Proceedings of the Thirty-Sixth International Conference on Automated Planning and Scheduling (ICAPS 2026)

Learning Numeric Action Models with Anytime Guarantees

Diego Aineto[1], **Enrico Scala**[2]

[1]Universitat Politècnica de València
[2]Università degli Studi di Brescia
dieaigar@vrain.upv.es, enrico.scala@unibs.it

Abstract

Model-based planning requires a predictive model of the world, yet such models are often hand-crafted. We address this by presenting a framework that learns PDDL action models with numeric fluents automatically from demonstrations. Our approach compactly represents the infinite space of linear numeric preconditions and effects consistent with the data, learning each action modularly and exploiting simpler structures when present. The framework provides anytime soundness and completeness guarantees with respect to the true hidden model, yielding conservative under-approximations and optimistic over-approximations at every stage of learning. Our findings across standard benchmark domains shows the advantages of our framework over existing approaches.

Introduction

Automated planning studies how to equip an agent with the ability to select and execute actions so as to achieve a goal-directed behavior (Ghallab, Nau, and Traverso 2004). A dominant approach is *model-based* planning: the agent relies on a predictive model of the world, expressed in a formal language, to generate a sequence of actions that achieves a user-specified goal. This naturally raises a fundamental question: how can such models be acquired automatically?

Action model learning addresses this problem. In this paper we study it in the setting of models represented in the PDDL language (Haslum et al. 2019), focusing on the fragment where actions constrain and update *numeric* state variables. Numeric fluents are essential for capturing real-world phenomena—including resources, timing, geometry, and continuous dynamics—and underpin many practical planning applications (Vallati et al. 2016; Kiam et al. 2020; Alon et al. 2024; Aineto et al. 2023).

While extensive prior work has addressed propositional domains (Yang, Wu, and Jiang 2007; Amir and Chang 2008; Zhuo et al. 2010; Cresswell, McCluskey, and West 2013; Mourão et al. 2012; Zhuo and Kambhampati 2013; Aineto, Celorrio, and Onaindia 2019; Bonet and Geffner 2020; Verma, Marpally, and Srivastava 2021; Xi, Gould, and Thiébaux 2024; Bachor and Behnke 2024; Gösgens, Jansen, and Geffner 2025; Lamanna et al. 2025), learning *numeric* action models remains under-explored. PlanMiner (Segura-Muros, Pérez, and Fernández-Olivares 2021) learns numeric action models from partially observed trajectories and noisy data, but does not provide any guarantees. N-SAM (Mordoch, Juba, and Stern 2023) extends the Safe Action Model learning framework (Stern and Juba 2017; Juba and Stern 2022) to linear and polynomial numeric domains, ensuring soundness but not completeness. To the best of our knowledge, these are the only approaches that directly target the learning of numeric action models, and none provide a principled way to derive models having *both* soundness and completeness guarantees w.r.t. the *true* hidden model.

Our contribution. We present NASCAL (*Numeric Anytime Sound and Complete Action Learning*), the first framework for learning numeric action models with anytime soundness and completeness guarantees. Rather than learning a single model, NASCAL maintains a compact representation of all (potentially infinitely many) models consistent with the observed data, learning each action's numeric preconditions and effects independently. From this representation, we extract two approximations of the unknown true model: (1) a **sound** model, a conservative under-approximation that allows only transitions guaranteed to be valid; and (2) a **complete** model, an optimistic over-approximation that admits every transition that cannot yet be ruled out. Together, these approximations provide strong **anytime guarantees**: after any number of demonstrations, the agent can act safely trusting the sound model while exploring intelligently using the complete one. Thus, NASCAL supports deployment *during* learning, not only after learning converges.

Theoretically, we establish the correctness of the resulting sound and complete models. Empirically, we evaluate NASCAL on standard numeric planning benchmarks, showing that both models provide strong predictive and planning performance early in the learning process, and offer, in our setting, advantages over existing methods for learning numeric action models.

Preliminaries

Linear Numeric Planning

We adopt *Linear Numeric Planning* (LNP) models as the target representation for learning.

Syntax. An LNP model is a tuple $M = \langle F, X, A, \mathsf{pre}, \mathsf{eff} \rangle$, where $F = \{f_1, \ldots, f_n\}$ and $X = \{x_1, \ldots, x_m\}$ are propositional and numeric variables, respectively, and A is the set of actions. A state s is a full assignment from F to the Booleans $\mathbb{B} = \{\top, \bot\}$ and from X to the rationals $\mathbb{Q}$, giving rise to the state space $S = \mathbb{B}^n \times \mathbb{Q}^m$. For any variable $v \in F \cup X$ we write $s[v]$ for its value in s.

Propositional conditions have the form $f = b$ and propositional assignments have the form $f := b$, where $f \in F$ and $b \in \mathbb{B}$. A *numeric expression* is a pair $\xi = (\boldsymbol{w}, b) \in \mathbb{Q}^m \times \mathbb{Q}$, representing the linear expression $\boldsymbol{w} \cdot \boldsymbol{x} + b$. Its evaluation in state s is $s[\xi] = \boldsymbol{w} \cdot s[\boldsymbol{x}] + b$, where $s[\boldsymbol{x}] = [\, s[x_1]; \ldots; s[x_m] \,]$. Numeric conditions are inequalities $\xi \geq 0$, and numeric assignments are updates $x := \xi$. For convenience, we denote the sets of linear expressions, conditions, and assignments by $\mathit{Lin} = \mathbb{Q}^m \times \mathbb{Q}$, $\mathit{Cnd} = (F \times \mathbb{B}) \cup \mathit{Lin}$, and $\mathit{Asg} = (F \times \mathbb{B}) \cup (X \times \mathit{Lin})$.

The relations $\mathsf{pre} \subseteq A \times \mathit{Cnd}$ and $\mathsf{eff} \subseteq A \times \mathit{Asg}$ map each action a to its *precondition* $\mathsf{pre}(a) = \{cnd : (a, cnd) \in \mathsf{pre}\}$ and *effect* $\mathsf{eff}(a) = \{asg : (a, asg) \in \mathsf{eff}\}$. We assume actions are *well-formed*: $\mathsf{eff}(a)$ contains at most one assignment per variable.

Semantics. A state s satisfies the precondition of action a, written $s \models \mathsf{pre}(a)$, iff (i) $s[f] = b$ for all propositional conditions $f = b$ in $\mathsf{pre}(a)$ and (ii) $s[\xi] \geq 0$ for all numeric conditions $\xi \geq 0$ in $\mathsf{pre}(a)$.

If $s \models \mathsf{pre}(a)$, action a is executable in s yielding the state $s' = \mathit{suc}(s, \mathsf{eff}(a))$, where $s'[f] = b$ if $(f, b) \in \mathsf{eff}(a)$ and $s'[f] = s[f]$ otherwise; similarly, $s'[x] = s[\xi]$ if $(x, \xi) \in \mathsf{eff}(a)$ and $s'[x] = s[x]$ otherwise.

Thus an LNP model $M = \langle F, X, A, \mathsf{pre}, \mathsf{eff} \rangle$ induces the transition system

$$T_M = \{\, (s, a, s') : s \models \mathsf{pre}(a) \ \wedge \ s' = \mathit{suc}(s, \mathsf{eff}(a)) \,\}.$$

Planning tasks. An LNP task is $\langle M, s_0, G \rangle$, where M is an LNP model, $s_0 \in S$ is the initial state, and $G \subseteq \mathit{Cnd}$ is the goal condition. A plan $\pi = (a_1, \ldots, a_k)$ provides a solution iff $(s_{i-1}, a_i, s_i) \in T_M$ for all $i = 1, \ldots, k$, and $s_k \models G$. We write $\Pi(\langle M, s_0, G \rangle)$ for the set of solution plans.

