Faster Policy Adaptation in Environments with Exogeneity: A State Augmentation Approach

Zhuoshu Li
Washington University in St. Louis
zhuoshuli@wustl.edu

Zhitang Chen
Noah's Ark Lab, Huawei Technologies
chenzhitang2@huawei.com

Pascal Poupart
University of Waterloo
ppoupart@uwaterloo.ca

Sanmay Das
Washington University in St. Louis
sanmay@wustl.edu

Yanhui Geng
Noah's Ark Lab, Huawei Technologies
geng.yanhui@huawei.com

ABSTRACT

The reinforcement learning literature typically assumes fixed state transition functions for the sake of tractability. However, in many real-world tasks, the state transition function changes over time, and this change may be governed by exogenous variables outside of the control loop. This can make policy learning difficult. In this paper, we propose a new algorithm to address the aforementioned challenge by embedding the state transition functions at different timestamps into a Reproducing Kernel Hilbert Space; the exogenous variable, as the cause of the state transition evolution, is estimated by projecting the embeddings into the subspace that preserves maximum variance. By augmenting the observable state vector with the estimated exogenous variable, standard RL algorithms such as Q-learning are able to learn faster and better. Experiments with both synthetic and real data demonstrate the superiority of our proposed algorithm over standard and advanced variants of Q-learning algorithms in dynamic environments.

KEYWORDS

Reinforcement learning; Q-learning; environments with exogeneity

ACM Reference Format:
Zhuoshu Li, Zhitang Chen, Pascal Poupart, Sanmay Das, and Yanhui Geng. 2018. Faster Policy Adaptation in Environments with Exogeneity: A State Augmentation Approach. In *Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), Stockholm, Sweden, July 10–15, 2018,* IFAAMAS, 9 pages.

1 INTRODUCTION

In Reinforcement Learning (RL), the environmental dynamics are often assumed to be fixed (i.e., the transition function does not evolve with time). When this assumption is violated, as it often is in practice, learning becomes challenging; past transitions stop being representative of current and future transitions, and RL methods therefore need to relearn the policy. While an agent is relearning its policy, performance is negatively affected.

In this paper, we develop a new RL algorithm for environments where the evolution of the state transition is caused by a continuous latent exogenous variable that varies with time, where the evolution of the variable itself is not affected by the action. Our algorithm (1) tracks in an online fashion and recovers the latent exogenous variable which causes changes to the observable transition function, and (2) augments the observable part of the process with the inferred exogenous variable. The inferred variable acts as an explicit signal and enables the RL algorithm to quickly adapt to the evolution of the system dynamics, improving the learning speed. Although the basic idea appears straight-forward, the challenge lies in tracking the variation of the system's state transition function, which is by nature a conditional distribution of high dimensional continuous variables and is difficult to estimate. In this paper, we leverage recent advances in Reproducing Kernel Hilbert space (RKHS) embeddings [11, 13, 29] to represent arbitrary conditional distributions in a non-parametric way. The variation of the state transition caused by the exogenous variable can be quantified by measuring the Hilbert-Schmidt norm of the difference of the conditional mean embeddings of the transition distributions at different time steps. We propose an approach that finds a proximal exogenous variable by a nonlinear projection of the conditional mean embeddings at different time steps into a manifold that preserves the maximum variance of those embeddings as well as an additive term that rewards higher autocorrelation of the latent variable over time. This regularization term implicitly improves the predictability of the process governing the exogenous variable.

We evaluate the effectiveness of our approach in three dynamic domains, including a stochastic toy task, a traditional pendulum control benchmark from the RL literature [3, 5, 30], and a network resource allocation problem with real data from a network equipment manufacturer. We show that our approach finds exogenous variables that cause changes to the observable transition function, and that help Q-learning find better policies while adapting to changes faster.

2 RELATED WORK

RL autonomously learns the policy for sequential decision-making by interacting with environments. It has demonstrated many successes over the years in training autonomous agents in complex, real world problems [18, 20, 21, 31]. RL problems in dynamic environments have been actively investigated. One stream of research considers environments with multiple agents that are simultaneously learning and therefore introducing environment dynamics evolution [1, 15, 18, 24]. These approaches adjust the learning rate and the amount of experience replay to mitigate the impact of time evolving state transitions. They do not identify the exogenous variables that are the source of the evolution, nor do they resolve it.

