Learning and Discovery of Predictive State Representations in Dynamical Systems with Reset

Michael R. James and Satinder Singh
Computer Science and Engineering
University of Michigan
Ann Arbor, MI 48109
{mrjames,baveja}@umich.edu

Abstract

Predictive state representations (PSRs) have been recently proposed as a new way of modeling controlled dynamical systems. PSR-based models use *predictions* of the observable outcomes of *tests* that could be done on the system as their state representation, and have model parameters that define how the predictive state representation changes over time as actions are taken and observations noted. Learning PSR-based models requires solving two subproblems: 1) *discovery* of the tests whose predictions constitute state, and 2) *learning* the model parameters that define the dynamics. So far, there have been no results available on the discovery subproblem while for the learning subproblem an approximate-gradient algorithm has been proposed [7] with mixed results (it works on some domains and not on others). In this paper, we provide the first PSR-discovery algorithm and a new learning algorithm for the special class of controlled dynamical systems that have a *reset* operation. We provide preliminary experimental verification of our algorithms.

1 Introduction

Predictive state representations (PSRs; Littman, Sutton, & Singh [4]) have been recently proposed as a new way (inspired by the work of Jaeger [2] and Rivest & Schapire [6]) of building models of controlled dynamical systems. PSRs capture the state of such a system as a vector of predictions or outcome probabilities for tests (or experiments) that one can do on the system. A test is a sequence of action-observation pairs and its prediction is the probability of the test's observation-sequence happening if the test's action-sequence were to be executed on the system. PSR-based models have parameters that define how the predictive state representation changes over time as actions are executed and observations noted. A novel[1] aspect of PSR-based models is that their state is expressed entirely in

[1]The state in history-window based or n^{th}-order Markov models is also expressed in terms of observable quantities but these models are less flexible than PSR models [4] and won't be considered further in this paper.

terms of observable quantities. In contrast, partially observable Markov decision process or POMDP-based models express state in terms of probability distributions over unobservable and often hypothetical underlying-states of the system (see, e.g., [5, 3]). Despite this key difference, PSR-models are as flexible and powerful as POMDP models; indeed, Littman et al. showed that any dynamical system that can be modeled as a POMDP can also be modeled as a PSR of size no larger than that of the POMDP model. On the other hand, because of this key difference, there is a pressing and yet unmet need to develop algorithms for learning PSR-models from streams of experience because existing POMDP-based model-learning methods scale quite poorly [4, 8].

Learning a PSR-based model from data has two important subproblems [4]: 1) *discovery* of the tests whose predictions constitute state, and 2) *learning* the model parameters that define how the predictive state representation is updated over time as actions are executed and observations noted. So far, there have been no results for the discovery problem, while for the learning problem an approximate gradient-based algorithm has been proposed by Singh et al. [7] with mixed results (it works on some domains and not on others). In this paper, we present the first PSR-discovery algorithm and a new learning algorithm, both for the restricted class of dynamical systems that have a reset. We present evaluations of our algorithms on the same domains (with an added reset) that the previous gradient algorithm got mixed results on.

2 Preliminaries

We consider finite, discrete-time, stationary, and controlled dynamical systems, henceforth simply dynamical systems, that accept actions from a discrete set $\mathcal{A}$, and produce observations from a discrete set $\mathcal{O}$. Letting $a_i \in \mathcal{A}$ and $o_i \in \mathcal{O}$ denote the action and observation at time i, the probability of some *history* $h = \{a^1 o^1 a^2 o^2 \ldots a^n o^n\}$ is the conditional probability of the observation sequence $o^1 o^2 \ldots o^n$ being obtained if the action sequence $a^1 a^2 \ldots a^n$ were executed from time 1 onwards, i.e., $P(h) = prob(o_1 = o^1, o_2 = o^2, \ldots, o_n = o^n | a_1 = a^1, a_2 = a^2, \ldots, a_n = a^n)$. A dynamical system is characterized by a probability distribution over all possible histories, $P : \{\mathcal{AO}\}^* \rightarrow [0, 1]$. A *test*, like a history, is also a sequence of action-observation pairs, but unlike a history, a test is not constrained to start at time 1. The conditional probability of a test $t = a^1 o^1 a^2 o^2 \ldots a^m o^m$ at some history h (w.l.o.g., of length n), is $P(t|h) = P(ht)/P(h) = prob(o_{n+1} = o^1, o_{n+2} = o^2, \ldots, o_{n+m} = o^m | h, a_{n+1} = a^1, a_{n+2} = a^2, \ldots, a_{n+m} = a^m)$, i.e., it is the probability of obtaining the test's observation-sequence if the test's action-sequence were executed from history h onwards. For ease of exposition, henceforth, we will abbreviate expressions like the right-hand side of the definition of $p(t|h)$ above as simply $prob(o^1 o^2 \ldots o^m | h a^1 a^2 \ldots a^m)$.

