Predictive State Models for Prediction and Control in Partially Observable Environments

Ahmed Hefny
Machine Learning Department
Carnegie Mellon University
ahefny@cs.cmu.edu

Zita Marinho
Robotics Institute
Carnegie Mellon University
zmarinho@cmu.edu

Carlton Downey
Machine Learning Department
Carnegie Mellon University
cmdowney@cs.cmu.edu

Wen Sun
Robotics Institute
Carnegie Mellon University
wensun@cs.cmu.edu

Siddhartha Srinivasa
Robotics Institute
Carnegie Mellon University
ss5@andrew.cmu.edu

Geoffrey Gordon
Machine Learning Department
Carnegie Mellon University
ggordon@cs.cmu.edu

Abstract: State estimation is an integral part of inference in partially observable environments. In this paper we propose predictive state controlled models (PSCMs); a method to estimate the state of a partially observable environment. PSCMs extend two-stage regression for predictive state representations to controlled environments. They enjoy several favorable qualities: They can represent controlled environments which can be affected by actions, they have a scalable and theoretically justified learning algorithm, and they use a non-parametric representation that is suitable for non-linear dynamics. We show promising results for the proposed method in two settings: learning to predict observations in an environment controlled by an external agent, and learning to control the environment using reinforcement learning.

Keywords: Predictive State Representations, Dynamical Systems, Reinforcement Learning

1 Introduction

Many important tasks in robotics involve reasoning about dynamic environments. Two of these tasks are prediction (estimating future observations of an environment that is controlled by an external agent) and control (learning a control policy from previous interactions with the environment in order to maximize cumulative rewards).

In a partially observable environment, it is crucial to obtain an appropriate state representation. This state provides a summary of previous observations and actions that is sufficient to decide on an action or make a prediction. Maintaining the state representation is also known as filtering.

In this work we propose predictive state controlled models (PSCMs) as a method to obtain a state representation. PSCMs combines the following three concepts: (1) predictive state representations: where the state is represented by a conditional distribution of future observations given future actions. Expressing the state in terms of observable quantities enables consistent initialization through the two-stage regression method, which is free of local optima, (2) Hilbert space embedding of distributions: which represents conditional distributions as infinite dimensional operators, allowing for non-parametric representation of non-linear dynamics, and (3) Random Fourier features and random projections: which enable scalable learning and inference by approximating kernel operations.

To our knowledge, this is the first filtering method for partially observable controlled systems that simultaneously enjoys these qualities: a learning algorithm with no local optima, flexibility to represent non-linear systems and efficient training and inference. In the following sections we describe the aforementioned concepts in more detail. We then describe our proposed method and demonstrate its performance in learning to predict and learning to control.

1st Conference on Robot Learning (CoRL 2017), Mountain View, United States.

2 Predictive State Representations and Two Stage Regression

Predictive state representations (PSRs) are a class of models for filtering in dynamical systems. In a latent state model such as a Hidden Markov model, the filtering algorithm maintains a belief state in terms of a latent variable (e.g. as a distribution over HMM states). A predictive state filter, on the other hand, maintains the state as a distribution over future observations. This distribution can be represented by a vector of expected sufficient statistics of future observations, i.e.:

$$q_t \equiv \mathbb{E}[\psi(o_{t:t+k-1}) \mid o_{1:t-1}],$$

where q_t is the predictive state at time t and ψ is a feature function that is computed from k future observations [1]. In order to maintain the state, we need to learn a state update function

$$q_{t+1} = f(q_t, o_t)$$

By representing the state in terms of an observable quantity (observation features), predictive states allow for learning a filter by reduction to supervised learning [1, 2]. To learn the state update, we follow the approach in [2]: we define the *extended* predictive state p_t to represent the distribution of the next $k+1$ observations, i.e.

$$p_t = \mathbb{E}[\xi(o_{t:t+k}) \mid o_{1:t-1}],$$

where ξ is an extended feature function. We assume an unknown linear map W such that $p_t = Wq_t$ and a fixed conditioning function f_{filter} such that $q_{t+1} = f_{\text{filter}}(p_t, o_t)$. The definition of f_{filter} follows from the choice of ψ and ξ (e.g. if ψ and ξ produce indicator vectors, it follows that q_t and p_t are probability tables and f_{filter} amounts to applying Bayes rule).

