Reconciling Rewards with Predictive State Representations [*] [†]

Andrea Baisero , **Christopher Amato**
Northeastern University, Boston, Massachusetts, USA
{baisero.a, c.amato}@northeastern.edu

Abstract

Predictive state representations (PSRs) are models of controlled non-Markov observation sequences which exhibit the same generative process governing POMDP observations without relying on an underlying latent state. In that respect, a PSR is indistinguishable from the corresponding POMDP. However, PSRs notoriously ignore the notion of rewards, which undermines the general utility of PSR models for control, planning, or reinforcement learning. Therefore, we describe a sufficient and necessary *accuracy* condition which determines whether a PSR is able to accurately model POMDP rewards, we show that rewards can be approximated even when the accuracy condition is not satisfied, and we find that a non-trivial number of POMDPs taken from a well-known third-party repository do not satisfy the accuracy condition. We propose *reward-predictive state representations* (R-PSRs), a generalization of PSRs which accurately models both observations and rewards, and develop value iteration for R-PSRs. We show that there is a mismatch between optimal POMDP policies and the optimal PSR policies derived from approximate rewards. On the other hand, optimal R-PSR policies perfectly match optimal POMDP policies, reconfirming R-PSRs as accurate stateless generative models of observations and rewards.

1 Introduction

Predictive state representations (PSRs) are models of controlled observation sequences which exhibit the same generative properties as partially observable Markov decision processes (POMDPs) [Littman and Sutton, 2002; Singh *et al.*, 2004]. Compared to POMDP models, PSRs lack an underlying latent state; instead, the system state is grounded in predicted likelihoods of future observations. The structure of PSRs only involves observable quantities, therefore learning a PSR model is generally considered to be simpler than learning a latent-state model such as a POMDP [Wolfe *et al.*, 2005]. Hence, significant research effort has been collectively spent on PSR model learning [Singh *et al.*, 2003; James and Singh, 2004; Rosencrantz *et al.*, 2004; Wolfe *et al.*, 2005; Wiewiora, 2005; Bowling *et al.*, 2006; McCracken and Bowling, 2006; Boots *et al.*, 2011]. Likewise, a number of control and reinforcement learning methods have been successfully adapted to PSRs, e.g., policy iteration [Izadi and Precup, 2003], incremental pruning [James *et al.*, 2004], point-based value iteration [James *et al.*, 2006; Izadi and Precup, 2008], and Q-learning [James *et al.*, 2004].

While PSRs are a promising alternative to POMDPs for modeling observation sequences, they notoriously lack the ability to model rewards appropriately. In fact, we show that PSRs are able to represent only a specific subset of reward functions representable by finite POMDPs. Prior work in PSR-based control can be split in two groups depending on how reward modeling is addressed: (a) In the first group, a linear PSR reward function which encodes the appropriate task is directly given and/or assumed to exist [Izadi and Precup, 2003; Izadi and Precup, 2008; Boots *et al.*, 2011], which means that these methods cannot be used to solve problems which cannot be modeled by vanilla PSRs. Further, directly designing PSR rewards which are not grounded to a latent state is extremely unintuitive, compared to designing POMDP rewards grounded on state. (b) In the second group, a *reward-aware* PSR (RA-PSR) models rewards by making them explicitly observable and an integral part of the agent's observable history [James *et al.*, 2004; James *et al.*, 2006]. However, in partially observable sequential decision making problems such as POMDPs, rewards are a construct meant to rank agent behaviors based on how well they solve a task, rather than an intrinsic aspect of the environment like states and observations. In fact, it is not uncommon for rewards to be available during offline training but not during online execution. Further, an agent which is able to observe rewards will be able to condition its behavior based on rewards and/or infer the latent state using rewards, which is not generally allowed when solving POMDPs. Therefore, while RA-PSRs are able to represent reward functions which are not representable by vanilla PSRs, they are still unable to represent the full range of control tasks representable by POMDPs, which do not assume observable rewards.

In this work, we first develop the theory of PSR reward processes, and then a novel extension of PSRs which is able

[*] Appendix available at https://arxiv.org/abs/2106.03926.

[†] Code available at https://github.com/abaisero/rl-rpsr.

to model any reward process without making rewards explicitly observable by the agent. Our contributions are as follows: (a) We derive a sufficient and necessary *accuracy* condition which determines whether the rewards of a POMDP can be accurately represented by a PSR, and a linear approximation for when the accuracy condition is not satisfied; (b) We develop *reward-predictive state representations* (R-PSRs), a generalization of PSRs capable of representing the reward structure of any finite POMDP; (c) We adapt Value Iteration (VI) to R-PSRs; (d) We show that a non-trivial portion of common finite POMDPs, taken from a third-party repository, do not satisfy the accuracy condition; and (e) We evaluate the optimal policies derived by POMDPs, PSRs and R-PSRs according to the other's reward structure, confirming that even the best linear reward approximations in PSRs may catastrophically alter the underlying task, while the reward structure of R-PSRs perfectly matches that of the original POMDPs. Our work represents a significant step towards being able to use PSRs to model realistic environments which do not assume observable rewards, and opens the door for more complicated and scalable methods able to exploit the structure of the proposed R-PSR model.