A set of tests $Q = \{q^1 q^2 \ldots q^{|Q|}\}$ constitutes a PSR if its prediction-vector, $P(Q|h) = [P(q^1|h) P(q^2|h) \ldots P(q^{|Q|}|h)]$ is a sufficient statistic for all histories, i.e., for any test t, $P(t|h) = f_t(P(Q|h))$ for some function f_t — i.e., $P(Q|h)$ captures all the information in h relevant to predicting the future. We will distinguish such a set of tests that constitutes a PSR by calling it a *core* set of tests and reserving the label Q for it. In general, the functions f_t can be linear or non-linear. In *linear PSRs* the functions f_t are linear for all t, i.e.,

$$\forall t \; P(t|h) = P^T(Q|h) w_t, \quad \text{for some weight vector } w_t \text{ of dimension } |Q| \times 1 \tag{1}$$

(by default vectors are assumed to be column vectors and so the transpose operator, $\cdot^T$, is used to obtain row vectors). In non-linear PSRs f_t is not linear for at least some t. On taking action a from some history h and observing o, one can update the prediction-vector for Q as follows: for each $q^i \in Q$

$$P(q^i|hao) = \frac{P(aoq^i|h)}{P(ao|h)} = \frac{f_{aoq^i}(P(Q|h))}{f_{ao}(P(Q|h))} = \frac{P^T(Q|h) m_{aoq^i}}{P^T(Q|h) m_{ao}} \tag{2}$$

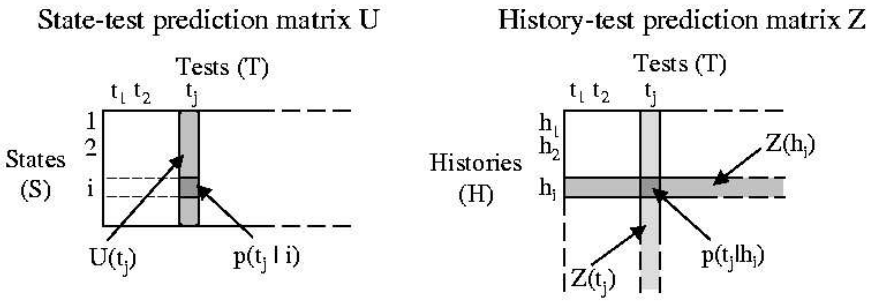

Figure 1: Graphical representation of state-test prediction matrix and history-test prediction matrix, with associated definitions

where we used the fact that aoq^i is just another test, and where the last equality above applies only to the linear PSR case. Therefore, a linear-PSR model is specified by core-set Q and the *model-parameters* which are the the weight vectors m_{aot} (all of size $|Q| \times 1$) for all $a \in \mathcal{A}$, all $o \in \mathcal{O}$ and for all $t \in Q \cup \phi$ (where the null string ϕ was added to take care of the parameters in the denominator of the right-hand side of Equation 2). Learning a PSR model from data involves two subproblems: the **discovery** of a core-set Q, and **learning** the model-parameters.

