LEARNING THE OBSERVATION FUNCTION IN POMDPS: IDENTIFIABILITY THROUGH STATE INVARIANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Policies for *partially observable Markov decision processes* (POMDPs) act on state observations. The *observation function* governs the probabilities of emitting observations at each state. We study the following problem: Identify the *unique* probabilities of an *unknown* observation function from a dataset of observations generated under an *unknown* policy. However, the underlying observation function of a POMDP is not identifiable in general. We therefore define a class of observation functions that we show to be identifiable. The key property of these functions is that the emission probability for every observation is invariant across all states that can emit it, for example, the probabilities of sensor failures are invariant. We call this class of observation functions *state-invariant*. We provide an optimization-based approach to learn the observation function from data: we formulate the problem as a nonlinear program (NLP) and convexify the NLP toward a disciplined convex program (DCP) solvable in polynomial time. We prove that the solutions to the optimization problems are *probably approximately correct* (PAC) with respect to the dataset. Our experiments support the theoretical findings.

1 INTRODUCTION

Partially observable Markov decision processes (POMDPs) (Åström, 1965) model sequential decision-making problems in which an agent cannot directly observe the underlying state. Instead, the agent receives observations and selects actions using a *policy* that depends on the observations seen so far. The *observation function* specifies the probability of emitting each observation in a given state and thus, for example, provides a natural way to model imperfect sensors in robotics (Thrun et al., 2005).

Planning can be performed on a POMDP when the model is fully specified (Kurniawati et al., 2008; Smith & Simmons, 2012; Katt et al., 2017; Andriushchenko et al., 2026). In practice, however, parts of the model are often unknown. We consider the setting where the observation function is unknown and must be estimated from a dataset of observation histories generated under an unknown policy. For principled POMDP planning, it is not enough to find *any* observation function that fits the data: one that differs from the ground-truth model can yield different optimal policies, even if it fits the data perfectly (see Appendix A). Thus, the goal is to find the ground-truth observation probabilities, a task known as *identification*. The core challenge is that observation-histories (with no hidden-state information) are ambiguous: multiple observation functions may induce the same distribution over observation histories, meaning that multiple models may fit the same data (Allman et al., 2009; Finesso, 1990; Gilbert, 1959; Tune et al., 2013). Hence, we require restrictions that ensure that the ground-truth is *identifiable*. Moreover, even just estimating any observation function consistent with the data can be both statistically (Krishnamurthy et al., 2016b; Xiong et al., 2022; Golowich et al., 2022) and computationally intractable (Terwijn, 2002; Mossel & Roch, 2005). Together, these challenges motivate the study of observation function classes that are *efficiently identifiable*.

Example 1. Fig. 1 **a)** exemplifies a common form of partial observability that, as we show below, is efficiently identifiable. A drone, initially located in the upper-left cell, must fly to the lower-right cell. When attempting to move in a cardinal direction, the drone succeeds with probability 0.7 or gets relocated in any of the three other directions with probability 0.1 due to wind disturbances. The state has two features: location and local occlusion level (written in upper-right corner of each cell). The drone observes its location through a GPS and the occlusion level through an optical sensor, but at occlusion level x, the GPS signal is lost with probability $x/3$. Hence, the observation (ERR, x) of a lost GPS signal at occlusion level x is emitted with the same probability $x/3$ by every state that can emit it, a dependence we can exploit to *identify* the loss probability for each occlusion level.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

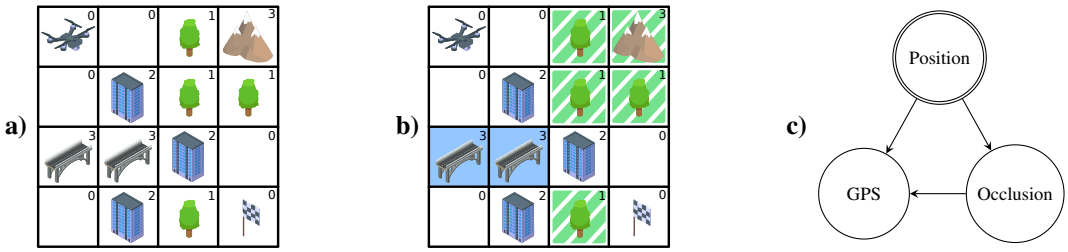


Figure 1: Drone-navigation example with unreliable GPS and occlusion sensors. Example 1 describes the POMDPs illustrated in **a)** and **b)**. **c)** shows the causal relations governing the partial observability.

Fig. 1 **b)** extends the example with error codes: high humidity in the forest (dashed green cells) can short-circuit both sensors, and the optical sensor can fail in the darkness under the bridge (solid blue cells). Each error is again equally likely in every state that can provoke it, which, perhaps surprisingly, still suffices for identification. Intuitively, by estimating the error probability of the GPS, we can estimate the likelihood of being in a humid cell, and thus obtain the probability of short circuiting.

We introduce *state-invariance* (SI), the property illustrated in the example: an observation function is SI if every state capable of emitting a given observation emits it with equal probability. SI is inspired by the missing-data literature, namely the standard assumption that data is *missing at random* (MAR) (Rubin, 1976), which requires that the probability of a feature going missing depends only on observed values. The GPS loss in Fig. 1 **a)** is MAR because it depends on the observed occlusion level, as shown in the causal graph in Fig. 1 **c)**. The error codes (like "GPS error" or "short-circuit") in Fig. 1 **b)** are *not* MAR, as MAR observations functions (and missingness functions in general) can only determine whether to omit or reveal features (Wendland et al., 2025). Thus, SI generalizes MAR, see Appendix B for a formal proof.

Identifiability under the MAR assumption is well established in the missing-data literature (Mohan & Pearl, 2021). For observation functions of POMDPs, however, identification from a dataset of histories was left open even for MAR in the first work connecting missingness and POMDPs (Wendland et al., 2025). The more general SI property has, to the best of our knowledge, not been considered so far.

Contributions. We introduce the class of SI observation functions (Section 4.2) and show that its members are efficiently identifiable from observation histories generated under an unknown policy.

- *Identification.* We provide a nonlinear program (NLP) to identify SI observation functions (Section 4.3) that is probably approximately correct (PAC, Section 4.4): for every $\varepsilon, \delta > 0$, an explicit dataset size N suffices for the approximation error to be at most ε with probability $1 - \delta$.
- *Efficiency.* Since general NLPs are intractable (Canny, 1988; Hillar & Lim, 2013), we relax the NLP into a disciplined convex program (DCP, Grant et al., 2006) that is solvable in polynomial time and inherits the PAC guarantees (Section 5.1). Further, we reformulate the NLP and DCP into "time-flattened" variants, which yield smaller optimization problems , but strip the system of all temporal aspects (Section 5.2); and we extend the NLP and DCP into algorithms with improved asymptotic runtime under model sparsity (Section 5.3).
- *Experimental validation.* We evaluate our methods on simulated data, on missing-data benchmarks from Wendland et al. (2025), and on a drone case study similar to Fig. 1 (Section 6).

2 Related Work

Missing data. Missing data makes statistical analysis and machine learning biased unless the reason for which said data is missing is accounted for (Schafer & Graham, 2002; Little & Rubin, 2019; Rubin, 1976). There are assumptions other than MAR that inspired SI: For instance, Heitjan & Rubin (1991); Gill & Grünwald (2008) consider coarsening at random (CAR), which extends the MAR assumption to coarse data problems—data being available only in broad, grouped, or imprecise categories rather than exact values. Such extensions and respective techniques are designed in contexts orthogonal to POMDP planning and, to the best of our knowledge, existing methods are not applicable in our setting. Wendland et al. (2025) study missing data problems in the context of POMDPs, but use assumptions that are more restrictive than MAR, while we generalize MAR.

HMM learning. A POMDP under a fixed policy induces a hidden Markov model (HMM). As for POMDPs, learning HMMs from observations is generally intractable (Abe & Warmuth, 1992; Terwijn, 2002). In practice, polynomial-time approximations based on expectation maximization (e.g., Baum-Welch) (Rabiner, 2002; Baum et al., 1970) and Bayesian inference (Stolcke & Omohundro, 1992) are common. Polynomial-time learning guarantees are possible if the transition matrix has a non-zero determinant (Mossel & Roch, 2005). Identification approaches for HMMs (i.e., recovering the unique underlying model) typically employ sufficient conditions such as separability assumptions (Hsu et al., 2012), specific tree topologies (Pearl & Tarsi, 1986; Steel et al., 1998), and assumptions on the (marginal) distribution of the observations (Liu et al., 2023). However, HMM learning under a fixed policy captures only policy-specific observation dynamics and are insufficient to learn the model's observation function in isolation (see Appendix A). Moreover, they rely on domain-specific assumptions and are not particularly suitable for POMDP planning. Therefore, we evaluate the classical Baum-Welch in Section 6 and consider other works in this direction orthogonal.

MDP and POMDP learning. Prior work on learning fully observable MDPs focuses on estimating transition dynamics, using methods such as expectation maximization (Bacci et al., 2021), the L^*-algorithm (Tappler et al., 2019), state merging (Mao et al., 2012), and confidence/uncertainty set estimation (Wiering, 1999; Strehl & Littman, 2004; 2008; Jaksch et al., 2010). In POMDPs, most work studies model learning without requiring identification, either to compute near-optimal policies with PAC guarantees (Krishnamurthy et al., 2016a; Guo et al., 2016; Liu et al., 2022; Lee et al., 2023; Uehara et al., 2023; Even-Dar et al., 2005) or to minimize regret (Jafarnia-Jahromi et al., 2022; Kwon et al., 2021; 2024). Unlike our work, these approaches aim for PAC guarantees on the resulting policies and do not aim to identify the model, typically relying on strong assumptions on the transition and observation dynamics. To the best of our knowledge, work on POMDP identification remains sparse and lacks PAC guarantees, instead focusing on heuristic approaches (Ho et al., 2025).

3 Partially observable Markov decision processes (POMDPs)

A *probability distribution* over a finite set X is a function $\mu\colon X \to [0,1]$ with $\sum_{x\in X}\mu(x) = 1$. The set of such μ is $\mathrm{Distr}(X)$. The *support* of μ is $\mathrm{supp}(\mu) = \{x \in X \mid \mu(x) > 0\}$. We use $\boldsymbol{x} \sim \mu$ to denote that random variable $\boldsymbol{x}$ is induced by μ, and often boldface random variables. Given $\sigma\colon X \to \mathrm{Distr}(Y)$, we let $\sigma(y \mid x) := \sigma(x)(y)$. The indicator $\mathbf{1}_{\varphi}$ returns 1 if predicate φ holds and 0 otherwise. The Kleene closure of a set X is $X^* := \bigcup_{i\in\mathbb{N}} X^i$.

Definition 3.1. A *partially observable Markov decision process* (POMDP; Åström, 1965) is a tuple $\mathcal{M} = (S, A, T, Z, O, b_{\text{init}})$, where S, A, and Z are finite sets called *state space*, *action space*, and *observation space*, respectively, $T\colon S \times A \to \mathrm{Distr}(S)$ is the *transition function*, $O\colon S \to \mathrm{Distr}(Z)$ is the *observation function*, and $b_{\text{init}} \in \mathrm{Distr}(S)$ is the *initial state distribution*.

Assumption 1. The observation function $O\colon S \to \mathrm{Distr}(Z)$ is action-independent without loss of generality via standard transformation; see Appendix G for details.

For a finite horizon of length $L \in \mathbb{N}$, histories are sequences $(\boldsymbol{z}^{(1)}, \boldsymbol{a}^{(1)}, \ldots, z^{(L)}, \boldsymbol{a}^{(L)}) \in (Z \times A)^L$. An agent selects actions based on the history of observations and actions. A *policy* is a map $\pi\colon Z \times (A \times Z)^* \to \mathrm{Distr}(A)$, and the space of such policies is $\Pi := \mathrm{Distr}(A)^{Z\times(A\times Z)^*}$.

Definition 3.2 (Induced Markov chain)**.** A POMDP $\mathcal{M}$ with a policy $\pi \in \Pi$ induces a stochastic process over states, actions, and observations. In particular, it induces a probability measure over finite state sequences $\{\boldsymbol{s}^{(t)}\}_{t=1}^{L}$ and observation sequences $\{\boldsymbol{z}^{(t)}\}_{t=1}^{L}$, via the dynamics $\boldsymbol{s}^{(1)} \sim b_{\text{init}}$, $\boldsymbol{a}^{(t)} \sim \pi(\boldsymbol{z}^{(1)}, \boldsymbol{a}^{(1)}, \ldots, \boldsymbol{z}^{(t)})$, $\boldsymbol{s}^{(t+1)} \sim T(\boldsymbol{s}^{(t)}, \boldsymbol{a}^{(t)})$, $\boldsymbol{z}^{(t)} \sim O(\boldsymbol{s}^{(t)})$. The induced stochastic process is a labeled Markov chain (Åström, 1965; Desharnais, 1999), where the probability of visiting state s' from s is given by the transition function $T^{\pi}(s' \mid s) := \mathbb{P}(\boldsymbol{s}^{(t+1)} = s' \mid \boldsymbol{s}^{(t)} = s) = \sum_{a\in A} T(s' \mid s, a)\,\pi(a \mid s^{(1)}, a^{(1)}, \ldots, s)$. Hence, visiting state s initially has probability $\mathbb{P}(\boldsymbol{s}^{(1)} = s) = b_{\text{init}}(s)$, and inductively $\mathbb{P}(\boldsymbol{s}^{(t+1)} = s') = \sum_{s\in S}\mathbb{P}(\boldsymbol{s}^{(t)} = s)\,T^{\pi}(s' \mid s)$ for subsequent time steps. Observing z at time t has probability $\mathbb{P}(\boldsymbol{z}^{(t)} = z) = \sum_{s\in S} O(z \mid s)\mathbb{P}(\boldsymbol{s}^{(t)} = s)$.

4 Identification of state invariant observation functions

In this section, we introduce the observation function identification problem in Section 4.1. In Section 4.2, we introduce the *state-invariance* (SI) condition, a sufficient assumption that enables

efficient identification. Then, Section 4.3 introduces a non-linear program (NLP) formulation that solves the identification problem. Finally, Section 4.4 provides proofs of (statistical) correctness.

4.1 Problem statement

We aim to identify the observation function from a given dataset. Since datasets are finite, we cannot identify the exact underlying observation function with absolute certainty. Instead, our identification shall be done probably approximately correctly (PAC) (Valiant, 1984; Ashok et al., 2019).

PAC-identification problem

We are given a dataset $\mathcal{D} \in (Z^L)^N$ of N traces (or histories) of observations of length $L \geqslant |S|$, sampled from the model by an unknown behavior policy $\pi_{\text{beh}} \in \Pi$. The state, action, and observation spaces S, A, and Z are known, whereas the observation function O, transition function T, and initial distribution b_{init} are unknown. The task is to compute an estimate $\hat{O}$ of the observation function that is *probably approximately correct*: for all $\varepsilon, \delta > 0$, there exists $N \in \mathbb{N}$ such that, for a dataset $\mathcal{D}$ of size N, the estimate $\hat{O}$ obtained from $\mathcal{D}$ satisfies, with probability at least $1 - \delta$ (*probably*), that $\|\hat{O} - O\| \leqslant \varepsilon$ (*approximately correct*).

Theorem 4.8 in Section 4.4 provides the technically detailed description of our PAC guarantees.

Remark 4.1. The dataset does not need to include actions, given that the observation function is action-independent by Assumption 1. For action-dependent observations, see Appendix G.

Assumption 2. The behavior policy is Markovian: $\mathbb{P}_{\pi_{\text{beh}}}(a \mid s^{(0)}, a^{(0)}, \ldots, s^{(t)}) = \mathbb{P}_{\pi_{\text{beh}}}(a \mid s^{(t)})$.