2 Background

Notation. We use bracket notation $[v]_i$ and $[M]_{ij}$ to index vectors and matrices. $[M]_{i:}$ and $[M]_{:j}$ respectively indicate the i-th row vector and the j-th column vector of M. We use $\Delta\mathcal{X}$ to indicate the set of probability distributions over set $\mathcal{X}$, and boldface $\boldsymbol{x}$ to indicate a random variable. To avoid ambiguity, we will sometimes denote quantities associated with POMDPs using superscript $^{(b)}$, those associated with PSRs using $^{(p)}$, and those associated with R-PSRs using $^{(r)}$.

2.1 POMDPs

A POMDP [Cassandra *et al.*, 1994] is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, R, \gamma \rangle$ consisting of: state, action, and observation spaces $\mathcal{S}$, $\mathcal{A}$, $\mathcal{O}$; transition function $T\colon \mathcal{S} \times \mathcal{A} \to \Delta\mathcal{S}$; observation function $O\colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \Delta\mathcal{O}$; reward function $R\colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$; and discount factor $\gamma \in [0, 1]$. We focus on finite POMDPs, whose sets $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ are finite; consequently, POMDP functions and related quantities can be represented as matrices and vectors, e.g., the reward matrix $[R]_{ij} \doteq R(s = i, a = j)$.

Interactions, Histories, and Beliefs. We define an *interaction* $ao \in \mathcal{A} \times \mathcal{O}$ as an action-observation pair representing a single exchange between agent and environment, and its *generative* matrix $G_{ao} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ as $[G_{ao}]_{ij} \doteq \Pr(s' = i, o \mid s = j, a)$, which encodes both state transition and observation emission probabilities. A *history* $h \equiv a_1o_1a_2o_2\ldots$ is a sequence of interactions, and the cumulative observable past. We use *string concatenation* to denote the concatenation of histories and/or interactions, e.g., h_1h_2 or hao. We denote the space of all histories as $\mathcal{H} \doteq (\mathcal{A} \times \mathcal{O})^*$, and the empty history as ε. A *belief* $b\colon \mathcal{H} \to \Delta\mathcal{S}$ is the distribution over states following history h, i.e., vector $[b(h)]_i \doteq \Pr(s = i \mid h)$. We define the history reward function $R(h, a) \mapsto \mathbb{E}\left[R(\boldsymbol{s}, a) \mid h\right] = b(h)^\top [R]_{:a}$.

2.2 Predictive State Representations

A PSR is a discrete-time controlled dynamical system which lacks the notion of a latent state [Littman and Sutton, 2002; Singh *et al.*, 2004]; rather, the system state is composed of the predictive probabilities of hypothetical futures called *tests*.

Tests and their Probabilities. A *test* $q \equiv a_1o_1a_2o_2\ldots$ is—like a history—a sequence of interactions. We denote the space of all tests as $\mathcal{Q} \doteq (\mathcal{A} \times \mathcal{O})^*$, and the empty test as ε. Although histories and tests are structurally equivalent, they differ semantically in that the former refer to the past, and the latter to hypothetical futures. We generalize the *generative* matrix to tests via $G_q = \cdots G_{a_2o_2}G_{a_1o_1}$. The action and observation sequences associated with a test q are denoted respectively as $a_q \in \mathcal{A}^*$ and $o_q \in \mathcal{O}^*$. A *test probability* $p(q \mid h) \doteq \Pr(o_q \mid h, a_q)$ is the probability of the test observations o_q if the test actions a_q are taken from the history h as a starting point. Test probabilities are the core quantity modeled by a PSR. A *linear* predictive state $p(h) \in \mathbb{R}^{|\mathcal{Q}^\dagger|}$ (where $\mathcal{Q}^\dagger$ is a *core* set of tests, defined later) is a representation of history h such that test probabilities are linear in $p(h)$, i.e., $p(q \mid h) = p(h)^\top m_q$, where $m_q \in \mathbb{R}^{|\mathcal{Q}^\dagger|}$ is the *parameter vector* associated to test q. Littman and Sutton [2002] show the vectorized form of test probabilities based on beliefs (using a less general version of the generative matrix G_q),

$$p(q \mid h) = b(h)^\top G_q^\top \bar{1}\,, \tag{1}$$

where the linear products represent an implicit marginalization over the sequence of latent states.

Outcome Vectors. The *outcome* of a test $u(q) \in [0, 1]^{|\mathcal{S}|}$ is a vector indicating the test probabilities from each state as a starting point, i.e., $[u(q)]_i \doteq \Pr(o_q \mid s = i, a_q)$. Outcomes can be defined recursively via the generative matrix,

$$u(\varepsilon) = \bar{1}\,, \tag{2}$$

$$u(aoq) = G_{ao}^\top u(q)\,. \tag{3}$$

Combining Equations (1) to (3) results in $p(q \mid h) = b(h)^\top u(q)$, i.e., the test probability given a history is the expectation of test probabilities given each state. A set of tests is said to be *linearly independent* iff the respective outcome vectors are linearly independent, and any maximal set of linearly independent tests is called a *core set*, denoted as $\mathcal{Q}^\dagger$. While there are infinite core sets, they all share the same size $|\mathcal{Q}^\dagger|$, called the PSR *rank*, which is upper-bounded by $|\mathcal{Q}^\dagger| \leq |\mathcal{S}|$ [Littman and Sutton, 2002].