In building a POMDP representation of a dynamical system one starts by hypothesizing a set $\mathcal{S}$ of unobservable states. For any history h, the counterpart to the prediction vector $P(Q|h)$ is the belief-state $b(\mathcal{S}|h) = [prob(1|h)\ prob(2|h)\ \ldots\ prob(|\mathcal{S}||h)]$ that represents the probability of the system being in each underlying state of $\mathcal{S}$ at history h. The model-parameters are a set of transition-probability matrices T^a (for all $a \in \mathcal{A}$) such that T^a_{ij} is the probability of a transition to state $j \in \mathcal{S}$ upon taking action a in state $i \in \mathcal{S}$, and a set of diagonal observation probability matrices $O^{a,o}$ for all $a \in \mathcal{A}$ and $o \in \mathcal{O}$ such that $O^{a,o}_{ii}$ is the probability of receiving observation o upon taking action a and transitioning to state i. On taking action a in history h and observing o, the belief-state can be updated as follows:

$$b^T(\mathcal{S}|hao) = \frac{b^T(\mathcal{S}|h)T^aO^{a,o}}{b^T(\mathcal{S}|h)T^aO^{a,o}1_{|\mathcal{S}|}}$$

where 1_n is a $n \times 1$ vector of all 1's. Learning a POMDP model from data also involves discovery and learning subproblems, where the discovery problem is to figure out how many states to use, while the learning problem is to estimate the model-parameters.

We can relate PSR-models to POMDP-models using a **conceptual** *state-test prediction matrix* U (see Figure 1) whose rows correspond to the states in $\mathcal{S}$ and whose columns correspond to all possible tests arranged in order of increasing size and within a size in some lexicographic order (we let $\mathcal{T}$ denote the set of all possible tests in this ordering). The entry U_{ij} is the prediction for the test t_j (the one associated with column j), given that the state of the system is i. Then, by construction, for any h and any t_j, $P(t_j|h) = b^T(S|h)U(t_j)$, where $U(t_j)$ is the j^{th} column of U, and for any set of tests C, the vector $P^T(C|h) = b^T(S|h)U(C)$ where $U(C)$ is the sub-matrix corresponding to the tests in C. Thus, the matrix U allows us to translate belief-states into prediction vectors.

We can also motivate and derive core-tests Q from the U matrix as follows. Let M be the set of tests corresponding to a maximal-set of linearly-independent columns of U. Then M must be core-tests for a linear-PSR representation, i.e., M must be Q, because for any

test t, $U(t) = U(M)w_t$, for some vector of weights w_t (by construction of M the column in U for any test must be linearly dependent on the sub-matrix $U(M)$), and therefore for all t, $p(t|h) = b^T(S|h)U(t) = b^T(S|h)U(M)w_t = P^T(M|h)w_t$ which is exactly the requirement for M to constitute a linear-PSR (see Equation 1). Note that the number of linearly independent columns is the rank of the matrix U and is therefore upper-bounded by the number of rows $|\mathcal{S}|$. Thus, through this conceptual matrix U we have proven that for any POMDP, there exists a linear-PSR with $\leq |\mathcal{S}|$ core-tests and that the linearly-independent columns of U correspond to a core-set Q. Littman et al. also prove by a constructive argument that there exists a core-set Q in which no test is longer than $|\mathcal{S}|$. These results together suggest the power and flexibility of PSRs as a representation.

3 Discovery and Learning using History-Test Prediction Matrix

In this section we derive a discovery and learning algorithm for dynamical systems with reset and provide some partial analysis supporting our algorithms.

First, note that if we could somehow estimate from action-observation sequence data the state-test prediction matrix U, defined above, then we could discover core-tests Q by finding the linearly-independent columns of the estimated U. But, we cannot directly estimate U because its rows correspond to a set $\mathcal{S}$ that is unobservable. A key insight in this paper is to replace the state-test prediction matrix U used in the original PSR paper [4] by a new *history-test prediction matrix* $\mathcal{Z}$ that as we will show can be estimated from data in certain classes of dynamical systems (e.g., those with reset), and that has many of the same desirable properties that U has. The history-prediction matrix $\mathcal{Z}$ (see Figure 1) has columns corresponding to the tests in $\mathcal{T}$ just as for U, but has rows corresponding to all possible histories $\mathcal{H}$ instead of the hypothetical $\mathcal{S}$ as in U. Note that $\mathcal{H} = \mathcal{T} \cup \phi$ where ϕ is the null history. The entry $\mathcal{Z}_{ij} = p(t_j|h_i)$, is the prediction of the test t_j given history h_i. Note that $\mathcal{Z}$, unlike U, is expressed entirely in terms of observable quantities and thus, as we will show next, can be directly used to derive discovery and learning algorithms.