To learn W, we apply two-stage regression: In **stage 1**, we collect training examples as triplets (ψ_t, ξ_t, h_t) where $\psi_t = \psi(o_{t:t+k-1})$, $\xi_t = \xi(o_{t:t+k})$ and $h_t = h(o_{t-l:t-1})$ indicate future, extended future and history features. We then build regression models to predict

$$\hat{q}_t \equiv \mathbb{E}[\psi_t \mid h_t], \qquad \hat{p}_t \equiv \mathbb{E}[\xi_t \mid h_t]. \tag{1}$$

In **stage 2**, we use $\hat{q}_t$ and $\hat{p}_t$ predicted by stage 1 models to learn W by solving a regression problem

$$\hat{p}_t \approx W\hat{q}_t. \tag{2}$$

Hefny et al. [2] prove the consistency of this algorithm assuming that stage 1 regression models are consistent. Extending this framework to controlled systems requires changing the predictive state q_t to encode the *conditional* distribution of future observations $o_{t:t+k-1}$ conditioned on future actions $a_{t:t+k-1}$. This representation makes the state independent of the policy used to generate actions. We discuss the new representation and its implications in the following sections.

3 Hilbert State Embeddings of Distributions

As mentioned in the previous section, we need a state representation that encodes a conditional distribution. For a discrete system with a few observations and actions, a suitable representation is a conditional probability table. In this section we describe how to extend the notion of a conditional probability table to the continuous case through Hilbert space embedding (HSE) of distributions [3, 4, 5]. Let $k_{\mathcal{X}}$ be a kernel function defined on $\mathcal{X} \times \mathcal{X}$. This function is associated with a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_{\mathcal{X}}$ and a feature function $\phi_{\mathcal{X}} : \mathcal{X} \mapsto \mathcal{H}_{\mathcal{X}}$ such that $k(x_1, x_2) = \langle \phi_{\mathcal{X}}(x_1), \phi_{\mathcal{X}}(x_2) \rangle_{\mathcal{H}_{\mathcal{X}}}$, where $\langle ., . \rangle_{\mathcal{H}_{\mathcal{X}}}$ denotes inner product in $\mathcal{H}_{\mathcal{X}}$.

Given a random variable $X \in \mathcal{X}$, the *mean map* of X is defined as $\mu_X \equiv \mathbb{E}[\phi_{\mathcal{X}}(X)]$. If the kernel $k_{\mathcal{X}}$ is *universal* (e.g. the Gaussian RBF), then the mean map μ_X is a sufficient representation of the distribution of X. Given two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, the *covariance operator* of X and Y is defined as $C_{XY} \equiv \mathbb{E}[\phi_{\mathcal{X}}(X) \otimes \phi_{\mathcal{Y}}(Y)]$, where $\otimes$ denotes outer product. The covariance operator is a sufficient representation of the joint distribution $P(X, Y)$. The conditional operator $\mathcal{W}_{X|Y}$ is a linear operator satisfying

$$\mu_{X|Y=y} \equiv \mathbb{E}[\phi_{\mathcal{X}}(x) \mid Y = y] = \mathcal{W}_{X|Y}\phi_{\mathcal{Y}}(y) \equiv \text{apply}(\mathcal{W}_{X|Y}, y).$$

[1] The choice of k depends on the observability of the system. A system is k-observable if each belief state defines a unique distribution of the next k observations. In practice, k can be chosen using cross-validation.

According to kernel Bayes rule [6], the following relation holds [2]

$$\mathcal{W}_{X|Y} = C_{XY} C_{YY}^{-1}.$$

Given the above discussion, a natural representation of the state is the conditional operator $\mathcal{W}_{o_{t:t+k-1}|a_{t:t+k-1}}$. This is the essence of Hilbert space embedding of predictive state representations (HSE-PSRs) [7]. Since $\mathcal{W}_{o_{t:t+k-1}|a_{t:t+k-1}}$ is typically infinite dimensional, Boots et al. [7] use a Gram matrix formulation for training and inference, resulting in costly procedure whose computational and space complexity that scale polynomially in the number of training examples. To obtain a two-stage regression method that scales linearly with training examples, we need an efficient approximation of kernel operations and we need an adaptation of stage 1 regression to estimate conditional states. We discuss these issues in Sections 3.1 and 4.2 respectively.

3.1 Kernel Approximation for Scalable Learning and Inference

To avoid the need for a costly Gram matrix formulation, we resort to kernel approximations. Kernel approximations attempt to replace the infinite dimensional vector $\phi_{\mathcal{X}}(x)$ with a finite dimensional approximation $\hat{\phi}_{\mathcal{X}}(x) \in \mathbb{R}^D$ such that $k_{\mathcal{X}}(x_1, x_2) \approx \hat{\phi}_{\mathcal{X}}(x_1)^\top \hat{\phi}_{\mathcal{X}}(x_2)$. In this work, we use Random Fourier features (RFF) approximation [8]. RFF approximation is simple to implement but it is data independent and typically requires D to be very large to give an acceptable approximation. Therefore we post-process RFF vectors by projecting them on a p dimensional space using randomized PCA [9]. Under this representation, mean maps are p-dimensional vectors and covariance operators are $p \times p$ matrices. States are also $p \times p$ conditional matrices but we use PCA again to project them into p-dimensional vectors in order to reduce the number of parameters.