Predictive States. The outcome matrix $U \in [0, 1]^{|\mathcal{S}|\times|\mathcal{Q}^\dagger|}$ of a core set $\mathcal{Q}^\dagger$ is the column-wise stacking of the core outcome vectors $\{u(q) \mid q \in \mathcal{Q}^\dagger\}$. By the definition of a core set, the outcome $u(q)$ of any non-empty test $q \neq \varepsilon$ is a linear combination of the core outcome matrix U columns (else the core set would not be maximal), i.e., $u(q) \in \operatorname{col} U$ and, because UU^+ is the projection onto $\operatorname{col} U$, then $u(q) = UU^+u(q)$. Consequently, $p(q \mid h) = b(h)^\top u(q) = b(h)^\top UU^+u(q) = p(h)^\top m_q$, where $p(h)^\top \doteq b(h)^\top U$ is the predictive state, and $m_q \doteq U^+u(q)$ is the parameter vector of q. Each dimension of $p(h)$ is itself the test probability of a core test, i.e., $[p(h)]_i = p(q_i \mid h)$, where q_i is the i^{th} core test.

Emissions and Dynamics. Observation probabilities are $\Pr(o \mid h, a) = p(ao \mid h) = p(h)^\top m_{ao}$, while the predictive state dynamics are $p(hao) = \left(p(h)^\top M_{ao}\right) / \left(p(h)^\top m_{ao}\right)$, where M_{ao} is the column-wise stacking of the extended core test parameters $\{m_{aoq} \mid q \in \mathcal{Q}^\dagger\}$.

2.3 Value Iteration

Ultimately, we wish to compare POMDP, PSR and R-PSR models by comparing their respective induced optimal policies. Although a number of modern solution methods could be used for this purpose, we employ simple value iteration (VI) as an arguably necessary stepping stone, leaving more modern methods for future work. Value iteration (VI) is a family of dynamic programming algorithms which estimate the optimal value function $V^{*(k)}(h)$ for increasing horizons k, from which optimal actions can be derived. Variants have been developed for POMDPs and PSRs [James *et al.*, 2004; Boots *et al.*, 2011]. A k-horizon policy tree π is composed of an initial action a_π and a $(k-1)$-horizon sub-policy tree π_o for each possible observation. We denote the space of k-horizon policies as $\Pi^{(k)}$.

Value iteration for POMDPs (POMDP-VI) [Cassandra *et al.*, 1994] is based on the linearity of policy tree value functions in the belief state, $V_\pi(h) = b(h)^\top \alpha_\pi^{(b)}$, where $\alpha_\pi^{(b)}$ is the *alpha vector* representing the values of π. The optimal k-horizon value function is the maximum over policy value functions, $V^{*(k)}(h) = \max_{\pi \in \Pi^{(k)}} b(h)^\top \alpha_\pi^{(b)}$, and is notably piecewise linear and convex (PWLC) [Smallwood and Sondik, 1973]. POMDP-VI iteratively computes the alpha vectors α_π of policy trees $\pi \in \Pi^{(k)}$ for increasing horizons using $\alpha_\pi = \left[R^{(b)}\right]_{:a_\pi}$ if $k = 1$, and $\alpha_\pi = \left[R^{(b)}\right]_{:a_\pi} + \gamma \sum_o G_{ao}^\top \alpha_{\pi_o}$ if $k > 1$. In practice, some alpha vectors are dominated by others, and can be pruned to mitigate the exponential growth [Cassandra *et al.*, 1997].

Value iteration can be adapted to PSRs (PSR-VI) [James *et al.*, 2004; Boots *et al.*, 2011] under a linear PSR reward function $R^{(p)}(h, a) \doteq p(h)^\top \left[R^{(p)}\right]_{:a}$. Many properties of POMDP-VI remain valid for PSR-VI, including the linearity of value functions $V_\pi^{(p)} = p(h)^\top \alpha_\pi^{(p)}$, and the PWLC property of the optimal k-horizon value function. PSR-VI iteratively computes the alpha vectors $\alpha_\pi^{(p)}$ of policy trees $\pi \in \Pi^{(k)}$ for increasing horizons using $\alpha_\pi = \left[R^{(p)}\right]_{:a_\pi}$ if $k = 1$, and $\alpha_\pi = \left[R^{(p)}\right]_{:a_\pi} + \gamma \sum_o M_{ao} \alpha_{\pi_o}$ if $k > 1$.

3 The Failure of PSRs as Reward Models

Given any finite POMDP, the respective PSR state $p(h)$ holds sufficient information to represent the probability of future observations. In this section, we show that the same cannot be said about being able to represent future rewards. We develop theory regarding a PSR model's (in)ability to accurately model POMDP rewards.

3.1 Theory of PSR Reward Accuracy

Proposition 1. *For any finite POMDP and its respective PSR, a (linear or non-linear) function $f(p(h), a) \mapsto R^{(b)}(h, a)$ does not necessarily exist. (proof in Appendix).*

For tractability reasons, we consider the family of linear PSR rewards represented by matrix $R^{(p)} \in \mathbb{R}^{|\mathcal{Q}^\dagger| \times |\mathcal{A}|}$, such that $R^{(p)}(h, a) \doteq p(h)^\top \left[R^{(p)}\right]_{:a}$. Ideally, it would always be possible to express the true rewards $R^{(b)}(h, a)$ in such a way. Unfortunately, this is not always possible.

