The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Learning Predictive State Representations from Non-Uniform Sampling

Yuri Grinberg,[1] Hossein Aboutalebi,[2]
Melanie Lyman-Abramovitch,[2] Borja Balle,[3]* Doina Precup[2]
[1] National Research Council of Canada
[2] McGill University, Canada
[3] Amazon Research Cambridge, UK

Abstract

Predictive state representations (PSR) have emerged as a powerful method for modelling partially observable environments. PSR learning algorithms can build models for predicting all observable variables, or predicting only some of them conditioned on others (e.g., actions or exogenous variables). In the latter case, which we call conditional modelling, the accuracy of different estimates of the conditional probabilities for a fixed dataset can vary significantly, due to the limited sampling of certain conditions. This can have negative consequences on the PSR parameter estimation process, which are not taken into account by the current state-of-the-art PSR spectral learning algorithms. In this paper, we examine closely conditional modelling within the PSR framework. We first establish a new positive but surprisingly non-trivial result: a conditional model can never be larger than the complete model. Then, we address the core shortcoming of existing PSR spectral learning methods for conditional models by incorporating an additional step in the process, which can be seen as a type of matrix denoising. We further refine this objective by adding penalty terms for violations of the system dynamics matrix structure, which improves the PSR predictive performance. Empirical evaluations on both synthetic and real datasets highlight the advantages of the proposed approach.

Introduction

Modelling stochastic sequential data is of significant interest in many scientific disciplines, and many approaches have been proposed in machine learning to deal with this problem. In this paper, we are concerned mainly with models which are capable of making predictions about future observations, given a stream of data seen so far. This is an important problem in many domains, including reinforcement learning (where one wants to predict future returns), medical classification (where one may want to predict the future risk of a patient becoming critically ill), etc.

Hidden Markov models (HMM) (Rabiner 1989) and their extensions such as partially observable Markov decision processes (POMDP) are popular models for this task. They represent conditional models of observations given latent states of the system, also incorporating action/input variables in the case of POMDPs. Historically, expectation maximization (EM) has been used to learn such models, with mixed success. In the last decade, spectral learning methods have emerged as an alternative to EM with competitive performance and stronger theoretical guarantees, such as consistency (Boots, Siddiqi, and Gordon 2011) and finite sample bounds (Hsu, Kakade, and Zhang 2012). Such methods can in fact be applied to learn about other classes of models with latent variables, such as weighted finite automata, HMM mixtures, latent junction trees, etc. (Parikh et al. 2012), (Balle et al. 2014),(Subakan, Traa, and Smaragdis 2014). In this paper, we focus on using spectral learning for an alternative representation called (linear) predictive state representations (PSR) (Littman et al. 2002), (Singh, James, and Rudary 2004), which is used to describe the evolution of system observations possibly conditioned on actions of an agent. The development of consistent learning algorithms for PSRs has made it possible to use them for sequential data modelling in large and realistic problems (Boots, Siddiqi, and Gordon 2011), (Boots and Gordon 2011), (Ong, Grinberg, and Pineau 2013); (Grinberg, Precup, and Gendreau 2014), (Hamilton, Fard, and Pineau 2013).

While learning PSR models has been applied to both problems without actions (HMM-like) and with actions (POMDP-like), the latter case deserves a more careful investigation. We focus on the problem of conditional modelling of a subset of variables within the PSR framework, while other variables are treated as exogenous or actions. We call such a model a *conditional* PSR as opposed to a *complete* PSR, which models all the variables jointly. We first derive a new result showing that the conditional PSR will never have a larger dimension than the complete PSR, regardless of the dependency between variables. Then, we examine the current state-of-the-art PSR learning method and show that its parameter estimation procedure suffers when the conditioned variable (action or exogenous variable) is not sampled uniformly at random. It is crucial to address this problem, as in conditional models some conditions may never be achievable or occur very rarely, thus crippling existing techniques. We introduce a new, theoretically justified, optimization objective which addresses this problem. This optimization is then incorporated as an intermediate step in existing PSR spectral learning. We evaluate the proposed approach on a

*B. Balle completed this work while at McGill University.

synthetic problem as a predictor; a simulated robot environment as a predictor, and as a model that is part of the reinforcement learning method; and on a real dataset collected from a robot, as a predictor. The modified learning algorithm exhibits better sample efficiency and the potential to produce more compact models.

Background

We use bold lowercase letters to represent vectors, capital letters for matrices, and curly letters for both spaces and random variables (the difference is clear from the context), with regular lowercase letters representing values for variables. Occasionally we use double indexing, in which case subscript indices represent different values of the variables and superscripts identify the time step.

Let $\mathcal{A}$ and $\mathcal{B}$ be discrete finite spaces, with $\mathcal{A}$ being an input and $\mathcal{B}$ an observation. A model that can predict $\mathcal{B}$ given variable $\mathcal{A}$ is denoted by $\mathcal{B}|\mathcal{A}$. A model that predicts both is denoted by $\mathcal{A} \times \mathcal{B}$. Several PSR parametrizations to model $\mathcal{B}|\mathcal{A}$ have been proposed in the literature (Littman et al. 2002), (Singh, James, and Rudary 2004), (Rosencrantz, Gordon, and Thrun 2004). We follow (Wiewiora 2007), which defines a linear PSR as a tuple $\left\{m_\epsilon \in \mathbb{R}^n, \left\{M_{ab} \in \mathbb{R}^{n \times n}_{ab \in \mathcal{A} \times \mathcal{B}}\right\}, m_0 \in \mathbb{R}^n\right\}$, s.t.:

$$\forall h \in (\mathcal{A} \times \mathcal{B})^* : P(b \in \mathcal{B} | h, a \in \mathcal{A}) = m(h)^T M_{ab} m_\epsilon \quad (1)$$

where $m(h)$ is a state of the PSR after observing history h (m_0 is the initial state), and can be computed recursively as:

$$m(h, ab) = \frac{m(h)^T M_{ab}}{m(h)^T M_{ab} m_\epsilon}$$

A PSR whose action space has a single action represents a stochastic process without input variables.