Remark 4.2. Assumption 2 is for convenience. Our results extend to datasets sampled by non-Markovian behavior policies, see Appendix F.3; however, such policies induce exponentially large Markov chains, making analysis needlessly tedious. The assumption is imposed only on the behavior policy, not on any other policy interacting with the POMDP. The assumption hence amounts to having a cooperative (yet unknown) behavior policy that avoids introducing history-dependent bias into the data. For instance, any (observation-based) memoryless policy $\pi_{\text{beh}} : Z \to \text{Distr}(A)$ suffices.

4.2 Introducing the state-invariance condition

We introduce the notion of state-invariance (SI)—a restriction on the observation function of a POMDP that enables its identification from a dataset of observations.

Definition 4.3. We say that observation function $O \colon S \to \text{Distr}(Z)$ is *state-invariant* (SI) if and only if there is a relation $\preceq \subseteq Z \times S$ where, for every observation z and state s, if $z \npreceq s$ then $O(z \mid s) = 0$, and if $z \preceq s$ and there exists another state s' such that $z \preceq s'$ then $O(z \mid s) = O(z \mid s')$.

Relation $z \preceq s$ indicates that state s *can* emit observation z (i.e., that $O(z \mid s)$ may be nonzero). Additionally, the state-invariance criterion states that, if two states s, s' can emit the same observation z, then they do so with the same probability. We provide two examples:

Example 2 (Toy model)**.** Let $S = \{s_1, s_2, s_3\}$ and $Z = \{z_1, z_2, z_3, z_4\}$. Then, we define the SI observation function O as follows: $O(z_1 \mid s_1) = 0.5$, $O(z_2 \mid s_1) = 0.5$, $O(z_1 \mid s_2) = 0.5$, $O(z_3 \mid s_2) = 0.3$, $O(z_4 \mid s_2) = 0.2$, $O(z_2 \mid s_3) = 0.5$, $O(z_3 \mid s_3) = 0.3$, $O(z_4 \mid s_3) = 0.2$.

Example 3 (Occluded navigation)**.** We elaborate on the model from Figure 1 a) and refer to Section 1 for intuition. The state space is $S := \{1, \ldots, 4\}^2$, and observation space is $Z := (\{1, \ldots, 4\}^2 \cup \mathsf{ERR}) \times \{0, \ldots, 3\}$. Movement is probabilistic, as detailed in Figure 1 a). Here, the admittance matrix encodes the fact that GPS signal may or may not be lost. For example, state $s = (3, 2, 1)$ has that $(3, 2, 1) \preceq s$, $(\mathsf{ERR}, 1) \preceq s$, but $(1, 4, 3) \npreceq s$. We confirm that the model is SI: $O((\mathsf{ERR}, z) \mid s) = z/3$ for any state $s \succeq (\mathsf{ERR}, z)$, and $O((x, y, z) \mid s') = 1 - z/3$ for any $s' \succeq (x, y, z)$.

As discussed in Section 1 and Appendix A, without any assumptions, the observation cannot generally be identified from observation histories alone. We therefore introduce the following assumption.

Assumption 3. The underlying observation function is SI and relation $\preceq$ is known.

Relation $\preceq$ being known is a natural assumption. In missing data problems (Rubin, 1976), this assumption is inherently present: $\preceq$ encodes the mere fact that data may or may not be missing (see Appendix B for formal details). Thus, all missing-data problems implicitly specify the $\preceq$ relation. As

another example, in Example 3, the fact that a state may or may not emit a GPS error fully describes this relation, and it is natural to inherently know this structural aspect of the model. We note that $\preceq$ alone does not give us the exact GPS error probabilities, nor does it reveal whether this probability is nonzero for state-observation pairs in this relation. In fact, in Example 3, we can let $(\mathsf{ERR}) \preceq (1,1)$, yet cell $(1,1)$ has zero occlusion, hence $O((\mathsf{ERR},0) \mid (x,y)) = 0/3 = 0$.

4.3 A NONLINEAR PROGRAM FOR IDENTIFICATION OF SI OBSERVATION FUNCTIONS

To learn the state-invariant (SI) observation function of a POMDP, we are given a dataset $\mathcal{D} \in (Z^L)^N$ of N observation sequences $\rho \in Z^L$. We start by computing the proportion $c_z^{(t)}$ at which each observation $z \in Z$ appears in the dataset $\mathcal{D}$ at each time step t for each observation sequence ρ:

$$c_z^{(t)} := |\mathcal{D}|^{-1} \sum_{\rho \in \mathcal{D}} \mathbf{1}_{\rho(t)=z}. \tag{1}$$

Recall the induced Markov chain in Definition 3.2. We use $c_z^{(t)}$ as a consistent estimator of the true probability $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$ of observing z at time t, which we formally confirm later in Theorem 4.8.

Toward an optimization problem. We introduce variables $p_s^{(t)} \in [0,1]$ to estimate the probability $\mathbb{P}(\boldsymbol{s}^{(t)} = s)$ of visiting state s at time t, and $q_z \in [0,1]$ to estimate $O(z \mid \tilde{s})$, where $\tilde{s}$ is any state such that $z \preceq \tilde{s}$, given that the probabilities are invariant in any state that can emit z due to SI. The number of variables is thus $|S| \cdot L + |Z|$. Assuming the variables approximately equal the intended values, $p_s^{(t)} \approx \mathbb{P}(\boldsymbol{s}^{(t)} = s)$ and $q_z \approx O(z \mid \tilde{s})$, then for any z and $\tilde{s} \preceq z$, the quantity $q_z \sum_{s \in S\colon z \preceq s} p_s^{(t)}$ approximates $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$, which is precisely the quantity estimated by $c_z^{(t)}$. For the complete derivation, refer to the proof of Lemma 4.7 in Appendix C.1. Based on the above, we let the NLP minimize the distance between $q_z \cdot \sum_{s \in S\colon z \preceq s} p_s^{(t)}$ and value $c_z^{(t)}$ obtained from the dataset in (1). Additionally, we impose constraints on the problem to ensure that the probabilities sum to one.

Optimization Problem 1 (Main NLP for PAC identification, $\texttt{NLP}_{\texttt{main}}$)**.**

$$\min_{q_z, p_s^{(t)}} \quad \sum_{t=1}^{L} \sum_{z \in Z} \left(\left(q_z \sum_{s \in S\colon z \preceq s} p_s^{(t)} \right) - c_z^{(t)} \right)^2, \tag{2}$$

$$\textit{s.t.} \quad \sum_{s \in S} p_s^{(t)} = 1, \quad \textit{for } t = 1, \ldots, L \tag{3}$$

$$\sum_{z \in Z\colon z \preceq s} q_z = 1, \quad \forall s \in S. \tag{4}$$

Example 4 ($\texttt{NLP}_{\texttt{main}}$)**.** Recall the SI observation function in Example 2. Suppose the horizon length is 1, and the dataset yields $c_{z_1}^{(1)} = 0.4$, $c_{z_2}^{(1)} = 0.35$, $c_{z_3}^{(1)} = 0.15$, and $c_{z_4}^{(1)} = 0.1$. Then, (2) becomes

$$\min_{q_z, p_s^{(1)}} \Big[\big(q_{z_1} (p_{s_1}^{(1)} + p_{s_2}^{(1)}) - c_{z_1}^{(1)} \big)^2 + \big(q_{z_2} (p_{s_1}^{(1)} + p_{s_3}^{(1)}) - c_{z_2}^{(1)} \big)^2 + \big(q_{z_3} (p_{s_2}^{(1)} + p_{s_3}^{(1)}) - c_{z_3}^{(1)} \big)^2 + \big(q_{z_4} (p_{s_2}^{(1)} + p_{s_3}^{(1)}) - c_{z_4}^{(1)} \big)^2 \Big].$$

With the addition of the constraints in (3) and (4), this NLP has a unique solution: $p_{s_1} = 0.5$, $p_{s_2} = 0.3$, $p_{s_3} = 0.2$, $q_{z_1} = 0.5$, $q_{z_2} = 0.5$, $q_{z_3} = 0.3$, and $q_{z_4} = 0.2$. Notice that the q variables match the observation probabilities in Example 2.

Complexity. $\texttt{NLP}_{\texttt{main}}$ is a nonconvex bilinear program (see (2)) and thus generally NP-hard (Hillar & Lim, 2013). Section 5.1 presents a convex relaxation that allows polynomial-time approximation.

4.4 PROVING CORRECTNESS OF THE NLP FOR (PAC) IDENTIFICATION

The variable domains and the constraints of $\texttt{NLP}_{\texttt{main}}$ form a compact feasible set; therefore, a global minimum always exists (Bertsekas, 1997). Even though the objective approaches zero as the variables approach the intended probabilities, this alone is not proof of the non-existence of other variable valuations that also minimize the objective. We next show that PAC identification for $\texttt{NLP}_{\texttt{main}}$ is correct even for approximate quantities from (1).

Assumption 4. All states in S are eventually visited by the behavior policy.

Remark 4.4. Assumption 4 is prevalent in the literature, referred to as ergodicity (Douc et al., 2011). If a state is never visited, no guarantees on its emissions can be provided given that no data can be sampled from it. Moreover, we restrict our attention to observations that appear in the dataset. For any unobserved z, we let all our algorithms implicitly set its estimated emission probability to 0 across all states, and it is subsequently ignored. We note that this is done without loss of generality, as, given enough data, only those observations that truly have an emission probability of 0 will be ignored due to Assumption 4.

We start by considering the exact case, where the $c_z^{(t)}$ attain their limit values, $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$. Using Lemma 4.5, Theorem 4.6 shows that $\texttt{NLP}_{\texttt{main}}$ has a unique global optimum, achieved when $q_z = O(z \mid \tilde{s})$ (for some $\tilde{s} \succeq z$). Note that there is no guarantee on the values of individual $p_s^{(t)}$ variables, as our interest is in learning the observation function O. Find the proofs in Appendix C.1.

Lemma 4.5 (Uniqueness)**.** *Suppose that $\forall z \in Z$ there exists $t \leqslant L$ where $c_z^{(t)} > 0$, and suppose that for both $p_z^{(t)}, q_z$ and $\hat{p}^{(t)}, \hat{q}_z$ it holds that (2) evaluates to 0 while satisfying (3) and (4). Then, we have that $\hat{q}_z = q_z$ for all z.*

Theorem 4.6 (Correct in the limit)**.** *Let $c_z^{(t)} = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$ for all $z \in Z$, then, $\tilde{p}_s^{(t)} := \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s)$ and $\tilde{q}_z := O(z \mid \tilde{s})$, for any $\tilde{s}$ where $z \preceq \tilde{s}$, are the unique global optimum of $\texttt{NLP}_{\texttt{main}}$.*

We now turn to the approximate case. Lemma 4.7 implies that, if $\hat{c}_z^{(t)}$ is approximately close to $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$, then the global minimum $\hat{q}_z$ of $\texttt{NLP}_{\texttt{main}}$ is approximately equal to the true observation emission probabilities, $O(z \mid s)$. We use this fact to provide PAC-style guarantees (Valiant, 1984; Ashok et al., 2019) in Theorem 4.8, where we show that for any $\varepsilon, \delta > 0$, no matter how small, we can find a dataset size N such that, solving $\texttt{NLP}_{\texttt{main}}$ using $c_z^{(t)}$ obtained from said dataset by (1) yields a global minimum $\hat{q}_z$ that is at most ε off from the true observation emission probability, $O(z \mid s)$, with a probability of at least $1 - \delta$. Proofs are in Appendix C.2.

Lemma 4.7 (Approximate correctness)**.** *Fix $c_z^{(t)}$ such that for all $z \in Z$ there exists $t \leqslant L$ with $c_z^{(t)} > 0$. Suppose that the global optimum $(p_s^{(t)}, q_z)$ of $\texttt{NLP}_{\texttt{main}}$ with $c_z^{(t)}$ attains objective value 0. Then, for every $\varepsilon > 0$, there exists $\varepsilon' > 0$ such that if $\max_{z \in Z} |\hat{c}_z^{(t)} - c_z^{(t)}| \leqslant \varepsilon'$, any minimizer $(\hat{p}, \hat{q})$ of $\texttt{NLP}_{\texttt{main}}$ with $\hat{c}_z^{(t)}$ satisfies $|q_z - \hat{q}_z| \leqslant \varepsilon$ for all $z \in Z$.*

Theorem 4.8 (PAC identification)**.** *For $\varepsilon, \delta > 0$, there exists a size of the dataset $|\mathcal{D}| = N$, such that, if we fix $c_z^{(t)}$ as in (1), then any global minimum $\hat{p}_s^{(t)}, \hat{q}_z$ of $\texttt{NLP}_{\texttt{main}}$ satisfies that, for all $z \in Z$ and $s \in S$ where $z \preceq s$, $\mathbb{P}_{\mathcal{D}}\left(|O(z \mid s) - \hat{q}_z| \leqslant \varepsilon\right) \geqslant 1 - \delta$.*

5 Toward efficient identification

We convexify $\texttt{NLP}_{\texttt{main}}$ into a disciplined convex program (DCP; Grant et al., 2006) (Section 5.1). Then, we introduce a time-flattened variant of the problem to reduce the number of variables and constraints (Section 5.2), and provide optimizations to handle sparse transition dynamics (Section 5.3).

5.1 A convex relaxation of the NLP

DCP is a framework to construct optimization problems that are guaranteed to be convex and solvable in polynomial time (Grant et al., 2006). It imposes a set of rules for constructing and combining expressions such that any optimization problem that adheres to said rules is DCP compliant and thus inherits said guarantees.

We relax $\texttt{NLP}_{\texttt{main}}$ to obtain a DCP, such that the PAC property from Section 4.4 carries over. To achieve this encoding, we make use of the fact that, as shown in Theorem 4.8, letting $c_z^{(t)}$ as in (1) means that $q_z \sum_{s \in S:\ z \preceq s} p_s^{(t)} \approx c_z^{(t)}$ as the number of samples in the dataset increases, where $p_s^{(t)}$ and q_z are the global minima of $\texttt{NLP}_{\texttt{main}}$.

If $c_z^{(t)} = 0$ for every t, we can simply let $q_z = 0$ as argued in Remark 4.4. Else, for all t where $c_z^{(t)} > 0$, we have $\sum_{s \preceq z} \mathbb{P}(\boldsymbol{s}^{(t)} = s) > 0$, and thus we can express q_z in terms of $p_s^{(t)}$ and $c_z^{(t)}$

$q_z \approx c_z^{(t)} (\sum_{s \leq z} p_s^{(t)})^{-1}$. By substituting it in (4), we obtain a constraint solely on $c_z^{(t)}$ and $p_s^{(t)}$,

$$\sum_{z \leq s} \frac{c_z^{(t)}}{\sum_{z \leq s'} p_{s'}^{(t)}} \approx 1, \quad \forall s \in S. \tag{5}$$

However, a solution to (5) may not exist unless $c_z^{(t)}$ takes its limit value $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$. Thus, as (5) is an approximation rather than a strict equality, we relax it by introducing slack variables $r_s^{(t)} \in \mathbb{R}$ that bound the error in (5) from above, and minimize the squared sum of $r_s^{(t)}$ to find the solution that minimizes said error to obtain a DCP.

Optimization Problem 2 (DCP relaxation for PAC identification, $\texttt{DCP}_{\texttt{main}}$)**.**

$$\min_{p_s^{(t)}, r_s^{(t)}} \sum_{t=1}^{L} \sum_{s \in S} \left(r_s^{(t)}\right)^2, \tag{6}$$

$$\text{s.t.} \sum_{s \in S} p_s^{(t)} = 1, \quad \forall t \in \{1, \ldots, L\} \tag{7}$$

$$\left[\sum_{z \in Z, z \leq s} \frac{c_z^{(t)}}{\sum_{s' \in S, z \leq s} p_{s'}^{(t)}} \right] - 1 \leqslant r_s^{(t)}, \quad \forall t \in \{1, \ldots, L\}, \forall s \in S. \tag{8}$$

We verify that $\texttt{DCP}_{\texttt{main}}$ is DCP compliant, and refer to Appendix D.3 for the proof.