Theorem 1 (Accurate Linear PSR Rewards)**.** *A POMDP reward matrix $R^{(b)}$ can be accurately converted to a PSR reward matrix $R^{(p)}$ iff every column of $R^{(b)}$ is linearly dependent on the core outcome vectors (the columns of U). If this condition is satisfied, we say that the PSR is* accurate*, and $R^{(p)} = U^+ R^{(b)}$ accurately represents the POMDP rewards. (proof in Appendix).*

As a direct consequence of Theorem 1, a PSR can only accurately model a sub-space of POMDP rewards $R^{(b)} \in \{UW \mid W \in \mathbb{R}^{|\mathcal{Q}^\dagger| \times |\mathcal{A}|}\}$. Also, full-rank PSRs are always accurate, while low-rank PSRs—often praised for their representational efficiency—are most likely to suffer from this issue. Corollary 1 shows the reverse problem does not exist.

Corollary 1. *Assuming that a PSR can be represented by a finite POMDP, then any PSR rewards $R^{(p)}$ are accurately represented by POMDP rewards $R^{(b)} = UR^{(p)}$. (proof in Appendix).*

Next, we consider whether it is possible to formulate an appropriate approximation of POMDP rewards for a non-accurate PSR. Ideally, we care for PSR rewards which induce the optimal policy and/or values most similar to those of the POMDP. However, it is extremely challenging to fully analyze the effects of rewards on policies (which is the control problem itself); therefore, we will use reward errors as a proxy. Note that, while this methodology is extremely simple, it is also imperfect, since small reward errors may lead to large policy errors, while large reward errors may lead to small policy errors. While it is possible to consider preferences over histories/beliefs which should result in a lower approximation error (e.g., the reachable histories/beliefs), that kind of prior knowledge is not common. Therefore, we consider reward approximations where the approximation errors for all histories/beliefs are weighted uniformly.

Theorem 2 (Approximate Linear PSR Rewards)**.** *The linear approximation of POMDP rewards for non-accurate PSRs which results in the lowest reward approximation error is $R^{(p)} \doteq U^+ R^{(b)}$. (proof in Appendix).*

Notably, Theorems 1 and 2 share the same expression for $R^{(p)}$, which is unsurprising since an accurate estimate is just an errorless approximate estimate. The ability to compute approximate rewards via Theorem 2 begs the question of whether they are accurate enough to serve as a basis for control. Unfortunately, the connection between PSR rewards and optimal policy is not as straightforward and interpretable as in POMDPs, since the rewards are encoded in terms of test probabilities rather than states. To answer this question quantitatively, one can solve both the POMDP and the approximate PSR, and compare the respective optimal policies/values directly; however, this approach is very expensive. Luckily, Corollary 2 provides a simple qualitative approach which allows one to interpret approximate PSR rewards directly by *reconstructing* equivalent POMDP rewards.

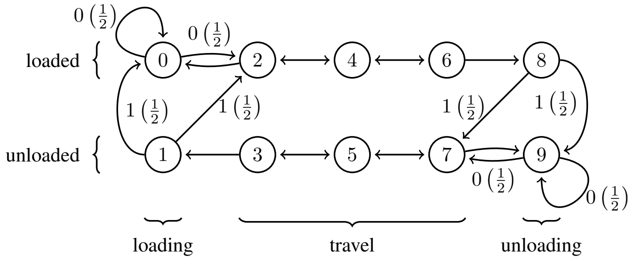

Figure 1: Load/unload domain. Rows indicate the agent's status (*loaded* or *unloaded*), and columns indicate the agent's position (observed as *loaded*, *travel*, and *unloaded*). Movements (*left* and *right*) are deterministic, and non-zero rewards are shown as "x (y)", where x is the POMDP reward, and y is the PSR approximation.

Core test $q \in \mathcal{Q}^\dagger$	Outcome $u(q)^\top$
left loading	(1 1 1 1 0 0 0 0 0 0)
right travel	(1 1 1 1 1 1 0 0 0 0)
right unloading	(0 0 0 0 0 0 1 1 1 1)
right travel, left loading	(1 1 0 0 0 0 0 0 0 0)
left travel, right travel	(0 0 0 0 1 1 1 1 0 0)

Table 1: Core tests $\mathcal{Q}^\dagger$ and respective outcome vectors for load/unload, found using a breadth-first variant of the search algorithm by Littman and Sutton [2002].

Corollary 2. *$\tilde{R}^{(b)} \doteq UU^+R^{(b)}$ is the reconstructed POMDP-form of the PSR approximation $R^{(p)}$ of the true POMDP rewards $R^{(b)}$. $\tilde{R}^{(b)}$ and $R^{(b)}$ are equal iff the accuracy condition is satisfied. (proof in Appendix).*

Next, we provide a detailed case study demonstrating the theory developed here. In Section 5 we show empirically that, if there is any approximation error at all, then there is a high chance that the policy has been altered catastrophically.

3.2 A Case Study of Approximate PSR Rewards

We use the *load/unload* domain (POMDP in Appendix) shown in Figure 1 as a case study to show a catastrophic failure of approximate PSR rewards. The agent navigates a corridor of 5 cells under partial observability of its own position and whether it is carrying a load or not. Rewards are given for loading (when not loaded) at one end of the corridor, and unloading (when loaded) at the other end. The task is to keep moving back and forth between one end of the corridor (loading) and the other (unloading).