Most of the learning algorithms for linear PSRs are based on estimating a portion of the system dynamics matrix (SDM) (Singh, James, and Rudary 2004), (Boots, Siddiqi, and Gordon 2011), a matrix that contains all the information about the observable behaviour of the environment. Its rows correspond to histories (sequences of past actions and observations) and columns correspond to future action-observation sequences, called tests; the entries are probabilities of observations in the corresponding history-test pair given their sequence of actions. The spectral learning algorithm for PSRs (Boots, Siddiqi, and Gordon 2011) proceeds by computing the singular value decomposition (SVD) of a portion of the matrix obtained by sampling from the environment, and then computes the PSR parameters based on the results of SVD. Let $\hat{P}_{\mathcal{H},\mathcal{T}} = USV^T$ be an estimated matrix for tests $\mathcal{T}$ and histories $\mathcal{H}$, with its truncated SVD decomposition. Let $\hat{P}_{\mathcal{H},ab\mathcal{T}}$ be a collection of estimated matrices whose columns correspond to $\{ab, t | t \in \mathcal{T}\}$ for all $ab \in \mathcal{A} \times \mathcal{B}$; and $\hat{p}_\mathcal{H}$ be a vector of estimated probabilities of histories. The algorithm computes PSR parameters as:

- m_0^T is the first row of US,
- m_ϵ: solution to the equation $(\hat{P}_{\mathcal{H},\mathcal{T}} V) m_\epsilon = \hat{p}_\mathcal{H}$,
- M_{ab}: solution to the equation $(\hat{P}_{\mathcal{H},\mathcal{T}} V) M_{ab} = \hat{P}_{\mathcal{H},ab\mathcal{T}}$.

$\mathcal{B}\|\mathcal{A}$	a_2b_2	a_3b_3
$\triangle$	$\mathrm{P}(b_2\|a_2)$	$\mathrm{P}(b_3\|a_3)$
a_1b_1	$\mathrm{P}(b_1\|a_1) \cdot \mathrm{P}(b_2\|a_1b_1a_2)$	$\mathrm{P}(b_1\|a_1) \cdot \mathrm{P}(b_3\|a_1b_1a_3)$
...	...	...

$\mathcal{A} \times \mathcal{B}$	a_2b_2	a_3b_3
$\triangle$	$\mathrm{P}(a_2b_2)$	$\mathrm{P}(a_3b_3)$
a_1b_1	$\mathrm{P}(a_1b_1a_2b_2)$	$\mathrm{P}(a_1b_1a_3b_3)$
...	...	...

Figure 1: Fragments of the SDMs corresponding to $\mathcal{B}|\mathcal{A}$ (top) and $\mathcal{A} \times \mathcal{B}$ (bottom). The symbol $\triangle$ represent an empty history.

As long as $\mathcal{T}$ and $\mathcal{H}$ are sufficiently large and $\hat{P}_{\mathcal{H},\mathcal{T}}$,$\hat{P}_{\mathcal{H},ab\mathcal{T}}$, $\hat{p}_\mathcal{H}$ are well estimated, the algorithm is guaranteed to produce an exact linear PSR representation of the environment (Boots, Siddiqi, and Gordon 2011).

Dimensions of Conditional PSR

Suppose that we only want to model some of the possible observations, not all of them. In this case, we are interested in building a conditional model $\mathcal{B}|\mathcal{A}$, where $\mathcal{A}$ could be actions, or other variables that we want to consider exogenous (perhaps because they are difficult to model). We would expect intuitively that a conditional distribution would be easier to model than a full joint, hence the conditional PSR should be more compact (or at least not larger) than the complete PSR.

In this section we state the main theorem that backs up this intuition. Surprisingly, proving this result is more complicated than it seems at first glance. In earlier PSR papers, the standard tool to show that one (sub-)system is at most as big as another has been the direct comparison of their SDMs. For example, in (Wolfe, James, and Singh 2008), (Ong, Grinberg, and Pineau 2013), the authors directly show that the column space of one SDM spans the column space of another SDM. Unfortunately, this line of attack does not work in our case. To get an intuition for why this is the case, consider the following example, depicted in Fig. 1. Let $\mathcal{A} = \{a_1, a_2, a_3\}$ and $\mathcal{B} = \{b_1, b_2, b_3\}$. Consider a fragment of the SDM of $\mathcal{B}|\mathcal{A}$ and one from the SDM of $\mathcal{A} \times \mathcal{B}$. Looking at the first row of $\mathcal{B}|\mathcal{A}$, it is not clear how to write it as a linear combination of rows from $\mathcal{A} \times \mathcal{B}$, since in general $P(a_2) \neq P(a_3)$. The situation is similar with respect to the first column of $\mathcal{B}|\mathcal{A}$. As a result, there is no direct way to show that the column/row spaces of one SDM are within the corresponding spaces of the other.

Theorem 1. *Consider an $\mathcal{A} \times \mathcal{B}$-valued stochastic process that can be represented by a k-dimensional linear PSR. Then, the conditional linear PSRs representing $\mathcal{B}|\mathcal{A}$ and $\mathcal{A}|\mathcal{B}$ models are each at most k-dimensional.*

The proof is provided in the Appendix. While the above theorem considers stochastic processes that output two variables only for simplicity, the extension to larger number of variables is straightforward.

Spectral Learning for Conditional PSRs

The existing literature on learning PSRs treats the modelling of conditional and complete models very similarly (Boots, Siddiqi, and Gordon 2011), (Boots and Gordon 2011). The single issue that has been identified earlier for learning conditional PSRs is that the original estimators of the SDM entries are biased and not consistent (Bowling et al. 2006). In that work, corrected estimators were developed, which can also be seen as importance-weighting of the original estimators (Boots, Siddiqi, and Gordon 2011). Assume for the rest of this section, that $\mathcal{A}$ is the exogenous variable and we want to learn the model $\mathcal{B}|\mathcal{A}$. In that case, an entry in the SDM that corresponds to the sequence $\langle a^1b^1, a^2b^2, ..., a^mb^m \rangle$ is estimated from:

$$\prod_{i=1}^{m} \frac{\#a^1b^1...a^ib^i}{\#a^1b^1...b^{i-1}a^i}, \tag{2}$$

which is an unbiased estimator [Bowling et al., 2006] of $\prod_{i=1}^{m} P(b^i|a^1b^1, ..., a^{i-1}b^{i-1}, a^i)$.