Action Model Learning

Action model learning is the task of automatically acquiring an agent's action model from *demonstrations* of its behavior. Demonstrations are collected from executions under the (hidden) true model M^*, such as observed plans or random walks. We distinguish between *positive demonstrations*, corresponding to transitions in T_{M^*}, and *negative demonstrations*, which represent failed action executions. A negative demonstration is indicated by a special post-state $\bot$, denoting that action a cannot be executed in state s.

Definition 1 (Demonstration)**.** *A demonstration is a triple $d = \langle s, a, s' \rangle$, consisting of a pre-state $s \in S$, an action $a \in A$, and a post-state $s' \in S \cup \{\bot\}$.*

Definition 2 (Consistency)**.** *An action model M is* consistent *with a positive demonstration $d = \langle s, a, s' \rangle$ if $(s, a, s') \in T_M$, and with a negative demonstration $d = \langle s, a, \bot \rangle$ if $(s, a, s'') \notin T_M$ for all $s'' \in S$.*

Let $\mathcal{M}$ be a hypothesis space containing at least one model equivalent to M^* (i.e., one with $T_M = T_{M^*}$). Given demonstrations D, let $\mathcal{M}_D \subseteq \mathcal{M}$ denote the models that are consistent with all demonstrations. In most formulations of action model learning, the solution to the learning task is required only to be consistent with D (Aineto, Jiménez, and Onaindia 2022). However, consistency alone provides no guarantees about the behavior of the learned model outside the demonstrated data. A model may be perfectly consistent with D yet still commit either type of error: it may allow transitions that are impossible in the true model (false positives) or forbid transitions that are actually valid (false negatives). Thus, one cannot predict how the learned model will behave with respect to the true model.

Sound and Complete Learned Models

To support planning with formal assurances while M^* remains unknown, we consider learned models that are sound or complete with respect to the true model.

Definition 3 (Soundness and Completeness)**.** *Let M and M' be two action models. M is* sound *with respect to M' iff $T_M \subseteq T_{M'}$, and* complete *with respect to M' iff $T_M \supseteq T_{M'}$.*

Soundness guarantees that every transition—and therefore every plan—admitted by M is truly possible under M'. Completeness guarantees that M preserves every transition that is genuinely possible under M', ensuring it never declares a feasible plan impossible.

Theorem 1 (Aineto and Scala (2024))**.** *Let M^* be the true model. If M is sound w.r.t. M^* and M' is complete w.r.t. M^*, then for any initial state s_0 and goal condition G:*

- $\Pi(\langle M, s_0, G \rangle) \subseteq \Pi(\langle M^*, s_0, G \rangle)$;
- $\Pi(\langle M', s_0, G \rangle) \supseteq \Pi(\langle M^*, s_0, G \rangle)$.

Thus, planning with a **sound** learned model is conservative but safe, while planning with a **complete** learned model is optimistic, ensuring no truly executable plan is rejected.

Learning Numeric Action Models with Anytime Guarantees

Our framework, NASCAL, provides learned numeric action models that *always* satisfy one of these two properties. Specifically, NASCAL enforces *anytime guarantees*: after any number of demonstrations, the learner must output both a sound and a complete approximation of the true model M^*. We build on the theoretical framework of Aineto and Scala (2024), which shows how sound and complete models can be derived from $\mathcal{M}_D$, the set of all action models consistent with the demonstrations. Our contribution extends this foundational work, originally restricted to propositional domains, to the far richer setting of LNP models.

Problem Statement

We consider the action model learning task $\langle \mathcal{M}, D \rangle$, specialized here to LNP models. Demonstrations arrive online as a sequence $D = \langle d_1, d_2, \ldots \rangle$, where each $d_i = \langle s, a, s' \rangle$ is either a positive or negative demonstration.

Inputs. The learner receives the demonstration sequence D together with the parameters defining the hypothesis space $\mathcal{M}$. Specifically:

- The propositional variables F, numeric variables X, and action set A.
- For each action $a \in A$, a matrix $\boldsymbol{W}^a = [\boldsymbol{w}_1; \ldots; \boldsymbol{w}_t] \in \mathbb{R}^{t \times m}$, whose rows specify the *admissible orientations* of inequalities.

Thus, the candidate models in $\mathcal{M}$ are all LNP action models over (F, X, A) whose numeric preconditions respect these orientation constraints: each inequality in $\mathsf{pre}(a)$ has the form $\boldsymbol{w} \cdot \boldsymbol{x} + b \geq 0$ where $\boldsymbol{w}$ is a row of $\boldsymbol{W}^a$ and $b \in \mathbb{R}$. Hence, $\boldsymbol{W}^a$ fixes the orientations, while the learner selects the offsets b.

These orientation matrices encode prior knowledge about the orientations of the true preconditions and must include them to ensure correctness. When they are unknown, one can instead start from coarse supersets of candidate orientations, from which redundant inequalities are eliminated during learning. In the limit, including all such orientations removes the bias entirely, so NASCAL effectively learns over the full linear numeric hypothesis space. Thus, the orientation matrices enable interpolation between full knowledge (exact orientations) and weak prior knowledge (large candidate sets), allowing NASCAL to trade prior structure for convergence speed while retaining formal guarantees.

Solution. After observing any prefix $D_t = \{d_1, \ldots, d_t\}$ of demonstrations, the learner must return:

- a *sound* model M_S^t such that $T_{M_S^t} \subseteq T_{M^*}$, and
- a *complete* model M_C^t such that $T_{M_C^t} \supseteq T_{M^*}$,

These guarantees would be trivial without a notion of optimality. For instance, the empty transition system is always sound, but useless; likewise, the model admitting all transitions is always complete, but uninformative. Following Aineto and Scala (2024), we therefore require M_S^t and M_C^t to be the *tightest* sound and complete approximations modulo the demonstrations. Formally:

$$T_{M_S^t} = \bigcap_{M' \in \mathcal{M}_{D_t}} T_{M'}, \tag{1}$$

$$T_{M_C^t} = \bigcup_{M' \in \mathcal{M}_{D_t}} T_{M'}. \tag{2}$$

That is, M_S^t includes exactly those transitions common to *all* consistent models, and M_C^t includes every transition that appears in *some* consistent model. These are the strongest sound and complete models given the available data.

Approach Overview

Building on these theoretical results, our approach provides an anytime method for computing the optimal sound and complete approximations for LNP domains. The method proceeds in two steps:

- **Computation of all consistent models.** We introduce a compact representation of $\mathcal{M}_{D_t}$ and show how it can be updated online as new demonstrations arrive.
- **Derivation of sound and complete models.** We show how to extract the corresponding optimal models M_S^t and M_C^t that satisfy the anytime guarantees.

The following sections develop these two steps in detail.

Computation of All Consistent Models

We decompose the computation of $\mathcal{M}_{D_t}$ into simpler subproblems. A key observation is that demonstrations constrain the behavior of each action independently: a demonstration $\langle s, a, s' \rangle$ provides information only about a. Moreover, within a single action, *preconditions* and *effects* play disjoint semantic roles—the former determine when a may be executed, while the latter determine its successor state. This separation motivates learning preconditions and effects independently for each action.