The domain has $|\mathcal{S}| = 10$ states, but its PSR rank is $|\mathcal{Q}^\dagger| = 5$. Table 1 shows a core set $\mathcal{Q}^\dagger$ and the respective outcome vectors $u(q)$. By Theorem 1 there is an entire subspace of reward functions which cannot be accurately represented, and there is a chance that the corresponding PSR model is not accurate; we will verify that this is indeed the case. We note that the outcomes in Table 1 are also the columns of matrix U, and that every pair of rows in U (columns in Table 1) is identical. This means that the PSR is unable to make any distinction between states 0 and 1, 2 and 3, etc, which will cause a problem, because the POMDP reward function specifically needs to differentiate between states 0 and 1, and 8 and 9.

Next, we show the POMDP rewards $R^{(b)}$ (note that only states 1 and 8 emit rewards),

$$R^{(b)} = \begin{pmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \end{pmatrix}^\top , \quad (4)$$

the PSR approximation (Theorem 2) $R^{(p)} = U^+R^{(b)}$,

$$R^{(p)} = \begin{pmatrix} 0.5 & -0.5 & -0.5 & 0.5 & 0.5 \\ 0.5 & -0.5 & -0.5 & 0.5 & 0.5 \end{pmatrix}^\top , \quad (5)$$

and the POMDP reconstruction (Corollary 2) $\tilde{R}^{(b)} = UR^{(p)}$ (note that states 0, 1, 8 and 9 emit rewards),

$$\tilde{R}^{(b)} = \begin{pmatrix} 0.5 & 0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.5 \end{pmatrix}^\top . \quad (6)$$

The approximate rewards $R^{(p)}$ are hard to interpret, and it is not obvious that something is wrong. However, looking at the reconstructed rewards $\tilde{R}^{(b)}$ (also shown in parentheses in Figure 1), we notice that the first two and the last two columns—respectively corresponding to states 0 and 1, and 8 and 9—have the same values, indicating that the PSR has lost the ability to distinguish between those states. After all, the only reason why states 0 and 1 are separate states in load/unload is because they lead to different rewards; they are equivalent in all other aspects, hence the PSR "efficiently" merges them. More critically, the optimal behavior prescribed by $\tilde{R}^{(b)}$ (read $R^{(p)}$) has changed catastrophically from that of $R^{(b)}$. With $R^{(b)}$, the optimal behavior is to move back-and-forth between the corridor ends; with $\tilde{R}^{(b)}$ (read $R^{(p)}$), the optimal behavior is to reach one end and stay there.

4 Reward-Predictive State Representations

In this section, we introduce *reward-predictive state representations* (R-PSRs), a generalization of PSRs which accurately models both the observation processes *and* the reward processes of all finite POMDPs. The derivation of R-PSRs resembles closely that of PSRs, with a few key differences: (a) a token action is used to unify observation and reward emissions, (b) tests are extended by a final action used which switches between observation and reward emissions, (c) test probabilities are generalized to include reward information.

Extended Action-Space. We define an extended action-space $\mathcal{Z} \doteq \mathcal{A} \cup \{\zeta\}$, where ζ is a token action, and extend the reward function such that $R(s, \zeta) = 1$ for every state s. Token action ζ (with its reward) is a construct which will allow us to define a single model of both observation and reward emissions; however, we are not changing the space of actions available to the agent, and ζ cannot be used to interact with the environment, nor can it be part of a history or test.

Intents and their Expectations. In R-PSRs, the system state incorporates the expected rewards of hypothetical futures we call *intents*. We define an *intent* qz as a test q followed by an extended action z, and the space of all intents as $\mathcal{I} \doteq \mathcal{Q} \times \mathcal{Z}$. An *intent reward* $r(qz \mid h) \doteq \Pr(q \mid h)R(hq, z)$

Algorithm 1 Depth-first search of a maximal set of linearly independent intents $\mathcal{I}^\dagger$.

```
Require: $q \in \mathcal{Q}$ (optional, default $\varepsilon$)
Require: Independent intents $I \subset \mathcal{I}$ (optional, default $\varnothing$)
Ensure: Maximal set of independent intents which either belong to $I$ or are extensions of $q$.
  function ISearchDFS($q, I$)
      for all $z \in \mathcal{Z}$ do
          if $u(qz)$ independent of $\{u(i) \mid i \in I\}$ then
              for all $a \in \mathcal{A}$, $o \in \mathcal{O}$ do
                  $I \leftarrow$ ISearchDFS($aoq, I \cup \{qz\}$)
      return $I$
```

is the test probability multiplied by the expected reward obtained by the intent action following the concatenated history-test. Intent rewards are the core quantity modeled by R-PSRs, and a *linear* reward-predictive state $r(h) \in \mathbb{R}^{|\mathcal{I}^\dagger|}$ (where $\mathcal{I}^\dagger$ is a *core* set of intents, defined later) is a representation of history h such that intent rewards are linear in $r(h)$, i.e., $r(qz \mid h) = r(h)^\top m_{qz}$, where $m_{qz} \in \mathbb{R}^{|\mathcal{I}^\dagger|}$ is the *parameter vector* associated to intent qz. A focal property of $r(qz \mid h)$, provided by the token action ζ, is that it generalizes both test probabilities and history rewards,

$$R(h, \zeta) = 1 \qquad \Longrightarrow \qquad r(q\zeta \mid h) = \Pr(q \mid h)\,, \tag{7}$$

$$\Pr(\varepsilon \mid h) = 1 \qquad \Longrightarrow \qquad r(\varepsilon a \mid h) = R(h, a)\,. \tag{8}$$