Proc. of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2018), M. Dastani, G. Sukthankar, E. André, S. Koenig (eds.), July 10–15, 2018, Stockholm, Sweden.

Another line of research considers domains where the environment dynamics are caused by different modes or contexts with a number of modes known a priori [7, 9] or learned on the fly [3, 27]. Partial models of the environment are learned for each mode and new modes/contexts are created when the environment dynamics are detected. However these approaches are restricted to problems with a small number of discrete modes.

Time evolving dynamics of state transitions can also be found in partially observable domains where the state transition of the observable part of the process is time varying, but becomes fixed when augmented with latent exogenous variables. A significant amount of work focuses on model-free reinforcement learning techniques that directly estimate a policy [2, 16, 22]. Since the state transition of the observable part of the process changes over time, the last observation is insufficient for selecting actions optimally. These methods typically augment the last observation with a finite discrete memory in which additional information from prior observations is stored to improve the decision process. Model-based approaches have also been explored to explicitly learn the latent state representation and associated dynamics of the process [12, 25, 33]. These approaches often assume that the set of possible states is known and fixed, but non-parametric approaches exist that grow the number of hidden states [8, 23] or use suffix trees of arbitrary depth as substitutes for states [32]. Closely related, predictive state representations can also be learned from past observations to predict future observations [4, 17, 28]. Very recently, with the advent of deep reinforcement learning, recurrent neural networks have also been used to learn the hidden part of a process that is fed to a deep Q-network to improve the accuracy of reinforcement learning in partially observable domains [14].

While most previous work assumes discrete modes, discrete hidden states, a parametric form for the dynamics, or training samples with true states available, we focus on processes with a continuous latent exogenous variable and leverage RKHS embeddings for conditional distributions to avoid any parametric assumptions on the dynamics and the availability of true state information.

3 PROBLEM DEFINITION

We formally define the type of RL problem in environments with exogeneity considered in this paper. The evolution of latent exogenous and observable processes over time is depicted in Figure 1, where

- $\mathbf{v}_t \in \mathcal{V} = \mathbb{R}^d$ represents the latent exogenous variable at t, which controls the evolution of transition functions and is changing with time t. Here, we assume that $\mathbf{v}_t$ is a stationary process, that is, its own transition function does not evolve with time.
- $\mathbf{o}_t \in \mathcal{O} = \mathbb{R}^{d_o}$ is the observed vector at t. Without loss of generality, we treat this vector as a mixture of two underlying processes including an endogenous component $\boldsymbol{\eta}_t$, whose evolution is not affected by $\mathbf{v}_t$ but controlled by the policy and a semi-endogenous one $\boldsymbol{\zeta}_t$, which is affected by $\mathbf{v}_t$ in addition to the policy. The algorithm proposed in this paper is to recover an underlying process $\boldsymbol{\chi}_t = [\boldsymbol{\eta}_t, \boldsymbol{\zeta}_t]^T$ for which there exists a separation matrix $\mathbf{M}^{-1}$ such that $\boldsymbol{\chi}_t = \mathbf{M}^{-1}\mathbf{o}_t$.
- $\mathbf{a}_t \in \mathcal{A} = \mathbb{R}^{d_a}$ is the action at t;

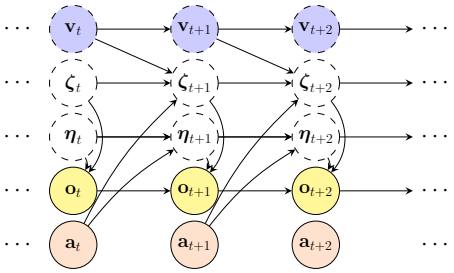

Figure 1: An illustration of MDPs in the dynamic environment with exogenous control variables, where the exogenous control variables are not observable.

- $p_t(\mathbf{o}_{t+1}|\mathbf{o}_t, \mathbf{a}_t)$ denotes the transition of the observations from $\mathbf{o}_t$ to $\mathbf{o}_{t+1}$ and it is time dependent (i.e., affected by latent exogenous variables) due to latent variable $\mathbf{v}_t$;
- $p(\mathbf{v}_{t+1}|\mathbf{v}_t)$ is the transition of the latent variable $\mathbf{v}_t$, which is assumed to be stationary.

