010.

[7] Dimitris Bertsimas and Vineet Goyal. On the power and limitations of affine policies in two-stage adaptive optimization. *Math Programming*, 134:491–531, 2012.

[8] Dimitris Bertsimas, Dan A Iancu, and Pablo A Parrilo. Optimality of affine policies in multi-stage robust optimization. *Mathematics of Operations Research*, 35(2):363–394, 2010.

[9] Dimitris Bertsimas, Vineet Goyal, and Brian Y Lu. A tight characterization of the performance of static solutions in two-stage adjustable robust linear optimization. *Mathematical Programming*, 150(2):281–319, 2015.

[10] Xin Chen and Yuhan Zhang. Uncertain linear programs: Extended affinely adjustable robust counterparts. *Operations Research*, 57(6):1469–1482, 2009.

[11] Omar El Housni and Vineet Goyal. On the optimality of affine policies for budgeted uncertainty sets. *Mathematics of Operations Research*, 2021.

[12] Michael P Friedlander. Exact regularization of linear programs. *Department of Computer Science Tech. Rep.*, 2005.

[13] Angelos Georghiou, Angelos Tsoukalas, and Wolfram Wiesemann. Robust dual dynamic programming. *Operations Research*, 67(3):813–830, 2019.

[14] Ali Haddad-Sisakht and Sarah M Ryan. Conditions under which adjustability lowers the cost of a robust linear program. *Annals of Operations Research*, 269(1):185–204, 2018.

[15] Omar El Housni, Ayoub Foussoul, and Vineet Goyal. Lp-based approximations for disjoint bilinear and two-stage adjustable robust optimization. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 223–236. Springer, 2022.

[16] Dan A. Iancu, Mayank Sharma, and Maxim Sviridenko. Supermodularity and affine policies in dynamic robust optimization. *Operations Research*, 61(4):941–956, 2013.

[17] Ji Liu, Steve Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. An asynchronous parallel stochastic coordinate descent algorithm. In *International Conference on Machine Learning*, pages 469–477. PMLR, 2014.

[18] Haihao Lu and Brad Sturt. On the sparsity of optimal linear decision rules in robust inventory management. *arXiv preprint arXiv:2203.10661*, 2022.

[19] Ahmadreza Marandi and Dick den Hertog. When are static and adjustable robust optimization problems with constraint-wise uncertainty equivalent? *Math. Program.*, 170(2):555–568, 2018.

[20] David Simchi-Levi, Nikolaos Trichakis, and Peter Yun Zhang. Designing response supply chain against bioattacks. *Operations Research*, 67(5):1246–1268, 2019.

[21] Srikrishna Sridhar, Stephen Wright, Christopher Re, Ji Liu, Victor Bittorf, and Ce Zhang. An approximate, efficient lp solver for lp rounding. *Advances in Neural Information Processing Systems*, 26, 2013.

[22] Ningji Wei and Peter Zhang. Adjustability in robust linear optimization. *Mathematical Programming*.

[23] İhsan Yanıkoğlu, Bram L Gorissen, and Dick den Hertog. A survey of adjustable robust optimization. *European Journal of Operational Research*, 277(3):799–813, 2019.