The intent reward function can be expressed in a vectorized form similar to that of PSRs (proof in Appendix),

$$r(qz \mid h) = b(h)^\top G_q^\top \left[R^{(b)}\right]_{:z}\,. \tag{9}$$

Outcome Vectors. The *outcome* of an intent $u(qz) \in \mathbb{R}^{|\mathcal{S}|}$ is a vector indicating the intent rewards from each state as starting point, i.e., $[u(qz)]_i = \mathbb{E}\left[\mathrm{R}(\boldsymbol{s'}, a) \mid s = i, q\right]$. Outcomes can be defined recursively using the generative matrix,

$$u(\varepsilon a) = \left[R^{(b)}\right]_{:a}\,, \tag{10}$$

$$u(aoqz) = G_{ao}^\top u(qz)\,. \tag{11}$$

Combining Equations (9) to (11) results in $r(qz \mid h) = b(h)^\top u(qz)$, i.e., the intent reward given a history is the expectation of intent rewards given each state. A set of intents is said to be *linearly independent* iff the respective outcome vectors are linearly independent, and any maximal set of linearly independent intents is called a *core set*, denoted as $\mathcal{I}^\dagger$. R-PSR core sets share similar properties to PSR core sets: there are infinite core sets which share the same size $|\mathcal{I}^\dagger|$, called the R-PSR rank, which is upper-bounded by $|\mathcal{I}^\dagger| \leq |\mathcal{S}|$. Note that the PSR rank and the R-PSR rank are not necessarily equal. Algorithm 1 is a depth-first search algorithm to find a core set, analogous to the original PSR search algorithm [Littman and Sutton, 2002], but adapted for R-PSRs (see Appendix for a more efficient breadth-first variant).

Reward-Predictive States. The outcome matrix $U \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{I}^\dagger|}$ of a core set $\mathcal{I}^\dagger$ is the column-wise stacking of the core outcome vectors $\{u(qz) \mid qz \in \mathcal{I}^\dagger\}$. By the definition of a core set, the outcome $u(qz)$ of any intent qz is a linear combination of the core outcome matrix U columns (else the core set would not be maximal), i.e., $u(qz) \in \operatorname{col} U$ and, because UU^+ is the projection onto $\operatorname{col} U$, then $u(qz) = UU^+ u(qz)$. Consequently, $r(qz \mid h) = b(h)^\top u(qz) = b(h)^\top UU^+ u(qz) = r(h)^\top m_{qz}$ where $r(h)^\top \doteq b(h)^\top U$ is the reward-predictive state, and $m_{qz} \doteq U^+ u(qz)$ is the parameter vector of qz. Each dimension of $r(h)$ is itself the intent reward of a core intent, i.e., $[r(h)]_i = r(qz \mid h)$, where qz is the i^{th} core intent.

Emissions and Dynamics. According to Equations (7) and (8), observation probabilities and reward emissions are $\Pr(o \mid h, a) = r(ao\zeta \mid h) = r(h)^\top m_{ao\zeta}$ and $R(h, a) = r(\varepsilon a \mid h) = r(h)^\top m_{\varepsilon a}$ respectively. Therefore, the test-less intent parameters $\{m_{\varepsilon a} \mid a \in \mathcal{A}\}$ constitute the columns of reward matrix $R^{(r)}$. The reward-predictive state dynamics are $r(hao) = \left(r(h)^\top M_{ao}\right) / \left(r(h)^\top m_{ao\zeta}\right)$, where M_{ao} is the column-wise stacking of the extended core intent parameters $\{m_{aoqz} \mid qz \in \mathcal{I}^\dagger\}$ (proof in Appendix).

Value Iteration for R-PSRs. Value iteration can be adapted to R-PSRs (R-PSR-VI). Both the derivation and the equation to compute the alpha vectors $\alpha_\pi^{(r)}$ are identical to those of PSR-VI [James *et al.*, 2004; Boots *et al.*, 2011], also shown in Section 2.3; the only difference being that parameters $R^{(r)}$ and $M_{ao}^{(r)}$ are used, whose values correctly represent the POMDP/R-PSR rewards.

5 Evaluation

We perform empirical evaluations to confirm the theory developed in this work, the issues with PSRs, and the validity of R-PSRs. We show that (a) a non-trivial portion of finite POMDPs used in classical literature do not satisfy the accuracy condition of Theorem 1, i.e., this is a common problem occurring in common domains; (b) PSR-VI based on inaccurate approximate PSR rewards results in catastrophically sub-optimal policies, i.e., approximate rewards are not viable for control; (c) R-PSRs are accurate reward models; and (d) R-PSR-VI results in the same optimal policies as POMDP-VI. Our evaluation involves a total of 63 unique domains: 60 are taken from Cassandra's POMDP page [Cassandra, 1999], a repository of classic finite POMDPs from the literature; 2 are the well-known *load/unload* [Meuleau *et al.*, 1999] and *heaven/hell* [Bonet, 1998]; and the last one is *float/reset* [Littman and Sutton, 2002].