But first, some notation to allow reference to submatrices of U and $\mathcal{Z}$. For any set of tests C and any set of histories/states A: 1) $\mathcal{Z}(C)$ and $U(C)$ will refer to the submatrices of $\mathcal{Z}$ and U respectively containing the columns corresponding to C, 2) $\mathcal{Z}(A)$ and $U(A)$ will refer to the submatrices of $\mathcal{Z}$ and U respectively containing all the rows corresponding to A, and 3) $\mathcal{Z}(C, A)$ and $U(C, A)$ will be the submatrix defined as the intersection of $\mathcal{Z}(C)$ and $\mathcal{Z}(A)$ (respectively, the intersection of $U(C)$ and $U(A)$). Also, let Q_T denote the tests corresponding to the linearly independent columns and let Q_H be the histories corresponding to the linearly independent rows of $\mathcal{Z}$.

Lemma 1 *For any dynamical system, the tests corresponding to the linearly-independent columns of $\mathcal{Z}$ constitute a linear-PSR. Furthermore, the size of the linear-PSR derived from $\mathcal{Z}$ will be no more than the size of the linear-PSR derived from U.*

Proof For any test t, $\mathcal{Z}(t) = \mathcal{Z}(Q_T)w_t$ by the definition of Q_T. $P(t|h)$ is the element corresponding to row h in vector $\mathcal{Z}(t)$, while $P^T(Q_T|h)$ is the row of $\mathcal{Z}(Q_T)$ corresponding to h. Therefore, $P(t|h) = P^T(Q_T|h)w_t$, which in turn implies that the set Q_T constitutes a linear-PSR (see Equation 1).

Let the rank of U be k and let the rank of $\mathcal{Z}$ be n. Then, for all $h \in Q_H$, $\mathcal{Z}(h) = b^T(S|h)U(\mathcal{T}) = w_h^T U'(\mathcal{T})$ where $U'(\mathcal{T})$ is a $(k \times \infty)$ matrix derived by stripping away the linearly dependent rows in $U(\mathcal{T})$ and w_h is a vector of length k. Therefore, the submatrix $\mathcal{Z}(Q_H) = W_{Q_H}^T U'(\mathcal{T})$ where the weight matrix $W_{Q_H}^T$ has dimension $n \times k$ and is composed by piling up the row vectors w_h^T for every $h \in Q_H$. Clearly, the rank of W_{Q_H} is an upper bound on the rank of $\mathcal{Z}(Q_H)$ and hence n must be $\leq k$. $\square$

Lemma 1 shows that Q_T constitutes a linear-PSR, and hence we will refer to Q_T as core-tests. By analogy we will refer to Q_H as core histories.

Lemma 2 *The set of belief-state vectors corresponding to the set Q_H of core-histories are linearly independent.*

Proof Let H be an arbitrary finite set of histories, and let $b(\mathcal{S}|H)$ be the associated $|\mathcal{S}| \times |H|$ belief-state matrix (each column is a belief-state). If the belief-state for some history h, denoted $b(\mathcal{S}|h)$ is linearly dependent on the set of belief-states associated with the histories in H, i.e., if $b^T(S|h) = w^T b^T(\mathcal{S}|H)$, then $\mathcal{Z}(h) = P^T(\mathcal{T}|h) = b^T(\mathcal{S}|h)U(\mathcal{T}) = w^T b^T(\mathcal{S}|H)U(\mathcal{T}) = w^T P(\mathcal{T}|H) = w^T \mathcal{Z}(H)$. In other words if the belief-state for some history h is linearly dependent on the belief-states of some set of histories H, then the row corresponding to h in $\mathcal{Z}$ will be linearly dependent on the rows corresponding to histories H in $\mathcal{Z}$. Contrapositively, this implies that if we find a set of rows of $\mathcal{Z}$ that are linearly independent of each other then their associated belief-states are linearly independent. □

Lemma 2 is interesting because it shows that core-histories are a kind of *basis* set for the space of histories $\mathcal{H}$, in that the belief-state vectors corresponding to core-histories span the full space of feasible belief-state vectors.