Definition 4 (Precondition consistency)**.** *A precondition $p \subseteq Cnd$ is consistent with a positive demonstration $\langle s, a, s' \rangle$ iff $s \models p$, and with a negative demonstration $\langle s, a, \bot \rangle$ iff $s \not\models p$.*

Definition 5 (Effect consistency)**.** *An effect $e \subseteq Asg$ is consistent with a positive demonstration $\langle s, a, s' \rangle$ iff $suc(s, e) = s'$. Negative demonstrations do not constraint effects.*

For each action $a \in A$, let D_t^a be the demonstrations in D_t whose action label is a. We define the *consistent precondition space* and *consistent effect space* of a as:

$$\mathsf{Pre}_{D_t}^a = \left\{ p \subseteq Cnd \;\middle|\; p \text{ consistent with every } d \in D_t^a \right\},$$

$$\mathsf{Eff}_{D_t}^a = \{ e \subseteq Asg \mid e \text{ consistent with all positive } d \in D_t^a \}.$$

Proposition 2 (Decomposition of model consistency)**.** *A model $M = \langle F, X, A, \mathsf{pre}, \mathsf{eff} \rangle$ is consistent with D_t iff for every $a \in A$,*

$$\mathsf{pre}(a) \in \mathsf{Pre}_{D_t}^a \qquad \textit{and} \qquad \mathsf{eff}(a) \in \mathsf{Eff}_{D_t}^a.$$

Thus computing $\mathcal{M}_{D_t}$ reduces to computing $\mathsf{Pre}_{D_t}^a$ and $\mathsf{Eff}_{D_t}^a$ for each a independently. Finally, within both preconditions and effects, the propositional and numeric components are independent: propositional conditions and assignments affect only Boolean fluents, while numeric ones affect numeric variables. Since the propositional part can be handled by existing techniques (Aineto and Scala 2024), the remainder of this work focuses on the numeric components.

Learning Numeric Preconditions

We now describe how to learn the *consistent precondition space* $\mathsf{Pre}_{D_t}^a$ for an action $a \in A$, i.e., the set of all numeric preconditions consistent with the demonstrations observed up to time t. Each hypothesis is represented in matrix form $(\boldsymbol{W}^a, \boldsymbol{b})$, where $\boldsymbol{W}^a = [\boldsymbol{w}_1; \ldots; \boldsymbol{w}_t]$ collects the admissible orientation vectors supplied in the input, and $\boldsymbol{b} \in \mathbb{R}^t$ contains the unknown offsets of the corresponding inequalities. Geometrically, each hypothesis $(\boldsymbol{W}^a, \boldsymbol{b})$ defines a (closed) convex polytope (the higher-dimensional analogue of a polygon):

$$P_{\boldsymbol{W}^a, \boldsymbol{b}} = \{ s \in S : \boldsymbol{W}^a \, s[\boldsymbol{x}] + \boldsymbol{b} \geq 0 \},$$

i.e., the convex set of states that satisfy all inequalities.

A hypothesis $(\boldsymbol{W}^a, \boldsymbol{b})$ permits execution in s exactly when $s[\boldsymbol{x}] \in P_{\boldsymbol{W}^a, \boldsymbol{b}}$. Therefore: (i) a positive demonstration $\langle s, a, s' \rangle$ requires $s[\boldsymbol{x}] \in P_{\boldsymbol{W}^a, \boldsymbol{b}}$; (ii) a negative demonstration $\langle s, a, \bot \rangle$ requires $s[\boldsymbol{x}] \notin P_{\boldsymbol{W}^a, \boldsymbol{b}}$.

Hypothesis space and boundary representation. We restrict attention to hypotheses compatible with the admissible orientations $\boldsymbol{W}^a$ supplied as input. The *hypothesis space of preconditions* for action a is therefore

$$\mathsf{Pre}^a = \{(\boldsymbol{W}^a, \boldsymbol{b}) : \boldsymbol{b} \in \mathbb{R}^t\},$$

i.e., all choices of offsets for the admissible orientation vectors $\boldsymbol{W}^a$. This space is ordered by set inclusion:

$$(\boldsymbol{W}^a, \boldsymbol{b}) \leq (\boldsymbol{W}^a, \boldsymbol{b}') \iff P_{\boldsymbol{W}^a, \boldsymbol{b}} \subseteq P_{\boldsymbol{W}^a, \boldsymbol{b}'},$$

which holds exactly when $\boldsymbol{b} \leq \boldsymbol{b}'$ elementwise.

Our goal is to compute $\mathsf{Pre}^a_{D_t} \subseteq \mathsf{Pre}^a$, i.e., the hypotheses consistent with all demonstrations in D^a_t. By exploiting the fact that $\mathsf{Pre}^a_{D_t}$ is partially ordered, we obtain a compact representation that remains finite throughout learning. This representation consists of the *lower* and *upper* boundaries, denoted $\mathcal{L}^a_{D_t}$ and $\mathcal{U}^a_{D_t}$, respectively. Here, $\mathcal{L}^a_{D_t}$ contains the minimal elements of $\mathsf{Pre}^a_{D_t}$ and $\mathcal{U}^a_{D_t}$ its maximal ones (w.r.t. $\leq$). A hypothesis $(\boldsymbol{W}^a, \boldsymbol{b})$ belongs to $\mathsf{Pre}^a_{D_t}$ iff there exist $(\boldsymbol{W}^a, \boldsymbol{b}') \in \mathcal{L}^a_{D_t}$ and $(\boldsymbol{W}^a, \boldsymbol{b}'') \in \mathcal{U}^a_{D_t}$ such that $\boldsymbol{b}' \leq \boldsymbol{b} \leq \boldsymbol{b}''$.

Initially, both boundaries contain a single hypothesis (the most restrictive and the most permissive one) and they remain finite throughout learning. The update rules presented below maintain $\mathcal{L}^a_{D_t}$ as a singleton, while $\mathcal{U}^a_{D_t}$ may grow but only ever contains finitely many maximal elements. Thus, the pair $(\mathcal{L}^a_{D_t}, \mathcal{U}^a_{D_t})$ provides a finite and explicit description of the potentially infinite set $\mathsf{Pre}^a_{D_t}$.

Initialization. Before observing any demonstration, all hypotheses are consistent. The boundary sets are therefore initialized to the extreme hypotheses:

$$\mathcal{L} = \{(\boldsymbol{W}^a, -\boldsymbol{\infty})\}, \qquad \mathcal{U} = \{(\boldsymbol{W}^a, +\boldsymbol{\infty})\},$$

where $\pm\boldsymbol{\infty}$ denote t-dimensional vectors of $\pm\infty$. The lower element corresponds to the empty polytope (no state satisfies it), and the upper to $\mathbb{R}^m$ (all states satisfy it). Together, these bounds represent the entire hypothesis space Pre^a.

Updating with positive demonstrations. Intuitively, a positive demonstration $\langle s, a, s' \rangle$ requires the lower bound to *expand just enough* to accept the observed pre-state s. We capture this with a *generalization operator* that minimally adjusts the offsets.