However, beyond the need to correct the entry-wise estimators of the SDM, there is a deeper issue with learning conditional models using standard spectral learning. To see this, consider the rank-k SDM approximation problem, which is solved using SVD as the central step of the algorithm:

$$\textbf{argmin}_{Q,R} ||\hat{P}_{\mathcal{H},\mathcal{T}} - QR||_F^2 \tag{3}$$

where $Q \in \mathbb{R}^{\mathcal{H}\times k}$, $R \in \mathbb{R}^{k\times\mathcal{T}}$, $||.||_F$ stands for Frobenius norm. Clearly, this optimization treats every entry in $\hat{P}_{\mathcal{H},\mathcal{T}}$ equally. However, a careful look at the above estimator reveals that not all entries are created equal. For example, if $P(a_1)$ is small compared to $P(a_2)$, the Monte-Carlo estimates of $P(b_1|a_1)$ are going to be more noisy than the estimates of $P(b_1|a_2)$. Taking this to extreme, if $P(a_1) = 0$, the corresponding SDM entries will be filled with a default value of 0. Those entries will have the same effect on the objective (3) as the other entries, which is a sub-optimal strategy at best.

To address this problem, instead of solving (3), we propose to solve a modified optimization problem, given by the entry-wise weighted Frobenius norm:

$$\begin{aligned} \textbf{argmin}_{Q,R} ||\hat{P}_{\mathcal{H},\mathcal{T}} - QR||_{F,W}^2 \\ \equiv \sum_{ij} w_{ij}([\hat{P}_{\mathcal{H},\mathcal{T}}]_{ij} - [QR]_{ij})^2 \end{aligned} \tag{4}$$

where the weight matrix $[W]_{ij} = w_{ij}$ identifies the importance of the entries. The question becomes how to set the weights so they reflect the accuracy of the Monte-Carlo estimates (2). One natural choice is the reciprocal of the variance of the corresponding estimators. In fact, using a simplifying assumption, these weights guarantee that the solution to (4) is a maximum likelihood estimate.

Corollary 2 (Known result)**.** *Let $Y = X + Z$ be an observation matrix of size $m \times n$ where X is fixed and Z is an i.i.d noise matrix, with $z_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$. Then, the maximum likelihood estimator for X for any hypothesis set $\mathbb{M} \subset \mathbb{R}^{m\times n}$ is:*

$$\textbf{\textit{argmin}}_{X\in\mathbb{M}} \sum_{ij} \frac{1}{\sigma_{ij}^2}(y_{ij} - x_{ij})^2.$$

Last but not least, it is well known that $\hat{P}_{\mathcal{H},\mathcal{T}}$ is a very structured matrix, but the optimization objective (4) does not account for that. Structural violations of this matrix increase the chances that the learned PSR will produce invalid probabilities, e.g. negative probabilities or probabilities that do not sum to 1. This problem shows up in the original spectral learning method as well (eq. (3) does not enforce the matrix structure), however to a lesser extent compared to the solution obtained from objective (4). Therefore, we propose to explicitly penalize structural violations in the objective function. Note that this approach is of independent interest to spectral learning methods in general. In the experimental section we will highlight the benefits of those penalties. To introduce the penalties, let h_it_j be the concatenation of sequences h_i and t_j. The structural properties of $\hat{P}_{\mathcal{H},\mathcal{T}}$ are:

- All entries are in the interval $[0, 1]$,
- If $h_1t_1 = h_2t_2$ then $\hat{P}_{\mathcal{H},\mathcal{T}}(h_1, t_1) = \hat{P}_{\mathcal{H},\mathcal{T}}(h_2, t_2)$,
- $\sum_{b\in\mathcal{B}} \hat{P}_{\mathcal{H},\mathcal{T}}(h, tab) = \hat{P}_{\mathcal{H},\mathcal{T}}(h, t);\ \ \forall hta \in \mathcal{H}\times\mathcal{T}\times\mathcal{A}$.

In our experiments, the first property was rarely violated, so we ignored it for convenience. For the second property, let C be a set of sublists C_i containing all the row-column indexes of history-test tuples whose probability should be the same in $\hat{P}_{\mathcal{H},\mathcal{T}}$, i.e.

$$\begin{aligned} C = \{C_i|\ \langle index[(h_1, t_1)], index[(h_2, t_2)]\rangle \in C_i \Leftrightarrow \\ h_1t_1 = h_2t_2;\ \ for\ \ i \neq j\ \ C_i \cap C_j = \emptyset\}. \end{aligned}$$

For the third property, note that it is actually a sum over the columns of $\hat{P}_{\mathcal{H},\mathcal{T}}$, so we need to enforce it on R no matter what the Q values are. Let $\langle Y_k, j_k\rangle_{k=1}^{l}$ be a collection of indexes encoding this constraint, specifically:

$$\forall k : \textstyle\sum_{y\in Y_k} [\hat{P}_{\mathcal{H},\mathcal{T}}]_{:,y} - [\hat{P}_{\mathcal{H},\mathcal{T}}]_{:,j_k} = 0.$$

Combining the above definitions, we further augment eq. (4) by adding the penalties as follows:

$$\begin{aligned} \textbf{argmin}_{Q,R} \textstyle\sum_{i,j} \left\| \hat{P}_{\mathcal{H},\mathcal{T}} - QR \right\|_{F,W}^2 \\ +\lambda_1 \left[\textstyle\sum_{k\in|C|} \sum_{\langle i,j\rangle\in C_k} (q_{x_i}^T r_{y_i} - q_{x_j}^T r_{y_j})^2 \right] \\ +\lambda_2 \textstyle\sum_{k=1}^{l} \left\| \sum_{y\in Y_k} R_{:,y} - R_{:,j_k} \right\|^2 \end{aligned} \tag{5}$$

where x_i, y_i denote the row and column indexes of element i in $\hat{P}_{\mathcal{H},\mathcal{T}}$ respectively. Note that these new terms in the objective do not change the nature or hardness of the optimization problem.