Lemma 5.1. *$\texttt{DCP}_{\texttt{main}}$ meets all requirements for DCP compliance specified in Grant et al. (2006).*

5.2 Flattening the optimization problems across time

We modify $\texttt{NLP}_{\texttt{main}}$ by flattening the problem and averaging the dataset over time. This transformation aims to reduce the number of variables and constraints, thus speeding up the solving time. The idea is to introduce *time-flattened* variables $p_s \in [0, 1]$ to approximate $\mathbb{P}_\pi(\boldsymbol{s}^{(\boldsymbol{\tau})} = s)$ for $s \in S$, where $\boldsymbol{\tau}$ is a uniformly random variable over $\{1, \ldots, L\}$, i.e., $\mathbb{P}(\boldsymbol{\tau} = t) = 1/L$ for $t = 1, \ldots, L$. Similarly, we define flattened $c_z \in [0, 1]$ for estimating $\mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})} = z)$ as follows:

$$c_z := |\mathcal{D}|^{-1} L^{-1} \sum_{t=1}^{L} \sum_{\rho \in \mathcal{D}} \mathbf{1}_{\rho(t)=z}. \tag{9}$$

Variables $q_z \in [0, 1]$ remain the same as in $\texttt{NLP}_{\texttt{main}}$, estimating probability $O(z \mid \tilde{s})$ for any $\tilde{s} \succeq z$, because this quantity is time-independent by definition. Analogous to the reasoning in Section 4.3, we use the fact that, for any $z \in Z$, we have $c_z \approx \mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})} = z) = O(z \mid \tilde{s}) \sum_{s \in S, z \leq s} \mathbb{P}_\pi(\boldsymbol{s}^{(\boldsymbol{\tau})} = s)$, to minimize the distance between $q_z \cdot \sum_{s \in S:\ z \leq s} p_s$ and c_z. We define the following objective

$$\min_{p_s, q_z} \sum_{z \in Z} \left((q_z \sum_{s \in S:\ z \leq s} p_s) - c_z \right)^2,$$

which, together with box-constraints analogous to (3) and (4), provides $\texttt{NLP}_{\texttt{flat}}$. Find the explicit optimization problem in Appendix D.1.

Correctness. Lemma D.1 shows that $c_z \approx \mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})} = z)$ as the size of the dataset increases; the proof is in Appendix D.2. Recall that, by construction, if $c_z = \mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})} = z)$ then $q_z = O(z \mid \tilde{s})$ (for any $\tilde{s} \succeq z$) and $p_s = \mathbb{P}_\pi(\boldsymbol{s}^{(\boldsymbol{\tau})} = s)$ is a global optimum of $\texttt{NLP}_{\texttt{flat}}$. Lemma D.1 is sufficient to carry over the proofs in Theorems 4.6 and 4.8 by removing the dependence on t, due to the near-identical structure of the NLPs. In fact, $\texttt{NLP}_{\texttt{main}}$ is identical to the time-flattened $\texttt{NLP}_{\texttt{flat}}$ when $L = 1$.

Relaxation into DCP, complexity, and approximation error. We adapt the DCP relaxation $\texttt{DCP}_{\texttt{main}}$ of $\texttt{NLP}_{\texttt{main}}$ into a time-flattened variant by ignoring time and computing c_z as in (9), see Appendix D.4. We refer to Appendix D.5 for a detailed technical analysis of the runtime complexity of the flattened program. In summary, the reduced number of variables and constraints in the flattened $\texttt{NLP}_{\texttt{flat}}$ compared with $\texttt{NLP}_{\texttt{main}}$ induces an asymptotic runtime speedup. Similarly, the convexified $\texttt{DCP}_{\texttt{flat}}$ has improved runtime over $\texttt{DCP}_{\texttt{main}}$. Given that c_z is obtained as in (9) by averaging over time steps, information is lost in the process of flattening, and a higher estimation error is expected for finite datasets. Indeed, we observe this phenomenon in our experimental evaluation (Figure 4c).

5.3 Exploiting sparsity of the POMDP

When the transition function is *sparse* (see Definition E.1), only a small fraction of the states can be reached at any given time step. However, $\texttt{NLP}_{\texttt{main}}$ still introduces a variable $p_s^{(t)}$ for every state $s \in S$ and every step t. We use two techniques to exploit this structure. Full details, programs and proofs are in Appendix E.

Pruning. The observation counts already show which states can be reached. If z is unseen in step t, then, given enough data, we can assert that no state that can emit z is reachable in exatcly t steps. Variable $p_s^{(t)}$ can then be set to 0 and *pruned*. We therefore replace S by the set of *visited states*

$$\hat{R}^{(t)} := \{s \in S \mid \exists z \in Z \colon z \preceq s \text{ and } c_z^{(t)} > 0\},$$

which approximates the set of reachable states $R^{(t)} := \{s \in S \mid \mathbb{P}(\boldsymbol{s}^{(t)} = s) > 0\}$. The resulting program, $\texttt{NLP}_{\texttt{pruned}}$, is $\texttt{NLP}_{\texttt{main}}$ with every sum over S restricted to $\hat{R}^{(t)}$ (see Appendix E.1).

Iterative solving. We define $\texttt{NLP}_{\texttt{iterative}}$ by splitting $\texttt{NLP}_{\texttt{main}}$ along the time axis. We solve a sequence of smaller programs $P^{(1)}, \ldots, P^{(L)}$. Program $P^{(t)}$ introduces a new variable q_z only for observations seen for the first time at step t. Every observation seen earlier has its value fixed to the solution of an earlier program. Each program contains only q variables from step t, and p variables over $\hat{R}^{(t)}$. We stop when every observed z has been assigned a value. Find details in Appendix E.2.

Guarantees and complexity. When $\hat{R}^{(t)} = R^{(t)}$, $\texttt{NLP}_{\texttt{pruned}}$ is equivalent to $\texttt{NLP}_{\texttt{main}}$. $\texttt{NLP}_{\texttt{iterative}}$ is correct by induction over t, using Lemma 4.5. With finite data, $\hat{R}^{(t)} = R^{(t)}$ holds with probability approaching 1 as $N \to \infty$, so both variants inherit the PAC guarantees of Theorem 4.8. Both variants admit convex relaxations analogous to Section 5.1. We do not flatten over time as in Section 5.2, because flattening would undo the per-step sparsity. Under both notions of sparsity in Definition E.1, the iterative approach has better asymptotic runtime than $\texttt{NLP}_{\texttt{main}}$ and $\texttt{NLP}_{\texttt{flat}}$. For the convex relaxations, the asymptotic speedup is milder (see Appendix E.4).

6 Empirical analysis

Algorithms. We evaluate three baseline methods for estimating the observation function: a uniformly random observation function, the `AsMAR` algorithm of Wendland et al. (2025), and the Baum-Welch algorithm (Baum et al., 1970). Our approaches are the fundamental $\texttt{NLP}_{\texttt{main}}$ (Section 4.3) with three improvements for scalability: convex relaxation (Section 5.1), time-flattening (Section 5.2), and exploiting sparsity (Section 5.3). Rather than benchmarking every combination, we isolate each improvement incrementally: $\texttt{NLP}_{\texttt{main}}$ vs. $\texttt{NLP}_{\texttt{flat}}$ for flattening, $\texttt{NLP}_{\texttt{flat}}$ vs. $\texttt{DCP}_{\texttt{flat}}$ for convex relaxation, and $\texttt{NLP}_{\texttt{main}}$ vs. $\texttt{NLP}_{\texttt{iterative}}$ for exploiting sparsity. We do not evaluate the remaining combinations ($\texttt{DCP}_{\texttt{main}}$, $\texttt{DCP}_{\texttt{iterative}}$) and the pruned variant of exploiting sparsity ($\texttt{NLP}_{\texttt{pruned}}$, $\texttt{DCP}_{\texttt{pruned}}$). Appendix H.1 provides implementation details.

Setup. Our *benchmarks* are (i) synthetic, randomly generated POMDPs with varying structures and sizes (2) a benchmark inspired by the example in Figure 1; see Appendix H.2 for details. We run the algorithms for various dataset sizes and record the following *metrics:* (1) time for approximating the observation function (2) the mean-squared error (MSE) between the estimate and the ground truth. Details on the full setup are in Appendix H. The anonymized source code is publicly available[1].

Correctness. For synthetically generated SI-POMDPs in Fig. 2a and a benchmark similar to Example 3 in Fig. 2b we observe that our algorithms approximate the ground truth observation functions with increasing dataset size. In contrast, the Baum-Welch baseline achieves a poor approximation error, confirming its lack of identifiability guarantees (Section A). The `AsMAR` baseline's error plateaus as it is limited to a subset of MAR observation functions.

Solving time improvements. As shown on synthetically generated SI-POMDPs, our improvements such as time flattening (Fig. 2d), convexification (Fig. 2e), and exploiting transition sparsity (Fig. 2f) reduce solving time by up to three orders of magnitude. The convex relaxation requires more samples for the same approximation quality, but catches up for acceptable dataset sizes, whereas $\texttt{NLP}_{\texttt{iterative}}$ incurs a substantially larger error (Fig. 2a; Fig. 4c in the appendix). With $\texttt{DCP}_{\texttt{flat}}$, we solve instances with $|S \times A| = 2000$ and $|Z| = 2000$ within ten seconds, confirming practicality.

[1] https://anonymous.4open.science/r/SI-POMDPs

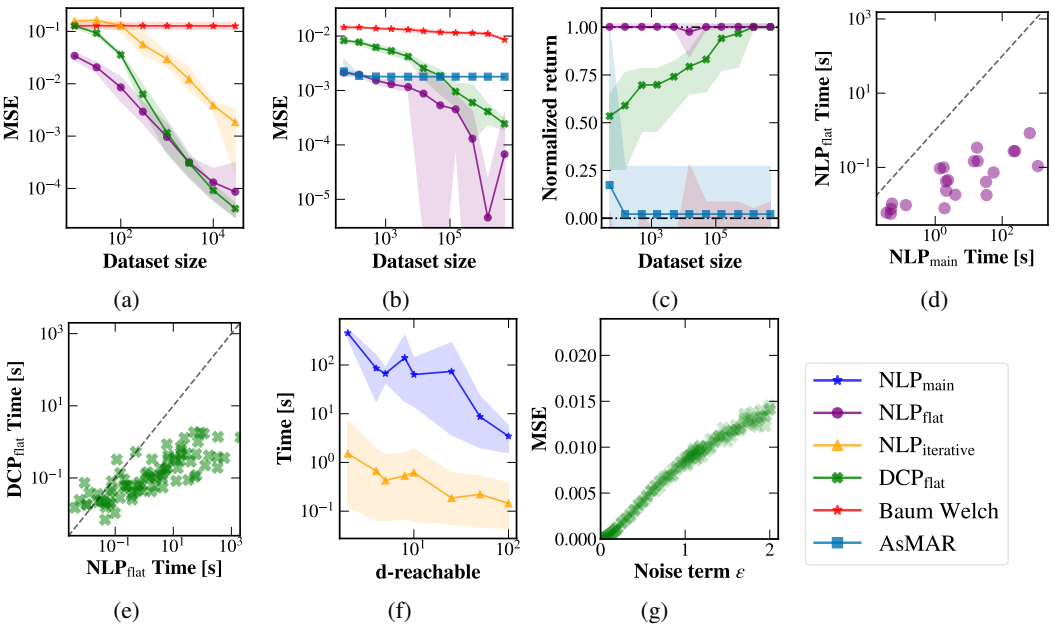


Figure 2: The empirical validation using implementations of the NLP and DCP variations as well as the baselines Baum Welch and AsMAR. **(a),(d)-(g)** show results for synthetically generated POMDPs and **(b),(c)** for a POMDP similar to Example 3/Figure 1. *Lines: mean; shaded area: min–max ranges.*

Policy performance. To empirically verify that identification matters for planning, we (i) estimate the observation function with our approaches and the non-identifying baselines, (ii) compute an ε-optimal policy in each estimated POMDP, and (iii) evaluate it on the original POMDP. Fig. 2c shows the return normalized between the optimal policy (1) and a policy computed from a random observation function (0). Our approaches provide good estimates, whereas `AsMAR` stays near 0 for all dataset sizes, and Baum-Welch does worse than random and is omitted for readability.

Impact of violating the SI assumption. We evaluate the impact of adding uniform noise ε to all probabilities in the observation function (see Appendix H.1 for details) and thereby making it non-SI. Fig. 2g depicts the MSE between the non-SI ground truth and the observation function computed by `DCP_flat` and shows that for small violations, the resulting approximation error remains small.

In conclusion, our evaluation demonstrates that our approaches correctly approximate SI observation functions, thereby allowing to compute optimal policies for such POMDPs even if the observation function is unknown. Further, all efficiency improvements reduce runtime as expected.

7 Conclusions and Future Work

We introduce a novel family of PAC-identifiable POMDPs and propose several learning algorithms formulated as optimization problems. In particular, we present a bilinear nonlinear program and prove its correctness and PAC guarantees. We then relax it into a disciplined convex program, enabling polynomial-time solutions. We further improve performance via time-flattened, pruned, and iterative reformulations and discuss alternative settings. Our empirical results support the theoretical claims.

Limitations and future work. Our current approach relies on generic solvers, leaving room for improvement with custom solvers, which defer to future work. Another direction is to handle non-Markovian behavior policies with compact memory representations (e.g., finite-state controllers (Meuleau et al., 1999)) while avoiding the exponential blowup of general history-based policies. Future work may also explore generalizations of the state-invariance criterion introduced here. For example, one could consider an approximate notion where $z \preceq s, s'$ implies $|O(z \mid s) - O(z \mid s')|$ is bounded rather than zero. Moreover, the current admittance relation $\preceq \in \{0,1\}^{Z \times S}$ could be extended to richer structures such as $\preceq \in \mathbb{N}^{Z \times S}$ or $\preceq \in \mathbb{R}^{Z \times S}$.

AI USE STATEMENT

We used LLMs to assist with editing the manuscript and writing code for the empirical analysis. The research idea, the problem formulation as a nonlinear program, and all derived results, including the reformulation as a disciplined convex program, were developed solely by the human brains of the authors. All AI-assisted code and AI-assisted edits of the human-written text was reviewed by the authors. This means that no text was written by an LLM from scratch. We, the authors, take full responsibility for the content in this paper, including text, claims, and artifacts produced with the help of generative AI.

REPRODUCIBILITY STATEMENT

All code required for conducting experiments is included in our repository (see Section 6). The final hyperparameters used for each model and algorithm are given in Table 1 in Appendix H and are also provided in the repository (see our `README.md`). The computing infrastructure in which the experiments have been carried out is described in Appendix H. All datasets were created via random policies interacting with POMDPs and can be recreated with the code in our repository. All assumptions are provided in Assumptions 1– 4, and the theoretical problem setting is described in Section 4.1. Complete proofs of all theoretical results are provided in Appendices C–G.

REFERENCES

Naoki Abe and Manfred K Warmuth. On the computational complexity of approximating distributions by probabilistic automata. *Machine learning*, 9(2):205–260, 1992.

Elizabeth S Allman, Catherine Matias, and John A Rhodes. Identifiability of parameters in latent structure models with many observed variables. *Ann. Statist.*, 37(6A):3099–3132, 2009. doi: 10.1214/09-AOS689.