In Figure 1, given that $\mathbf{v}_t$ is not observable, $\mathbf{o}_{t+1} = f(\mathbf{o}_t, \mathbf{a}_t, \mathbf{v}_t, \mathbf{e}_s^t) = f_t(\mathbf{o}_t, \mathbf{a}_t, \mathbf{e}_s^t)$, where $\mathbf{e}_s^t$ is some external noise. Furthermore, $\mathbf{v}_{t+1} = g(\mathbf{v}_t, \mathbf{e}_z^t)$, where $\mathbf{e}_z^t$ represents noise at time t.

Proposition 3.1. *If the latent exogenous variable $\mathbf{v}_t$ is accurately estimated and concatenated with the observation vector, the time varying state transition reduces to a fixed one.*

The proof is relatively easy. If $\mathbf{v}_t$ is accurately estimated and concatenated with the observed vector, then the transition function becomes $p(\mathbf{o}_{t+1}, \mathbf{v}_{t+1}|\mathbf{o}_t, \mathbf{v}_t, \mathbf{a}_t)$. Since $\mathbf{o}_{t+1}$ and $\mathbf{v}_{t+1}$ are conditionally independent given $\mathbf{o}_t, \mathbf{v}_t, \mathbf{a}_t$, we obtain the following factorization $p(\mathbf{o}_{t+1}, \mathbf{v}_{t+1}|\mathbf{o}_t, \mathbf{v}_t, \mathbf{a}_t) = p(\mathbf{v}_{t+1}|\mathbf{v}_t)p(\mathbf{o}_{t+1}|\mathbf{o}_t, \mathbf{v}_t, \mathbf{a}_t)$ where $p(\mathbf{o}_{t+1}|\mathbf{o}_t, \mathbf{v}_t, \mathbf{a}_t)$ is a fixed conditional distribution and the state transition function of the overall process is time invariant.

We propose an approach based on kernel conditional mean embeddings to recover and predict the latent exogenous variable that we concatenate with the observable process to speed up policy adaptation. Throughout the paper, we follow some basic assumptions about the evolution of exogenous variables that are commonly made in the literature [3, 7, 27]: (1) The transitions of latent exogenous variables are stationary and Markovian; (2) Latent exogenous variables cannot be directly observed; (3) Latent exogenous variable transitions are independent of the control system's responses.

4 THE STATE AUGMENTATION ALGORITHM

The key idea of our approach is to use a Reproducing Kernel Hilbert space (RKHS) embedding to represent the state transition functions at different time steps and find a nonlinear projection of those conditional mean embeddings into a manifold that preserves the maximum variance, which gives us a proximal estimation of the exogenous variable. Then we augment the observable process with the learned exogenous variable to speed up the RL process. We start by illustrating how to use RKHS embeddings to represent transition functions, namely conditional distributions of state transitions, and also show that these conditional mean embeddings preserve

differences between the conditional distributions, which provide the foundation of our paper. We then present a detailed derivation of our approach, which consists of two phases: offline learning of the latent process and online updating. The offline phase initializes the nonlinear function, which characterizes the latent exogenous process on the subspace that preserves the maximum variance and explains the time evolution of the state transition function. In the online updating phase, the nonlinear function is revised each time a new sample $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$ is collected, and we use it to estimate the current value of the latent exogenous variable.

4.1 Characterizing Transition Functions in RKHS: Why and How?

As we focus on the time evolving transition function (conditional distributions), we need to capture the changes of the transition function. We use an operator $\mathcal{U}_{Y|X}$, which is an embedding of a conditional distribution $p(\mathbf{y}|\mathbf{x})$ in RKHS. Suppose we have kernels $k(\mathbf{x}, \cdot)$ and $k(\mathbf{y}, \cdot)$ for variables $\mathbf{x}$ and $\mathbf{y}$, with the corresponding RKHS $\mathcal{H}_X$ and $\mathcal{H}_Y$, respectively. The operator $\mathcal{U}_{Y|X}$ is defined as $\mathcal{U}_{Y|X} := C_{YX} C_{XX}^{-1}$, where C_{YX} and C_{XX} are the cross-covariance and covariance operators respectively [10]. Alternatively, $k(\mathbf{x}, \cdot)$ can also be viewed as a feature map denoted by $\phi(\mathbf{x})$ where $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$. If L is the number of samples, the empirical estimate of $\mathcal{U}_{Y|X}$ is

$$\hat{\mathcal{U}}_{Y|X} = \Gamma_Y (K_X + \epsilon L I)^{-1} \Phi_X^T,$$

where ϵ is a regularization parameter and Γ_Y, Φ_X, and K_X are the feature matrix on Y, feature matrix on X, and the kernel matrix on X, respectively [29].