The solution to the new optimization problem (5), which we call low rank matrix denoising, is then used by the spectral learning algorithm instead of the SVD decomposition; the subsequent conditional PSR parameter estimation process remains unchanged. Note that the choice of the optimization problem in (5) as well as the choice of the weights lead to the following meaningful but easily overlooked consistency–like result.

Proposition 3. *Let the sets $\mathcal{H}$ and $\mathcal{T}$ be sufficiently large*[1]*, and assume the rank of QR is not less than the rank of the*

[1] A typical requirement to obtain consistency results, which requires the submatrix $P_{\mathcal{H},\mathcal{T}}$ to have the same rank as the SDM.

*SDM representing $\mathcal{B}|\mathcal{A}$. Then, in the limit of infinite data, $QR = P_{\mathcal{H},\mathcal{T}}$ is a **global** optimum of the problem* (5).

Proof. First, due to the assumption on the rank of QR, $QR = P_{\mathcal{H},\mathcal{T}}$ is achievable. Further, note that the penalties for structure violations in (5) are 0, since $P_{\mathcal{H},\mathcal{T}}$ is a valid submatrix of the SDM. Finally, as the weights are the reciprocal of the variance of the estimators, they remain constant for missing entries (missing due to the possible non-stochasticity of the policy), and grow to infinity for the other entries. This is, however, equivalent to maintaining bounded weights for the non-missing entries and weights that go to zero for missing entries (by normalizing (5)). As $\hat{P}_{\mathcal{H},\mathcal{T}} \to P_{\mathcal{H},\mathcal{T}}$ entry-wise, for non-missing entries, we therefore have that setting $QR = P_{\mathcal{H},\mathcal{T}}$ makes the objective (5) $\to 0$ in the limit. □

The above result is meaningful precisely because it does not hold for non-stochastic policies for the standard spectral learning method. In the case of non-stochastic policies, the missing entries in the SDM will still be treated as meaningful entries (typically containing the value of 0).

To solve the problem (5) we use gradient descent and initialize the matrices Q and R with the solution obtained from the SVD, since this approach has been proven optimal in certain matrix completion problems (Jain, Netrapalli, and Sanghavi 2013; Gunasekar et al. 2013). The pseudo-code is shown in Alg. 1 box below.

Algorithm 1: Denoising Algorithm

Data: $\hat{P}_{\mathcal{H},\mathcal{T}} = U^T S V \in R^{m \times n}$, step size α , threshold ϵ, reg. parameters λ_1, λ_2

Let $f(Q,R) = \sum_{i,j} ||\hat{P}_{\mathcal{H},\mathcal{T}} - QR||^2_{F,W} + \lambda_1(\sum_{k\in|C|}\sum_{i,j\in C_k}(q_{x_i}^T r_{y_i} - q_{x_j}^T r_{y_j})^2) + \lambda_2 \sum_{k=1}^{l} || \sum_{y\in Y_k} R_{:,y} - R_{:,j_k}||^2$

Result: Rank-k matrix R

Initialize: $Q = U^T S$; $R = V$;

Until: convergence

$Q = Q + \alpha \nabla_Q f(Q,R)$; $R = R + \alpha \nabla_R f(Q,R)$;

Experiments

We conducted several experiments on real and simulated environments in order to evaluate our denoising algorithm, in comparison to the standard PSR learning approach. In all experiments we set the Frobenious norm weights to be the number of samples used to estimate each entry as a proxy to the inverse variance.

One State Environment

First, we consider a simple environment with one state, two actions ($a \in \{right, left\}$) and two observations ($o \in \{0,1\}$). The policy is a two-state automaton with states labeled A and B, A being the initial state. Figure 2 depicts the behavior of the environment and the policy. The observations obtained from the environment are probabilistic, depending on the action observed, parameterized with k. The policy states cycle ($A \to B$, $B \to A$) after each time step.

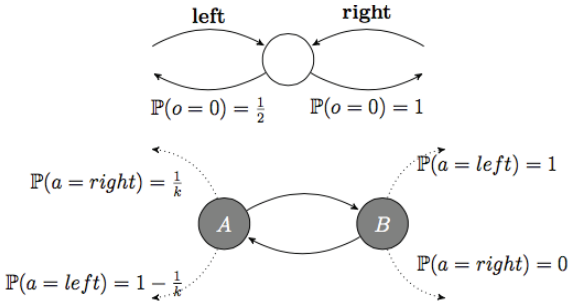


Figure 2: Synthetic problem setup: one-state environment (top) and two-state policy(bottom)

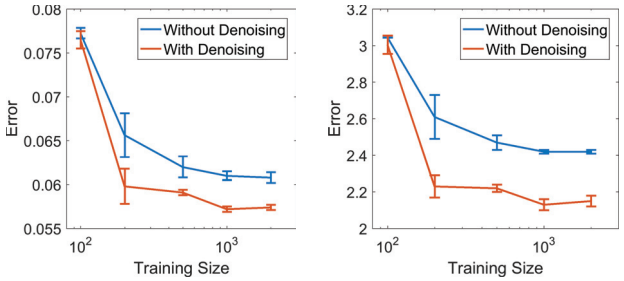
The results for $k = 100$ are depicted in Figure 3 for PSR dimensions 1 and 2, showing one-step prediction errors and the difference in Frobenius norm between the true and estimated SDM matrices, as a function of the number of trajectories used for training. The curves represent means and standard errors over 5 independent runs. The prediction errors are measured through the L_1-norm between the one-step predicted and true probabilities of the test data consisting of 3000 trajectories of length 2 (2 action-observation pairs) generated under a random policy, and then averaged out.

Due to the non-stationary and biased nature of the policy, the SDM matrix without denoising will always remain rank-2 matrix even in the infinite data case, despite representing a one-state environment. The denoising algorithm suggests that the rank-1 SDM matrix fits the problem better than rank-2 matrix, as expected, by achieving better performance on the SDM matrix error and comparable performance on the prediction error. This is in addition to its superior performance to the standard approach across both setups.