Roman Andriushchenko, Alexander Bork, Milan Češka, Sebastian Junges, Joost-Pieter Katoen, and Filip Macák. Search and explore: symbiotic policy synthesis in pomdps. *Formal Methods in System Design*, 68(1):4, 2026.

Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

Pranav Ashok, Jan Kretínský, and Maximilian Weininger. PAC statistical model checking for markov decision processes and stochastic games. In *CAV (1)*, volume 11561 of *Lecture Notes in Computer Science*, pp. 497–519. Springer, 2019.

Karl Johan Åström. Optimal control of markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965.

Giovanni Bacci, Anna Ingólfsdóttir, Kim G Larsen, and Raphaël Reynouard. Active learning of markov decision processes using baum-welch algorithm. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 1203–1208. IEEE, 2021.

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, 41(1):164–171, 1970.

Logan Beal, Daniel Hill, R Martin, and John Hedengren. Gekko optimization suite. *Processes*, 6(8): 106, 2018. doi: 10.3390/pr6080106.

Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3): 334–334, 1997.

Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

Carlos E. Budde, Arnd Hartmanns, Tobias Meggendorfer, Maximilian Weininger, and Patrick Wienhöft. Sound statistical model checking for probabilities and expected rewards. In *TACAS (1)*, volume 15696, pp. 167–190. Springer, 2025.

John Canny. Some algebraic and geometric computations in pspace. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pp. 460–467, 1988.

Krishnendu Chatterjee, Martin Chmelík, Raghav Gupta, and Ayush Kanodia. Optimal cost almost-sure reachability in POMDPs. 234:26–48, 2016.

Stephen A Cook. Towards a complexity theory of synchronous parallel computation. In *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*, pp. 219–244. 2023.

Benoit Courty, Victor Schmidt, Sasha Luccioni, Goyal-Kamal, MarionCoutarel, Boris Feld, Jérémy Lecourt, LiamConnell, Amine Saboni, Inimaz, supatomic, Mathilde Léval, Luis Blanche, Alexis Cruveiller, ouminasara, Franklin Zhao, Aditya Joshi, Alexis Bogroff, de Lavoreille, Hugues, Niko Laskaris, Edoardo Abati, Douglas Blank, Ziyao Wang, Armin Catovic, Marc Alencon, Michal Stechly, Christian Bauer, de Araújo, Lucas Otávio N., JPW, and MinervaBooks. CodeCarbon: v2.4.1, May 2024.

George B Dantzig. Linear programming. *Operations research*, 50(1):42–47, 2002.

Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester. *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

Josée Desharnais. Labelled markov processes. 1999.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Randal Douc, Eric Moulines, Jimmy Olsson, and Ramon Van Handel. Consistency of the maximum likelihood estimator for general hidden markov models. 2011.

Maxim Egorov, Zachary N. Sunberg, Edward Balaban, Tim A. Wheeler, Jayesh K. Gupta, and Mykel J. Kochenderfer. POMDPs.jl: A framework for sequential decision making under uncertainty. *Journal of Machine Learning Research*, 18(26):1–5, 2017.

Eyal Even-Dar, Sham M. Kakade, and Yishay Mansour. Reinforcement learning in pomdps without resets. In *IJCAI*, pp. 690–695. Professional Book Center, 2005.

Lorenzo Finesso. *Consistent estimation of the order for Markov and hidden Markov chains*. University of Maryland, College Park, 1990.

Edgar J Gilbert. *On the identifiability problem for functions of finite Markov chains*. Sandia Corporation, 1959.

Richard D Gill and Peter D Grünwald. An algorithmic and a geometric characterization of coarsening at random. 2008.

Noah Golowich, Ankur Moitra, and Dhruv Rohatgi. Learning in observable pomdps, without computationally intractable oracles. *Advances in neural information processing systems*, 35: 1458–1473, 2022.

Michael Grant, Stephen Boyd, and Yinyu Ye. Disciplined convex programming. In *Global optimization: From theory to implementation*, pp. 155–210. Springer, 2006.

Zhaohan Daniel Guo, Shayan Doroudi, and Emma Brunskill. A pac rl algorithm for episodic pomdps. In *Artificial Intelligence and Statistics*, pp. 510–518. PMLR, 2016.

Daniel F Heitjan and Donald B Rubin. Ignorability and coarse data. *The annals of statistics*, pp. 2244–2253, 1991.

Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):1–39, 2013.

Michelle Ho, Arec Jamgochian, and Mykel J Kochenderfer. Model identification adaptive control with ρ-pomdp planning. In *2025 11th International Conference on Control, Decision and Information Technologies (CoDIT)*, volume 1, pp. 1981–1986. IEEE, 2025.

Daniel J. Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *J. Comput. Syst. Sci.*, 78(5):1460–1480, 2012.

Mehdi Jafarnia-Jahromi, Rahul Jain, and Ashutosh Nayyar. Online learning for unknown partially observable MDPs. In *AISTATS*, Proceedings of Machine Learning Research, pp. 1712–1732. PMLR, 2022.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11:1563–1600, 2010.

Sammie Katt, Frans A Oliehoek, and Christopher Amato. Learning in pomdps with monte carlo tree search. In *International Conference on Machine Learning*, pp. 1819–1827. PMLR, 2017.

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *NIPS*, pp. 1840–1848, 2016a.

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *NIPS*, pp. 1840–1848, 2016b.

Hanna Kurniawati, David Hsu, Wee Sun Lee, et al. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Robotics: Science and systems*, volume 2008. Zurich, Switzerland, 2008.

Jeongyeol Kwon, Yonathan Efroni, Constantine Caramanis, and Shie Mannor. RL for latent MDPs: Regret guarantees and a lower bound. In *NeurIPS*, pp. 24523–24534, 2021.

Jeongyeol Kwon, Yonathan Efroni, Shie Mannor, and Constantine Caramanis. Prospective side information for latent mdps. In *ICML*, Proceedings of Machine Learning Research, pp. 25755–25783. PMLR / OpenReview.net, 2024.

Jonathan Lee, Alekh Agarwal, Christoph Dann, and Tong Zhang. Learning in pomdps is sample-efficient with hindsight observability. In *ICML*, Proceedings of Machine Learning Research, pp. 18733–18773. PMLR, 2023.

Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2019.

Qinghua Liu, Alan Chung, Csaba Szepesvári, and Chi Jin. When is partially observable reinforcement learning not scary? In *COLT*, Proceedings of Machine Learning Research, pp. 5175–5220. PMLR, 2022.

Ying Liu, Steven Andrew Culpepper, and Yuguo Chen. Identifiability of hidden markov models for learning trajectories in cognitive diagnosis. *psychometrika*, 88(2):361–386, 2023.

Hua Mao, Yingke Chen, Manfred Jaeger, Thomas D Nielsen, Kim G Larsen, and Brian Nielsen. Learning markov decision processes for model checking. *arXiv preprint arXiv:1212.3873*, 2012.

Nicolas Meuleau, Leonid Peshkin, Kee-Eung Kim, and Leslie Pack Kaelbling. Learning finite-state controllers for partially observable environments. In *UAI*, pp. 427–436. Morgan Kaufmann, 1999.

Karthika Mohan and Judea Pearl. Graphical Models for Processing Missing Data. *Journal of the American Statistical Association*, 116(534):1023–1037, 2021.

Elchanan Mossel and Sébastien Roch. Learning nonsingular phylogenies and hidden markov models. In *STOC*, pp. 366–375. ACM, 2005.

Masashi Okamoto. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the Institute of Statistical Mathematics*, 10(1):29–35, 1959.

Judea Pearl and Michael Tarsi. Structuring causal trees. *Journal of Complexity*, 2(1):60–77, 1986.

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 2002.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

Trey Smith and Reid Simmons. Heuristic search value iteration for pomdps. *arXiv preprint arXiv:1207.4166*, 2012.

Mike Steel, Michael D Hendy, and David Penny. Reconstructing phylogenies from nucleotide pattern probabilities: a survey and some new results. *Discrete Applied Mathematics*, 88(1-3):367–396, 1998.

Andreas Stolcke and Stephen Omohundro. Hidden markov model induction by bayesian model merging. *Advances in neural information processing systems*, 5, 1992.

Alexander L Strehl and Michael L Littman. An empirical evaluation of interval estimation for markov decision processes. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pp. 128–135. IEEE, 2004.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Martin Tappler, Bernhard K Aichernig, Giovanni Bacci, Maria Eichlseder, and Kim G Larsen. L^*-based learning of Markov decision processes. In *International Symposium on Formal Methods*, pp. 651–669. Springer, 2019.

Sebastiaan Terwijn. On the learnability of hidden markov models. In *ICGI*, volume 2484 of *Lecture Notes in Computer Science*, pp. 261–268. Springer, 2002.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. Intelligent robotics and autonomous agents. MIT Press, 2005.

Paul Tune, Hung X Nguyen, and Matthew Roughan. Hidden markov model identifiability via tensors. In *2013 IEEE International Symposium on Information Theory*, pp. 2299–2303. IEEE, 2013.

Masatoshi Uehara, Ayush Sekhari, Jason D. Lee, Nathan Kallus, and Wen Sun. Computationally efficient PAC RL in pomdps with latent determinism and conditional embeddings. In *ICML*, Proceedings of Machine Learning Research, pp. 34615–34641. PMLR, 2023.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Joshua Wendland, Markel Zubia, Roman Andriushchenko, Maris F L Galesloot, Milan Ceska, Henrik von Kleist, Thiago D Simão, Maximilian Weininger, and Nils Jansen. Missingness-MDPs: Bridging the theory of missing data and POMDPs. In *Eighteenth European Workshop on Reinforcement Learning*, 2025.

Marco Alexander Wiering. *Explorations in efficient reinforcement learning*. PhD thesis, University of Amsterdam, 1999.

Yi Xiong, Ningyuan Chen, Xuefeng Gao, and Xiang Zhou. Sublinear regret for learning pomdps. *Production and Operations Management*, 31(9):3491–3504, 2022.

A IDENTIFICATION IS NECESSARY FOR POMDP PLANNING

This section justifies the need for identification of the observation function for optimal policy synthesis in POMDPs. We consider the hardest setting to make this argument: the transition dynamics and behavior policy are known, and the reward is observation-based. The argument becomes easier if the transition dynamics and behavior policy are unknown, as the model becomes more ambiguous and hence even harder to identify. Observation-based rewards make it harder to justify the need for identifiability in contrast with the more standard state-based rewards, because if another observation also fits the data, then observation-based reward functions will yield the same reward distributions. As will be shown later, the key is that the same reward distributions are generated only under the behavior policy (the policy that generated the data), and may thus disagree on the optimal policy. We note that in any of the cases mentioned above, state invariance always ensures the identifiability of the observation function.

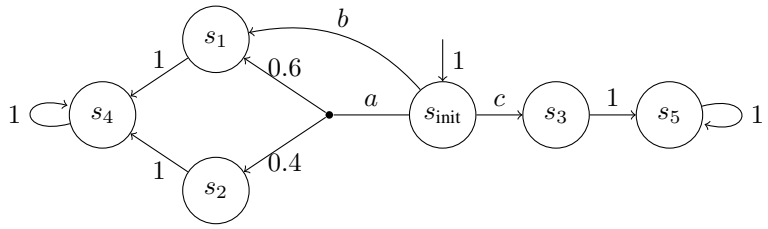


Figure 3: The transition function of a tiny POMDP where identification is necessary for optimal policy synthesis.

Consider a POMDP $\mathcal{M} = (S, A, T, Z, O, b_{\text{init}})$ with states $S = \{s_{\text{init}}, s_1, s_2, s_3, s_4, s_5\}$, observations $Z = \{z_{\text{init}}, z_1, z_2, z_3, z_4, z_5\}$, and actions $A = \{a, b, c\}$. The transition dynamics, T and b_{init}, are given graphically in Figure 3. The observation probabilities are $O(z_{\text{init}} \mid s_{\text{init}}) = 1$, $O(z_1 \mid s_1) = O(z_1 \mid s_2) = 0.2$, $O(z_2 \mid s_1) = O(z_2 \mid s_2) = 0.8$, $O(z_3 \mid s_3) = 1$, $O(z_4 \mid s_4) = 1$, and $O(z_5 \mid s_5) = 1$. Notice O is SI with $z_{\text{init}} \preceq s_{\text{init}}$, $z_1, z_2 \preceq s_1, s_2$, $z_3 \preceq s_3$, $z_4 \preceq s_4$, and $z_5 \preceq s_5$ (with $z \not\preceq s$ for any other state-observation pair). Define an observation-based reward function $R(z_1) = 1$, $R(z_3) = 0.3$, all other rewards set to 0, and let the objective be the undiscounted infinite-horizon mean payoff of said function:

$$J_O(\pi) := \lim_{t\to\infty} \frac{1}{t} \sum_{t'=1}^{t} \mathbb{E}_{\boldsymbol{z}^{(t)}} \left[R(\boldsymbol{z}^{(t)}) \right]$$

Next, define the (known) behavior policy $\pi_{\text{beh}}(a \mid z_{\text{init}}) = 0.5$, $\pi_{\text{beh}}(c \mid z_{\text{init}}) = 0.5$. We can construct a new (non-SI) observation function, $\hat{O}$, that induces the same observation probabilities under π_{beh}. Due to the transition topology, if we do not enforce SI, the only equations $\hat{O}$ needs to satisfy are

$$\begin{aligned}
\mathbb{P}_{\hat{O}}(z_{\text{init}}^{(1)}) &= 1, \\
\mathbb{P}_{\hat{O}}(z_1^{(2)} \mid \boldsymbol{a}^{(1)} = a) &= 0.6\hat{O}(z_1 \mid s_1) + 0.4\hat{O}(z_1 \mid s_2) = 0.2, \\
\mathbb{P}_{\hat{O}}(z_2^{(2)} \mid \boldsymbol{a}^{(1)} = a) &= 0.6\hat{O}(z_2 \mid s_1) + 0.4\hat{O}(z_2 \mid s_2) = 0.8, \\
\mathbb{P}_{\hat{O}}(z_3^{(2)} \mid \boldsymbol{a}^{(1)} = c) &= \hat{O}(z_3 \mid s_3) = 1, \\
\mathbb{P}_{\hat{O}}(z_4^{(3)} \mid \boldsymbol{a}^{(1)} = a) &= \hat{O}(z_4 \mid s_4) = 1, \\
\mathbb{P}_{\hat{O}}(z_5^{(3)} \mid \boldsymbol{a}^{(1)} = c) &= \hat{O}(z_5 \mid s_5) = 1.
\end{aligned}$$

These equations force $\hat{O}(z_{\text{init}} \mid s_{\text{init}}) = 1$, $\hat{O}(z_3 \mid s_3) = 1$, $\hat{O}(z_4 \mid s_4) = 1$, and $\hat{O}(z_5 \mid s_5) = 1$. However, there is some slack on $(\hat{O}(z_1 \mid s_1), \hat{O}(z_1 \mid s_2)) =: (v_1, v_2)$; they lay in the following line segment

$$\{x, y \in [0, 1] \mid 0.6x + 0.4y = 0.2\}, \tag{10}$$

For general POMDPs, the respective set is a simplex. Fixing v_1, v_2 yields $\hat{O}(z_2 \mid s_1) = 1 - v_1$ and $\hat{O}(z_2 \mid s_2) = 1 - v_2$.

Any $\hat{O}$ obtained from picking a point in (10) yields an observation function with an identical distribution on observation histories to the original O under π_{beh}. Therefore, if we sample a dataset from the original model using π_{beh}, as the number of samples tends to infinity, any $\hat{O}$ in the line segment (including O itself) will fit the data arbitrarily well. We thus talk about $\hat{O}$ that lay on (10).