As the difference of the transition function at any different time steps is used to capture the time variation of the process dynamics, the difference of the corresponding embeddings needs to quantify the difference of the original conditional distributions. We denote the difference between the conditional distributions $p_m(\mathbf{y}|\mathbf{x})$ and $p_n(\mathbf{y}|\mathbf{x})$ by $\mathcal{D}[p_m(\mathbf{y}|\mathbf{x})||p_n(\mathbf{y}|\mathbf{x})]$, then we have,

Theorem 4.1. *The divergence of two conditional distributions $\mathcal{D}[p_m(\mathbf{y}|\mathbf{x})||p_n(\mathbf{y}|\mathbf{x})]$ is preserved by the Hilbert-Schmidt norm of the difference of two corresponding conditional mean embeddings $||\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)}||^2$, i.e. $\mathcal{D}[p_m(\mathbf{y}|\mathbf{x})||p_n(\mathbf{y}|\mathbf{x})] \propto ||\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)}||^2$, where*

$$\mathcal{D}[p_m(\mathbf{y}|\mathbf{x})||p_n(\mathbf{y}|\mathbf{x})] := \int_{\mathbf{y}} \Big(\int_{\mathbf{x}} p_m(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x} - p_n(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x} \Big)^2 d\mathbf{y}$$

can be interpreted as the infinite dimensional Euclidean space distance between two conditional distributions with respect to a reference distribution $\tau(\mathbf{x}) = \frac{1}{|\mathcal{X}|}$ in $\mathcal{X}$.

Proof. We begin by stating the relation between $\mathcal{D}[p_m(\mathbf{x})||p_n(\mathbf{x})]$ and $||\mu_X^{(m)} - \mu_X^{(n)}||^2$, which is the relation between the difference of two marginal distributions and their corresponding embeddings. This will serve as a foundation for the analysis of the difference of two conditional distributions. We first define

$$\mathcal{D}[p_m(\mathbf{x})||p_n(\mathbf{x})] := \int_{\mathbf{x}} (p_m(\mathbf{x}) - p_n(\mathbf{x}))^2 d\mathbf{x}. \tag{1}$$

Chen *et al.* [6] showed that for any two probability densities $p_m(\mathbf{x})$ and $p_n(\mathbf{x})$, if $\mu_X^{(m)}$ and $\mu_X^{(n)}$ are the corresponding RKHS embeddings, $\mathcal{D}[p_m(\mathbf{x})||p_n(\mathbf{x})]$ is proportional to $||\mu_X^{(m)} - \mu_X^{(n)}||^2$.

Similar to Equation (1), we define a divergence between two conditional distributions as follows:

$$\mathcal{D}[p_m(\mathbf{y}|\mathbf{x})||p_n(\mathbf{y}|\mathbf{x})] = \int_{\mathbf{y}} \Big(\int_{\mathbf{x}} p_m(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x} - \int_{\mathbf{x}} p_n(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x} \Big)^2 d\mathbf{y}.$$

Now, we have the marginal distribution

$$\bar{p}_m(\mathbf{y}) = \int_{\mathbf{x}} p_m(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x}, \tag{2}$$

where $\tau(\mathbf{x})$ is the reference distribution, $\tau(\mathbf{x}) = \frac{1}{|\mathcal{X}|}$, which is a uniform distribution in the domain $\mathcal{X}$. We can also get $\bar{p}_n(\mathbf{y}) = \int_{\mathbf{x}} p_n(\mathbf{y}|\mathbf{x}) \frac{1}{|\mathbf{x}|} d\mathbf{x}$. Then using $\bar{\mu}_Y^{(m)}$, which denotes the embedding of $\bar{p}_m(\mathbf{y})$, we have,

$$\bar{\mu}_Y^{(m)} = \int_{\mathbf{y}} \Big(\int_{\mathbf{x}} p_m(\mathbf{y}|\mathbf{x})\tau(\mathbf{x})d\mathbf{x} \Big) \phi(\mathbf{y}) d\mathbf{y}.$$