Lemma 3 *The set of tests and histories corresponding to the linearly independent columns and rows of any submatrix of $\mathcal{Z}$ are subsets of core-tests and core-histories respectively.*

Proof A set of columns (rows) that are linearly independent in any submatrix of $\mathcal{Z}$ are also linearly independent in the full matrix $\mathcal{Z}$. The proof follows from Lemma's 1 and 2 □

Lemma 3 frees us from having to deal with the infinite matrix $\mathcal{Z}$ and instead allows us to consider finite, and hopefully small, submatrices of $\mathcal{Z}$ in building discovery and learning algorithms. We consider discovery first.

3.1 Discovery Algorithm

If we could estimate elements of $\mathcal{Z}$ from data, then we could estimate a large enough (we discuss what constitutes large enough below) submatrix of $\mathcal{Z}$ and find its linearly independent columns and thereby discover core-tests. So next we consider how to estimate an element of $\mathcal{Z}$, and this is where we motivate the need for a reset. The key is that in dynamical systems with reset we can repeat history and thereby repeatedly generate an *iid* sample from $p(t|h)$ for any (feasibly-sized) t and h as follows:

Algorithm *Sample* $p(t|h)$
generate h:
repeat until success
reset
execute action sequence in h
if actual observation sequence matches observation-sequence in h, then success else failure
end
try test t:
do action sequence in t
if actual observation sequence matches observation-sequence in t, then return 1 else return 0

By generating enough samples from some $\mathcal{Z}_{ij} = p(t_j|h_i)$ and averaging the samples obtained, we can generate arbitrarily accurate estimates of $\mathcal{Z}_{ij}$. The high-level idea of our discovery algorithm is to incrementally increase the size of the submatrix of $\mathcal{Z}$ to be estimated until a stopping condition is met. Our iterative algorithm maintains a set of currently found core-tests and core-histories in $\hat{Q}_T$ and $\hat{Q}_H$ respectively. At iteration i it considers all

one-step extensions $\mathcal{H}_i$ to the histories in $\hat{Q}_T$ and all one-step extensions $\mathcal{T}_i$ to the tests in $\hat{Q}_T$. It uses the sampling algorithm above to estimate the submatrix $\mathcal{Z}(\hat{Q}_H \cup \mathcal{H}_i, \hat{Q}_T \cup \mathcal{T}_i)$ and computes its linearly independent rows and columns and saves the corresponding histories and tests in Q_H and Q_T. It stops when an iteration adds no new tests and histories. We present our algorithm in pseudo-code form below:

Algorithm *Discovery*
$i \leftarrow 1$; $\hat{Q}_H \leftarrow \{\}$; $\hat{Q}_T \leftarrow \{\}$
do forever
$\mathcal{H}_i \leftarrow$ all one-step extensions of $\hat{Q}_H$
$\mathcal{T}_i \leftarrow$ all one-step extensions of $\hat{Q}_T$
estimate $\mathcal{Z}(\mathcal{H}_i \cup \hat{Q}_H, \mathcal{T}_i \cup \hat{Q}_T)$ (and denote estimate as $\hat{\mathcal{Z}}$)
Rank(i) $\leftarrow$ rank of $\hat{\mathcal{Z}}(\mathcal{H}_i \cup \hat{Q}_H, \mathcal{T}_i \cup \hat{Q}_T)$
$\hat{Q}_H \leftarrow$ histories for linearly independent rows in $\hat{\mathcal{Z}}(\mathcal{H}_i \cup \hat{Q}_H, \mathcal{T}_i \cup \hat{Q}_T)$
$\hat{Q}_T \leftarrow$ tests for linearly independent columns in $\hat{\mathcal{Z}}(\mathcal{H}_i \cup \hat{Q}_H, \mathcal{T}_i \cup \hat{Q}_T)$
if Rank(i) = Rank(i-1), return $\hat{Q}_H$, and $\hat{Q}_T$ **(stopping condition)**
else $i \leftarrow i + 1$

We conjecture that our discovery algorithm is provably correct (for accurate estimates of the submatrices of $\mathcal{Z}$) because it is a "two-dimensional" adaptation of Littman et al.'s provably correct "one-dimensional" algorithm for finding core tests given the unrealistic ability to generate columns of the U matrix. Here the two and one dimensions refer to the fact that $\mathcal{Z}$ stretches infinitely in both rows and columns while U stretches infinitely only in columns. We were unable, however, to complete the proof so far. We did directly test our conjecture on a number of controlled dynamical systems with the results reported in Section 4.1.