We introduce parameter $\alpha \in [0, 1]$ and define the line segment in (10) in terms of said parameter

$$\hat{O}(z_1 \mid s_1) = \alpha/3; \quad \hat{O}(z_1 \mid s_2) = (1 - \alpha)/2; \quad \alpha \in [0, 1]. \tag{11}$$

For instance, we can get $\hat{O}(z_1 \mid s_1) = 1/3$ from (11) by letting $\alpha = 1$, which gives us $\hat{O}(z_1 \mid s_2) = 0$.

From now on, we fix $\alpha = 1$ and its corresponding $\hat{O}$. Both O and $\hat{O}$ agree on the distribution of observation histories induced by π_{beh}, and thus they both assign it the same objective value

$J_O(\pi_{\text{beh}}) = J_O(\pi_{\text{beh}}) = 0.5 \cdot 0.2 + 0.5 \cdot 0.3 = 0.25$. However, they disagree on the valuations of other policies. In the original or ground-truth POMDP $\mathcal{M}$, it is easy to see that $\pi^*(c \mid z_{\text{init}}) = 1$ is optimal with a mean payoff of 0.3. By contrast, policy $\bar{\pi}(b \mid z_{\text{init}}) = 1$ is optimal with a mean payoff of $1/3$ if we use $\hat{O}$, but is suboptimal in the ground-truth model with a mean payoff of only 0.2.

Takeaway. If we perform expectation maximization, then line (10) forms the global maximum: function $\hat{O}$ maximizes expectation if and only if it lays on (10). Thus, even if expectation maximization (say, with Baum-Welch) manages to reach the global optimum, it may output any $\hat{O}$ laying in this continuum, which suggests that the algorithm's output is not a consistent estimator, in that, estimation error may not converge to zero as more data is provided. By assuming state-invariance, the simplex collapses into a single point, allowing the identification of O.

Finally, note that our experiments in Section 6 also demonstrate the suboptimal performance when failing to identify the ground-truth observation function.

B SI STRICTLY GENERALIZES MAR

Definition B.1 (Rubin (1976))**.** Let random variables $X = (X_1, \ldots, X_n) \sim \text{Distr}(\times_{i=1}^n \mathcal{X}_i)$ represent the underlying values of data, and $Y = (Y_1, \ldots, Y_n) \sim \text{Distr}(\times_{i=1}^n (\mathcal{X}_i \cup \bot))$ represent possibly missing data, that is, either $Y_i = X_i$ or $Y_i = \bot$ ("missing"). For convenience, also define sets $\mathcal{X} := \times_{i=1}^n \mathcal{X}_i$ and $\mathcal{Y} := \times_{i=1}^n (\mathcal{X}_i \cup \{\bot\})$. Then, X and Y are *missing at random* (MAR) if and only if for any underlying value vectors $x, x' \in \mathcal{X}$ and observation vector $y \in \mathcal{Y}$, it holds that if $x_i = x'_i$ for all i that $y_i = \bot$, then $\mathbb{P}(Y = y \mid X = x) = \mathbb{P}(Y = y \mid X = x')$.

Lemma B.2 (SI strictly generalizes MAR.)**.** *Let X and Y be MAR as defined in Definition B.1, with corresponding $\mathcal{X}$ and $\mathcal{Y}$, and let $O : \mathcal{X} \to \text{Distr}(\mathcal{Y})$ such that $\mathbb{P}(Y = y \mid X = x) = O(y \mid x)$. Then, O is SI. Moreover, not every SI O is MAR.*

Proof. We simply need to spell it out and the proof follows directly.

Suppose that X and Y are MAR, and define the following relation:

$$\preceq := \{(y, x) \in \mathcal{Y} \times \mathcal{X} \mid y_i = x_i \text{ or } y_i = \bot, \forall i = 1, \ldots, n\}.$$

Then, we immediately get the two conditions for SI:

- If $y \not\preceq x$, then for some i we have $y_i \neq x_i$ and $y_i \neq \bot$. Definition B.1 imposes that $Y_i = X_i$ or $Y_i = \bot$, so $\mathbb{P}(Y = y \mid X = x) = 0 = O(y \mid x)$.
- If $y \preceq x, x'$, then $\mathbb{P}(Y = y \mid X = x) = \mathbb{P}(Y = y \mid X = x')$ by Definition B.1, so $O(y \mid x) = O(y \mid x')$.

Lastly, any SI observation function without a missing data structure—like those in Figure 1 b) or in Example 2—is not MAR, so SI is a strict generalization of MAR. □

C PROOFS OF CORRECTNESS FROM SECTION 4.4

C.1 PROOFS OF EXACT CORRECTNESS

Lemma 4.5 (Uniqueness)**.** *Suppose that $\forall z \in Z$ there exists $t \leqslant L$ where $c_z^{(t)} > 0$, and suppose that for both $p_z^{(t)}, q_z$ and $\hat{p}^{(t)}, \hat{q}_z$ it holds that (2) evaluates to 0 while satisfying (3) and (4). Then, we have that $\hat{q}_z = q_z$ for all z.*

Proof. Since (2) evaluates to 0 for both p, q and $\hat{p}, \hat{q}$, we have that, $\forall z \in Z$ and every $t = 1, \ldots L$,

$$q_z \sum_{s \in S \colon z \preceq s} p_s^{(t)} = c_z^{(t)} \quad \text{and} \quad \hat{q}_z \sum_{s \in S \colon z \preceq s} \hat{p}_s^{(t)} = c_z^{(t)}. \tag{12}$$

By (4), for any $s \in S$, we know that $\sum_{z \in Z \colon z \preceq s} q_s = \sum_{z \in Z \colon z \preceq s} \hat{q}_s = 1$. Hence,

$$\sum_{z \in Z \colon z \preceq s} (q_z - \hat{q}_z) = 0,$$

and, for any given $t = 1, \dots L$, multiplying by $p_s^{(t)}$ yields

$$\sum_{s \in S} \left[p_s^{(t)} \left(\sum_{z \in Z \colon z \preceq s} (q_z - \hat{q}_z) \right) \right] = 0.$$

Given that these are finite sums, we can swap the summation order:

$$0 = \sum_{s \in S} \left[p_s^{(t)} \left(\sum_{z \in Z \colon z \preceq s} (q_z - \hat{q}_z) \right) \right] = \sum_{s \in S, z \in Z \colon z \preceq s} (q_z - \hat{q}_z) \cdot p_s^{(t)}$$
$$= \sum_{z \in Z} \left[(q_z - \hat{q}_z) \sum_{s \in S \colon z \preceq s} p_s^{(t)} \right].$$

We then do the same with $\hat{p}^{(t)}$ to get

$$\sum_{z \in Z} \left[(q_z - \hat{q}_z) \sum_{s \in S \colon z \preceq s} p_s^{(t)} \right] = 0 = \sum_{z \in Z} \left[(q_z - \hat{q}_z) \sum_{s \in S \colon z \preceq s} \hat{p}_s^{(t)} \right],$$

and, subtracting the two,

$$\sum_{z \in Z} \left[(q_z - \hat{q}_z) \left(\sum_{s \in S \colon z \preceq s} p_s^{(t)} - \sum_{s \in S \colon z \preceq s} \hat{p}_z^{(t)} \right) \right] = 0. \tag{13}$$

Picking t such that $c_z^{(t)} > 0$, we get that $\sum_{s \in S \colon z \preceq s} p_s^{(t)} > 0$, $\sum_{s \in S \colon z \preceq s} \hat{p}_s^{(t)} > 0$, and $q_z, \hat{q}_z > 0$ by (12), which allows us to rewrite

$$\left(\sum_{s \in S \colon z \preceq s} p_s^{(t)} - \sum_{s \in S \colon z \preceq s} \hat{p}_z^{(t)} \right) = c_z^{(t)} \left(\frac{1}{q_z} - \frac{1}{\hat{q}_z} \right) = c_z^{(t)} \left(\frac{q_z - \hat{q}_z}{q_z \hat{q}_z} \right). \tag{14}$$

Rewriting (14) into (13), we get the identity

$$\sum_{z \in Z} c_z^{(t)} \frac{(q_z - \hat{q}_z)^2}{q_z \hat{q}_z} = 0. \tag{15}$$

Since $c_z^{(t)} > 0$, we have $q_z = \hat{q}_z$ for all $z \in Z$. □

Theorem 4.6 (Correct in the limit)**.** *Let $c_z^{(t)} = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$ for all $z \in Z$, then, $\tilde{p}_s^{(t)} := \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s)$ and $\tilde{q}_z := O(z \mid \tilde{s})$, for any $\tilde{s}$ where $z \preceq \tilde{s}$, are the unique global optimum of* $\texttt{NLP}_{\texttt{main}}$.

Proof. Both constraints, (3) and (4) are immediately satisfied, as $\tilde{p}$ and $\tilde{q}$ are probabilities.

Fix $z \in Z$, and take any $s' \in S$ where $z \preceq s'$. Then, for $t = 1, \dots, L$,

$$\begin{aligned}
\tilde{q}_z \cdot \sum_{s \in S \colon z \preceq s} \tilde{p}_s^{(t)} &= O(z \mid s') \sum_{s \in S \colon z \preceq s} \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \\
&= O(z \mid s') \left(\sum_{s \in S \colon z \preceq s} \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) + 0 \right) \\
&=^{(\text{def. } \preceq)} O(z \mid s') \left(\sum_{s \in S \colon z \preceq s} \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) + \sum_{s \in S, z \not\preceq s} \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \right) \\
&= O(z \mid s') \sum_{s \in S} \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \\
&= \sum_{s \in S} O(z \mid s') \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \\
&=^{(\text{SI})} \sum_{s \in S} O(z \mid s) \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \\
&= \sum_{s \in S} \mathbb{P}(\boldsymbol{z}^{(t)} = z \mid \boldsymbol{s}^{(t)} = s) \mathbb{P}_\pi(\boldsymbol{s}^{(t)} = s) \\
&= \mathbb{P}(\boldsymbol{z}^{(t)} = z) \\
&= c_z^{(t)}.
\end{aligned}$$

Therefore, (2) evaluates to 0, and it is thus the unique global minimum of $\texttt{NLP}_{\texttt{main}}$ by Lemma 4.5. □

C.2 Proofs of approximate correctness

Lemma 4.7 (Approximate correctness). *Fix $c_z^{(t)}$ such that for all $z \in Z$ there exists $t \leqslant L$ with $c_z^{(t)} > 0$. Suppose that the global optimum $(p_s^{(t)}, q_z)$ of $\texttt{NLP}_{\texttt{main}}$ with $c_z^{(t)}$ attains objective value 0. Then, for every $\varepsilon > 0$, there exists $\varepsilon' > 0$ such that if $\max_{z\in Z} |\hat{c}_z^{(t)} - c_z^{(t)}| \leqslant \varepsilon'$, any minimizer $(\hat{p}, \hat{q})$ of $\texttt{NLP}_{\texttt{main}}$ with $\hat{c}_z^{(t)}$ satisfies $|q_z - \hat{q}_z| \leqslant \varepsilon$ for all $z \in Z$.*

Proof. Fix $\varepsilon > 0$ and let $\max_{z\in Z} |\hat{c}_z^{(t)} - c_z^{(t)}| \leqslant \varepsilon'$. The goal is to construct a bound on ε' with respect to ε that ensures $|q_z - \hat{q}_z| \leqslant \varepsilon$. Since (p, q) yields 0 in the objective of $\texttt{NLP}_{\texttt{main}}$ with parameters $c_z^{(t)}$, we have

$$q_z \sum_{s\in S:\ z\preceq s} p_s^{(t)} = c_z^{(t)} \tag{16}$$

for $t = 1, \ldots, L$, thus

$$\sum_{z\in Z} \left(q_z \sum_{s\in S:\ z\preceq s} p_s^{(t)} - \hat{c}_z^{(t)} \right)^2 = \sum_{z\in Z} (c_z^{(t)} - \hat{c}_z^{(t)})^2 \leqslant |Z|(\varepsilon')^2.$$

Since (p, q) yield a value $\leqslant |Z|(\varepsilon')^2$ in the objective, any global minimizer $(\hat{p}, \hat{q})$ of $\texttt{NLP}_{\texttt{main}}$ with coefficients $\hat{c}_z^{(t)}$ satisfies

$$\sum_{z\in Z} \left(\hat{q}_z \sum_{s\in S:\ z\preceq s} \hat{p}_s^{(t)} - \hat{c}_z^{(t)} \right)^2 \leqslant |Z|(\varepsilon')^2. \tag{17}$$

We now define helper variables to bound the error for each summand:

$$e_z := \hat{q}_z \sum_{s\in S:\ z\preceq s} \hat{p}_s^{(t)} - \hat{c}_z^{(t)}. \tag{18}$$

Thus, by (17), we have:

$$\sum_{z\in Z} e_z^2 \leqslant |Z|(\varepsilon')^2. \tag{19}$$

As in the proof of Lemma 4.5, by (4), we have the identity (refer to (13) for its derivation)

$$\sum_{z\in Z} (q_z - \hat{q}_z) \left(\sum_{s\in S:\ z\preceq s} p_s^{(t)} - \sum_{s\in S:\ z\preceq s} \hat{p}_s^{(t)} \right) = 0. \tag{20}$$

By (16) and (18), we have

$$\sum_{s\in S:\ z\preceq s} p_s^{(t)} = \frac{c_z^{(t)}}{q_z} \quad \text{and} \quad \sum_{s\in S:\ z\preceq s} \hat{p}_s^{(t)} = \frac{\hat{c}_z^{(t)} + e_z}{\hat{q}_z},$$

which we substitute into (20) to get

$$\sum_{z\in Z} (q_z - \hat{q}_z) \left(\frac{c_z^{(t)}}{q_z} - \frac{\hat{c}_z^{(t)} + e_z}{\hat{q}_z} \right) = 0. \tag{21}$$

Aiming for an identity similar to that in (15), we expand this expression as follows:

$$\begin{aligned}
\frac{c_z^{(t)}}{q_z} - \frac{\hat{c}_z^{(t)} + e_z}{\hat{q}_z} &= \frac{c_z^{(t)}\hat{q}_z - \hat{c}_z^{(t)} q_z + e_z q_z}{q_z \hat{q}_z} \\
&= \frac{c_z^{(t)}\hat{q}_z - \hat{c}_z^{(t)} q_z + e_z q_z + (c_z^{(t)} q_z - c_z^{(t)} q_z)}{q_z \hat{q}_z} \\
&= \frac{c_z^{(t)}(\hat{q}_z - q_z) - (c_z^{(t)} - \hat{c}_z^{(t)}) q_z + e_z q_z}{q_z \hat{q}_z} \\
&= c_z^{(t)} \frac{\hat{q}_z - q_z}{q_z \hat{q}_z} + \frac{c_z^{(t)} - \hat{c}_z^{(t)} - e_z}{\hat{q}_z}.
\end{aligned}$$

Substituting the expression back into (21),

$$\sum_{z\in Z} c_z^{(t)} \frac{(q_z - \hat{q}_z)^2}{q_z \hat{q}_z} - (q_z - \hat{q}_z) \frac{c_z^{(t)} - \hat{c}_z^{(t)} - e_z}{\hat{q}_z} = 0$$

Notice that, if $(q_z - \hat{q}_z)(c_z^{(t)} - \hat{c}_z^{(t)} - e_z)/\hat{q}_z = 0$ in the equation above, we precisely get (15) from the proof of the exact case (Lemma 4.5). Instead, since we are in the approximate case, this term may take a small but non-zero value, since both $|c_z^{(t)} - \hat{c}_z^{(t)}|$ and $|e_z|$ are small. The next part of this proof focuses on obtaining a bound on $|q_z - \hat{q}_z|$ using this insight.