By simple algebra,

$$\bar{\mu}_Y^{(m)} = \int_{\mathbf{x}} \mu_{Y|x}^{(m)} \tau(\mathbf{x})d\mathbf{x} = \mathcal{U}_{Y|X}^{(m)} \bar{\mu}_\tau,$$

where $\bar{\mu}_\tau$ is the mean embedding of the reference distribution. Then, the difference of the two embeddings is

$$\begin{aligned} ||\bar{\mu}_Y^{(m)} - \bar{\mu}_Y^{(n)}||^2 &= tr[(\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)}) \bar{\mu}_\tau \bar{\mu}_\tau^T (\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)})^T] \\ &\approx \frac{1}{N} tr(\bar{\mu}_\tau^T \bar{\mu}_\tau) ||\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)}||^2. \end{aligned}$$

The last approximate equality follows free independence [34]. As $||\bar{\mu}_Y^{(m)} - \bar{\mu}_Y^{(n)}||^2$ preserves the relative magnitude of $\int_{\mathbf{y}} (\bar{p}_m(\mathbf{y}) - \bar{p}_n(\mathbf{y}))^2 d\mathbf{y}$, this means that we can use $||\mathcal{U}_{Y|X}^{(m)} - \mathcal{U}_{Y|X}^{(n)}||^2$ to measure the difference between two conditional distributions $p_m(\mathbf{y}|\mathbf{x})$ and $p_n(\mathbf{y}|\mathbf{x})$. □

In this section, we show that the conditional mean embeddings preserve the distance of the conditional distributions, which is defined as the infinite dimensional Euclidean distance between two conditional distributions with respect to a reference distribution of the conditioning variable. Thus in order to find the exogenous variable that causes the time variation of the state transition function, we can estimate it by projecting the conditional mean embeddings into the manifold that preserves the maximum amount of variance.

4.2 Offline Estimation of the Exogenous Variable

Without loss of generality, suppose the observation $\mathbf{o}_t$ is a mixture of a endogenous subspace $\boldsymbol{\eta}_t$ and the semi-endogenous subspace $\boldsymbol{\zeta}_t$ as shown in Sec. 3, i.e. $\boldsymbol{o}_t = \mathbf{M}[\boldsymbol{\eta}_t, \boldsymbol{\zeta}_t]^T$. To accurately estimate the latent exogenous variable $\mathbf{v}_t$ that contributes to the time evolution of the state transition $p_t(\mathbf{o}_{t+1}|\mathbf{o}_t, \mathbf{a}_t)$, we need to find the subspace $\boldsymbol{\zeta}_t$ that preserves maximum variance of its corresponding transition function, i.e. $\boldsymbol{\zeta}_t = \mathbf{W}_d^T \mathbf{o}_t$. We now show how to find the projection that yields maximum variance, and initialize the estimation of the

latent variable as a nonlinear function of the conditional mean embedding that characterizes the latent exogenous variables in the offline learning phase.

Suppose the offline phase is from time 0 to t_{tr}, we first sample N instances $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$ with equal space in the observable trajectory and the n^{th} instance is denoted by t_n. At each t_n, the conditional mean embedding $\mathcal{U}^{(n)}_{\zeta_{t+1}|\zeta_t,\mathbf{a}_t}$ of the state transition function in the subspace obtained by the projection $\mathbf{W}_d$ can be estimated by

$$\hat{\mathcal{U}}^{(n)}_{\zeta_{t+1}|\zeta_t,\mathbf{a}_t} = \Gamma^{(n)}\Lambda^{1/2}(\Lambda^{1/2}\mathbf{K}^{(n)}_{\zeta_t,\mathbf{a}_t}\Lambda^{1/2} + \lambda^t\epsilon I)^{-1}\Lambda^{1/2}(\Phi^{(n)})^T, \tag{3}$$

where Λ is a diagonal matrix with $\lambda_{ii} = \lambda^{L-i}$, and λ is a decay factor and $\lambda \in (0,1)$. L is the number of observed samples before t_n which satisfies $\lambda^L \approx 0$. $\Gamma^{(n)}$ and $\Phi^{(n)}$ are the feature matrices, i.e.

