PAC Learnability of Scenario Decision-Making Algorithms: Necessary Conditions and Sufficient Conditions

Guillaume O. Berger, and Raphaël M. Jungers, *Senior Member, IEEE*

***Abstract*— We investigate the Probably Approximately Correct (PAC) property of scenario decision algorithms, which refers to their ability to produce decisions with an arbitrarily low risk of violating unknown safety constraints, provided a sufficient number of realizations of these constraints are sampled. While several PAC sufficient conditions for such algorithms exist in the literature—such as the finiteness of the VC dimension of their associated classifiers, or the existence of a compression scheme—it remains unclear whether these conditions are also necessary. In this work, we demonstrate through counterexamples that these conditions are not necessary in general. These findings stand in contrast to binary classification learning, where analogous conditions are both sufficient and necessary for a family of classifiers to be PAC. Furthermore, we extend our analysis to stable scenario decision algorithms, a broad class that includes practical methods like scenario optimization. Even under this additional assumption, we show that the aforementioned conditions remain unnecessary. Furthermore, we introduce a novel quantity, called the dVC dimension, which serves as an analogue to the VC dimension for scenario decision algorithms. We prove that the finiteness of this dimension is a PAC necessary condition for scenario decision algorithms. This allows to (i) guide algorithm users and designers to recognize algorithms that are not PAC, and (ii) contribute to a comprehensive characterization of PAC scenario decision algorithms.**

***Index Terms*— Randomized Algorithms, Scenario Design, Statistical Learning.**

I. INTRODUCTION

RISK-aware decision making is an important problem in engineering, encompassing many applications of great importance, such as control, energy planning, healthcare, etc. One key challenge in this problem is that the distribution of the uncertainty is usually unknown, so that one has to rely on observations to make a decision that has a low *risk* (probability of failure). The approach of *scenario decision making* [1]–[15] provides an effective way to address this problem by using the principle of sample-based methods. Concretely, this approach consists in sampling N realizations of the uncertainty, and making a decision based on these samples. The process is called a *scenario decision algorithm*, mapping finite sets of samples to decisions. Under some conditions on the problem and the algorithm, one can guarantee with high probability that if enough samples are provided to the algorithm, the returned decision has a low risk. This property is called the *PAC (Probably Approximately Correct)* property.

Several PAC sufficient conditions, i.e., conditions on the problem and the scenario decision algorithm ensuring that the algorithm is PAC, have been studied in the literature; see, e.g., [16] for a survey. These conditions can be grouped into two categories:[1] *complexity-based* conditions and *compression-based* conditions. Complexity-based conditions, such as those in [17]–[19], provide PAC guarantees for scenario decision problems whose decision space has bounded "complexity" such as bounded VC dimension or Rademacher complexity [16]. In this work, we focus on complexity-based PAC conditions that are based on the VC dimension, as in [18]. Compression-based conditions, such as those in [2], [3], [15], [20], provide PAC guarantees for scenario decision algorithms whose input can be "compressed", meaning that a subset of the samples provides the same decision as the complete set of samples [16]. Note that the results in [2], [3], [15] require additional assumptions, such as *stability* (see Definition 8) and *non-degeneracy* ([3, Definition 2.7]).

The objective of this work is to progress in the understanding of what makes a scenario decision algorithm PAC or not. Our starting point is to ask whether the above PAC sufficient conditions are also necessary. We show with counterexamples that this is not the case (Section III). We do this for general scenario decision algorithms, and for *stable* ones (with slightly weaker conclusions); see Table I for a summary. Designing and analyzing these counterexamples is the first main contribution of this work. The second main contribution is to provide a novel quantity, similar to the VC dimension, for scenario decision algorithms, and showing that finiteness of this quantity is a PAC necessary condition (Section IV). We also show that this PAC necessary condition is not sufficient.

This paragraph of the first footnote will contain the date on which you submitted your paper for review. This project has received funding from (i) the Belgian National Fund for Scientific Research (F.R.S.–FNRS), (ii) the European Research Council (ERC) [under the European Union's Horizon 2020 research and innovation programme, grant agreement no. 864017, project name: L2C], and (iii) the ARC (French Community of Belgium) [project name: SIDDARTA].

G. Berger is with ICTEAM institute, UCLouvain, Louvain-la-Neuve, Belgium. He is an FNRS Postdoctoral Researcher (e-mail: guillaume.berger@uclouvain.be).

R. Jungers is with ICTEAM institute, UCLouvain, Louvain-la-Neuve, Belgium. He is an FNRS honorary Research Associate (e-mail: raphael.jungers@uclouvain.be).

[1]Three categories in [16], but we merged "compression-based methods" and "scenario-based methods" because of their similarities.

	Infinite VC dim.	No Compr. Scheme	No Compr. Map	Finite dVC dim.
PAC	Sec. III-A	Sec. III-B	Sec. III-B	$\not\nearrow$
Stable PAC	Sec. III-A	Empty?	Sec. III-C	$\not\nearrow$
Not PAC	$\not\nearrow$ [18]	$\not\nearrow$ [21]	$\not\nearrow$ [21]	Sec. IV-A

TABLE I

SUMMARY OF OUR RESULTS AND COUNTEREXAMPLES (E.G., SEC. III-A CONTAINS AN ALGORITHM THAT IS PAC, AND STABLE PAC, AND HAS INFINITE VC DIMENSION). THE SYMBOL "$\not\nearrow$" MEANS THAT THE CONDITION OF THE ROW IMPLIES THE CONDITION OF THE COLUMN.