We start by moving

$$\sum_{z\in Z} c_z^{(t)} \frac{(q_z - \hat{q}_z)^2}{q_z \hat{q}_z} = \sum_{z\in Z} (q_z - \hat{q}_z) \frac{c_z^{(t)} - \hat{c}_z^{(t)} - e_z}{\hat{q}_z},$$

and, multiplying both sides by $\hat{q}_z$,

$$\sum_{z\in Z} c_z^{(t)} \frac{(q_z - \hat{q}_z)^2}{q_z} = \sum_{z\in Z} (q_z - \hat{q}_z)(c_z^{(t)} - \hat{c}_z^{(t)} - e_z). \tag{22}$$

We now upper bound the the right-hand side of (22) by taking its absolute value:

$$\sum_{z\in Z} (q_z - \hat{q}_z)(c_z^{(t)} - \hat{c}_z^{(t)} - e_z) \leqslant \sum_{z\in Z} |q_z - \hat{q}_z| \cdot (|c_z^{(t)} - \hat{c}_z^{(t)}| + |e_z|). \tag{23}$$

Using the Cauchy-Schwarz inequality and the bound from (19),

$$\sum_{z\in Z} |e_z| \leqslant \sqrt{|Z|} \sqrt{\sum_{z\in Z} e_z^2} \leqslant |Z| \varepsilon'.$$

Plugging this bound and the assumption that $|c_z^{(t)} - \hat{c}_z^{(t)}| \leqslant \varepsilon'$ into (23), we get

$$\sum_{z\in Z} (q_z - \hat{q}_z)(c_z^{(t)} - \hat{c}_z^{(t)} - e_z) \leqslant (1 + |Z|)\, \varepsilon' \sum_{z\in Z} |q_z - \hat{q}_z|, \tag{24}$$

Combining it back with (22),

$$\sum_{z\in Z} \frac{c_z^{(t)} (q_z - \hat{q}_z)^2}{q_z} \leqslant (1 + |Z|)\, \varepsilon' \sum_{z\in Z} |q_z - \hat{q}_z|,$$

which we simplify by letting $\check{Q} := \min_{z\in Z} \frac{c_z^{(t)}}{q_z}$ in

$$\check{Q} \sum_{z\in Z} (q_z - \hat{q}_z)^2 \leqslant (1 + |Z|)\, \varepsilon' \sum_{z\in Z} |q_z - \hat{q}_z|.$$

Using the Cauchy-Schwarz inequality again,

$$\check{Q} \sum_{z\in Z} (q_z - \hat{q}_z)^2 \leqslant (1 + |Z|)\, \varepsilon' \sqrt{|Z|} \sqrt{\sum_{z\in Z} (q_z - \hat{q}_z)^2}.$$

If $\sqrt{\sum_{z\in Z} (q_z - \hat{q}_z)^2} = 0$ we are done. Thus, we can divide both sides by this expression:

$$\check{Q} \sqrt{\sum_{z\in Z} (q_z - \hat{q}_z)^2} \leqslant (1 + |Z|) \sqrt{|Z|} \varepsilon',$$

which yields the final bound

$$|q_z - \hat{q}_z| \leqslant \frac{(1 + |Z|)\sqrt{|Z|}}{\check{Q}} \varepsilon'.$$

Choosing ε' such that

$$\varepsilon' \leqslant \varepsilon \left(\min_{z\in Z} \frac{c_z^{(t)}}{q_z} \right) \left((1 + |Z|)\sqrt{|Z|} \right)^{-1}$$

gives us $|q_z - \hat{q}_z| \leqslant \varepsilon$ for all $z \in Z$. □

Theorem 4.8 (PAC identification). *For $\varepsilon, \delta > 0$, there exists a size of the dataset $|\mathcal{D}| = N$, such that, if we fix $c_z^{(t)}$ as in (1), then any global minimum $\hat{p}_s^{(t)}, \hat{q}_z$ of $\texttt{NLP}_{\texttt{main}}$ satisfies that, for all $z \in Z$ and $s \in S$ where $z \preceq s$, $\mathbb{P}_{\mathcal{D}}\left(|O(z \mid s) - \hat{q}_z| \leqslant \varepsilon\right) \geqslant 1 - \delta$.*

Proof. We first provide an argument for the precision of every individual $c_z^{(t)}$, then generalize it to all $c_z^{(t)}$, and finally use Lemma 4.7 to prove the overall claim.

Every $c_z^{(t)}$ corresponds to a Bernoulli process (Dekking et al., 2005, Chapter 4.3), i.e., it is a sequence of binary random variables that are independent and identically distributed, namely with success probability $p = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$ to yield a 1 and failure probability $1 - p$ to yield a 0. $c_z^{(t)}$ is exactly the empirical success probability of this Bernoulli process.

By using, e.g., Okamoto's inequality (Okamoto, 1959, Theorem 1), we can relate the number of samples N for a Bernoulli process with the probability that the empirical success probability is ε-precise. Concretely, when

$$N \geqslant \frac{\ln(\frac{2}{\delta})}{2 \cdot \varepsilon^2},$$

then we know that $\mathbb{P}(|c_z^{(t)} - p| \geqslant \varepsilon) \leqslant \delta$, see (Budde et al., 2025, Section 3) for a more detailed discussion of this standard argument.

To extend this argument too all $c_z^{(t)}$, we uniformly distribute the overall confidence δ over all z and t, i.e., we set $\delta_z^{(t)} = \frac{\delta}{Z \cdot L}$. Then, for a given ε', the above argument yields that $c_z^{(t)}$ is ε'-close to $\mathbb{P}(\boldsymbol{z}^{(t)} = z)$ with probability at least $\delta_z^{(t)}$ when $N \geqslant \frac{\ln(\frac{2}{\delta_z^{(t)}})}{2 \cdot (\varepsilon')^2}$. By the union bound, we have that all $c_z^{(t)}$ are ε'-close with probability at least δ.

Finally, by choosing $\varepsilon' \leqslant \varepsilon \left(\min_{z \in Z} \frac{c_z^{(t)}}{q_z}\right) \left((1 + |Z|)\sqrt{|Z|}\right)^{-1}$, we can apply Lemma 4.7 to obtain that the q_z that solve $\texttt{NLP}_{\texttt{main}}$ are ε-close to the true observation probability. □

D On the time-flattened problem

This section concerns $\texttt{NLP}_{\texttt{flat}}$ from Section 5.2, and $\texttt{DCP}_{\texttt{flat}}$ from Section 5.1.

D.1 Explicit optimization problem

The explicit flattened NLP is the following:

Optimization Problem 3 (Time-flattened NLP, $\texttt{NLP}_{\texttt{flat}}$).

$$\min_{p_s, q_z} \sum_{z \in Z} \left((q_z \sum_{s \in S \colon z \preceq s} p_s) - c_z \right)^2, \tag{25}$$

$$\text{s.t.} \sum_{s \in S} p_s = 1, \tag{26}$$

$$\sum_{z \in Z \colon z \preceq s} q_z = 1, \quad \forall s \in S. \tag{27}$$

D.2 Correctness

Lemma D.1. *For any $\varepsilon, \delta > 0$, there exists a number of samples in the dataset $N = |\mathcal{D}|$ such that, letting c_z as in (9), we have $\mathbb{P}(|c_z - \mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})} = z)| \leqslant \varepsilon) = 1 - \delta$ for all $z \in Z$.*

Proof. The argument for this proof again works via Okamoto's inequality and relating the number of samples N to the required precision and confidence ε and δ as in the proof of Theorem 4.8. The only difference is that now we do not have a Bernoulli process for every z and t, but only for every z. Thus, the confidence is only distributed over observations (again using the union bound), not over time. Overall, by choosing

$$N \geqslant \frac{\ln(\frac{2}{\delta/|Z|})}{2 \cdot \varepsilon^2},$$

we ensure that every c_z is ε-close to the true probability $\mathbb{P}(\boldsymbol{z}^{(\boldsymbol{\tau})}$ with confidence at least δ. □

D.3 DCP COMPLIANCE

To show that $\texttt{DCP}_{\texttt{main}}$ is DCP compliant, we verify that all expressions are DCP compliant and are combined via valid DCP composition rules, as specified in Grant et al. (2006).

Lemma 5.1. *$\texttt{DCP}_{\texttt{main}}$ meets all requirements for DCP compliance specified in Grant et al. (2006).*

Proof. In particular, we use the function $f(x) \coloneqq 1/x$ in (8), which is DCP compliant when x is strictly positive and affine. Variables $p_s^{(t)} > 0$ are affine, and thus the argument of $f(x) = 1/x$, namely, $\sum_{s' \in S : z \preceq s} p_{s'}^{(t)}$, is affine and nonnegative. Scaling by constants $c_z^{(t)}$ and summation preserve DCP compliance. For each constraint, the left-hand side is convex and the right-hand side is affine, respectively, meaning that (8) is DCP compliant. Constraint (7) is affine and thus DCP compliant. Finally, the objective in (6) is a sum of squares of affine expressions and thus DCP compliant. □

D.4 CONVEXIFIED TIME-FLATTENED PROGRAM

Analogously to the relaxation in Section 5.1, we transform the time-flattened $\texttt{NLP}_{\texttt{flat}}$ to obtain a convex version that allows polynomial-time solutions while having PAC guarantees.

Optimization Problem 4 (Time-flattened DCP, $\texttt{DCP}_{\texttt{flat}}$)**.**

$$\min_{p_s, r_s} \sum_{s \in S} r_s^2, \tag{28}$$

$$\text{s.t.} \sum_{s \in S} p_s = 1, \tag{29}$$

$$\left[\sum_{z \in Z, z \preceq s} \frac{c_z}{\sum_{s' \in S, z \preceq s} p_{s'}} \right] - 1 \leqslant r_s, \quad \forall s \in S. \tag{30}$$

D.5 RUNTIME COMPLEXITY

$\texttt{NLP}_{\texttt{flat}}$ has $|S| + |Z|$ variables and $|S| + 1$ constraints, whereas the time-based version in $\texttt{NLP}_{\texttt{main}}$ has L times as many variables and constraints. For simplicity, since the number of constraints in our NLPs is proportional to the number of variables, we consider runtime only with respect to a problem size of $L(|S| + |Z|)$ for $\texttt{NLP}_{\texttt{main}}$, and $|S| + |Z|$ for $\texttt{NLP}_{\texttt{flat}}$.

Comparing the NLPs In practice, NLPs require exponential solving time in the size of the NLP (Canny, 1988), and we thus we assume that there exists a $k > 1$ where the runtime of an NLP of size v is k^v. Therefore, plugging in the sizes of $\texttt{NLP}_{\texttt{main}}$ and $\texttt{NLP}_{\texttt{flat}}$, we get an exponential asymptotic speedup of $k^{|S|+|Z|} \ll k^{L(|S|+|Z|)}$.

Comparing the DCPs In case of their convex counterparts, $\texttt{DCP}_{\texttt{main}}$ and $\texttt{DCP}_{\texttt{flat}}$, we still obtain a speedup. Given that DCP allows polynomial-time solutions Grant et al. (2006), we assume a runtime of v^k in the size v of the program, for some constant $k > 1$. We then get the respective runtimes $L^k(|S| + |Z|)^k$ and $(|S| + |Z|)^k$. If L is treated as a constant, this case yields a constant speedup, and both programs thus share the same asymptotic complexity. However, L^k is large in practice, and we should hence observe a significant empirical speedup.

E PROGRAMS FOR HANDLING SPARSE POMDPS

We start by introducing two common notions of sparsity that restrict the transition function of the Markov chain induced by the behavior policy π_{beh} (Definition 3.2).

Definition E.1 (Sparsity)**.** Transition function $T^{\pi_{\text{beh}}}$ is *d-branching* iff $\forall s \in S, |\text{supp}(T^{\pi_{\text{beh}}}(s))| \leqslant d$. Further, $T^{\pi_{\text{beh}}}$ has *d-reachable support* iff $|\{s \in S \mid \mathbb{P}(\boldsymbol{s}^{(t)} = s) > 0\}| \leqslant d$ for all t.

Intuitively, a transition function is d-branching if each state has at most d successors. Furthermore, the transition function has d-reachable support if, at every time step, the number of states reachable with nonzero probability is at most d, which is a strictly stronger condition since it argues about reachable states and not just direct successors.

E.1 Pruning redundant variables

To speed up runtime, we prune variables in $\texttt{NLP}_{\texttt{main}}$ that the transition function's sparsity renders redundant. For each $z \in Z$, we proceed as follows. If $c_z^{(t)} = 0$ for all t, we let $q_z = 0$ with the same reasoning as in Remark 4.4. Otherwise, we conclude that $q_z > 0$, and, for every t where $c_z^{(t)} = 0$, if the count is exact, i.e, $c_z^{(t)} = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$, then we conclude that $\sum_{s \in S\colon z \preceq s} \mathbb{P}(\boldsymbol{s}^{(t)} = s) = 0$, and we let $p_s^{(t)} = 0$ for all $s \in S$ where $z \preceq s$, and we deem variables $p_s^{(t)}$ *pruned*. With this idea in mind, we define the set of visited states as

$$\hat{R}^{(t)} := \{s \in S \mid \exists z \in Z, z \preceq s \text{ and } c_z^{(t)} > 0\}, \tag{31}$$

which approximates the set of reachable states, $R^{(t)} := \{s \in S \mid \mathbb{P}(\boldsymbol{s}^{(t)} = s) > 0\}$, except for any $z \in Z$ where $\sum_{t=1}^{L} c_z^{(t)} = 0$. Hence, we prune $\texttt{NLP}_{\texttt{main}}$ by iterating over $\hat{R}^{(t)}$ instead of S.

Optimization Problem 5 (Pruned NLP, $\texttt{NLP}_{\texttt{pruned}}$)**.**

$$\min_{q_z, p_s^{(t)}} \quad \sum_{t=1}^{L} \sum_{z \in Z} \left(\left(q_z \sum_{s \in \hat{R}^{(t)}\colon z \preceq s} p_s^{(t)} \right) - c_z^{(t)} \right)^2, \tag{32}$$

$$\textit{s.t.} \quad \sum_{s \in \hat{R}^{(t)}} p_s^{(t)} = 1, \quad \textit{for } t = 1, \ldots, L \tag{33}$$

$$\sum_{z \in Z\colon z \preceq s} q_z = 1, \quad \forall s \in \bigcup_{t=1}^{L} \hat{R}^{(t)}. \tag{34}$$

E.2 Solving the optimization problems iteratively

We extend $\texttt{NLP}_{\texttt{pruned}}$ from the previous section to obtain an iterative algorithm, by solving programs $P^{(t)}$ iteratively for $t = 1, \ldots, L$, reusing results from past iterations. We let $\hat{R}^{(t)}$ as in (31) and introduce the following sets. $\hat{U}^{(t)}$ approximates the set of currently seen but previously unseen observations and $\hat{W}^{(t)}$ tracks the set of reachable states that may emit any of the unseen observations: $\hat{U}^{(t)} := \{z \in Z \mid c_z^{(t)} > 0 \text{ and } \forall t' < t, c_z^{(t')} = 0\}$, and $\hat{W}^{(t)} := \{s \in \hat{R}^{(t)} \mid \exists z \in \hat{U}^{(t)}\colon z \preceq s\}$. We also define $U^{(t)}$ and $W^{(t)}$ analogously but with respect to $R^{(t)}$. As in $\texttt{NLP}_{\texttt{main}}$, we introduce variables $p_s^{(t)} \in [0, 1]$ to approximate $\mathbb{P}(\boldsymbol{s}^{(t)} = s)$, but this time only for $s \in \hat{R}^{(t)}$. Similarly, each $q_z \in [0, 1]$ approximates $O(z \mid s)$, but the variable is introduced into problem $P^{(t)}$ only when $z \in \hat{U}^{(t)}$. Then, we define each program $P^{(t)}$, for $t = 1, \ldots, L$, as

Optimization Problem 6 (Iterative NLP, $P^{(t)}$, $\texttt{NLP}_{\texttt{iterative}}$)**.**

$$\min_{q_z, p_s^{(t)}} \quad \sum_{z \in Z} \left(\left(\tilde{q}_z \sum_{s \in \hat{W}^{(t)}\colon z \preceq s} p_s^{(t)} \right) - c_z^{(t)} \right)^2, \tag{35}$$

$$\textit{s.t.} \quad \sum_{s \in \hat{R}^{(t)}} p_s^{(t)} = 1, \tag{36}$$

$$\sum_{z \in Z\colon z \preceq s} \tilde{q}_z = 1, \quad \forall s \in \bigcup_{t=1}^{L} \hat{R}^{(t)}. \tag{37}$$

Here, we let $\tilde{q}_z := q_z$ if $z \in \hat{U}^{(t)}$, and we let $\tilde{q}_z$ be the global minimum of $P^{(t-1)}$ otherwise. The algorithm may terminate early once the value of q_z is found for every $z \in \bigcup_t \hat{U}^{(t)}$.