Definition 6 (Generalization)**.** *For $(\boldsymbol{W}^a, \boldsymbol{b}) \in \mathsf{Pre}^a$ and positive demonstration $\langle s, a, s' \rangle$, define $gen(\boldsymbol{W}^a, \boldsymbol{b}, s) = (\boldsymbol{W}^a, \boldsymbol{b}')$ where*

$$b'_i = \begin{cases} b_i, & \text{if } \boldsymbol{w}_i \cdot s[\boldsymbol{x}] + b_i \geq 0, \\ -\boldsymbol{w}_i \cdot s[\boldsymbol{x}], & \text{otherwise.} \end{cases}$$

This yields the least hypothesis that satisfies $s[\boldsymbol{x}] \in P_{\boldsymbol{W}^a, \boldsymbol{b}'}$.

Definition 7 (Positive-update rules)**.** *Let $(\mathcal{L}, \mathcal{U})$ be the boundaries of $\mathsf{Pre}^a_{D_t}$. After observing a positive demonstration $\langle s, a, s' \rangle$, the boundaries are updated as follows:*

R1 *Generalize the lower bounds:*

$$\mathcal{L} \leftarrow \{\, gen(\boldsymbol{W}^a, \boldsymbol{b}, s) : (\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{L} \,\};$$

R2 *Remove upper bounds that reject the demonstration:*

$$\mathcal{U} \leftarrow \{(\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{U} : s[\boldsymbol{x}] \in P_{\boldsymbol{W}^a, \boldsymbol{b}}\}.$$

Updating with negative demonstrations. Intuitively, a negative demonstration $\langle s, a, \bot \rangle$ requires upper bounds to *contract just enough* to reject the pre-state s. Rejection can be enforced by tightening at least one inequality so that it becomes violated at s. We therefore define a *specialization operator* that minimally tightens a chosen constraint.

Definition 8 (Specialization)**.** *For $(\boldsymbol{W}^a, \boldsymbol{b}) \in \mathsf{Pre}^a$, negative demonstration $\langle s, a, \bot \rangle$, and index $j \in [t]$, define $spe(\boldsymbol{W}^a, \boldsymbol{b}, s, j) = (\boldsymbol{W}^a, \boldsymbol{b}')$, where*

$$b'_i = \begin{cases} -\boldsymbol{w}_j \cdot s[\boldsymbol{x}], & \text{if } i = j \text{ and } \boldsymbol{w}_j \cdot s[\boldsymbol{x}] + b_j \geq 0, \\ b_i, & \text{otherwise,} \end{cases}$$

This is the greatest hypothesis obtained from $(\boldsymbol{W}^a, \boldsymbol{b})$ by tightening constraint j just enough to reject $s[\boldsymbol{x}]$.

Remark. After specialization, the updated offset satisfies $b'_j = -\boldsymbol{w}_j \cdot s[\boldsymbol{x}]$, so $\boldsymbol{w}_j \cdot s[\boldsymbol{x}] + b'_j = 0$. Thus the specialized hypothesis places s exactly on the boundary of the polytope. This is intentional. Upper bounds represent the *outer surface* of all consistent hypotheses, and by tightening constraint j to equality we bring the bound to the limiting position where exclusion begins. This does not affect completeness but can nevertheless be refined with a *strict* constraint $\boldsymbol{w}_j \cdot \boldsymbol{x} + b'_j > 0$ rather than ≥ 0, effectively *peeling off the skin of the polytope*: the updated bound marks the exact boundary, while every strictly smaller hypothesis lying just inside it excludes the negative example.

Definition 9 (Negative-update rules)**.** *Let $(\mathcal{L}, \mathcal{U})$ be the boundaries of $\mathsf{Pre}^a_{D_t}$. After observing a negative demonstration $\langle s, a, \bot \rangle$, the boundaries are updated as follows:*

R3 *Specialize the upper bounds:*

$$\mathcal{U} \leftarrow \max\big\{ spe(\boldsymbol{W}^a, \boldsymbol{b}, s, j) : \ (\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{U}, \ j \in [t] \big\};$$

R4 *Remove lower bounds that accept the demonstration:*

$$\mathcal{L} \leftarrow \big\{ (\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{L} : s[\boldsymbol{x}] \notin P_{\boldsymbol{W}^a, \boldsymbol{b}} \big\}.$$

Note that **R3** may produce multiple maximal hypotheses, enlarging the upper boundary, since the elements of $\mathcal{U}$ are partially ordered. In contrast, the lower boundary remains a singleton, as no rule enlarges it. **R4** is included merely for completeness: if triggered, it would collapse the consistent space, signalling noise, which we do not assume.

Theorem 3 (Correctness)**.** *Let $(\mathcal{L}, \mathcal{U})$ be the boundaries of $\mathsf{Pre}^a_{D_t}$, d the demonstration at time $t+1$, and $(\mathcal{L}', \mathcal{U}')$ be the boundaries obtained by applying **R1**–**R4**. Then $(\mathcal{L}', \mathcal{U}')$ represent exactly the updated space $\mathsf{Pre}^a_{D_{t+1}}$.*

Proof sketch. We treat positive and negative demonstrations separately.

Positive demonstration. Let $d = \langle s, a, s' \rangle$. To accept s, bounds must be moved to a more general hypothesis, i.e., by increasing $\boldsymbol{b}$. For each lower bound $(\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{L}$, **R1** replaces $\boldsymbol{b}$ by the *least* vector $\boldsymbol{b}' \geq \boldsymbol{b}$ such that $\boldsymbol{W}^a s[\boldsymbol{x}] + \boldsymbol{b}' \geq \boldsymbol{0}$. **R2** removes upper bounds that reject s: by maximality, no strictly larger consistent hypothesis exists.

Negative demonstration. Let $d = \langle s, a, \bot \rangle$. To reject s, bounds must be moved to a more specific hypothesis,

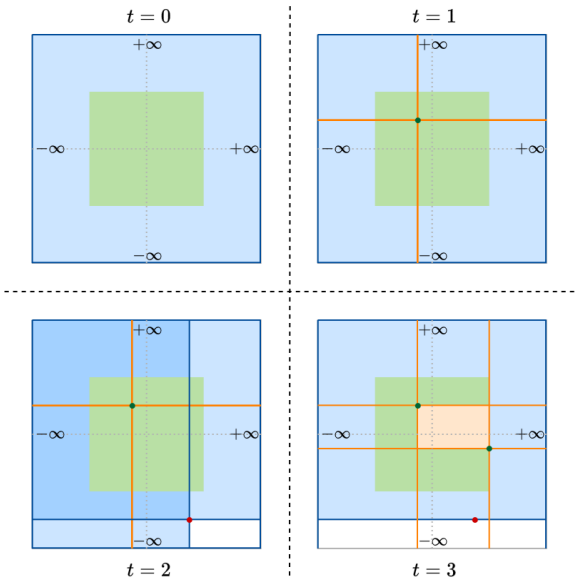

Figure 1: Evolution of lower (orange) and upper (blue) boundaries w.r.t the true precondition (green).

i.e., by decreasing some component of $\boldsymbol{b}$. Rule **R3** applies $spe(\boldsymbol{W}^a, \boldsymbol{b}, s, j)$, which decreases b_j just enough to make the j-th inequality exclude s, producing all minimally specialized consistent hypotheses below each $(\boldsymbol{W}^a, \boldsymbol{b}) \in \mathcal{U}$. Among these, we keep only the maximal ones. **R4** removes any lower bound that accepts s: by minimality, no strictly smaller consistent hypothesis exists.