4 Predictive State Controlled Models

We are now ready to specify PSCMs in more detail. As described in Sections 3, the predictive state q_t is a p-dimensional approximation of $\mathcal{W}_{o_{t:t+k-1}|a_{t:t+k-1}}$. The extended state p_t consists of two vectors. $p_t^{(1)}$ approximates $\mathcal{W}_{o_t, o_{t:t+k}|a_t, a_{t+1:t+k}}$ while $p_t^{(2)}$ approximates $\mathcal{W}_{o_t|a_t}$. [3]

4.1 State Update

Given a predictive state q_t, an action a_t and the resulting observation o_t, q_{t+1} is computed as follows: (1) We compute $p_t^{(1)} = W^{(1)} q_t$ and $p_t^{(2)} = W^{(2)} q_t$, where $W^{(1)}$ and $W^{(2)}$ are appropriate blocks of W in Equation 2. (2) Given $p_t^{(2)}$ we apply a_t to obtain a distribution over o_t represented by $C_{o_t o_t}$. (3) Given $p_t^{(1)}$ and $C_{o_t o_t}$ we use kernel Bayes rule to obtain $\mathcal{W}_{o_{t+1:t+k}|a_t, a_{t+1:t+k}, o_t}$. [4] (4) By applying o_t and a_t we obtain $\mathcal{W}_{o_{t+1:t+k}|a_{t+1:t+k}} \equiv q_{t+1}$.

4.2 Learning Model Parameters

The parameters $W^{(1)}$ and $W^{(2)}$ can be learnt using the two stage regression method described in Section 2. However, stage 1 regression to estimate $\hat{q}_t = \mathbb{E}[q_t \mid h_t]$ needs to be adapted to the conditional distribution representation [5]. We propose two approaches. In the **joint approach** we train regression models to estimate $\hat{C}_{o_{t:t+k-1} a_{t:t+k-1}} \equiv \mathbb{E}[\phi_O(o_{t:t+k-1}) \otimes \phi_A(a_{t:t+k-1}) \mid h_t]$ and $\hat{C}_{a_{t:t+k-1} a_{t:t+k-1}}$ and then use kernel Bayes rule to compute $\hat{q}_t = \hat{C}_{o_{t:t+k-1} a_{t:t+k-1}} \hat{C}^{-1}_{a_{t:t+k-1} a_{t:t+k-1}}$.

[2] It is informative to match HSE concepts to their discrete counterparts. If X and Y are discrete variables with cardinality m and we use the delta kernel $k(x_1, x_2) = 1(x_1 = x_2)$, it follows that $\phi(x)$ is an indicator (one-hot) vector, μ_X is a probability vector, C_{XY} is a joint probability table, and $\mathcal{W}_{X|Y}$ is a conditional probability table.

[3] In an abuse of notation we denote by $\mathcal{W}_{o_t|a_t}$ the operator that satisfies $C_{o_t, o_t} = \mathcal{W}_{o_t|a_t} \phi(a_t)$. The covariance C_{o_t, o_t} is needed to condition on o_t using kernel Bayes rule.

[4] We think of $p_t^{(1)}$ as a 4-mode tensor, with modes corresponding to $o_t, o_{t:t+k} \mid a_t$ and $a_{t+1:t+k}$, and we multiply the o_t mode with $C^{-1}_{o_t, o_t}$.

[5] We assume that training data is gathered from an exploratory blind policy.

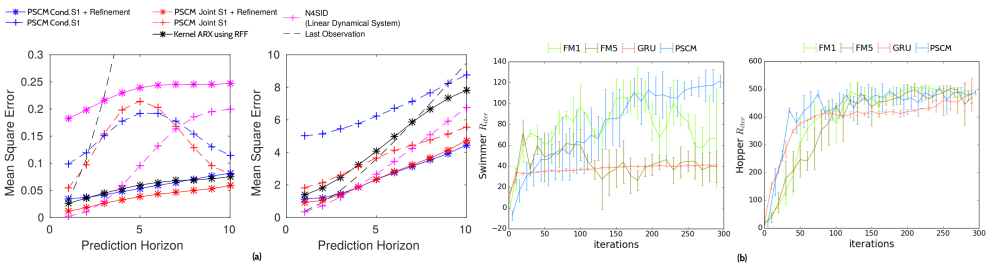

Figure 1: (a) Mean square error for 10-step prediction on synthetic model (left) and swimming robot simulation (right). Baselines with very high MSE are not shown for clarity. (b) average reward in best 3 out of 5 runs on swimmer (left) and hopper (right). FM1(FM5) models use the last observation(5 observations) as input to policy.