A. Connections with the Literature

To the best of our knowledge, this is the first work studying the necessity of PAC conditions for scenario decision algorithms. The algorithms in [17, Theorem 2.1] and [22, Example 23.1] are shown to have no compression scheme and be PAC-like (through VC theory); however, the notion of PAC used there is different since the algorithm returns a *set* of decisions, and PAC is defined with respect to the largest risk of a decision in the returned set; see also, e.g., [7, Def. 1]. The algorithm in [7, Appendix A] is shown to have an unbounded number of support constraints. With a bit of work, it can be shown to be a valid counterexample for Section III-C. More broadly, we believe that other examples in the literature could potentially serve as counterexamples in Section III. However, to the best of our knowledge, these examples have not been explicitly demonstrated to meet the required conditions. In contrast, our counterexamples are rigorously proven to do so. Furthermore, they have been carefully designed to be as simple as possible, serving two key purposes: (i) facilitating ease of analysis, and (ii) acting as prototypical examples of problem classes that satisfy the requirements; thereby guiding algorithm designers to identify appropriate PAC sufficient conditions for their problem. For instance, Section III-C exemplifies a nonconvex optimization problem with finite VC dimension.

Notation: For $n \in \mathbb{N}$, we let $[n] = \{1, \ldots, n\}$. For a set A, we let $A^* = \bigcup_{m=0}^{\infty} A^m$.

II. RISK-AWARE SCENARIO DECISION MAKING

We introduce the problem of risk-aware scenario decision making.[2] We assume given a set X of *decisions* and a set $Z \subseteq 2^X$ of *constraints* on X. Given a decision $x \in X$ and a constraint $z \in Z$, we say that x *satisfies* z if $x \in z$; otherwise, we say that x *violates* z.

Example 1: In the following optimization problem:

$$\min_{x \in \mathbb{R}^2} \|x\| \quad \text{s.t.} \quad a^\top (x - c) \le 1 \ \ \forall\, a \in \mathbb{R}^2,\ \|a\| \le 1,$$

where $c \in \mathbb{R}^2$ is fixed, the decision space is $X = \mathbb{R}^2$, and the constraint space is $Z = \{C(a) \subseteq X : a \in \mathbb{R}^2,\ \|a\| \le 1\}$, where $C(a) = \{x \in \mathbb{R}^2 : a^\top (x - c) \le 1\}$.

Finding a decision that satisfies all constraints z in Z is often intractable if Z is large or unknown. An approach to circumvent this—in the case where Z can be sampled—is to sample N constraints $z_1, \ldots, z_N$ from Z, and find a decision x that satisfies the sampled constraints. The sample-based problem is called the *scenario problem*, and the associated decision the *scenario decision*. Since the scenario problem is a relaxation of the original problem (aka. the *robust problem* [6]), one can generally not hope that the scenario decision satisfies *all* the constraints in Z. However, under some conditions (the study of necessary and sufficient such conditions is the object of this paper), one can ensure with high probability that the scenario decision satisfies all constraints in Z except possibly those in a subset of small measure. We formalize this below:

Definition 1: Given a probability measure P on Z, the *violation probability* (aka. *risk*) of a decision $x \in X$ w.r.t. P, denoted by $V_{\mathsf{P}}(x)$, is the probability that x violates a randomly chosen constraint $z \in Z$, i.e., $V_{\mathsf{P}}(x) \triangleq \mathsf{P}[\{z \in Z : x \notin z\}]$.

As mentioned above, scenario decision making consists in sampling N constraints from Z, and finding a decision that satisfies the sampled constraints. For instance, one can define the scenario decision as the decision in X that minimizes some cost function while satisfying the sampled constraints: this is the setting of *scenario optimization* [3], [15]. Hence, there is a mapping from tuples of constraints from Z (the samples) to decisions in X (the scenario decision):

Definition 2: A *scenario decision algorithm* is a function that given a tuple of constraints $(z_1, \ldots, z_N) \in Z^*$ returns a decision $x \in X$. Hence, it is a function $\mathcal{A} : Z^* \to X$.

Remark 1: We consider *tuples* of constraints, instead of *sets* of constraints, because in general the order and multiplicity of the sampled constraints may influence the scenario decision.

Example 2: Continuing Example 1, a scenario decision algorithm $\mathcal{A}$ can be defined as follows: given $z_i = C(a_i)$ for $i = 1, \ldots, N$, $\mathcal{A}(z_1, \ldots, z_N)$ is the optimal solution of

$$\min_{x \in \mathbb{R}^2} \|x\| \quad \text{s.t.} \quad a_i^\top (x - c) \le 1 \ \ \forall\, i \in [N].$$

This is an instance of scenario optimization.

The ability of a scenario decision algorithm to return a decision that has a low risk, provided enough constraints are sampled, is called the *PAC (Probably Approximately Correct) property*. More precisely, for any PAC algorithm, one can ensure with a predefined probability (called *confidence*) that if enough constraints are sampled, the risk of the decision is below a predefined *tolerance*:

Definition 3: Consider a scenario decision algorithm $\mathcal{A}$. We say that $\mathcal{A}$ is *PAC* if for any tolerance $\epsilon \in (0, 1)$ and any confidence $1 - \beta \in (0, 1)$, there is a sample size $N_\circ \in \mathbb{N}$ such that for any probability measure P on Z and any $N \ge N_\circ$, it holds with probability $1 - \beta$ that if one samples N constraints $(z_1, \ldots, z_N)$ i.i.d. according to P, then the decision returned by $\mathcal{A}$ has risk below ϵ, i.e.,

$$\mathsf{P}^N\left[\left\{\mathbf{z} \in Z^N : V_{\mathsf{P}}(\mathcal{A}(\mathbf{z})) > \epsilon\right\}\right] \le \beta,$$

where $\mathbf{z}$ is a shorthand notation for $(z_1, \ldots, z_N)$.