Next we turn to defining our learning algorithm.

3.2 Learning Algorithm

Here we assume that the discovery problem has already been addressed (with however many samples one could afford), as above, and so we have an approximation $\hat{Q}_T$ and $\hat{Q}_H$ to the true core-tests and true core-histories. The learning algorithm's goal is to estimate the model parameters — the various m vectors in the following prediction equations:

$$\forall a \in \mathcal{A}, o \in \mathcal{O},\ P(ao|Q_H) = P(Q_T|Q_H)m_{ao} \text{ and}$$
$$\forall a \in \mathcal{A}, o \in \mathcal{O}, t \in Q_T,\ P(aot|Q_H) = P(Q_T|Q_H)m_{aot}$$

Note that these are the same equations as in the update Equation 2 except that we have separated the numerator and the denominator and clustered the equations into a matrix form. Observe that $P(Q_T|Q_H)$ is the submatrix $\mathcal{Z}(Q_H, Q_T)$ which is by the definitions of Q_T and Q_H a square and invertible matrix. If we had solved the discovery problem exactly, then we would have the exact submatrix $\mathcal{Z}(Q_H, Q_T)$, else we would have an approximation (but still square and full rank and hence still invertible) $\hat{\mathcal{Z}}(\hat{Q}_T, \hat{Q}_H)$. We can also use the sampling algorithm of Section 3.1 to estimate the left-hand sides of the two sets of equations above. Thus, given all of the sampling-derived approximations to the various prediction quantities in the above set of equations, our learning algorithm estimates the parameters as follows:

$$\begin{aligned} \forall a \in \mathcal{A}, o \in \mathcal{O},\ \hat{m}_{ao} &= [\hat{\mathcal{Z}}(\hat{Q}_H, \hat{Q}_T)]^{-1}\hat{P}(ao|\hat{Q}_H) \text{ and} \\ \forall a \in \mathcal{A}, o \in \mathcal{O}, t \in Q_T,\ \hat{m}_{aot} &= [\hat{\mathcal{Z}}(\hat{Q}_H, \hat{Q}_T)]^{-1}\hat{P}(aot|\hat{Q}_H) \end{aligned} \quad (3)$$

Lemma 4 *Our learning procedure (Equations 3) converges to the true model parameters w.p.1. in the limit of an infinite amount of samples (under conditions on the svd value of $\mathcal{Z}(Q_H, Q_T)$).*

Proof Sketch (Brief): Clearly, the more samples we use to estimate the various predictions that go into the right-hand sides of Equation 3 above, the more accurate our predictions will be, and modulo well-behaved matrix inversion of $\mathcal{Z}(Q_H, Q_T)$, the more accurate our estimates of model-parameters will be.

4 Empirical Results

As mentioned earlier, the only previous learning algorithm for PSRs was a myopic gradient-based algorithm [7] and it obtained mixed results on a set of dynamical systems taken from Cassandra's web-page [1] on POMDPs, indicating that these systems collectively offer a range of difficulty. Hence we chose to test our algorithms on these same simulated systems (listed in the first column of Table 1) — note, however, that we added a reset action to each system that when executed takes the system back to an initial configuration.

Problem	*Core Tests*	*Num. Act.*	*Num. Obs.*	*Reset Learning*	*Myopic Learning*	*Discovery Worked?*
Tiger	2	3	2	0.000001	0.0000043	Yes
Paint	2	4	2	0.0000001	0.00001	Yes
Float-reset	5	2	2	0.000001	0.0001	Yes
Cheese Maze	11	4	7	0.000005	0.00037	Yes
Network	7	4	2	0.0004	0.00083	Yes
Bridge Repair	5	12	5	0.00015	0.0034	Yes
Shuttle	7	3	5	0.00026	0.027	Yes
4x3 Maze	10	4	6	0.00027	0.066	Yes

Table 1: Results for problems taken from Cassandra's page, as selected by [7]. For each system, the table presents the number of core tests, the number of actions, the number of observations, the error obtained on a resettable version of the system using our reset-learning algorithm, the smallest average error obtained by [7] with their myopic algorithm, and an indication of whether the discovery algorithm was able to discover a full set of core tests and core histories.