In both cases, the updated boundaries $(\mathcal{L}', \mathcal{U}')$ contain exactly the minimal and maximal hypotheses consistent with D_{t+1}, and therefore represent $\mathsf{Pre}^a_{D_{t+1}}$. □

Illustrative example. Figure 1 illustrates the evolution of $\mathcal{L}^a_{D_t}$ and $\mathcal{U}^a_{D_t}$ for $t = 0, \ldots, 3$ when learning a precondition of the form $(\pm x + c \geq 0) \wedge (\pm y + c \geq 0)$ (i.e., an axis-aligned box). Hypotheses in $\mathcal{L}^a_{D_t}$ are shown in orange, those in $\mathcal{U}^a_{D_t}$ in blue, and the true precondition in green. Positive demonstrations appear as green points, and negative ones as red points. At $t = 0$, the lower bound $\mathcal{L}^a_{D_t}$ accepts no state, while $\mathcal{U}^a_{D_t}$ accepts all states. At $t = 1$, a positive demonstration arrives; rule **R1** generalizes $\mathcal{L}^a_{D_t}$ just enough to accept its pre-state (a single point). At $t = 2$, a negative demonstration (red point) is received; rule **R3** produces two maximal specializations of $\mathcal{U}^a_{D_t}$ that reject it. In the Figure, the darker blue area is the intersection of the two lighter regions. Finally, at $t = 3$, another positive demonstration arrives: rule **R1** generalizes $\mathcal{L}^a_{D_t}$ to accept the smallest axis-aligned box containing both green points, while rule **R2** removes an inconsistent upper bound. Throughout the sequence, $\mathcal{L}^a_{D_t}$ always under-approximates the true precondition and $\mathcal{U}^a_{D_t}$ always over-approximates it, forming progressively tighter inner and outer envelopes.

Learning Numeric Effects

We now describe how to learn the *consistent effect space* $\mathsf{Eff}^a_{D_t}$ for an action $a \in A$, i.e., the set of all numeric effects consistent with the positive demonstrations observed up to time t. A positive demonstration $\langle s, a, s' \rangle$ reveals how the numeric state changes when a is applied, relating $s[\boldsymbol{x}]$ to $s'[\boldsymbol{x}]$. In LNP models, each variable $x \in X$ is updated by a linear function of the pre-state:

$$s'[x] = \boldsymbol{w}_x \cdot s[\boldsymbol{x}] + b_x, \tag{3}$$

where $\boldsymbol{w}_x \in \mathbb{Q}^m$ and $b_x \in \mathbb{Q}$ are unknown coefficients.

Thus, learning effects reduces to identifying, for each $x \in X$, the coefficients-bias pair $(\boldsymbol{w}_x, b_x)$ that satisfy the constraints imposed by the observed transitions. Each positive demonstration contributes a linear equation relating $s[\boldsymbol{x}]$ and $s'[x]$, and the set $\mathsf{Eff}^a_{D_t}$ can therefore be characterized by the solution space of the corresponding linear systems.

Linear systems from demonstrations. Let $D^{a,+}_t = \{(s_i, a, s'_i)\}_{i=1}^{\ell}$ be the positive demonstrations for action a observed up to time t. For each numeric variable $x \in X$, these demonstrations induce the linear system

$$\boldsymbol{A}_x \boldsymbol{\theta}_x = \boldsymbol{y}_x, \tag{4}$$

where $\boldsymbol{\theta}_x = [\boldsymbol{w}_x, b_x]^T$ contains the unknown coefficients,

$$\boldsymbol{A}_x = \begin{bmatrix} s_1[\boldsymbol{x}]^T & 1 \\ \vdots & \vdots \\ s_\ell[\boldsymbol{x}]^T & 1 \end{bmatrix}, \qquad \boldsymbol{y}_x = \begin{bmatrix} s'_1[x] \\ \vdots \\ s'_\ell[x] \end{bmatrix}.$$

$\boldsymbol{A}_x$ and $\boldsymbol{y}_x$ collect, respectively, the pre-state vectors and the observed post-state values, and each row encodes the constraint $s'_i[x] = \boldsymbol{w}_x \cdot s_i[\boldsymbol{x}] + b_x$ from demonstration i. The system admits either a unique solution or a parametric (infinite) family of solutions.

Solution space of numeric effects. For each $x \in X$, the coefficients consistent with $D^{a,+}_t$ form the solution set

$$\mathcal{S}_{a,x}(D^{a,+}_t) = \{\, \boldsymbol{\theta}_x \in \mathbb{Q}^{m+1} : \boldsymbol{A}_x \boldsymbol{\theta}_x = \boldsymbol{y}_x \,\}.$$

By standard linear algebra, its parametric form is

$$\mathcal{S}_{a,x}(D^{a,+}_t) = \Big\{\, \boldsymbol{\theta}_x(\boldsymbol{\lambda}) = \boldsymbol{\theta}^*_x + \sum_{i=1}^{k_x} \lambda_i \, \boldsymbol{v}_{x,i} \;:\; \lambda_i \in \mathbb{Q} \Big\},$$

where $\boldsymbol{\theta}^*_x$ is a particular solution, $\{\boldsymbol{v}_{x,1}, \ldots, \boldsymbol{v}_{x,k_x}\}$ is a basis of the null space of $\boldsymbol{A}_x$, and k_x is the number of free parameters (dimension of the null space). We write $\boldsymbol{\theta}_x(\boldsymbol{\lambda}) = \big(\boldsymbol{w}_x(\boldsymbol{\lambda}),\, b_x(\boldsymbol{\lambda})\big)$ for the corresponding weight and bias.

A choice of $\boldsymbol{\lambda}$ determines a concrete assignment

$$x := \boldsymbol{w}_x(\boldsymbol{\lambda}) \cdot \boldsymbol{x} + b_x(\boldsymbol{\lambda}) \qquad (x \in X),$$

and a consistent effect hypothesis for a is obtained by choosing such parameters independently for each x. Thus, the consistent effect space $\mathsf{Eff}^a_{D_t}$ is fully characterized by the families $\{\mathcal{S}_{a,x}(D^{a,+}_t)\}_{x \in X}$, without enumerating the (possibly infinite) set of concrete assignments they represent.

Special case: Simple Numeric Planning (SNP). A particularly relevant subclass of LNP is *Simple Numeric Planning* (SNP) (Hoffmann 2003; Scala et al. 2020), where numeric effects are restricted to additive updates of the form

$$x \coloneqq x + b_x.$$

This corresponds to fixing $\boldsymbol{w}_x$ to the unit vector on x and leaving only the bias b_x as an unknown. Each positive demonstration then yields a single linear constraint

$$s'_i[x] - s_i[x] = b_x,$$

so any single positive demonstration for action a fully determines b_x. Consequently, the solution space $\mathcal{S}_{a,x}(D_t^{a,+})$ becomes a singleton for every $x \in X$, and so does $\mathsf{Eff}^a_{D_t}$.

Illustrative example. Consider an action a with a single numeric variable x, whose effect is modeled in the general linear form $x \coloneqq wx + b$. Suppose we observe a single positive demonstration $\langle s, a, s' \rangle$ with $s[x] = 1$ and $s'[x] = 3$. The resulting linear system reduces to

$$w \cdot 1 + b = 3,$$

so the solution set is

$$\mathcal{S}_{a,x}(D_t^{a,+}) = \{(w, b) \in \mathbb{Q}^2 : w + b = 3\},$$

a line in coefficient space. In parametric form,

$$\boldsymbol{\theta}_x(\lambda) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \qquad \lambda \in \mathbb{Q}.$$

Each λ yields a distinct effect $x \coloneqq (3 - \lambda)x + \lambda$ consistent with the demonstration, and all of them belong to $\mathsf{Eff}^a_{D_t}$.