Remark 2: In this work, we focus on scenario algorithms that return a single decision, as opposed to scenario algorithms that return a set of decisions, such as those in [17], [22]. There, the violation probability of a set of decisions is defined as the largest violation probability among all decisions in the set. This framework is different from ours, and typically precludes

[2]The definitions and concepts presented in this section are classical, and can be found, e.g., in [13], [15], [18], [20].

the use of compression-based PAC results (defined in the next section) since set-valued scenario decision algorithms rarely admit a compression scheme. See also Remark 4.

In this work, we focus on scenario decision algorithms that are consistent, i.e., that return a decision that satisfies all the sampled constraints:

Definition 4: A scenario decision algorithm $\mathcal{A}$ is *consistent* if for all $(z_1, \ldots, z_N) \in Z^*$, $\mathcal{A}(z_1, \ldots, z_N) \in \bigcap_{i=1}^N z_i$.

Remark 3: A large class of scenario decision algorithms considered in the literature are consistent [2], [3], [8], [9], [13], [15], [23]. See also Example 2. Scenario decision algorithms that are not consistent include those in [4], [12], [14], [18], [20]. Nevertheless, since one of the main goals of this paper is to show that well-known PAC sufficient conditions (see below) are not necessary, there is no loss of generality in showing it under the stronger assumption of consistency.

A. PAC Sufficient Conditions

Sufficient conditions for being PAC, available in the literature, can be grouped into two categories: complexity-based conditions and compression-based conditions. We present one central condition for each category:

1) Complexity-Based Condition: This condition, introduced in [18], relies on the connection between scenario decision making and binary classification learning. The idea is that each decision $x \in X$ induces a classifier of the constraints z in Z based on whether $x \in z$, or not. One can then define the *VC dimension* of the set of classifiers induced by X (see Definition 5). It is well known that, in the context of binary classification learning, any set of classifiers with finite VC dimension enjoys PAC properties [24, Theorem 6.7]. By leveraging this result, [18] obtained PAC sufficient conditions for scenario decision algorithms, as explained below.

For each $x \in X$, we let $\mathcal{S}(x) = \{z \in Z : x \in z\}$ be the set of constraints in Z that x satisfies. So, x classifies Z into two classes: $\mathcal{S}(x)$ and $Z \setminus \mathcal{S}(x)$. The *range* of $\mathcal{A}$, denoted by $\mathcal{R}(\mathcal{A})$, is the set of all classifiers that it can return:

$$\mathcal{R}(\mathcal{A}) = \{\mathcal{S}(\mathcal{A}(\mathbf{z})) : \mathbf{z} \in Z^*\}.$$

We next recall the definition of the VC dimension of a set of classifiers, first introduced by [25]:

Definition 5: Let $C \subseteq 2^Z$ be a set of classifiers. A subset $Z' \subseteq Z$ is *shattered* by a C if for every classifier $T \subseteq Z'$, there is $S \in C$ such that $T = S \cap Z'$. The *VC dimension* of C is the supremum of all integers k for which there is a subset $Z' \subseteq Z$ of cardinality k that is shattered by C.

Theorem 1 ([18, Theorem 3]): Consider a consistent scenario decision algorithm $\mathcal{A}$. Assume that $\mathcal{R}(\mathcal{A})$ has finite VC dimension. Then, $\mathcal{A}$ is PAC.

2) Compression-Based Condition: Several compression-based PAC results have been proposed in the literature [15], [20], [23]. These results rely on the notion of compression reminded below, which may slightly vary across works. We first introduce the notion of compression used in [15], [23]. The idea is that a scenario decision algorithm $\mathcal{A}$ has compression size d if for any tuple of constraints $(z_1, \ldots, z_N) \in Z^N$, one can extract a subtuple of length at most d, from which $\mathcal{A}$ gives the same decision as from $(z_1, \ldots, z_N)$.

Definition 6: Consider a scenario decision algorithm $\mathcal{A}$. A *compression map* of capacity d for $\mathcal{A}$ is a function $\kappa : Z^* \to Z^{\leq d}$ satisfying that for every $(z_1, \ldots, z_N) \in Z^*$, (i) $\kappa(z_1 \ldots, z_N) = (z_{i_1}, \ldots, z_{i_r})$ for some integers $1 \leq i_1 < \ldots < i_r \leq N$, and (ii) $\mathcal{A}(\kappa(z_1, \ldots, z_N)) = \mathcal{A}(z_1, \ldots, z_N)$.

A slightly more general notion of compression, called here *compression scheme*, was introduced in [21] (see also [24, §30]) and used in [20] for scenario decision making:[3]

Definition 7: Consider a scenario decision algorithm $\mathcal{A}$. A *compression scheme* of capacity d for $\mathcal{A}$ is a pair (κ, ρ), where $\kappa : Z^* \to Z^{\leq d}$ is called the *compression map* and $\rho : Z^{\leq d} \to X$ the *reconstruction map*, satisfying that for every $(z_1, \ldots, z_N) \in Z^*$, (i) $\kappa(z_1 \ldots, z_N) = (z_{i_1}, \ldots, z_{i_r})$ for some integers $1 \leq i_1 < \ldots < i_r \leq N$, and (ii) $\rho(\kappa(z_1, \ldots, z_N)) = \mathcal{A}(z_1, \ldots, z_N)$.

Clearly, if $\mathcal{A}$ admits a compression map of capacity d, then it admits a compression scheme of capacity d.