Correctness. $\texttt{NLP}_{\texttt{pruned}}$ is correct in the limit case, $c_z^{(t)} = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$, as it means that $\hat{R}^{(t)} = R^{(t)}$, and the problem is thus equivalent to $\texttt{NLP}_{\texttt{main}}$. For the iterative algorithm, in the limit case of $c_z^{(t)} = \mathbb{P}(\boldsymbol{z}^{(t)} = z)$ we get $\hat{R}^{(t)} = R^{(t)}$, and thus $\hat{U} = U$ and $\hat{W} = W$. Therefore, for each $z \in \hat{U}^{(1)}$, program $P^{(1)}$ yields $q_z = O(z \mid s)$ due to Lemma 4.5, and, since $P^{(t+1)}$ reuses values from $P^{(t)}$, we get the (limit) correctness of every program inductively. In the approximate case, if $c_z^{(t)}$ is as in (1),

then $\mathbb{P}(\hat{R}^{(t)} = R^{(t)})$ approaches 1 as N approaches ∞, and thus both $\texttt{NLP}_{\texttt{pruned}}$ and $\texttt{NLP}_{\texttt{iterative}}$ inherit the PAC guarantees of $\texttt{NLP}_{\texttt{main}}$ proved in Theorem 4.8.

Reformulations. Similarly to the transformation in Section 5.1, we relax $\texttt{NLP}_{\texttt{pruned}}$ and $\texttt{NLP}_{\texttt{iterative}}$ into their counterparts $\texttt{DCP}_{\texttt{pruned}}$ and $\texttt{DCP}_{\texttt{iterative}}$ in Appendix E.3. We refrain from a time-flattened version (as in Section 5.2), as it contradicts exploiting sparsity.

Complexity. For both notions of sparsity introduced in Definition E.1, the iterative algorithm in $\texttt{NLP}_{\texttt{iterative}}$ has better asymptotic runtime than both $\texttt{NLP}_{\texttt{main}}$ and $\texttt{NLP}_{\texttt{flat}}$. In case of their convex counterparts, the iterative version still has better runtime in case of d-reachable support, whereas this advantage is lost in case of d-branching $T^{\pi_{\text{beh}}}$. We provide technical details in Appendix E.4.

E.3 Convexified programs for exploiting sparsity

We transform the pruned $\texttt{NLP}_{\texttt{pruned}}$ to obtain a DCP, analogously to Section 5.1.

Optimization Problem 7 (Pruned DCP, $\texttt{DCP}_{\texttt{pruned}}$)**.**

$$\min_{p_s^{(t)}, r_s^{(t)}} \sum_{t=1}^{L} \sum_{s \in \hat{R}^{(t)}} \left(r_s^{(t)}\right)^2, \tag{38}$$

$$\text{s.t.} \sum_{s \in \hat{R}^{(t)}} p_s^{(t)} = 1, \quad \forall t \in \{1, \dots, L\}, \tag{39}$$

$$\left[\sum_{z \in Z, z \preceq s} \frac{c_z^{(t)}}{\sum_{s' \in \hat{R}^{(t)}, z \preceq s} p_{s'}^{(t)}}\right] - 1 \leqslant r_s^{(t)}, \quad \forall t \in \{1, \dots, L\}, \forall s \in \hat{R}^{(t)}. \tag{40}$$

Now, we perform a similar transformation to convexify the sequence of NLPs in $\texttt{NLP}_{\texttt{iterative}}$ into a sequence of DCPs $\{Q^{(t)}\}_{t=1}^{L}$.

Optimization Problem 8 (Iterative DCP, $Q^{(t)}$, $\texttt{DCP}_{\texttt{iterative}}$)**.**

$$\min_{p_s^{(t)}, r_s^{(t)}} \sum_{s \in \hat{W}^{(t)}} \left(r_s^{(t)}\right)^2, \tag{41}$$

$$\text{s.t.} \sum_{s \in \hat{R}^{(t)}} p_s^{(t)} = 1, \tag{42}$$

$$\left[\sum_{z \in Z, z \preceq s} \frac{c_z^{(t)}}{\sum_{s' \in \hat{R}^{(t)}, z \preceq s} p_{s'}^{(t)}}\right] - 1 \leqslant r_s^{(t)}, \quad \forall s \in \bigcup_{t=1}^{L} \hat{R}^{(t)}. \tag{43}$$

E.4 Runtime complexity of Problems $\texttt{NLP}_{\texttt{iterative}}$ and $\texttt{DCP}_{\texttt{iterative}}$

***d*-reachable support** Suppose $T^{\pi_{\text{beh}}}$ has d-reachable support with $d \ll |S|$. With a similar simplification as in Appendix D.5, we assume that each program $P^{(t)}$ has a size of $\mathsf{size}(P^{(t)}) = d + |Z|$. As in Appendix D.5, we let each NLP be solved run in time $k^{\mathsf{size}(P^{(t)})}$ for some $k > 1$. Therefore, the sequence of NLPs $\{P^{(t)}\}$ is solved in time

$$\sum_{t=1}^{L} k^{\mathsf{size}(P^{(t)})} \leqslant \sum_{t=1}^{L} k^{d+|Z|} \leqslant L k^{d+|Z|} \ll k^{|S|+|Z|},$$

yielding an asymptotic speedup in comparison with the time-flattened version in $\texttt{NLP}_{\texttt{flat}}$, which requires $k^{|S|+|Z|}$ time (see Appendix D.5).

We now consider the case of their convex counterparts, Problems $\texttt{DCP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{iterative}}$. Again as in Appendix D.5, we use the runtime of $\mathsf{size}(P^{(t)})^k$ for some constant $k > 1$, to get

$$\sum_{t=1}^{L} \mathsf{size}(P^{(t)})^k \leqslant \sum_{t=1}^{L} (d + |Z|)^k \leqslant L(d + |Z|)^k.$$

As $\texttt{DCP}_{\texttt{flat}}$ requires time $(|S| + |Z|)^k$, we have

$$L(d + |Z|)^k \ll (|S| + |Z|)^k,$$

whenever $L \ll (|S| + |Z|)^k / (d + |Z|)^k$. Notice that this is a weak condition, usually satisfied in practice, because $(|S| + |Z|)^k \gg (d + |Z|)^k$.

d-branching Suppose $T^{\pi_{\text{beh}}}$ is d-branching with $d \ll |S|, |Z|$. We assume $|\text{supp}(b_{\text{init}})| = 1$ for simplicity, but the results carry over for other b_{init}, just with an additional term $|\text{supp}(b_{\text{init}})|$. Analogously to Appendix D.5, each program has a size of $\text{size}(P^{(t)}) = d^t$. The worst case (i.e., the $P^{(t)}$ with the greatest size) happens when every $P^{(t)}$ covers a fresh set of states in every time step. Recall that the program may terminate early once every observation probability is learned, and this is guaranteed to happen if every state in $\bigcup_t \hat{R}^{(t)}$ is reached at some point $n \leqslant L$. More specifically, the number of programs we need to solve in the worst case is bounded by

$$n = \arg\min_{n' \in \mathbb{N}} \sum_{i=0}^{n'} d^i \geqslant |S|.$$

As $\sum_{i=0}^{n'} d^i = \frac{d^{n'+1}-1}{d-1}$, we conclude $n = \lceil \log_d((d-1)|S|+1) \rceil - 1$. Since we are interested in the worst-case runtime, we simplify $n = \log_d((d-1)|S|+1) - 1$, as this ensures that program $P^{(n)}$ requires the longest possible runtime.

If each program can then be solved in time $f(\text{size}(P^{(t)}))$ for a monotonic f, we have

$$\sum_{t=1}^{n} f(\text{size}(P^{(t)})) \leqslant n f(d^n) = n f((1 + |S|(d-1))/d). \tag{44}$$

Assuming each NLP $P^{(t)}$ can be solved in exponential time as in Appendix D.5, $f(\text{size}(P^{(t)})) = k^{\text{size}(P^{(t)})}$ for some $k > 1$, we plug it into (44) to get that the solving time of the iterative sequence of NLPs is bounded by

$$n f((1 + |S|(d-1))/d) = n k^{((1+|S|(d-1))/d)} \ll k^{|S|+|Z|},$$

noting again that $\texttt{NLP}_{\texttt{flat}}$ requires $k^{|S|+|Z|}$ time (see Appendix D.5).

In case of their convex counterparts, $\texttt{DCP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{iterative}}$, we still obtain a speedup, although less dramatic. We plug $f(\text{size}(P^{(t)})) = \text{size}(P^{(t)})^k$ for some constant $k > 1$ into (44) to get

$$n f((1 + |S|(d-1))/d) = n((1 + |S|(d-1))/d)^k = \mathcal{O}(\log_d(|S|)|S|^k).$$

In practical terms, we may see a speedup in runtime when $((d-1))/d)^k$ is small, whereas the algorithm may not be beneficial otherwise.

F Alternative Settings

This section considers variants of the problem statement from Section 4.1, namely the cases where both π_{beh} and T are known, T is known but π_{beh} unknown, or π_{beh} and O are non-Markovian.

F.1 Known transition dynamics and behavior policy.

Here, we assume that we know T, b_{init}, and π_{beh}. Thus, we can calculate the exact probability of visiting state s at time step t by letting $P_s^{(t)} := \mathbb{P}(\boldsymbol{s}^{(t)} = s)$ as in Definition 3.2. Recall that $\texttt{NLP}_{\texttt{main}}$ in contrast involves the approximation of these probabilities through optimization variables $p_s^{(t)}$. By replacing $p_s^{(t)}$ from $\texttt{NLP}_{\texttt{main}}$ with the exact $P_s^{(t)}$, we obtain a linear program in Optimization Problem 9 that is efficiently solvable in practice (Dantzig, 2002). We additionally provide a quick approximation in Remark F.1.

For readability, we define constants

$$\kappa_z^{(t)} := \sum_{s \in S \colon z \preceq s} P_s^{(t)},$$

and rewrite these into $\texttt{NLP}_{\texttt{main}}$ to replace the sums of $p_s^{(t)}$ variables. The only discrepancy is that we will use a sum of absolute values in the objective instead of a sum of squares, which inherits the correctness properties from Section 4.4, to allow its easy conversion into a linear program (LP) later.

$$\begin{aligned} \min_{q_z} \quad & \sum_{t=1}^{L} \sum_{z \in Z} \left| \kappa_z^{(t)} q_z - c_z^{(t)} \right|, \\ \text{s.t.} \quad & \sum_{z \in Z \colon z \preceq s} q_z = 1, \quad \forall s \in S. \end{aligned}$$

To formulate it as an LP, we perform a standard transformation by replacing each absolute value term $\left|\kappa_z^{(t)} q_z - c_z^{(t)}\right|$ in the objective by introducing auxiliary variables $y_z^{(t)} \in [0,1]$ and enforcing linear constraints. The resulting LP is:

Optimization Problem 9 (Known model LP)**.**

$$\begin{aligned} \min_{q_z,\, y_z^{(t)}} \quad & \sum_{t=1}^{L} \sum_{z \in Z} y_z^{(t)}, \\ \textit{s.t.} \quad & y_z^{(t)} \geqslant \kappa_z^{(t)} q_z - c_z^{(t)}, \quad \forall z \in Z,\ t = 1, \dots L, \\ & y_z^{(t)} \geqslant c_z^{(t)} - \kappa_z^{(t)} q_z, \quad \forall z \in Z,\ t = 1, \dots, L, \\ & \sum_{z \in Z\colon\, z \preceq s} q_z = 1, \quad \forall s \in S. \end{aligned}$$

Remark F.1. Alternatively, to get a quick approximation that is still PAC, let the time-flattened

$$\kappa_z := L^{-1} \sum_{t=1}^{L} \kappa_z^{(t)} \text{ and } c_z := L^{-1} \sum_{t=1}^{L} c_z^{(t)},$$

and simply let $q_z := c_z / \kappa_z$. Since the resulting values may not satisfy $\sum_{z \in Z\colon\, z \preceq s} q_z = 1$, it may be convenient to normalize them.

F.2 Known transition dynamics with unknown behavior policy.

We discuss the case where T and b_{init} are known, but π_{beh} is still unknown. In general, this setting is as hard as the main one from Section 4.1. To see this, consider a POMDP where both T and π_{beh} are unknown, as in Section 4.1. Then we can transform it into a POMDP where a new behavior policy π'_{beh} encodes the probabilities of both T, b_{init} and π_{beh}, and we provide uniformly random transition dynamics instead, b'_{init} and T'. Then, learning the observation function while knowing neither T nor π_{beh} in the original POMDP is equivalent to learning it while knowing T' but not π'_{beh}.

NC-reduction for known transition dynamics Let $\mathcal{M} = (S, A, T, Z, O, b_{\text{init}})$ have an SI observation function, and let $\pi_{\text{beh}} \colon (Z \times A)^* \to A$ be a behavior policy. Then, we let $\tilde{\pi}(a \mid s) := \mathbb{P}_{\pi_{\text{beh}}}(a \mid s)$, which is well-defined by Remark 4.2. Now, we let POMDP $\mathcal{M}' = (S, A', T', Z, O, b_{\text{init}})$ where $A' := S$, $T'(s' \mid s, a) := \mathbf{1}_{s'=a}$, and define a new behavior policy for $s' \in A'$, $s \in S$,

$$\pi'_{\text{beh}}(s' \mid s) := \sum_{a \in A} \tilde{\pi}(a \mid s) T(s' \mid s, a).$$

Therefore, Markov chains $T^{\pi_{\text{beh}}}$ and $(T')^{\pi'_{\text{beh}}}$ are equivalent by construction, and the observation learning problem from Section 4.1 for $\mathcal{M}$ many-one reduces to the problem from Appendix F.2 for $\mathcal{M}'$. In fact, the reduction is NC-computable, as π'_{beh} can be computed with polynomial parallel processes that each use polylog time (Cook, 2023; Arora & Barak, 2009).

F.3 Non-Markovian behavior policy.

As we discuss in Remark 4.2, we assume the behavior policy is Markovian. In case the behavior policy π_{beh} is non-Markovian, that is, $\mathbb{P}_{\pi_{\text{beh}}}(a \mid s_0, a_0, \dots, s_t)$ does not necessarily equal $\mathbb{P}_{\pi_{\text{beh}}}(a \mid s_t)$, the state space of the Markov chain induced by π_{beh} is the product $S \times (Z \times A)^*$, rather than S alone. Given that our horizon is of length L, this Markov chain is finite, but it is exponential in L. An optimization problem analogous to $\texttt{NLP}_{\texttt{main}}$ can still be obtained by defining $p_{s,\rho}$ for each $(s, \rho) \in S \times (Z \times A)^*$, but the optimization problem has exponential size with respect to L. Then, state invariance restricts the states in the induced Markov chain, $\forall (s, \rho), (s', \rho') \in S \times (Z \times A)^*$, $\forall z \in Z$ if $z \preceq (s, \rho), (s', \rho')$, then $O(z \mid s) = O(z \mid s')$.