In the **conditional approach** we learn a function $\hat{q}_t = g(h_t)$ by directly solving the problem

$$\min_{g \in \mathcal{G}} \sum_t \|\text{apply}(g(h_t), a_{t:t+k-1}) - \phi_O(o_{t:t+k-1})\|^2.$$

4.3 Local Refinement by Backpropagation

Two stage regression algorithm is an instance of the method of moments, which is known for its statistical inefficiency. Therefore, it is common to use method of moments to initialize a local optimization procedure. Noting that a PSCM essentially defines a recurrent computation $q_{t+1} = f_{\text{cond}}(W^{(1)}q_t, W^{(2)}q_t, o_t, a_t)$ means hat we can optimize the parameters $W^{(1)}$ and $W^{(2)}$ using backpropagation through time [10] to minimize the objective $\sum_t \|\text{apply}(q_t, a_{t:t+k-1}) - o_{t:t+k-1}\|^2$. Our experiments show that this results in significant gains in prediction accuracy.

5 Experiments: Prediction

In this experiment, we test the ability of PSCM to learn a model of a controlled environment with continuous non-linear dynamics , and to predict future observations given future actions. We compare our method against a set of common filtering methods on a synthetic non-linear benchmark [7] and on a simulation of a swimming robot. In both cases, we test the ability to predict up to 10 observations in the future. Results are shown in Figure 1 (a). With local refinement, PSCMs with joint S1 outperforms other baselines, especially in long-range predictions.

6 Experiments: Reinforcement Learning

In this experiment, we employ PSCMs in a reinforcement learning (RL) setting. The RL agent consists of a PSCM and a reactive policy. The PSCM provides predictive states, which are used as inputs by the reactive policy. The reactive policy is represented by a feed-forward neural network that, given a predictive state, computes the mean vector and diagonal covariance matrix of a Gaussian distribution over actions.

We jointly train the PSCM and the reactive policy using alternating optimization: given a batch of trajectories generated by interaction with the environment using the current policy, the PSCM is updated to minimize prediction error using back-propagation through time as described in Section 4.3 while the reactive policy is updated by a step of Trust-region Policy Optimization algorithm (TRPO) [11].

We applied our method to two partially observable environments with continuous controls. These environments are based on Mujoco environments[12] from OpenAI Gym[13]; however, only joint positions are given to the agent as observations. Results are shown in Figure 1 (b). PSCM is competitive in the Walker environment and clearly outperforms other methods in the Swimmer environment.

7 Conclusion

We described a method for maintaining a state representation in a partially-observable controlled environment with continuous non-linear dynamics. Predictive state controlled models combine predictive state representations, Hilbert space embedding and kernel approximation to obtain a model with scalable and local minima-free learning and inference. Our experiments show that PSCMs constitute a promising method for prediction and control. For future work, we aim to develop exploration strategies for reinforcement learning using PSCMs.

References

[1] W. Sun, A. Venkatraman, B. Boots, and J. A. Bagnell. Learning to filter with predictive state inference machines. In *Proceedings of the 2016 International Conference on Machine Learning (ICML-2016)*, 2016.

[2] A. Hefny, C. Downey, and G. J. Gordon. Supervised learning for dynamical system learning. In *NIPS*, 2015.

[3] A. Smola, A. Gretton, L. Song, and B. Schlkopf. A hilbert space embedding for distributions. In *In Algorithmic Learning Theory: 18th International Conference*, 2007.

[4] L. Song, J. Huang, A. J. Smola, and K. Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *ICML*, 2009.

[5] J. Langford, R. Salakhutdinov, and T. Zhang. Learning nonlinear dynamic models. In *ICML*, 2009.

[6] K. Fukumizu, L. Song, and A. Gretton. Kernel bayes' rule: Bayesian inference with positive definite kernels. *Journal of Machine Learning Research*, 14(1), 2013.

[7] B. Boots, A. Gretton, and G. J. Gordon. Hilbert Space Embeddings of Predictive State Representations. In *UAI*, 2013.

[8] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*. 2008.

[9] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 2011.

[10] P. Werbos. Backpropagation through time: what does it do and how to do it. In *Proceedings of IEEE*, volume 78, pages 1550–1560, 1990.

[11] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In D. Blei and F. Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1889–1897. JMLR Workshop and Conference Proceedings, 2015.

[12] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

[13] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.