Theorem 2 ([24, Theorem 30.2]): Consider a consistent scenario decision algorithm $\mathcal{A}$. Assume that $\mathcal{A}$ admits a compression scheme or map. Then, $\mathcal{A}$ is PAC.

B. Stable Scenario Decision Algorithms

Stability is an important property of many practical scenario decision algorithms, such as scenario optimization [2]–[4], [9], [13], [18], [20]; see also [15], [16], [23], [27] and Example 2. This property captures the fact that if the returned decision satisfies some unsampled constraint, then adding this constraint to the set of sampled constraints does not change the decision:

Definition 8: A consistent scenario decision algorithm $\mathcal{A}$ is *stable* if for all $(z_1, \ldots, z_{N+1}) \in Z^{\geq 1}$, $\mathcal{A}(z_1, \ldots, z_N) \in z_{N+1}$ implies that $\mathcal{A}(z_1, \ldots, z_{N+1}) = \mathcal{A}(z_1, \ldots, z_N)$.

In the next section, we study the necessity of PAC sufficient conditions, for both stable and non-stable scenario decision algorithms.

III. The VC- and Compression-Based Sufficient Conditions Are Not Necessary

The main topic of this paper is to address the key question of whether the sufficient conditions in Theorems 1 and 2 are also necessary. It turns out that the answer is negative for general scenario decision algorithms. Next, we consider the additional assumption of *stability*. We show that, even under this additional assumption, the answer is mostly negative. The following theorem summarizes these results (see also Table I):

Theorem 3: (i) There exist consistent stable PAC scenario decision algorithms whose range has infinite VC dimension. (ii) There exist consistent PAC scenario decision algorithms that do not admit a compression scheme. (iii) There exist stable consistent PAC scenario decision algorithms that do not admit a compression map.

[3]The definition used in [21], [26] is actually slightly more general than Definition 7, because—on top of the indices $\{i_1, \ldots, i_r\}$—an information q belonging to a *finite* information set Q is passed by the compression map to the reconstruction map. For the sake of simplicity and because our results directly extend to this extended framework, we focus on the slightly more restricted, but simpler, framework.

The question of existence of a stable PAC scenario decision algorithm without a compression scheme remains open.

We prove Theorem 3 by providing counterexamples in the next subsections. The counterexamples have been simplified to the maximum for the ease of analysis, but they capture the essence of algorithms used in practical problems. The purpose is to (i) allow algorithm users and designers to recognize which PAC sufficient condition may be better suited for the analysis of their algorithm, and (ii) build the foundations for a complete characterization of PAC scenario decision algorithms.

All scenario decision algorithms in the rest of this section are consistent. Hence, we will drop this adjective when referring to them.

A. Stable PAC Scenario Decision Algorithm with Infinite VC Dimension

To prove (i) in Theorem 3, we provide an example of stable PAC scenario decision algorithm $\mathcal{A} : Z^* \to X$ whose range has infinite VC dimension. The algorithm is a convex scenario program (like Example 2). The construction is inspired by [28] and is illustrated in Figure 1.

The decision space X is the Euclidean unit ball in $\mathbb{R}^2$. To present the constraint set, we need some preliminary constructions. Let $D \subseteq X$ be the open arc corresponding to the upper-right quarter of the unit circle, i.e., $D = \{(\cos(\alpha), \sin(\alpha)) : 0 < \alpha < \frac{\pi}{2}\}$. Let U be the set of all finite subsets of $\mathbb{N}_{>0}$, and let $\tau : U \to D \cup \{(1,0)\}$ be a one-to-one function satisfying $\tau(u) \in D$ if $u \neq \emptyset$ and $\tau(\emptyset) = (1,0)$.[4] Let $G = \{(m,i) \in \mathbb{N}^2 : i \leq m\}$, and $\xi : G \to 2^U$ be defined by $\xi(m,i) = \{u \subseteq [m] : i \in u\}$, i.e., $\xi(m,i)$ contains all the subsets of $[m]$ that contain i.[5] Finally, let $\sigma : G \to 2^X$ be defined by $\sigma(m,i) = \mathrm{convexhull}(\{(0,1)\} \cup \{\tau(u) : u \in \xi(m,i)\})$, i.e., $\sigma(m,i)$ is the convex hull of the point $(0,1)$ and the points $\tau(u)$ with $u \in \xi(m,i)$. Note that σ is one-to-one. With these preliminaries, we define the constraint set by $Z = \{\sigma(m,i) : (m,i) \in G\} \cup \{\mathbb{R} \times [y,1] : y \in [0,1]\}$. Finally, we define the algorithm $\mathcal{A} : Z^* \to X$ as follows: on input $(z_1, \ldots, z_N) \in Z^*$, let

$$\mathcal{A}(z_1, \ldots, z_N) = \arg\max_{(x_1,x_2) \in X \cap \bigcap_{i=1}^N z_i} x_1.$$

It is easy to see that $\mathcal{A}$ is well defined, i.e., the argmax exists and is unique.

Proposition 1: The VC dimension of $\mathcal{R}(\mathcal{A})$ is infinite.