4.1 Empirical Results on Discovery

The main ambiguities about our otherwise straightforward discovery algorithm of Section 3.1 are whether the stopping condition is correct, and whether we could only consider one-step extensions of the already found linearly-independent tests. Because we have not yet analyzed our algorithm theoretically we chose to experimentally verify the validity of our algorithm independent of the sampling-inaccuracy issues. To that end, when we ran our discovery algorithm, instead of sampling to obtain estimates of the submatrices of $\mathcal{Z}$ as called for by the algorithm, we computed the exact submatrices using the POMDP parameter specification of the system. Thus our experiments reported here did not test how estimation errors from finite samples affect the discovery of core-tests, though keeping in mind that our sampling algorithm of Section 3.1 generates i.i.d. samples one would hope for this not to be an issue for large sample size. The last column of Table 1 asks whether our algorithm worked on each system, i.e., when our algorithm stopped whether it had found the tests constituting a linear-PSR. For each system the answer was a Yes! We conjecture that this would be true in general.

4.2 Empirical Results on Learning

The learning algorithm assumed the discovery results of the previous experiments. We cycled through all the prediction quantities relevant to Equation 3, generating a sample for

each quantity using the sampling algorithm of Section 3.1. In this manner we collected N samples for each prediction quantity, and then computed an estimate of the model-parameters using matrix-inversion as in Equation 3. The number of samples N varied between $50,000$ and $500,000$ depending on the problem. Then we tested to see how good our estimates were using the same error function used by Singh et al. for their results with the myopic-gradient algorithm. The error numbers reported in column 5 of Table 1 are averaged over 100 test runs with our final estimated parameters. Each test run chooses actions randomly and is of length $10,000$ time steps. The error for each run is $\frac{1}{10,000} \sum_{t=1}^{10,000} \sum_{o \in \mathcal{O}} [p(o|h_t) - \hat{p}(o|h_t)]^2$, where $p(o|h_t)$ is the true probability of observation o in history h_t and $\hat{p}(o|h_t)$ is the corresponding estimated probability that uses our estimated model parameters to maintain the prediction vectors (as in Equation 2). Column 6 of the results table presents the error obtained in [7] using their myopic algorithm. While noting that our results are not directly comparable to the results with the myopic algorithm because we used a reset action and they did not, we find that our learning algorithm worked well on all the problems while the myopic learning algorithm did not.

5 Conclusion

Replacing the state-test prediction matrix U used in the original PSR paper with the history-prediction matrix $\mathcal{Z}$ opens up new possibilities for learning and discovery algorithms. In this paper we focused on the class of controlled dynamical systems with reset and presented a discovery algorithm (a first for PSRs on a general class of systems) as well as a new learning algorithm. Our empirical results on the learning subproblem show that our algorithm was able to find good model parameters on the dynamical systems where the previous myopic algorithm had difficulty. Our empirical results on discovery, though preliminary, offer an empirical validation of the essential ideas in our discovery algorithm. As future work we intend to continue our empirical investigation of our algorithms as well as developing more sophisticated sampling strategies than the one presented here, e.g., sampling strategies that incorporate variance-reduction techniques.

References

[1] Tony Cassandra. Tony's POMDP page. `http://www.cs.brown.edu/research/ai/pomdp/index.html`, 1999.

[2] Herbert Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6):1371–1398, 2000.

[3] Michael L. Littman. *Algorithms for Sequential Decision Making*. PhD thesis, Brown University, 1996.

[4] Michael L. Littman, Richard S. Sutton, and Satinder Singh. Predictive representations of state. In *Advances In Neural Information Processing Systems 14*, 2001.

[5] William S. Lovejoy. A survey of algorithmic methods for partially observed markov decision processes. *Annals of Operations Research*, 28(1):47–65, 1991.

[6] Ronald L. Rivest and Robert E. Schapire. Diversity-based inference of finite automata. *Journal of the ACM*, 41(3):555–589, May 1994.

[7] Satinder Singh, Michael L. Littman, Nicholas E. Jong, David Pardoe, and Peter Stone. Learning predictive state representations. In *The Twentieth International Conference on Machine Learning (ICML-2003)*, 2003. To appear.

[8] Hagit Shatkay and Leslie P. Kaelbling. Learning topological maps with weak local odometric information. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 1997.