We found that 8 out of these 63 domains—a non-trivial amount—do not satisfy the accuracy condition. Table 2 shows the reward errors between original and reconstructed POMDP rewards. We note that all domains where accurate PSR rewards are not possible also have high relative errors, which implies that inaccurate PSRs are unlikely to be only *mildly* inaccurate. VI is a fairly expensive algorithm which does not scale well with long planning horizons and is thus not suitable for all problems; convergence to a steady optimal value function was possible within a reasonable timeframe for only 6 of the 8 domains. For each of these 6 domains, we run 4 different policies for 1000 episodes of 100 steps: the uniform policy, and the policies respectively obtained by POMDP-VI, PSR-VI, and R-PSR-VI. Every

	4x3	*heaven/hell*	*iff*	*line4-2goals*	*load/unload*	*paint*	*parr*	*stand-tiger*
d_∞	1.0	1.0	48.93	0.6	0.5	1.33	1.0	65.0
rel-d_∞	1.0	1.0	0.75	0.75	0.5	1.33	0.5	0.65

Table 2: PSR reward errors. Measure $d_\infty \doteq \|R^{(b)} - \tilde{R}^{(b)}\|_\infty$ is the ℓ_∞ distance between the POMDP rewards and their reconstruction. Relative measure rel-$d_\infty \doteq {}^{d_\infty}/_{\|R^{(b)}\|_\infty}$ is normalized w.r.t. the scale of POMDP rewards. The R-PSR reward errors (omitted) are all zero.

Domain	Model	Random	POMDP-VI	PSR-VI	R-PSR-VI
heaven/hell	POMDP/R-PSR	0.0 ± 0.1	$\mathbf{1.4 \pm 0.0}$	0.0 ± 0.0	$\mathbf{1.4 \pm 0.0}$
	PSR	$\mathbf{-0.0 \pm 0.0}$	$\mathbf{-0.0 \pm 0.0}$	$\mathbf{-0.0 \pm 0.0}$	$\mathbf{-0.0 \pm 0.0}$
line4-2goals	POMDP/R-PSR	$\mathbf{0.4 \pm 0.0}$	$\mathbf{0.4 \pm 0.0}$	$\mathbf{0.4 \pm 0.0}$	$\mathbf{0.4 \pm 0.0}$
	PSR	$\mathbf{4.0 \pm 0.0}$	$\mathbf{4.0 \pm 0.0}$	$\mathbf{4.0 \pm 0.0}$	$\mathbf{4.0 \pm 0.0}$
load/unload	POMDP/R-PSR	1.2 ± 0.5	$\mathbf{4.5 \pm 0.1}$	0.6 ± 0.2	$\mathbf{4.5 \pm 0.1}$
	PSR	4.0 ± 1.0	2.6 ± 0.1	$\mathbf{9.1 \pm 0.5}$	2.6 ± 0.1
paint	POMDP/R-PSR	-4.2 ± 1.4	$\mathbf{3.3 \pm 0.3}$	0.0 ± 0.0	$\mathbf{3.3 \pm 0.3}$
	PSR	-3.2 ± 1.0	1.0 ± 0.9	$\mathbf{3.3 \pm 0.0}$	1.0 ± 1.0
parr	POMDP/R-PSR	4.3 ± 1.7	$\mathbf{7.1 \pm 0.0}$	6.5 ± 1.8	$\mathbf{7.1 \pm 0.0}$
	PSR	4.3 ± 0.8	3.6 ± 0.0	$\mathbf{6.3 \pm 0.0}$	3.6 ± 0.0
stand-tiger	POMDP/R-PSR	-122.3 ± 43.1	$\mathbf{49.2 \pm 23.4}$	0.0 ± 0.0	$\mathbf{49.8 \pm 23.2}$
	PSR	-122.7 ± 26.4	-151.1 ± 17.6	$\mathbf{0.0 \pm 0.0}$	-150.2 ± 18.0

Table 3: Policy return estimates for each policy (columns) by each model (rows), where the identical POMDP and R-PSR rows were merged. Means and standard deviations shown as $\mu \pm \sigma$. Bold text indicates, for each model, the highest performing policy.

action-observation sequence is then evaluated by the POMDP, PSR, and R-PSR reward models. Table 3 shows the return estimates for POMDPs, PSRs, and R-PSRs; Note that the POMDP and R-PSR rows are combined since they contain the same values.

In this context, the POMDP represents the true task, which the PSR and R-PSR also attempt to encode; hence, the POMDP rows show how well each model's respective policy solves the original task, while the PSR rows show how the PSR's (inaccurate) encoded task evaluates each model's respective policy. Notably, POMDPs and R-PSRs consistently agree with high numerical precision on the return values of all trajectories, whereas PSRs consistently disagree. The POMDP-VI and R-PSR-VI policies achieve the same values throughout all experiments, i.e., they consistently converge to the same policies. Since the POMDP encodes the true task, both POMDP-VI and R-PSR-VI represent the optimal policy which solves that task. With the singular exception of *line4-2goals*, a trend appears where the PSR-VI policy is sub-optimal and, in the case of *load/unload*, is even worse than the random policy (see Section 3.2). Vice versa, the POMDP-VI/R-PSR-VI policies are sub-optimal according to the PSR model; notably, the random policy performs better than the POMDP-VI/R-PSR-VI policies in 3 out of 6 cases, which underlines just how much the task encoded by the PSRs have diverged from their original form.