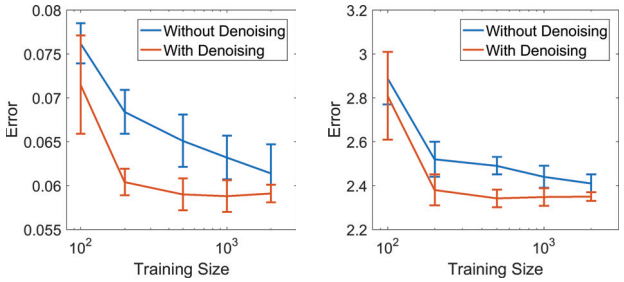
Torus Environment

The torus environment represents a simulated navigation task with stochastic partial observability, shown schematically in Fig. 4. There are 100 states representing locations on the torus, with observations being a color sampled from a stochastic observation model that depends on the state. The actions move the agent along the 4 available directions, either 1 or 2 steps (giving 8 possible actions). However, going up or down is only possible at certain locations (see Fig. 4a). The robot chooses any valid action with equal probability. While this is a synthetic environment, it captures the characteristic of real environment where certain actions may not always be available, or may be too costly to take even during the exploration stage, thus inducing a non-uniform sampling over the trajectories.

The results of applying the standard spectral learning and

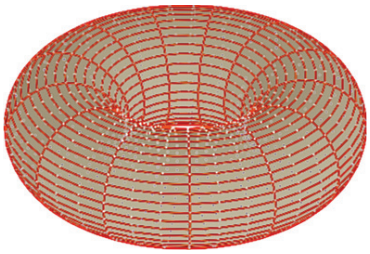


(a) Pred. error on test data (b) SDM matrix error

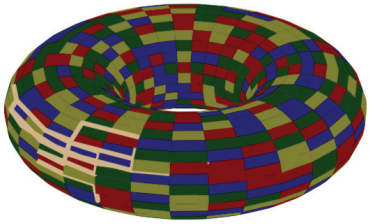


(c) Pred. error on test data (d) SDM matrix error

Figure 3: Comparison of standard vs "with denoising" algorithms: performance on learning one-state environment using rank-1 SDM (top) and rank-2 SDM (bottom) matrices.



(a) The white dots represent possible robot states, the red lines represent possible paths the robot can traverse



(b) Traversing the torus: yellow line - robot exploration, color patches - sampled observations

Figure 4: Details of the torus environment

spectral learning with denoising are shown in Figure 5, for problem setups with 4 observations and 20 observations. The SDM was constructed using histories up to length 3 and tests up to length 2. The error measures used are as in the previous task. In all cases, we used rank-15 SDM, as it produced better performance for both the standard and denoising methods. The error is computed using test data similarly to the first experiment, containing 5000 trajectories of length 4 obtained from the same exploratory policy.

Planning in Torus environment

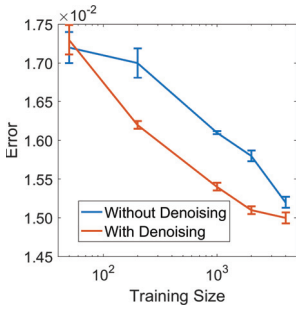
To further show that the PSR models learned using spectral learning with denoising are useful, we evaluate their utility as a state representation for planning in the Torus environments with 8 observations and 4 actions. The rewards [-3,-2,-1,0,1,2,3,4] are assigned to observations 1 to 8. The objective is to maximize the total reward after taking 1000 steps. Standard spectral learning and spectral learning with denoising were used to model the environment first, and then Fitted-Q iteration (FQI) (Ernst, Geurts, and Wehenkel 2005) was used to learn the optimal policy on a single trajectory of size 1500 generated from a uniformly random policy. At each iteration of FQI, we used Extremely randomized trees (Geurts, Ernst, and Wehenkel 2006) to do the fitting step. The experiment was repeated 10 times, and the average reward for varying training data sizes (for model learning) are shown in Figure 7a. The performance of our proposed approach is superior to its counterpart. We also included the performance of the random policy as a baseline for benchmarking purposes (a flat line around the value 900). The performance degradation in standard spectral learning can be attributed to lack of proper handling of "missing" entries in the SDM. As more data is available, more "missing" entries appear in the SDM matrix due to the constraint on policy actions (see Figure 7b), resulting in a less accurate model that is less useful for planning. On the other hand, the denoising approach handles the missing entries fairly well by design.

Robot sensing

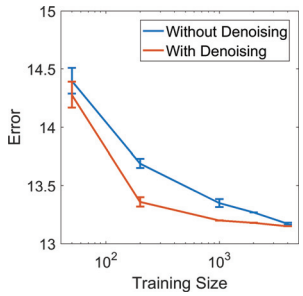
In this final experiment, we evaluate the proposed method on the real data obtained from the robot cyclically exploring his environment using 24 sensors (Freire et al. 2009). As the gathered data was continuous, we discretized it using the same approach as in (Rosencrantz, Gordon, and Thrun 2004). We sampled 100 observations at random and then relabelled the rest with the label of nearest sampled observation in L_2-norm. Performance was evaluated using 5-fold cross validation, such that each fold represents a new trajectory of the robot. A rank-40 SDM was used to compute PSR parameters, as increasing the rank did not have any notable effect on either of the methods. As the true model is not accessible, we used log-likelihood of the test data as performance measure. Figure 6 shows the average log-likelihoods for two methods, with the denoising algorithm clearly outperforming the standard approach.

Discussion

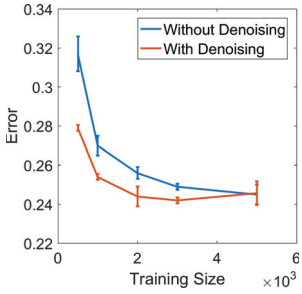
We considered the problem of building conditional models in the PSR framework when the action(s)/exogenous variable(s) are sampled non-uniformly. This is a shared characteristic of many real domains, whether because the exogenous variable has its own complicated dynamics or because it is risky and/or expensive to collect data following a uniformly random exploration policy. Coupled with the fact that



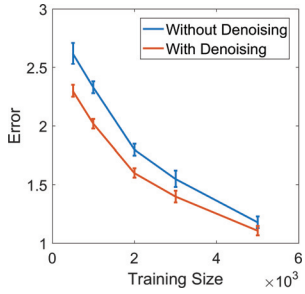
(a) Pred. error on test data



(b) SDM matrix error

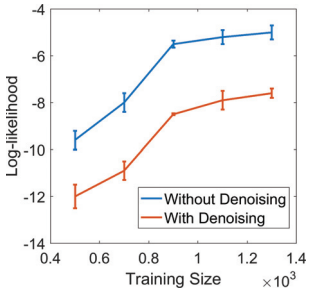


(c) Pred. error on test data

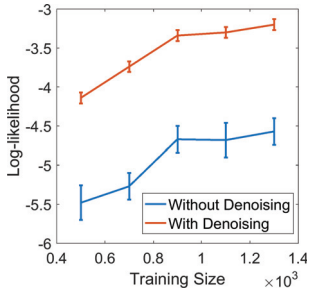


(d) SDM matrix error

Figure 5: The results of applying denoising algorithm on torus environment: 4 (top) and 20 (bottom) observations.