On a side note, if we assume that the behavior policy has a compact memory space, e.g., such as in certain *finite-state controllers* (Meuleau et al., 1999), it may induce polynomial-sized Markov chains.

G Action-independent observation functions are enough

We assume an action-independent observation function, $O : S \to \text{Distr}(Z)$, purely for simplicity and without loss of generality. If the observation function depends on actions, $O : S \times A \to \text{Distr}(Z)$,

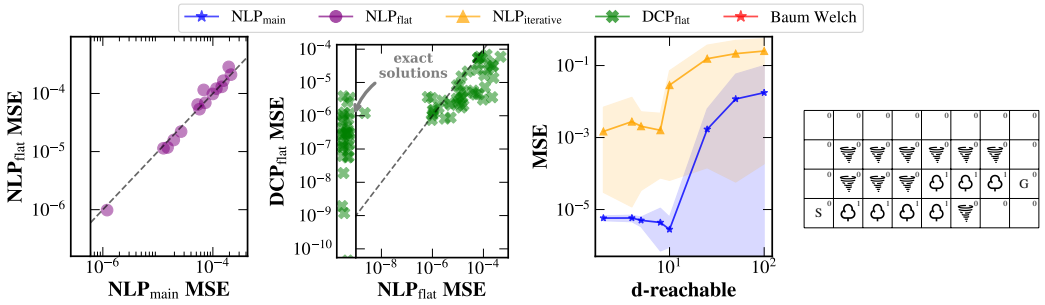


Figure 4: Additional results of the empirical validation. Lines: mean; shaded area: min–max range. **(d)** Map for experiment 6.

one can augment the state space to incorporate the last performed action via a standard transformation (Chatterjee et al., 2016). In short, given an action-dependent POMDP $\mathcal{M} = (S, A, T, Z, O, b_{\text{init}})$, we transform it into an equivalent action-independent POMDP by simply defining states $S' := S \times A$, observations $Z' := Z \times A$, transitions $T'((s', a') \mid (s, a), a') = T(s' \mid s, a')$, initial distribution $b'_{\text{init}}((s, *)) = b_{\text{init}}(s)$ with some dummy action $* \in A$, and observation function $O'((z, a) \mid (s, a)) = O(z \mid s, a)$ that reveals the last-performed action.

Since action information is embedded into the observation space via this transformation, notice that a dataset of the form $\mathcal{D} \in (Z'^L)^N$ is already sufficient, where Z' is the transformed observation space.

We note that state invariance does not become a stronger assumption if observations depend on actions: SI gives $O(z, a \mid s, a) = O(z, a \mid s', a)$ for any $z \preceq s, s'$, which is in reality slightly weaker as a is rigid.

Lastly, we note that if rewards, costs, or other objective-related information is available to us, then it may be beneficial to also treat them as part of the observation and include them in the dataset to obtain an even weaker notion of SI where rewards can also reveal state information.

H Empirical Analysis

This appendix provides additional details on the experiments presented in Section 6. Section H.1 details the experiments and the technical setup. Section H.2 describes the synthetic benchmark generation.

H.1 Experimental Details

Software setup. We use the solver `GEKKO` (v1.3.0) (Beal et al., 2018) for NLP implementations (`NLP_main`, `NLP_flat`, `NLP_iterative`) and `CVXPY` (v1.8.2) (Diamond & Boyd, 2016) in combination with the solver `MOSEK` (v11.10) for DCP implementations (`DCP_flat`). For the baseline implementations we employ the code of (Wendland et al., 2025) for `AsMAR` and provide our own implementation of Baum-Welch. We compute policies with SARSOP (Kurniawati et al., 2008), using the `pomdpsol` implementation via `SARSOP.jl` (v0.5.8)[2] in the `POMDPs.jl` framework (Egorov et al., 2017) (v0.9.6) in Julia 1.11.6 (Bezanson et al., 2017).

Hardware setup. All experiments were conducted on workstations equipped with an AMD Ryzen ThreadRipper PRO 5965WX (24-core, 3.8 GHz) CPU, 512 GB ECC DDR4 RAM, and a 2 TB PCIe 4.0 NVMe SSD.

Energy consumption. We track energy consumption using `CodeCarbon` (v2.8.1) (Courty et al., 2024). Across all experiments, we consumed a total of 195 kWh, corresponding to approximately 74.3 kg CO_2eq.

Table 1 lists the (hyper-)parameters used in each experiment. Across experiments, we vary the history length of each trace, the number of traces in the dataset, the problem formulation, the solver tolerance,

[2] https://github.com/JuliaPOMDP/SARSOP.jl

the maximum allotted solving time, and the maximum number of solver iterations. We highlight the configuration specific to each experiment below.

Experiment 1 – $\texttt{NLP}_{\texttt{main}}$ vs $\texttt{NLP}_{\texttt{flat}}$. We generate synthetic SI POMDPs for all combinations of $|S|$ and $|Z|$ listed in Table 1. For each POMDP, we approximate the observation function by solving $\texttt{NLP}_{\texttt{main}}$ and $\texttt{NLP}_{\texttt{flat}}$. Fig. 2d reports the results regarding solving time. Figure 4a compares the approximatino error between the two approaches. The differences are negligible for the experiment's parameters. Most points lie close to the diagonal.

Experiment 2 – $\texttt{NLP}_{\texttt{flat}}$ vs $\texttt{DCP}_{\texttt{flat}}$. We generate synthetic SI POMDPs for all combinations of $|S|$ and $|Z|$, using observation function initialization techniques (1) and (2). For each POMDP, we solve $\texttt{NLP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{flat}}$, and compare both solving time (Fig. 2e) and approximation error (Fig. 4b). The left region of Fig. 4b shows that $\texttt{NLP}_{\texttt{flat}}$ recovers exact solutions while $\texttt{DCP}_{\texttt{flat}}$ retains nonzero error, reflecting the relaxation introduced by the convex reformulation.

Experiment 3 – Influence of dataset size. We fix the sizes of the state and observation spaces and generate a POMDP whose observation function follows initialization techniques (1) and (2). We then approximate the observation function with $\texttt{NLP}_{\texttt{flat}}$, $\texttt{NLP}_{\texttt{iterative}}$, and $\texttt{DCP}_{\texttt{flat}}$ for varying dataset sizes. We repeat the experiment over three seeds and report the results in Fig. 2a.

Experiment 4 – Influence of model sparsity. We fix the sizes of the state and observation spaces and generate SI POMDPs with observation functions following initialization techniques (1), (2), and (3). The transition function is generated with varying d according to the d-reachable support property (Definition E.1). We approximate the observation function by solving $\texttt{NLP}_{\texttt{main}}$ and $\texttt{NLP}_{\texttt{iterative}}$; solving times are shown in Fig. 2f and approximation errors in Fig. 4c. The MSE increases with d, as expected when both the POMDP size and the dataset size remain fixed.

Experiment 5 – SI-violation. In this experiment we analyze the sensitivity of our algorithms to POMDPs that violate the SI-property. We begin by initializing an SI-POMDP with the observation function following the initialization technique (1). Then we uniformly add noise $\mathcal{U}(0, \varepsilon)$ to each probability $O(z \mid s, a)$ in the observation function, with $\varepsilon \in [0, 2]$. We then re-normalize again such that each $O(\cdot \mid s, a)$ is a valid probability distribution. We sample traces 10^6 from this non-SI-POMDP and use the resulting dataset as an input to our $\texttt{DCP}_{\texttt{flat}}$ algorithm. We measure the error of $\texttt{DCP}_{\texttt{flat}}$'s solution compare to the actual non-SI-POMDP using the MSE. We repeat this for two model sizes and show in Figure 2g that increasing ε increases the approximation error. We repeat the experiment with three seeds.

Experiment 6 – Effect on policy performance. In this experiment we show the effect of the approximation error of the observation function on the resulting policy performance. We define an SI-POMDP similar to Example 3, a grid world where each cell has varying occlusion probability (see Figure 4d). The drone here has headwinds, meaning with prob 0.3 it remains in the same cell. The cells with tornadoes are final states and incur a negative reward of -10. There is a step cost of -0.04 and reward when reaching the goal "G" of 1.0. The observation function is defined as in Example 3: the occlusion feature is missing with probability 0.2, in which case the GPS is missing too; otherwise the GPS is missing with probability 0.9 in dark cells and never in clear cells. Hence only a planner that knows the true observation function recognizes that it cannot localize itself at the turn and takes the safe clear route. The resulting SI-POMDP has $|S \times A| = 128$, $|Z| = 99$.
We generate datasets of varying sizes with a uniformly random policy. The observation function is approximated using $\texttt{NLP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{flat}}$ as well as the baselines Baum Welch and `AsMAR` (Wendland et al., 2025). `AsMAR` is able to identify a subset of SI observation functions which the authors refer to as *simple MAR*. Figure 2b clearly shows how `AsMAR` fails to learn this SI observation function as the error plateaus while the dataset size increases. $\texttt{NLP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{flat}}$ outperform `AsMAR` and Baum-Welch with a sufficiently large dataset size.
With the approximated observation functions we use SARSOP (Kurniawati et al., 2008) to compute the resulting policies on the benchmark. We run SARSOP with a target precision of 10^{-3} and a timeout of 120 s, and evaluate each policy on the true SI-POMDP over 1000 episodes. In Fig. 2c we normalize the resulting policies' values, where $y = 0$ is a SARSOP policy using an observation function that is initialized uniformly at random, and $y = 1$ is SARSOP's upper bound on the value of the optimal policy, computed with the true observation function. With sufficient data $\texttt{NLP}_{\texttt{flat}}$ and $\texttt{DCP}_{\texttt{flat}}$ converge to the optimal policy, while `AsMAR` performs similar to the policy with the uniformly random observation function. Baum-Welch performs even worse than this policy ($y < 0$)

Table 1: Parameters for the experimental results. The numbers for O correspond to the observation function initialization techniques.

Experiment	**1**	**2**	**3**	**4**	**5**	**6**
Problem	$\texttt{NLP}_{\texttt{main}}$, $\texttt{NLP}_{\texttt{flat}}$	$\texttt{NLP}_{\texttt{flat}}$, $\texttt{DCP}_{\texttt{flat}}$	$\texttt{NLP}_{\texttt{flat}}$, $\texttt{NLP}_{\texttt{iterative}}$, $\texttt{DCP}_{\texttt{flat}}$	$\texttt{NLP}_{\texttt{main}}$, $\texttt{NLP}_{\texttt{iterative}}$	$\texttt{DCP}_{\texttt{flat}}$	$\texttt{NLP}_{\texttt{flat}}$, $\texttt{DCP}_{\texttt{flat}}$
State-Actions $\lvert\mathbf{S}\times\mathbf{A}\rvert$	10, 50, 100, 200, 400, 500, 700	10, 50, 100, 200, 400, 500, 700, 1000, 1500, 2000	500	200	200, 500	128
Observations $\lvert\mathbf{Z}\rvert$	10, 50, 100, 200, 400, 500, 700	10, 50, 100, 200, 400, 500, 700, 1000, 1500, 2000	665	222	222, 555	99
O structure	(1)	(1), (2)	(1), (2)	(1), (2), (3)	(2)	Example 3
T structure	uniform at random	uniform at random	uniform at random	d-reachable with $d \in \{5, 10, 25, 50\}$	uniform at random	Section H.1
History length	10	100	10	10	10	50
Traces	10^4	10^3	$\{1, 3\} \cdot 10^i$ for $i = 1, \ldots, 5$, and 10^6	10^5	10^6	$\{10^0, \ldots, 10^5\}$
Tolerance	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-6}
Max time [s]	1200	1200	18000	1200	1200	600
Max iterations	10^6	10^6	10^6	10^6	10^6	250

and is therefore mostly outside the plotted range. We repeat the experiment with six seeds and report the mean and the min–max range over seeds.

H.2 Benchmark generation

Our experiments use synthetic SI POMDPs. We parameterize each benchmark by the sizes of the state and observation spaces and by the structure of the observation and transition functions. We describe each generation procedure below.

Remark H.1 (Reduction to Hidden Markov Models)**.** While our theoretical results are stated for POMDPs, the synthetic benchmarks generated below use transition functions that depend only on the current state, i.e., $T(s' \mid s)$ rather than $T(s' \mid s, a)$. Together with the action-independence of O established in Section 3, this is equivalent to fixing an arbitrary policy π and studying the induced Hidden Markov Model. The approximated observation function $\hat{O}$ is therefore consistent with the true O under this fixed policy. This restriction simplifies data generation without affecting the validity of our empirical conclusions, which concern identifiability of O rather than policy-dependent quantities.

Observation Functions. We generate observation functions adhering to the SI property in the following three ways.

1. **Uniformly random with repair:** We partition Z into Z_1 and Z_2 such that $|Z_2| = |S|$ (assuming $|Z| \geqslant |S|$), which gives us a bijective map $f\colon Z_2 \to S$. For every $z \in Z_1$, we generate its admittance matrix ($\preceq$) uniformly randomly. Then, for each $s \succeq z$, we assign $O(z \mid s)$ a uniformly random value between 0 and $1/|Z|$. To ensure that $\sum_{z \in Z} O(z \mid s) = 1$, we use Z_2 in a "repair" step. For each $z \in Z_2$, we let state $s := f(z)$ be the only state that admits z, and we assign $O(z \mid s) := 1 - \sum_{z' \in Z_1} O(z' \mid s)$.

2. **Constant probability with no repair:** We fix $k \in [0, 1]$, and generate a uniformly random admittance matrix $\preceq$ where $|\{z \in Z \mid z \preceq s\}| = k$. Then, we simply let $O(z \mid s) = 1/k$ for all $z \preceq s$.

3. **Non-constant probability with no repair:** We fix some values for $q_z \in [0, 1]$ for each $z \in Z$. In practice, we let $q_{z_i} := 1/i$ for enumeration $\{z_i\}_{i=1}^{|Z|}$ to allow $\preceq$ to take diverse shapes while still adding up to 1. Then, for each $s \in S$, we do as follows. First, we generate random $v_z \in \{0, \dots, M\}$, for a large $M \in \mathbb{N}$. Then, we solve the knapsack-like problem

$$\begin{aligned} \min_{x_z \in \{0,1\}} \quad & \sum_{z \in Z} x_z v_z \\ \text{s.t.} \quad & \sum_{z \in Z} q_z = 1. \end{aligned}$$

Lastly, we use the global minimum x^* of this problem to populate the column corresponding to s in the admittance matrix, that is, we let $x_z^* = 1 \Leftrightarrow z \preceq s$. For every $s \in S$, a new knapsack-like problem is solved for newly randomly generated $\{v_z\}_{z \in Z}$ to ensure that O is both SI and random.

Transition Functions. We generate transition functions in the following two ways.

- **Uniformly at random:** We generate transition probabilities uniformly at random for each state, resulting in a fully connected state space, meaning that $T(s' \mid s) > 0$ for all $s \in S$.
- **d-reachable support** (Cyclic Grouping)**:** Partition the $|S|$ states into d groups $G_1, G_2, \dots, G_d$ of approximately equal size. States in group G_i transition exclusively to states in group $G_{(i+1)}$, with no within-group or cross-group edges except forward to the next group (the last group G_d maps back to the first group G_0). Each state receives a randomized distribution over all target states. This guarantees the d-reachable support (Definition E.1) of the transition function that the support remains confined to maximum d many states at each timestep. This approach ensures strong connectivity of states, meaning that all states are reachable.