These results reaffirm not only the theory developed in this document, i.e., that PSRs are equivalent to POMDPs only in relation to their observation process and not their reward process, but also that this is a common problem which causes significant control issues. Further, the results confirm the validity of the developed R-PSR theory, and the equivalence between POMDPs and R-PSRs. Overall, this confirms that R-PSRs are better suited for control, compared to vanilla PSRs.

6 Conclusions

In this work, we presented theoretical results on the accuracy of PSR rewards relative to POMDP rewards, identified a sufficient and necessary condition which determines whether a PSR can accurately represent POMDP rewards, and derived the closest linear approximate rewards for non-accurate PSRs. We also showed empirically that reward approximations are likely to warp the implied task in undesirable ways. Therefore, we proposed R-PSRs, a generalization of PSRs which encodes reward values jointly with test probabilities, and solves the reward modeling problem of vanilla PSRs while remaining faithful to the idea of grounding a system state representation on non-latent quantities, and avoiding the pitfalls of observable rewards. R-PSRs combine the benefits of both POMDPs and PSRs: Compared to POMDPs, R-PSRs do not rely on a latent state, which makes model learning easier and grounded in non-latent quantities; Compared to PSRs, R-PSRs are able to model a wider range of tasks, and the reward structure of any finite POMDP. In future work, we aim to adapt more learning and planning algorithms to R-PSRs, and address the *discovery* problem, i.e., the problem of learning a core set of intents from sample interactions.

Acknowledgments

This research was funded by NSF award 1816382.

References

[Bonet, 1998] Blai Bonet. Solving large POMDPs using real time dynamic programming. In *Proceedings of AAAI Fall Symposium on POMDPs*, 1998.

[Boots *et al.*, 2011] Byron Boots, Sajid M. Siddiqi, and Geoffrey J. Gordon. Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research*, 30(7):954–966, 2011.

[Bowling *et al.*, 2006] Michael Bowling, Peter McCracken, Michael James, James Neufeld, and Dana Wilkinson. Learning predictive state representations using non-blind policies. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 129–136. ACM, 2006.

[Cassandra *et al.*, 1994] Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the 12th AAAI National Conference on Artificial Intelligence*, volume 94, pages 1023–1028, 1994.

[Cassandra *et al.*, 1997] Anthony R. Cassandra, Michael L. Littman, and Nevin Lianwen Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pages 54–61, 1997.

[Cassandra, 1999] Anthony R. Cassandra. Tony's POMDP file repository page, 1999.

[Izadi and Precup, 2003] Masoumeh T. Izadi and Doina Precup. A planning algorithm for predictive state representations. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1520–1521, 2003.

[Izadi and Precup, 2008] Masoumeh T. Izadi and Doina Precup. Point-based planning for predictive state representations. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 126–137. Springer, 2008.

[James and Singh, 2004] Michael R. James and Satinder Singh. Learning and discovery of predictive state representations in dynamical systems with reset. In *Proceedings of the 21st International Conference on Machine Learning*, page 53. ACM, 2004.

[James *et al.*, 2004] Michael R. James, Satinder Singh, and Michael L. Littman. Planning with predictive state representations. In *Proceedings of the International Conference on Machine Learning and Applications*, pages 304–311. IEEE, 2004.

[James *et al.*, 2006] Michael R James, Ton Wessling, and Nikos Vlassis. Improving approximate value iteration using memories and predictive state representations. In *Proceedings of the 21st National Conference on Artificial Intelligence*, volume 1, pages 375–380, 2006.

[Littman and Sutton, 2002] Michael L. Littman and Richard S. Sutton. Predictive representations of state. In *Advances in Neural Information Processing Systems*, volume 14, pages 1555–1561, 2002.

[McCracken and Bowling, 2006] Peter McCracken and Michael Bowling. Online discovery and learning of predictive state representations. In *Advances in Neural Information Processing Systems*, pages 875–882, 2006.

[Meuleau *et al.*, 1999] Nicolas Meuleau, Leonid Peshkin, Kee-Eung Kim, and Leslie Pack Kaelbling. Learning finite-state controllers for partially observable environments. In *Proceedings of the 15th conference on Uncertainty in artificial intelligence*, pages 427–436, 1999.

[Rosencrantz *et al.*, 2004] Matthew Rosencrantz, Geoff Gordon, and Sebastian Thrun. Learning low dimensional predictive representations. In *Proceedings of the 21st International Conference on Machine Learning*, page 88. ACM, 2004.

[Singh *et al.*, 2003] Satinder P. Singh, Michael L. Littman, Nicholas K. Jong, David Pardoe, and Peter Stone. Learning predictive state representations. In *Proceedings of the 20th International Conference on Machine Learning*, pages 712–719, 2003.

[Singh *et al.*, 2004] Satinder Singh, Michael R. James, and Matthew R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Proceedings of the 20th conference on Uncertainty in Artificial Intelligence*, pages 512–519. AUAI Press, 2004.

[Smallwood and Sondik, 1973] Richard D. Smallwood and Edward J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations research*, 21(5):1071–1088, 1973.

[Wiewiora, 2005] Eric Wiewiora. Learning predictive representations from a history. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 964–971, 2005.

[Wolfe *et al.*, 2005] Britton Wolfe, Michael R. James, and Satinder Singh. Learning predictive state representations in dynamical systems without reset. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 980–987. ACM, 2005.