$$\Phi^{(n)} = [\phi(\zeta_{t_n+1-L}, \mathbf{a}_{t_n+1-L}), \ldots, \phi(\zeta_{t_n}, \mathbf{a}_{t_n})]$$

and

$$\Gamma^{(n)} = [\phi(\zeta_{t_n+2-L}), \ldots, \phi(\zeta_{t_n+1})].$$

We also have

$$\Omega^{(n)} = [\phi(\zeta_{t_n+1-L}), \ldots, \phi(\zeta_{t_n})],$$

and

$$\Theta^{(n)} = [\phi(\mathbf{a}_{t_n+1-L}), \ldots, \phi(\mathbf{a}_{t_n})].$$

Thus the Gram matrices from the feature matrices are $\mathbf{K}^{(n)}_{\zeta_t} = \Omega^{(n)^T}\Omega^{(n)}$, $\mathbf{K}^{(n)}_{\mathbf{a}_t} = \Theta^{(n)^T}\Theta^{(n)}$ and $\mathbf{K}^{(n)}_{\zeta_t,\mathbf{a}_t} = \mathbf{K}^{(n)}_{\zeta_t} \odot \mathbf{K}^{(n)}_{\mathbf{a}_t}$ which is the element-wise product. For the rest of the paper, we use $\mathcal{U}^{(n)}$ to represent the conditional mean embedding $\mathcal{U}^{(n)}_{\zeta_{t+1}|\zeta_t,\mathbf{a}_t}$ of t_n in the offline learning phase, and $\mathcal{U}_t$ to represent the conditional mean embedding $\mathcal{U}_{\zeta_{t+1}|\zeta_t,\mathbf{a}_t}$ at any time t. $\hat{\mathcal{U}}^{(n)}$ and $\hat{\mathcal{U}}_t$ are the corresponding empirical estimations respectively.

Finding Proximal Exogenous Variable with Maximum Variance. To estimate the latent exogenous variable, we assume that $\mathbf{v}_t$ can be approximated by a proximal exogenous variable as a non-linear function of $\mathcal{U}_t$, i.e. $f(\mathcal{U}_t)$. One can interprete $f(\mathcal{U}_t)$ as a nonlinear projection of the conditional mean embeddings to a manifold where the difference amongst the conditional mean embeddings of all time steps is maximized. A similar approach is also proposed by Zhang *et al.* ([2015]) to deal with causal discovery tasks with non-stationarity.

To find the nonlinear function, we map the conditional mean embedding to its own RKHS $\mathcal{H}_U$, where $\mathcal{H}_U$ is a RKHS spanned by $\phi(\mathcal{U}_t)$, i.e. $\mathcal{U}_t \mapsto \phi(\mathcal{U}_t)$, and $\phi(\mathcal{U}_t)$ is the centered feature mapping of $\mathcal{U}_t$. We assume that $f \in \mathcal{H}_U$ such that

$$\mathbf{v}_t = f(\mathcal{U}_t) = \boldsymbol{\alpha}^T\phi(\mathcal{U}_t), \tag{4}$$

here $\boldsymbol{\alpha}$ is a coefficient vector to be estimated. We note that $\boldsymbol{\alpha}$ lies in the span of $\phi(\mathcal{U}^{(0)}), \cdots, \phi(\mathcal{U}^{(N-1)})$ and thus we have $\boldsymbol{\alpha} = \sum_{i=0}^{N-1}\tilde{\alpha}_i\phi(\mathcal{U}^{(i)}) = \Phi_U\tilde{\boldsymbol{\alpha}}$, where $\Phi_U = \left[\phi(\mathcal{U}^{(0)}), \ldots, \phi(\mathcal{U}^{(N-1)})\right]$ and $\tilde{\boldsymbol{\alpha}}$ is the vector of coordinates to be estimated. Thus, the variance of $f(\mathcal{U}_t)$ can be calculated as

$$\hat{\sigma}^2_f = \frac{1}{N}\boldsymbol{\alpha}^T\Phi_U\Phi_U^T\boldsymbol{\alpha} = \frac{1}{N}\tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U^2\tilde{\boldsymbol{\alpha}},$$
$$s.t.\ \|f\|^2_{\mathcal{H}_U} = \tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U\tilde{\boldsymbol{\alpha}} = 1,$$

where $[\mathbf{K}_U]_{m,n} = k(\hat{\mathcal{U}}^{(m)}, \hat{\mathcal{U}}^{(n)})$, $k(\cdot,\cdot)$ is a positive definite kernel on the conditional mean embedding.