Under the SNP restriction, however, we enforce $x \coloneqq x + b$, i.e., $w = 1$, so the same demonstration gives

$$1 \cdot 1 + b = 3 \quad \Rightarrow \quad b = 2.$$

Thus $\mathsf{Eff}^a_{D_t}$ reduces to the singleton $x \coloneqq x + 2$, illustrating how structural restrictions in the target language (LNP vs. SNP) shrink the consistent effect space from a full parametric family to a single solution.

Derivation of Sound and Complete Models

This section describes how to derive optimal sound and complete approximations of the unknown true model from the learned spaces of consistent preconditions and effects.

Extracting the Sound Model

By Aineto and Scala (2024, Theorem 6), the optimal sound model is the one whose transition system is exactly the intersection of all transition systems of models consistent with the demonstrations (Eq. (1)). Thus, for each action a, we must retain exactly those transitions that appear in *every* consistent model.

As shown in the previous section, the space $\mathsf{Pre}^a_{D_t}$ of consistent preconditions has a single lower-bound element. We therefore write $\mathcal{L}^a_{D_t} = \{\, p^a_{\min} \,\}$, where $p^a_{\min}$ is the most restrictive precondition consistent with all demonstrations of a. Likewise, $\mathsf{Eff}^a_{D_t}$ contains all effect hypotheses consistent with the positive demonstrations of a.

A transition (s, a, s') appears in the intersection $\bigcap_{M' \in \mathcal{M}_{D_t}} T_{M'}$ iff both: (i) all consistent models agree that a is executable in s, and (ii) all consistent models agree on the resulting successor s'. Condition (i) holds exactly when $s \models p^a_{\min}$. Condition (ii) holds exactly when $\mathsf{Eff}^a_{D_t}$ contains a single hypothesis, i.e., when the effect of a is uniquely determined for every numeric variable. This leads directly to the following construction.

Definition 10 (Sound model)**.** *For each action $a \in A$, let $\mathcal{L}^a_{D_t} = \{p^a_{\min}\}$ be the minimal precondition and let $\mathsf{Eff}^a_{D_t}$ be the set of consistent numeric effects. The sound model $M^t_S = \langle F, X, A, \mathsf{pre}, \mathsf{eff} \rangle$ is defined by*

$$\mathsf{pre}(a) = \begin{cases} p^a_{\min}, & \text{if } |\mathsf{Eff}^a_{D_t}| = 1, \\ \bot, & \text{otherwise}, \end{cases}$$

$$\mathsf{eff}(a) = \begin{cases} e^a_{\text{uniq}}, & \text{if } |\mathsf{Eff}^a_{D_t}| = 1, \\ \emptyset, & \text{otherwise}, \end{cases}$$

where e^a_{uniq} is the unique element of $\mathsf{Eff}^a_{D_t}$ and $\bot$ denotes an unsatisfiable precondition.

Intuitively, when multiple effects remain possible, no transition is common across all consistent models, so the sound model disables the action entirely.

Theorem 4. *M^t_S satisfies $T_{M^t_S} = \bigcap_{M' \in \mathcal{M}_{D_t}} T_{M'}$.*

Proof sketch. ($\subseteq$) If $|\mathsf{Eff}^a_{D_t}| > 1$, then different consistent models produce different successors when executing a, so no transition involving a appears in the intersection; M^t_S also excludes all such transitions by setting $\mathsf{pre}(a) = \bot$. If $|\mathsf{Eff}^a_{D_t}| = 1$, all consistent models share the effect e^a_{uniq}. Since $p^a_{\min}$ is satisfied by every consistent precondition hypothesis, any transition admitted by M^t_S is admitted by all models in $\mathcal{M}_{D_t}$.

($\supseteq$) Since M^t_S itself is a model in $\mathcal{M}_{D_t}$, the intersection of all transition systems is a subset of $T_{M^t_S}$. □

Extracting the Complete Model

By Aineto and Scala (2024, Theorem 7), the optimal complete model is the one whose transition system is exactly the union of all transition systems of models consistent with the demonstrations (Eq. (2)). Thus, for each action $a \in A$, the complete model must include every transition (s, a, s') that appears in *at least one* consistent model.

A transition (s, a, s') belongs to this union iff (i) some consistent precondition for a is satisfied in s, and (ii) some consistent effect hypothesis maps s to s'. The precondition part is straightforward as it corresponds to allowing execution of a in any state admitted by at least one of the maximally permissive preconditions in $\mathcal{U}^a_{D_t}$. The effect part is more delicate: $\mathsf{Eff}^a_{D_t}$ may contain infinitely many consistent effects. To remain within deterministic setting, we encode the choice of an effect hypothesis *inside the state*. The remainder of the section describes how to construct such a model, handling preconditions and effects independently.

Preconditions. Completeness requires that a be applicable in any state that is admitted by some consistent precondition. Since $\mathsf{Pre}^a_{D_t}$ is upward-closed under inclusion, it suffices to consider only the maximally permissive hypotheses in $\mathcal{U}^a_{D_t}$. We therefore introduce, for each $p \in \mathcal{U}^a_{D_t}$, a syntactic variant a^p of action a whose precondition is exactly p; all such variants share the same effect (constructed below). This ensures that a is applicable in precisely the states where at least one consistent model permits it.

Parametric numeric effects. The effect for action a is encoded by the parametric assignments

$$x \coloneqq \boldsymbol{w}_x(\boldsymbol{\lambda}) \cdot \boldsymbol{x} + b_x(\boldsymbol{\lambda}) \qquad (x \in X),$$

where varying the parameters $\boldsymbol{\lambda}$ ranges over consistent effects in $\mathsf{Eff}^a_{D_t}$. Thus, by choosing different values of $\boldsymbol{\lambda}$, one can generate *every* effect hypothesis consistent with the demonstrations. To enable the complete model to realize any such effect at execution time, we make the parameters explicit state variables and introduce actions that modify them.

Auxiliary parameters and control actions. For each action $a \in A$, variable $x \in X$, and free parameter index $i \in \{1, \ldots, k_x\}$, we introduce a numeric variable $\lambda_{a,x,i}$ encoding the i-th free term of $\mathcal{S}_{a,x}(D_t^+)$. To control the granularity of parameter updates, we also introduce two numeric variables $\delta_N, \delta_D \in \mathbb{N}_{>0}$ and interpret $\Delta = \delta_N/\delta_D$ as a dynamically adjustable step size.

We add two classes of auxiliary actions:

- *parameter-update actions*:

$$\lambda_{a,x,i} \coloneqq \lambda_{a,x,i} \pm \frac{\delta_N}{\delta_D};$$

- *step-size refinement actions*:

$$\delta_N \coloneqq \delta_N \pm 1, \qquad \delta_D \coloneqq \delta_D \pm 1.$$

These affect only the auxiliary variables, not the original problem state. From any initial value (e.g. $\delta_N = \delta_D = 1$), any rational step size can be reached by finitely many refinements, and arbitrary parameter vectors $\boldsymbol{\lambda}$ can be encoded using repeated parameter updates.