Proof: Fix $k \in \mathbb{N}$ and let $Z' = \{\sigma(k,i) : i \in [k]\}$. We show that Z' is shattered by $\mathcal{R}(\mathcal{A})$. For that, fix $T \subseteq Z'$, and we will show that there is $\mathbf{z} \in Z^*$ such that $\mathcal{S}(\mathcal{A}(\mathbf{z})) \cap Z' = T$. Since σ is one-to-one, there is $u \subseteq [k]$ such that $T = \{\sigma(k,i) : i \in u\}$. By definition of σ, it holds that for all $i \in [k]$, $\tau(u) \in \sigma(k,i)$ if and only if $\sigma(k,i) \in T$. Hence, to conclude the proof, we need to show that $\tau(u) = \mathcal{A}(z)$ for some $z \in X^*$. This is the case for $z := \mathbb{R} \times [\tau_2(u)), 1]$ where $\tau_2(u)$ is the second component of $\tau(u)$. Hence, the VC dimension of $\mathcal{R}(\mathcal{A})$ is at least k. Since k was arbitrary, this concludes the proof. ■

[4]Such a function is given for instance by $\tau(u) = (\cos(\alpha(u)), \sin(\alpha(u)))$, where $\alpha(u) = \frac{\pi}{2} n(u)/(1 + n(u))$ and $n(u) = \sum_{i \in u} 2^{i-1}$.

[5]E.g., $\xi(3,2) = \{\{2\},\{1,2\},\{2,3\},\{1,2,3\}\}$.

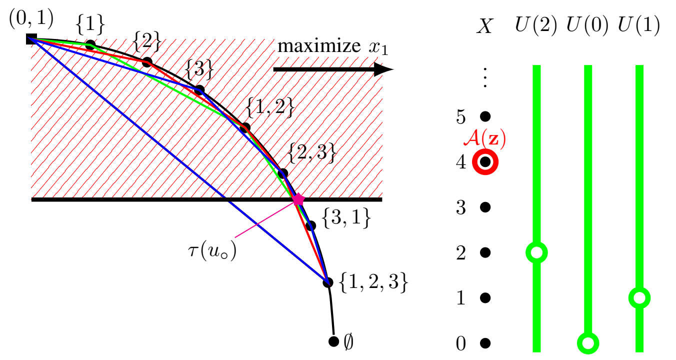

Fig. 1. *Left.* Illustration of the convex scenario program in Section III-A. Black dots: $\tau(u)$ for $u \subseteq [3]$. Colored sets: $\sigma(3,i)$ for $i \in [3]$. Hatched set: the set $\mathbb{R} \times [y, 1]$ where $y = \tau_2(u_\circ)$. *Right.* Illustration of the algorithm in Section III-B, with $\mathbf{z} = (U(2), U(0), U(1))$.

The algorithm $\mathcal{A}$ is a convex scenario optimization program in finite dimension (the dimension of X is finite). Hence, it is stable and PAC by classical results in convex scenario optimization; see, e.g., [20, Theorem 2].

B. PAC Scenario Decision Algorithm without Compression Scheme

To prove (ii) in Theorem 3, we provide an example of PAC scenario decision algorithm $\mathcal{A} : Z^* \to X$ that does not admit a compression scheme.[6] See Figure 1 for an illustration.

Let $X = \mathbb{N}$. For each $a \in \mathbb{N}$, let $U(a) = X \setminus \{a\}$ be the constraint on X requiring that $x \neq a$. Let $Z = \{U(a) : a \in \mathbb{N}\}$. Define the algorithm $\mathcal{A} : Z^* \to X$ as follows: on input $(z_1, \ldots, z_N) \in Z^*$, where for each $i \in [N]$, $z_i = U(a_i)$, let $\mathcal{A}(z_1, \ldots, z_N) = 1 + \sum_{i=1}^N a_i$.

Proposition 2: $\mathcal{A}$ does not admit a compression scheme.

Proof: Let $k \in \mathbb{N}_{>d}$ and $A = \{2^0, 2^1, \ldots, 2^{k-1}\}$. For every $S \subseteq 2^{\mathbb{N}}$, define $\sigma(S) = \sum_{s \in S} s$. Note that for any $S_1 \subseteq A$ and $S_2 \subseteq A$, $S_1 \neq S_2$ implies $\sigma(S_1) \neq \sigma(S_2)$.[7] Let $T = \{U(a) : a \in A\}$. The definition of $\mathcal{A}$ implies that $|\{\mathcal{A}(\mathbf{z}) : \mathbf{z} \in T^{\leq k}\}| \geq |\{\sigma(S) : S \subseteq A\}| = 2^k$.

For a proof by contradiction, let us assume that (κ, ρ) is a compression scheme of capacity d for $\mathcal{A}$. By definition of a compression scheme, it holds that $\{\mathcal{A}(\mathbf{z}) : \mathbf{z} \in T^N\} \subseteq \{\rho(\mathbf{z}) : \mathbf{z} \in T^{\leq d}\}$. It follows that $|\{\mathcal{A}(\mathbf{z}) : \mathbf{z} \in Z^N\}| \leq \sum_{r=0}^d \binom{k}{r} \leq (k+1)^d$. For k large enough, $(k+1)^d < 2^k$. This is a contradiction, concluding the proof. ■

We show that $\mathcal{A}$ is PAC by showing that its range has VC dimension 1:

Proposition 3: The VC dimension of $\mathcal{R}(\mathcal{A})$ is 1.

Proof (sketch): For any $x \in X$ and $Z' \subseteq Z$, $\mathcal{S}(x) \cap Z'$ excludes at most one constraint from Z': namely, $\mathcal{S}(x) \cap Z' = Z' \setminus \{U(x)\}$. This precludes the existence of a shattered set of size > 1. ■

Remark 4: Examples of *set-valued* scenario decision algorithms without a compression scheme, that enjoy *PAC-like properties*, are available in the literature; see, e.g., [17], [22].

[6]In fact, most PAC scenario optimization programs with sample-dependent objective function (e.g., [19]) are expected to satisfy this property. Here, we prove it rigorously for an example that has been crafted for ease of analysis.

[7]Indeed, S_i is the "binary" representation of $\sigma(S_i)$.

However, the definition of PAC is different (cf. Remark 2), so that these examples are not valid examples for our problem.