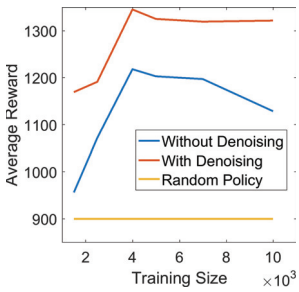


(a) Testing error

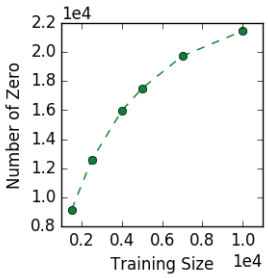


(b) Training error

Figure 6: The results of applying denoising algorithm on the robot sensing task



(a) Planning performance



(b) Number of missing entries in the SDM

Figure 7: Evaluation of learned PSR models for the planning task in Torus environment

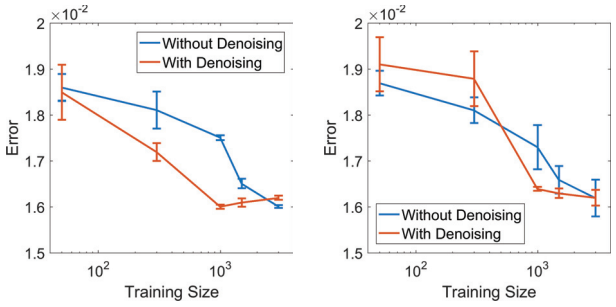


Figure 8: Prediction error as a function of the training size for the 4 observations Torus experiment with $\lambda_2 = 0$. Left: $\lambda_1 = 10$, Right: $\lambda_1 = 0.005$.

the amount of training data is typically limited, it is imperative to use the data efficiently when building these models. We proved that the dimension of a conditional model is never larger than the dimension of a complete PSR, regardless of the dependencies between the variables. Further, we showed that the standard PSR spectral learning algorithm is suboptimal in a non-uniform sampling scenario. We proposed to modify the original algorithm to explicitly account for non-uniform sampling in a provably reasonable way and performed experimental evaluations. Finally, the proposed algorithm includes the feature of building a PSR model that maintains the SDM structure, which is of independent interest. We observed that the effect of those penalties becomes more pronounced as the complexity of the domain increases. For example, the effect of the first penalty term on the PSR predictive performance in the torus experiment with 4 observations is seen in Figure 8.

Although the proposed optimization objective is a better objective to solve, how to solve it efficiently is an open question. Previous results show that the general problem of minimizing the weighted Frobenius norm is hard, however exploring the specific structure of the matrix to improve the search for the optimal solution is left for future work.

Appendix: Theorem 1 Proof

The proof is done in several steps. First, we make sure that conditional PSR is well defined even if the corresponding system dynamics matrix has infinite rank. Prop. 4 guarantees that a PSR representation is well defined under such circumstances. Second, based on the collection of conditional PSRs we define a new construct named *tensor PSR* and show that it has all important PSR-like properties (Prop. 6). Finally, we show that the states of conditional PSRs, complete PSR and tensor PSR relate to each other according to Fig. 9.

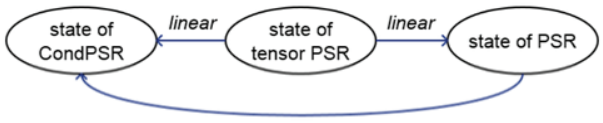


Figure 9: The relationship between the states of the PSR, Conditional PSR (CondPSR) and tensor PSR.

Proposition 4. *Any discrete-valued stochastic process with actions can be represented by a possibly infinite dimensional PSR with actions.*

Proof. The finite dimensional case is trivial. Suppose that the SDM corresponding to the above process (with actions) has infinite rank. To construct a well-defined PSR parametrization for this process, we start with choosing a set of core tests in the following iterative fashion. We loop over tests, starting from the shortest and proceeding to longer tests. A new test is added to the set of core tests only if the corresponding column is linearly independent from the columns matching the tests already in the set. Note that this choice of core tests (also seen as a basis for the column space) guarantees that a vector of coefficients representing any test of finite length will have finitely many nonzero entries.

Let $\mathbf{m}_t$ be a vector of coefficients for a test t. Note that $m_\epsilon = \sum_{b\in\mathcal{B}} \mathbf{m}_{ab}$ for any $a \in \mathcal{A}$.[2] Following (Wiewiora 2007), matrices $\mathbf{M}_{ab}$ are built of column vectors of coefficients that correspond to one-step extensions of core tests. Hence, PSR property (1) is well-defined, as all infinite sums are in fact finite, since they have only a finite number of nonzero elements. This yields a well-defined infinite- dimensional PSR parametrization that can be (conceptually) used to calculate any future predictions of the stochastic process. □

Definition 5. *Let $\mathcal{A}\times\mathcal{B}$-valued stochastic process be represented by a collection of two conditional PSRs $\mathcal{A}|\mathcal{B}$ and $\mathcal{B}|\mathcal{A}$ of dimensions $n \in \mathbb{N}\cup\infty$ respectively whose parameters are:*

$$\left\{\boldsymbol{p}^{\mathcal{A}}, \boldsymbol{m}^{\mathcal{A}}_{\epsilon}, \left\{\boldsymbol{M}^{\mathcal{A}}_{ba}\right\}_{ba\in\mathcal{B}\times\mathcal{A}}, \left\{\boldsymbol{m}^{\mathcal{A}}_{a}\right\}_{a\in\mathcal{A}}\right\},$$

$$\left\{\boldsymbol{m}^{\mathcal{B}}_{\epsilon}, \left\{\boldsymbol{M}^{\mathcal{B}}_{ab}\right\}_{ab\in\mathcal{A}\times\mathcal{B}}, \boldsymbol{m}^{\mathcal{B}}_{0}\right\}$$

respectively. A tensor PSR is a collection of parameters defined by:

- $\boldsymbol{c}_\epsilon \in \mathbb{R}^{mn} = \boldsymbol{m}^{\mathcal{A}}_{\epsilon} \otimes \boldsymbol{m}^{\mathcal{B}}_{\epsilon}$
- $\left\{\boldsymbol{C}_{ba} \in \mathbb{R}^{mn\times mn} = \boldsymbol{M}^{\mathcal{A}}_{ba} \otimes \boldsymbol{I}^m\right\}_{ba\in\mathcal{B}\times\mathcal{A}}$
- $\left\{\boldsymbol{C}_{ab} \in \mathbb{R}^{mn\times mn} = \boldsymbol{I}^n \otimes \boldsymbol{M}^{\mathcal{B}}_{ab}\right\}_{ab\in\mathcal{A}\times\mathcal{B}}$
- $\left\{\boldsymbol{c}_a \in \mathbb{R}^{mn} = \boldsymbol{m}^{\mathcal{A}}_{a} \otimes \boldsymbol{m}^{\mathcal{B}}_{0}\right\}_{a\in\mathcal{A}}$

Where $\boldsymbol{I}^k$ is a $k\times k$ identity matrix, and $\otimes$ represents represents the tensor product. Also, let $c(h)$ be a vector that represents a sufficient statistic of history (the state) of the tensor PSR, given that history h was observed.

Proposition 6. *Given a tensor PSR, for any $h = ab^{1:l} \in (\mathcal{A}\times\mathcal{B})$ we have:*

$$P(ab^{1:l}) = p(a^1).\boldsymbol{c}^T_{a^1}\boldsymbol{C}_{a^1b^1}\boldsymbol{C}_{b^1a^2}...\boldsymbol{C}_{b^{l-1}a^l}\boldsymbol{C}_{a^lb^l}\boldsymbol{c}_\epsilon \quad (6)$$

$$\boldsymbol{c}(h)^T \equiv \frac{\boldsymbol{c}^T_{a^1}\boldsymbol{C}_{a^1b^1}\boldsymbol{C}_{b^1a^2}...\boldsymbol{C}_{b^{l-1}a^l}\boldsymbol{C}_{a^lb^l}}{\boldsymbol{c}^T_{a^1}\boldsymbol{C}_{a^1b^1}\boldsymbol{C}_{b^1a^2}...\boldsymbol{C}_{b^{l-1}a^l}\boldsymbol{C}_{a^lb^l}\boldsymbol{c}_\epsilon} = \boldsymbol{m}^{\mathcal{A}}(h)^T \otimes \boldsymbol{m}^{\mathcal{B}}(h)^T \quad (7)$$

[2] This fact is easy to verify from the properties of PSR, assuming that it is minimal-dimensional (Wiewiora 2007)

where $\boldsymbol{m}^{\mathcal{A}}(h)$ and $\boldsymbol{m}^{\mathcal{B}}(h)$ are states of conditional PSRs $\mathcal{A}|\mathcal{B}$ and $\mathcal{B}|\mathcal{A}$ respectively after observing history h.

Proof.

$$\begin{aligned}
P(ab^{1:l}) &= P(a^1).P(a^2|ab^{1:1})...P(a^l|ab^{1:l-1}) \\
&\quad .P(b^1|a^1)...P(b^l|ab^{1:l-1}, a^l) \\
&= \mathbf{p}^{\mathcal{A}}(a^1).\mathbf{m}_{a^1}{}^{\mathcal{A}T}\mathbf{M}^{\mathcal{A}}_{b^1a^2}...\mathbf{M}^{\mathcal{A}}_{b^{l-1}a^l}\mathbf{m}^{\mathcal{A}}_{\epsilon} \\
&\quad .\mathbf{m}^{\mathcal{B}T}_0\mathbf{M}^{\mathcal{B}}_{a^1b^1}...\mathbf{M}^{\mathcal{B}}_{a^lb^l}\mathbf{m}^{\mathcal{B}}_{\epsilon} \\
&= \mathbf{p}^{\mathcal{A}}(a^1).\left[\mathbf{m}^{\mathcal{A}T}_{a^1}\mathbf{I}^m\mathbf{M}^{\mathcal{A}}_{b^1a^2}...\mathbf{I}^m\mathbf{M}^{\mathcal{A}}_{b^{l-1}a^l}\mathbf{m}^{\mathcal{A}}_{\epsilon}\right] \\
&\quad .\left[\mathbf{m}^{\mathcal{B}T}_0\mathbf{M}^{\mathcal{B}}_{a^1b^1}\mathbf{I}^n...\mathbf{M}^{\mathcal{B}}_{a^lb^l}\mathbf{I}^n\mathbf{m}^{\mathcal{B}}_{\epsilon}\right] \\
&= \mathbf{p}^{\mathcal{A}}(a^1).(\mathbf{m}^{\mathcal{A}\mathcal{T}}_{a^1}\otimes\mathbf{m}^{\mathcal{B}\mathcal{T}}_0).(\mathbf{I}^m\otimes\mathbf{M}^{\mathcal{B}}_{a^1b^1}) \\
&\quad ...(\mathbf{M}^{\mathcal{A}}_{b^{l-1}a^l}\otimes\mathbf{I}^n).(\mathbf{m}^{\mathcal{A}}_{\epsilon}\otimes\mathbf{m}^{\mathcal{B}}_{\epsilon}) \\
&= \mathbf{p}^{\mathcal{A}}(a^1).\mathbf{c}^T_{a^1}\mathbf{C}_{a^1b^1}\mathbf{C}_{b^1a^2}...\mathbf{C}_{b^{l-1}a^l}\mathbf{C}_{a^lb^l}\mathbf{c}_\epsilon
\end{aligned}$$

(7)*:* The proof is similar to (6). □

Proof of Theorem 1:

Let $C = \{\mathbf{c}_\epsilon, \mathbf{C}_{ba\in\mathcal{B}\times\mathcal{A}}, \mathbf{C}_{ab\in\mathcal{A}\times\mathcal{B}}, \mathbf{c}_{a\in\mathcal{A}}\}$ be a tensor PSR constructed from a possibly infinite-dimensional conditional PSRs $\mathcal{A}|\mathcal{B}$ and $\mathcal{B}|\mathcal{A}$. We now show that the state of a conditional PSR is a linear function of the state of the tensor PSR, which is therefore a linear function of the state of the PSR as well.