Definition 11 (Complete model)**.** *Let D_t be the demonstrations observed so far. For each action $a \in A$, let $\mathcal{U}^a_{D_t}$ be the upper boundary of its precondition space, and let $\{\mathcal{S}_{a,x}(D_t^+)\}$ be the parametric solution sets capturing $\mathsf{Eff}^a_{D_t}$. The complete model M^t_C has numeric variables*

$$X \cup \{\lambda_{a,x,i} : a \in A,\ x \in X,\ i = 1, \ldots, k_x\} \cup \{\delta_N, \delta_D\},$$

and actions $A' \cup A^\lambda \cup A^\delta$, where:

- $A' = \{\, a^p : a \in A,\ p \in \mathcal{U}^a_{D_t} \,\}$*. Each a^p has precondition* $\mathsf{pre}(a^p) = p$ *and, for every $x \in X$, effect*

$$x \coloneqq \boldsymbol{w}_x(\lambda) \cdot \boldsymbol{x} + b_x(\lambda),$$

  where $\lambda = (\lambda_{a,x,1}, \ldots, \lambda_{a,x,k_x})$.
- *A^λ contains all parameter-update actions.*
- *A^δ contains all step-size refinement actions.*

Let $T_{M^t_C}|_X$ denote the transitions restricted to the original numeric variables X.

Theorem 5. *$T_{M^t_C}$ satisfies $T_{M^t_C}|_X = \bigcup_{M' \in \mathcal{M}_{D_t}} T_{M'}$.*

Proof sketch. ($\supseteq$) Fix any transition $(s, a, s') \in T_{M'}$ for some $M' \in \mathcal{M}_{D_t}$. Its precondition is satisfied by some $p \in \mathsf{Pre}^a_{D_t}$, hence by some $p' \in \mathcal{U}^a_{D_t}$ with $p \le p'$, so $a^{p'}$ is executable in M^t_C. The effect used by M' corresponds to a choice of parameters $\{\lambda_{a,x,i}\}$; these values can be reached in M^t_C using step-size and parameter-update actions. Executing $a^{p'}$ then yields the same successor on X as M'.

($\subseteq$) Any transition produced by M^t_C arises from some a^p with $p \in \mathcal{U}^a_{D_t}$ under a particular assignment of the parameters $\{\lambda_{a,x,i}\}$. Each such assignment defines a consistent effect hypothesis in $\mathsf{Eff}^a_{D_t}$. Hence the resulting transition coincides with one produced by some $M' \in \mathcal{M}_{D_t}$. □

Experimental Evaluation

Our evaluation aims to assess the practical usefulness of the learned models by addressing two questions: (i) how accurate are the learned models, and (ii) how well do they approximate the true model for planning purposes.

(i) Model accuracy is measured using standard metrics: precision and recall. The sound model never admits false positives, but it may miss valid transitions (false negatives). Its precision is therefore always 1, and we evaluate it through recall, i.e., the fraction of valid transitions it admits. Dually, the complete model never rejects valid transitions, but may admit invalid ones (false positives). Its recall is always 1, and we evaluate it through precision, i.e., the fraction of invalid transitions it successfully excludes.

(ii) Approximation quality of the sound and complete model is measured by solving planning tasks optimally with each model and comparing the resulting plan lengths with those obtained under the true model. Since the sound model under-approximates the true model and the complete one over-approximates it, the plan lengths they produce form upper and lower bounds on the true optimal cost, respectively.

Baselines. We compare our sound model with NSAM, the current state-of-the-art method for learning numeric models with soundness guarantees. NSAM assumes less prior knowledge—notably, it does not assume known orientations of numeric preconditions; our goal is to quantify the benefit of leveraging this additional structural information.

Data generation and setup. We evaluate the approach across several domains from the International Numeric Planning Competition (Taitler et al. 2024): COUNTERS, BLOCK-GROUPING, ROVER, FO-COUNTER, and SAILING, together with a new domain, SHAPES, designed to stress-test precondition learning. SHAPES features complex preconditions, akin to SAILING but with richer geometry. The dataset contains positive demonstrations from satisficing plans and negative ones from random walks; full details in the supplementary material. We use 5-fold cross-validation and initialize each algorithm with a bootstrap set consisting of one positive demonstration per action. Code and datasets are available at https://github.com/daineto/NASCAL.

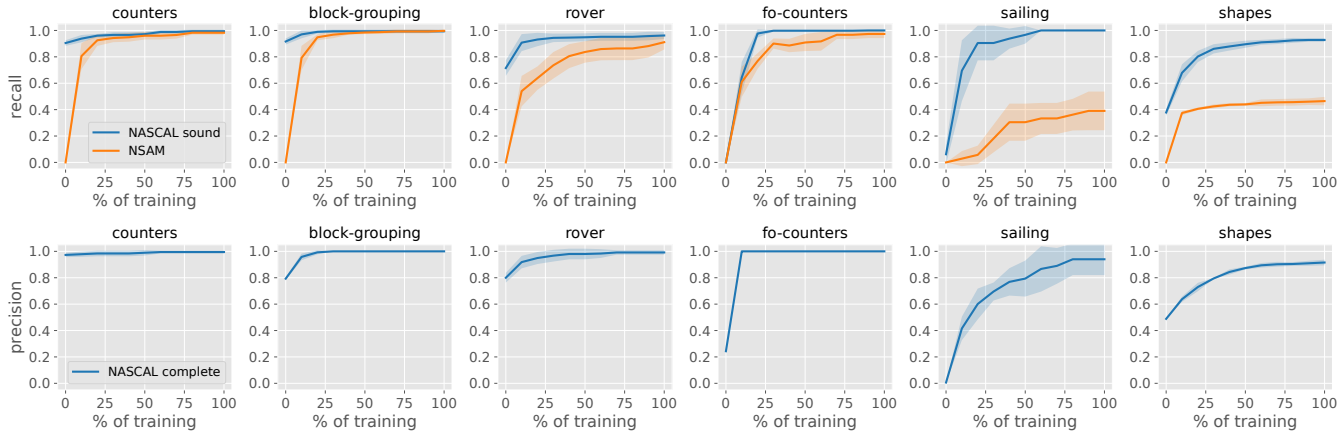

Figure 2: Recall of the sound models (top) and precision of the complete models (bottom) as a function of training size. Curves show the mean over 5-fold cross-validation; shaded regions show the variance.

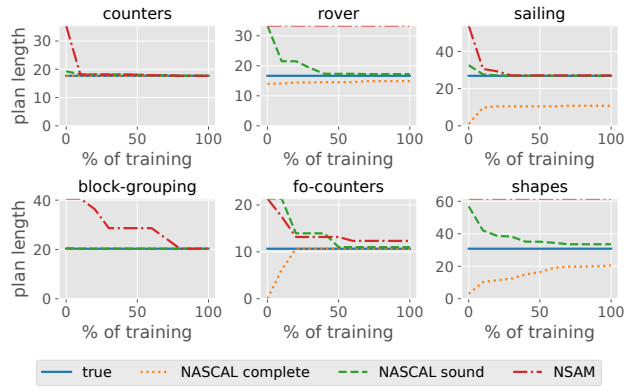

Figure 3: Avg. optimal plan length for the true, sound, and complete models vs. training size. Mean over 5 folds.