C. Stable PAC Scenario Decision Algorithm without Compression Map

To prove (iii) in Theorem 3, we provide an example of stable PAC scenario decision algorithm $\mathcal{A} : Z^* \to X$ that does not admit a compression map. The construction is similar to the one in Subsection III-B; hence, most of the details are omitted.

Let $X = \mathbb{N}$. For each $a \in \mathbb{N}$, let $U(a) = X \setminus \{a\}$. Let $Z = \{U(a) : a \in \mathbb{N}\}$. Define the algorithm $\mathcal{A} : Z^* \to X$ as follows: on input $(z_1, \ldots, z_N) \in Z^*$, let $\mathcal{A}(z_1, \ldots, z_N) = \min \bigcap_{i=1}^N z_i$.

Proposition 4: $\mathcal{A}$ does not admit a compression map.

Proof (sketch): Assume that κ is a compression map of capacity d for $\mathcal{A}$. For each $i \in \mathbb{N}_{>0}$, let $z_i = U(i-1)$. Observe that $\mathcal{A}(z_1, \ldots, z_{d+1}) = d$. Let $\kappa(z_1, \ldots, z_{d+1}) = \{i_1, \ldots, i_r\}$ with $r \leq d$. Note that $\mathcal{A}(z_{i_1}, \ldots, z_{i_r}) < d$, a contradiction. ■

It is clear that $\mathcal{A}$ is stable (as it is a nonconvex optimization program). A proof similar to the proof of Proposition 3 shows that $\mathcal{R}(\mathcal{A})$ has finite VC dimension. Hence, $\mathcal{A}$ is PAC.

IV. VC-Inspired PAC Necessary Condition for Scenario Decision Algorithms

In this section, we propose a new PAC *necessary* condition for scenario decision algorithms. This condition is inspired by the VC dimension and the associated *no-free-lunch theorem* [24]. More precisely, we introduce a novel quantity, that can be seen as the "VC dimension of a scenario decision algorithm", which we call *dVC dimension*, and we show that finiteness of this quantity is a PAC necessary condition. We also show with a counterexample that this condition is not a PAC *sufficient* condition.

Definition 9: Consider a scenario decision algorithm $\mathcal{A}$. A subset $Z' \subseteq Z$ is *shattered* by $\mathcal{A}$ if for every $\mathbf{z} \in (Z')^*$, it holds that $\mathcal{S}(\mathcal{A}(\mathbf{z})) \cap Z' = \mathrm{set}(\mathbf{z})$, where $\mathrm{set}(z_1, \ldots, z_N) := \{z_1, \ldots, z_N\}$. The *dVC dimension* of $\mathcal{A}$ is the supremum of all integers k for which there is a subset $Z' \subseteq Z$ of cardinality k that is shattered by $\mathcal{A}$.

Finiteness of the dVC dimension is a PAC necessary condition:

Theorem 4: Consider a scenario decision algorithm $\mathcal{A}$. Assume that $\mathcal{A}$ is PAC. Then, $\mathcal{A}$ has finite dVC dimension.

Proof: For a proof by contraposition, we assume that $\mathcal{A}$ has infinite dVC dimension, and we show that $\mathcal{A}$ is not PAC. Therefore, fix $\epsilon \in (0, \frac{1}{2})$. We show that for all $N \in \mathbb{N}$, there is P such that $\mathsf{P}^N\left[\{\mathbf{z} \in Z^N : V_{\mathsf{P}}(\mathcal{A}(\mathbf{z})) > \epsilon\}\right] = 1$. To show that, fix $N \in \mathbb{N}$, and let $Z' \subseteq Z$ be a finite set shattered by $\mathcal{A}$ with cardinality $|Z'| \geq 2N$. Let P be the discrete probability measure satisfying $\mathsf{P}\left[\{z\}\right] = 1/|Z'|$ for all $z \in Z'$. We show that for all $\mathbf{z} \in (Z')^N$, $V_{\mathsf{P}}(\mathcal{A}(\mathbf{z})) > \epsilon$. Indeed, fix $\mathbf{z} \in (Z')^N$. It holds that $V_{\mathsf{P}}(\mathcal{A}(\mathbf{z})) \geq \frac{1}{2}$ since $\mathcal{S}(\mathcal{A}(\mathbf{z})) \cap Z' = \mathrm{set}(\mathbf{z})$ (since Z' is shattered) and $|Z' \setminus \mathrm{set}(\mathbf{z})| \geq |Z'|/2$. Since N was arbitrary, this shows that $\mathcal{A}$ is not PAC. ■

However, finiteness of the dVC dimension is not a PAC sufficient condition, even for stable scenario decision algorithms. This is shown in the next subsection.

Remark 5: The notion of dVC dimension and Theorem 4 can be extended straightforwardly to set-valued scenario decision algorithms. However, note that for a certain class of set-valued algorithms, namely those that return all decisions that satisfy the sampled constraints (e.g., [17], [22]), finiteness of the VC dimension is already a PAC sufficient and *necessary* condition (consequence of the fundamental theorem of PAC learning [24, Theorem 6.7]).

A. Stable non-PAC Scenario Decision Algorithm with Finite dVC Dimension

We provide an example of stable consistent scenario decision algorithm $\mathcal{A} : Z^* \to X$ that has finite dVC dimension, but is not PAC.