Let $\mathbf{U} \in \mathbb{R}^{m\times mn} = \mathbf{m}^{\mathcal{A}T}_{\epsilon} \otimes \mathbf{I}^m$. Then, for any history $h \in (\mathcal{A}\times\mathcal{B})^*$, we have:

$$\begin{aligned}
\mathbf{Uc}(h) &= \left[\mathbf{m}^{\mathcal{A}T}_{\epsilon}\otimes\mathbf{I}^m\right].\left[\mathbf{m}^{\mathcal{A}}(h)\otimes\mathbf{m}^{\mathcal{B}}(h)\right] \\
&= \left[\mathbf{m}^{\mathcal{A}T}_{\epsilon}\mathbf{m}^{\mathcal{A}}(h)\right]\otimes\mathbf{m}^{\mathcal{B}}(h) = \mathbf{m}^{\mathcal{B}}(h)
\end{aligned}$$

where the last equality follows from the properties of the normalizer $\mathbf{m}_\epsilon$. Thus, a state of conditional PSR is a linear function of a state of tensor PSR. It is also clear from equations (1) and (2) that a prediction of any test is a linear function of a state of the tensor PSR, and in particular any core test of the PSR. Hence a state of the PSR is a linear function of a state of the tensor PSR as well.

Finally, let $s(h) \in \mathbb{R}^k$ be the state of the PSR of the combined stochastic process after seeing history h. Since it is possible to calculate any predictions using $s(h)$ and the PSR parameters, it is therefore possible to calculate conditional predictions that are core tests of conditional PSR $\mathcal{B}|\mathcal{A}$. Hence, we can define an operator G that takes the PSR state and produces the state of conditional PSR $\mathcal{B}|\mathcal{A}$. Then, for any history $h \in (\mathcal{A}\times\mathcal{B})^*$, we have:

$$\mathbf{Uc}(h) = G[s(h)] = G[\mathbf{c}(h)\mathbf{W}]$$

where $\mathbf{W}$ is the matrix that represents the coefficients of the core tests of the PSR with respect to the tensor PSR state. Hence, G is a linear operator, implying that the state space of conditional PSR $\mathcal{B}|\mathcal{A}$ is at most k-dimensional. The same argument holds regarding the state space of conditional PSR $\mathcal{A}|\mathcal{B}$. □

References

Balle, B.; Carreras, X.; Luque, F. M.; and Quattoni, A. 2014. Spectral learning of weighted automata. *Machine Learning* 96(1-2):33–63.

Boots, B., and Gordon, G. J. 2011. An online spectral learning algorithm for partially observable nonlinear dynamical systems. In *AAAI*.

Boots, B.; Siddiqi, S. M.; and Gordon, G. J. 2011. Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research* 30(7):954–966.

Bowling, M.; McCracken, P.; James, M.; Neufeld, J.; and Wilkinson, D. 2006. Learning predictive state representations using non-blind policies. In *Proceedings of the 23rd international conference on Machine learning*, 129–136. ACM.

Ernst, D.; Geurts, P.; and Wehenkel, L. 2005. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* 6(Apr):503–556.

Freire, A. L.; Barreto, G. A.; Veloso, M.; and Varela, A. T. 2009. Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *Robotics Symposium (LARS), 2009 6th Latin American*, 1–6. IEEE.

Geurts, P.; Ernst, D.; and Wehenkel, L. 2006. Extremely randomized trees. *Machine learning* 63(1):3–42.

Grinberg, Y.; Precup, D.; and Gendreau, M. 2014. Optimizing energy production using policy search and predictive state representations. In *Advances in Neural Information Processing Systems*, 2969–2977.

Gunasekar, S.; Acharya, A.; Gaur, N.; and Ghosh, J. 2013. Noisy matrix completion using alternating minimization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 194–209. Springer.

Hamilton, W. L.; Fard, M. M.; and Pineau, J. 2013. Modelling sparse dynamical systems with compressed predictive state representations. In *ICML (1)*, 178–186.

Hsu, D.; Kakade, S. M.; and Zhang, T. 2012. A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences* 78(5):1460–1480.

Jain, P.; Netrapalli, P.; and Sanghavi, S. 2013. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, 665–674. ACM.

Littman, M. L.; Sutton, R. S.; Singh, S.; et al. 2002. Predictive representations of state. *Advances in neural information processing systems* 2:1555–1562.

Ong, S. C.; Grinberg, Y.; and Pineau, J. 2013. Mixed observability predictive state representations. In *AAAI*.

Parikh, A. P.; Song, L.; Ishteva, M.; Teodoru, G.; and Xing, E. P. 2012. A spectral algorithm for latent junction trees. *arXiv preprint arXiv:1210.4884*.

Rabiner, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286.

Rosencrantz, M.; Gordon, G.; and Thrun, S. 2004. Learning low dimensional predictive representations. In *Proceedings of the twenty-first international conference on Machine learning*, 88. ACM.

Singh, S.; James, M. R.; and Rudary, M. R. 2004. Predictive state representations: A new theory for modeling dynamical systems. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, 512–519. AUAI Press.

Subakan, C.; Traa, J.; and Smaragdis, P. 2014. Spectral learning of mixture of hidden markov models. In *Advances in Neural Information Processing Systems*, 2249–2257.

Wiewiora, E. W. 2007. *Modeling probability distributions with predictive state representations*. University of California, San Diego.

Wolfe, B.; James, M. R.; and Singh, S. 2008. Approximate predictive state representations. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, 363–370. International Foundation for Autonomous Agents and Multiagent Systems.