Results: model accuracy. We report results only for actions with numeric preconditions, as these are the ones for which our framework and NSAM differ; the construction of numeric effects in the sound model follows the same principles for both methods. Figure 2 reports macro-averaged recall (sound models) and precision (complete model), averaged uniformly across actions to avoid bias toward frequently executed ones. Metrics are shown as training increases; higher values indicate better learning. NASCAL consistently reaches high recall faster than NSAM. In simpler domains such as COUNTERS and BLOCK-GROUPING the two methods behave similarly, while in domains with complex numeric preconditions— SAILING and SHAPES —the gap widens substantially. A per-action inspection reveals significant variability, especially in SHAPES, where visit actions (defined by complex numeric regions) are the hardest to learn. Precision improves steadily for our complete model, though at domain-dependent rates; again, SAILING and SHAPES are the most challenging due to their intricate numeric constraints.

Results: approximation quality. Figure 3 reports the average optimal plan lengths across 10 test instances obtained with the sound and complete models, compared against the true optimal cost. As expected, the sound model yields an upper bound on the true plan length, while the complete model yields a lower bound. In most domains, these bounds gracefully approach the true cost at different rates, whereas in COUNTERS and BLOCK-GROUPING both models converge almost immediately. This is because demonstrations fall either within or just outside the true numeric preconditions, yielding highly informative samples. By contrast, SAILING and SHAPES illustrate the opposite scenario: the agent moves in a largely unbounded numeric space, and many preconditions—especially for visit actions—must be inferred from states far from the true boundary, leading to slower convergence. This suggests qualitatively different classes of domains, a distinction we intend to formalize in future work. NSAM's sound model provides weaker approximations in 4 of the 6 domains; in ROVER and SHAPES its overly restrictive preconditions prevent any solution.

Conclusion

In this paper, we introduce a novel approach for learning numeric action models from demonstrations, providing anytime guarantees of soundness and completeness with respect to the true hidden model. To the best of our knowledge, this is the first fully specified framework for learning numeric action models; our sound variant also offers a complementary perspective to previous work (Mordoch, Juba, and Stern 2023). A key contribution of our formulation is its ability to leverage prior knowledge about the true model, expressed through the orientation of the bounded convex region. We demonstrate that incorporating this information significantly improves convergence. In future work, we plan to investigate how these anytime models can be integrated within an active learning loop, allowing them to request the most informative demonstrations, while equipping them with planning capabilities with full formal assurances.

Acknowledgments

Diego was partially supported by the GENERALITAT VALENCIANA Project PROMETEO CIPROM/2023/23. Enrico Scala has been supported by the Italian Ministry of University and Research within the PRIMA 2024 programme project "Optimizing Water Resources in Coastal Areas using Artificial Intelligence" (AI4WATER – D53C25000510006).

References

Aineto, D.; Celorrio, S. J.; and Onaindia, E. 2019. Learning action models with minimal observability. *Artificial Intelligence*, 275: 104–137.

Aineto, D.; Jiménez, S.; and Onaindia, E. 2022. A comprehensive framework for learning declarative action models. *Journal of Artificial Intelligence Research*, 74: 1091–1123.

Aineto, D.; and Scala, E. 2024. Action Model Learning with Guarantees. In *KR*, 801–811.

Aineto, D.; Scala, E.; Onaindia, E.; and Serina, I. 2023. Falsification of Cyber-Physical Systems Using PDDL+ Planning. In *ICAPS*, 2–6.

Alon, L.; Weitman, H.; Shleyfman, A.; and Kaminka, G. A. 2024. Planning to be Healthy: Towards Personalized Medication Planning. In *ECAI 2024*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, 4232–4239.

Amir, E.; and Chang, A. 2008. Learning partially observable deterministic action models. *Journal of Artificial Intelligence Research*, 33: 349–402.

Bachor, P.; and Behnke, G. 2024. Learning planning domains from non-redundant fully-observed traces: theoretical foundations and complexity analysis. In *AAAI*, 20028–20035.

Bonet, B.; and Geffner, H. 2020. Learning First-Order Symbolic Representations for Planning from the Structure of the State Space. In *ECAI*, 2322–2329.

Cresswell, S. N.; McCluskey, T. L.; and West, M. M. 2013. Acquiring planning domain models using LOCM. *The Knowledge Engineering Review*, 28(2): 195–213.

Ghallab, M.; Nau, D. S.; and Traverso, P. 2004. *Automated planning - theory and practice*. Elsevier. ISBN 978-1-55860-856-6.

Gösgens, J.; Jansen, N.; and Geffner, H. 2025. Learning lifted strips models from action traces alone: A simple, general, and scalable solution. In *ICAPS*, 189–197.

Haslum, P.; Lipovetzky, N.; Magazzeni, D.; and Muise, C. 2019. *An Introduction to the Planning Domain Definition Language*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Hoffmann, J. 2003. *Utilizing Problem Structure in Planning: A local search approach*, volume 2854. Springer Science & Business Media.

Juba, B.; and Stern, R. 2022. Learning Probably Approximately Complete and Safe Action Models for Stochastic Worlds. In *AAAI*, 9795–9804.

Kiam, J. J.; Scala, E.; Javega, M. R.; and Schulte, A. 2020. An AI-Based Planning Framework for HAPS in a Time-Varying Environment. In *ICAPS*, 412–420.

Lamanna, L.; Serafini, L.; Saetti, A.; Gerevini, A. E.; and Traverso, P. 2025. Lifted action models learning from partial traces. *Artificial Intelligence*, 339: 104256.

Mordoch, A.; Juba, B.; and Stern, R. 2023. Learning Safe Numeric Action Models. In *AAAI*, 12079–12086.

Mourão, K.; Zettlemoyer, L.; Petrick, R. P. A.; and Steedman, M. 2012. Learning STRIPS Operators from Noisy and Incomplete Observations. In *UAI*, 614–623.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2020. Subgoaling Techniques for Satisficing and Optimal Numeric Planning. *J. Artif. Intell. Res.*, 68: 691–752.

Segura-Muros, J. Á.; Pérez, R.; and Fernández-Olivares, J. 2021. Discovering relational and numerical expressions from plan traces for learning action models. *Applied Intelligence*, 51(11): 7973–7989.

Stern, R.; and Juba, B. 2017. Efficient, Safe, and Probably Approximately Complete Learning of Action Models. In *AAAI*, 4405–4411.

Taitler, A.; Alford, R.; Espasa, J.; Behnke, G.; Fiser, D.; Gimelfarb, M.; Pommerening, F.; Sanner, S.; Scala, E.; Schreiber, D.; Segovia-Aguas, J.; and Seipp, J. 2024. The 2023 International Planning Competition. *AI Mag.*, 45(2): 280–296.

Vallati, M.; Magazzeni, D.; Schutter, B. D.; Chrpa, L.; and McCluskey, T. L. 2016. Efficient Macroscopic Urban Traffic Models for Reducing Congestion: A PDDL+ Planning Approach. In *AAAI*, 3188–3194.

Verma, P.; Marpally, S. R.; and Srivastava, S. 2021. Asking the right questions: Learning interpretable action models through query answering. In *AAAI*, 12024–12033.

Xi, K.; Gould, S.; and Thiébaux, S. 2024. Neuro-symbolic learning of lifted action models from visual traces. In *ICAPS*, 653–662.

Yang, Q.; Wu, K.; and Jiang, Y. 2007. Learning action models from plan examples using weighted MAX-SAT. *Artificial Intelligence*, 171(2-3): 107–143.

Zhuo, H. H.; and Kambhampati, S. 2013. Action-model acquisition from noisy plan traces. In *IJCAI*, 2444–2450.

Zhuo, H. H.; Yang, Q.; Hu, D. H.; and Li, L. 2010. Learning complex action models with quantifiers and logical implications. *Artificial Intelligence*, 174(18): 1540–1569.