Let $X = 2^{[0,1]}$. For each $a \in [0, 1]$, let $U(a) = \{x \in X : a \in x\}$. Let $Z = \{U(a) : a \in [0, 1]\}$. Define the algorithm $\mathcal{A} : Z^* \to X$ as follows: on input $(z_1, \ldots, z_N) \in Z^*$, where for each $i \in [N]$, $z_i = U(a_i)$, if $0 \in \{a_i\}_{i=1}^N$, let $\mathcal{A}(z_1, \ldots, z_N) = \{a_i\}_{i=1}^N$; otherwise, let $\mathcal{A}(z_1, \ldots, z_N) = (0, 1]$.

Clearly, $\mathcal{A}$ is consistent. It is also not difficult to show that $\mathcal{A}$ is stable. We show that $\mathcal{A}$ has finite dVC dimension:

Proposition 5: $\mathcal{A}$ has dVC dimension at most 2.

Proof: Let $Z' \subseteq Z$ be a finite subset with cardinality at least 3. We show that Z' is not shattered by $\mathcal{A}$. Indeed, if $U(\{0\}) \notin Z'$, then clearly Z' is not shattered by $\mathcal{A}$ since for all $\mathbf{z} \in (Z')^*$, $\mathcal{A}(\mathbf{z}) = (0, 1]$, so that $\mathcal{S}(\mathcal{A}(\mathbf{z})) \cap Z' = Z'$. On the other hand, if $U(\{0\}) \in Z'$, take $a \in (0, 1]$, such that $U(a) \in Z'$, and let $\mathbf{z} = (U(a))$. Then, $\mathcal{A}(\mathbf{z}) = (0, 1]$, so that $\mathcal{S}(\mathcal{A}(\mathbf{z})) \cap Z' \neq \mathrm{set}(\mathbf{z})$ since $|Z'| > 0$. This shows that Z' is not shattered by $\mathcal{A}$, concluding the proof. ■

Proposition 6: $\mathcal{A}$ is not PAC.

Proof: Let $\epsilon = \beta = \frac{1}{4}$. Consider the continuous-discrete probability measure P' on $[0, 1]$ defined by $\mathsf{P}'[\{0\}] = \frac{1}{2}$, and for all $a \in (0, 1]$, $\mathsf{P}'[(0, a]] = a/2$. Consider the associated probability distribution on Z defined by: for all $A \subseteq [0, 1]$, $\mathsf{P}[\{U(a) : a \in A\}] = \mathsf{P}'[A]$. For all $N \in \mathbb{N}_{>0}$, the probability of sampling $\mathbf{z} \in Z^N$ such that $U(\{0\}) \in \mathrm{set}(\mathbf{z})$ is at least $\frac{1}{2}$: $\mathsf{P}[\{\mathbf{z} \in Z^N : U(\{0\}) \in \mathrm{set}(\mathbf{z})\}] = \mathsf{P}'[\{\mathbf{a} \in [0, 1]^N : 0 \in \mathrm{set}(\mathbf{a})] = 1 - (\frac{1}{2})^N \geq \frac{1}{2}$. Whenever $U(\{0\}) \in \mathrm{set}(\mathbf{z})$, the risk of $\mathcal{A}(\mathbf{z})$ is $\frac{1}{2}$, which is larger than ϵ. Hence, we conclude that for all $N \in \mathbb{N}_{>0}$, $\mathsf{P}[\{\mathbf{z} \in Z^N : V_{\mathsf{P}}(\mathcal{A}(\mathbf{z})) > \epsilon\}] \geq \frac{1}{2}$. The latter is larger than β. Thus, $\mathcal{A}$ is not PAC. ■

V. Example of Application of our Results

Consider the problem of shortest path planning from a starting location I to a target location T. See Figure 2 for an illustration. The path has to avoid a *random* obstacle, which takes the form of a barrier positioned at some unknown random angle from the middle point between I and T (see red lines in Figure 2). There is also a fixed known obstacle below I and T, namely the gray area in Figure 2. Positions of the random obstacle can be sampled. We consider two scenario decision algorithms: $\mathcal{A}_1$ that computes the shortest path between I and T, while avoiding the fixed and sampled obstacles (see magenta line in Figure 2); and $\mathcal{A}_2$ that computes the shortest *parabola* between I and T, while avoiding the fixed and sampled obstacles (see cyan line in Figure 2).

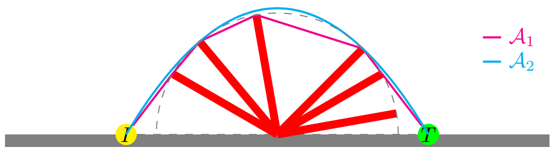

Fig. 2. Shortest path planning (from I to T) in the presence of a randomly positioned obstacle. The red lines represent N sampled positions of the obstacle. The gray area is a fixed, known obstacle.

It can be shown that the dVC dimension of $\mathcal{A}_1$ is infinite. Hence, this algorithm is not PAC (Theorem 4). On the other hand, it can be shown that $\mathcal{A}_2$ admits a compression map of capacity 1 (take any obstacle that touches the optimal parabola). Hence, this algorithm is PAC (Theorem 2).

VI. Conclusions

While PAC learning has been a longstanding major topic in Machine Learning, the development of a similar systematic framework for Optimization and Scenario Decision Making remains elusive. Various PAC sufficient conditions for scenario decision algorithms have been proposed, but the necessity of these conditions remains an open question. This work addresses this gap by providing counterexamples showing that these conditions are not necessary. Inspired by practical algorithms used in real-world applications, these counterexamples have been carefully simplified to isolate their essential features. In addition, we introduce a novel quantity, the dVC dimension, which serves as an analogue to the VC dimension for scenario decision algorithms. We prove that the finiteness of this dimension is a PAC necessary condition for scenario decision algorithms. Beyond its theoretical significance in advancing toward a complete characterization of PAC scenario decision algorithms, this work also offers practical insights for algorithm users and designers. It enables them to identify which PAC sufficient conditions are better suited for analyzing their algorithms—highlighting that some conditions may not hold while others do—, or to determine that their algorithms are not PAC by leveraging necessary conditions.

Looking ahead, we aim to address several open questions. First, we will investigate the existence of compression schemes for stable PAC scenario decision algorithms. Second, we will explore new sufficient and necessary PAC conditions, with the goal of narrowing the gap between the two and moving closer to a comprehensive characterization of PAC scenario decision algorithms.

References

[1] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.
[2] M. C. Campi and S. Garatti, "The exact feasibility of randomized solutions of uncertain convex programs," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1211–1230, 2008.
[3] G. C. Calafiore, "Random convex programs," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 3427–3464, 2010.
[4] M. C. Campi and S. Garatti, "A sampling-and-discarding approach to chance-constrained optimization: feasibility and optimality," *Journal of Optimization Theory and Applications*, vol. 148, no. 2, pp. 257–280, 2011.
[5] K. Margellos, P. Goulart, and J. Lygeros, "On the road between robust optimization and the scenario approach for chance constrained optimization problems," *IEEE Transactions on Automatic Control*, vol. 59, no. 8, pp. 2258–2263, 2014.
[6] P. M. Esfahani, T. Sutter, and J. Lygeros, "Performance bounds for the scenario approach and an extension to a class of non-convex programs," *IEEE Transactions on Automatic Control*, vol. 60, no. 1, pp. 46–58, 2015.
[7] S. Grammatico, X. Zhang, K. Margellos, P. Goulart, and J. Lygeros, "A scenario approach for non-convex control design," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 334–345, 2016.
[8] M. C. Campi and S. Garatti, "Wait-and-judge scenario optimization," *Mathematical Programming*, vol. 67, no. 1, pp. 155–189, 2018.
[9] M. C. Campi, S. Garatti, and F. A. Ramponi, "A general scenario theory for nonconvex optimization and decision making," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4067–4078, 2018.
[10] Y. Yang and C. Sutanto, "Chance-constrained optimization for nonconvex programs using scenario-based methods," *ISA transactions*, vol. 90, pp. 157–168, 2019.
[11] R. Rocchetta and L. G. Crespo, "A scenario optimization approach to reliability-based and risk-based design: soft-constrained modulation of failure probability bounds," *Reliability Engineering & System Safety*, vol. 216, no. 107900, 2021.
[12] L. Romao, K. Margellos, and A. Papachristodoulou, "Tight sampling and discarding bounds for scenario programs with an arbitrary number of removed samples," in *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, A. Jadbabaie, J. Lygeros, G. J. Pappas, P. Parrilo, B. Recht, C. J. Tomlin, and M. N. Zeilinger, Eds., vol. 144, 2021, pp. 312–323.
[13] S. Garatti and M. C. Campi, "Risk and complexity in scenario optimization," *Mathematical Programming*, vol. 191, pp. 243–279, 2022.
[14] L. Romao, A. Papachristodoulou, and K. Margellos, "On the exact feasibility of convex scenario programs with discarded constraints," *IEEE Transactions on Automatic Control*, vol. 68, no. 4, pp. 1986–2001, 2023.
[15] M. C. Campi and S. Garatti, "Compression, generalization and learning," *Journal of Machine Learning Research*, vol. 24, no. 339, pp. 1–74, 2023.
[16] R. Rocchetta, A. Mey, and F. A. Oliehoek, "A survey on scenario theory, complexity, and compression-based learning and generalization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 12, pp. 16 985–16 999, 2024.
[17] D. P. De Farias and B. Van Roy, "On constraint sampling in the linear programming approach to approximate dynamic programming," *Mathematics of Operations Research*, vol. 29, no. 3, pp. 462–478, 2004.
[18] T. Alamo, R. Tempo, and E. F. Camacho, "Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2545–2559, 2009.
[19] F. Lauer, "Margin-based scenario approach to robust optimization in high dimension," *IEEE Transactions on Automatic Control*, vol. 69, no. 10, pp. 7182–7189, 2024.
[20] K. Margellos, M. Prandini, and J. Lygeros, "On the connection between compression learning and scenario based single-stage and cascading optimization problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2716–2721, 2015.
[21] N. Littlestone and M. Warmuth, "Relating data compression and learnability," 1986, unpublished manuscript.
[22] R. El-Yaniv and Y. Wiener, "On the version space compression set size and its applications," in *Measures of Complexity*, V. Vovk, H. Papadopoulos, and A. Gammerman, Eds. Cham: Springer, 2015, pp. 341–357.
[23] S. Garatti and M. C. Campi, "The risk of making decisions from data through the lens of the scenario approach," *IFAC-PapersOnLine*, vol. 54, no. 7, pp. 607–612, 2021.
[24] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: from theory to algorithms*. Cambridge, UK: Cambridge University Press, 2014.
[25] V. N. Vapnik and A. Y. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability & Its Applications*, vol. 16, no. 2, pp. 264–280, 1971.
[26] S. Moran and A. Yehudayoff, "Sample compression schemes for VC classes," *Journal of the ACM*, vol. 63, no. 3, pp. 1–10, 2016.
[27] S. Hanneke, "The optimal sample complexity of PAC learning," *Journal of Machine Learning Research*, vol. 17, no. 38, pp. 1–15, 2016.
[28] N. Grelier, S. G. Ilchi, T. Miltzow, and S. Smorodinsky, "On the VC-dimension of half-spaces with respect to convex sets," *Discrete Mathematics & Theoretical Computer Science*, vol. 23, no. 3, 2021.