Regularization with Predictability. To guarantee that $[\mathbf{v}_t, \mathbf{o}_t]^T$ is Markovian, we need to ensure that the latent variable transition process of $\mathbf{v}_t$ is Markovian. Thus, we add a regularization term based on maximizing the autocorrelation σ^2_{+f} of any two consecutive instances t_n, t_{n+1} in the offline learning trajectory:

$$\sigma^2_{+f} = \mathbb{E}[\hat{\mathbf{v}}_{t_n}\hat{\mathbf{v}}_{t_{n+1}}] \approx \frac{1}{N}\sum_{i=0}^{N-1}\hat{\mathbf{v}}_{t_i}\hat{\mathbf{v}}^s_{t_i}, \tag{5}$$

where $\hat{\mathbf{v}}_{t_i}$ is the estimated proximal exogenous variable at time t_i and $\hat{\mathbf{v}}^s_{t_i}$ is the estimated proximal exogenous variable at time $t_i + 1$ correspondingly. Define

$$\mathbf{k}_U(\hat{\mathcal{U}}^{(i)}) = [k(\hat{\mathcal{U}}^{(0)}, \hat{\mathcal{U}}^{(i)}), \ldots, k(\hat{\mathcal{U}}^{(N-1)}, \hat{\mathcal{U}}^{(i)}]^T.$$

Then, by using the kernel trick and further expanding the above Equation (5), we get

$$\hat{\sigma}^2_{+f} = \frac{1}{N}\sum_{i=0}^{N-1}\tilde{\boldsymbol{\alpha}}^T\mathbf{k}_U(\hat{\mathcal{U}}^{(i)})(\tilde{\boldsymbol{\alpha}}^T\mathbf{k}_U(\hat{\mathcal{U}}^{(i+1)}))^T$$
$$= \frac{1}{N}\tilde{\boldsymbol{\alpha}}^T\frac{\mathbf{K}_U\mathbf{K}_U^{(+)} + \left(\mathbf{K}_U^{(+)}\right)^T\mathbf{K}_U}{2}\tilde{\boldsymbol{\alpha}},$$

where $\mathbf{K}_U^{(+)} = \left[k_U(\hat{\mathcal{U}}^{(1)}), \ldots, k_U(\hat{\mathcal{U}}^{(0)})\right]$. As $\mathbb{E}[\hat{\mathbf{v}}_{t_n}\hat{\mathbf{v}}_{t_{n+1}}]$ will be estimated based on data from multiple pairs of consecutive instances, a high autocorrelation implies that the process is "more predictable".

By combining these two parts, we have the following constrained optimization problem to learn the projection W_d and the vector of coordinates $\tilde{\alpha}$ for the non-linear function f,

$$\mathbf{W}_d^*, \tilde{\boldsymbol{\alpha}}^* = \operatorname{argmax}\hat{\sigma}^2_f + \gamma\hat{\sigma}^2_{+f}$$
$$s.t. \tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U\tilde{\boldsymbol{\alpha}} = 1, \text{and } \mathbf{W}_d^T\mathbf{W}_d = \mathbf{I}_d. \tag{6}$$

To solve the above optimization problem, we use an alternating optimization method. First, we fix $\mathbf{W}_d$ to be the value obtained in the last iteration. The optimization of $\tilde{\boldsymbol{\alpha}}$ reduces to a simple generalized eigen-decomposition problem. Second, we fix $\tilde{\boldsymbol{\alpha}}$ and optimize $\mathbf{W}_d$. The gradient of $\mathbf{W}_d$ is

$$\nabla\mathbf{W}_d = \underbrace{\frac{1}{N}\frac{\partial\tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U^2\tilde{\boldsymbol{\alpha}}}{\partial\mathbf{W}_d}}_{(a)} + \lambda\underbrace{\frac{1}{N}\frac{\partial\tilde{\boldsymbol{\alpha}}^T\mathbf{P}\tilde{\boldsymbol{\alpha}}}{\partial\mathbf{W}_d}}_{(b)}. \tag{7}$$

Here part (a) of Equation (7) can be derived as

$$(a) = \sum_i\sum_j tr\left(\frac{\partial\tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U^2\tilde{\boldsymbol{\alpha}}}{\partial[\mathbf{K}_U]_{i,j}}\right)\frac{\partial[\mathbf{K}_U]_{i,j}}{\partial\mathbf{W}_d}$$

where $\frac{\partial\tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U^2\tilde{\boldsymbol{\alpha}}}{\partial[\mathbf{K}_U]_{i,j}}$ and $\frac{\partial[\mathbf{K}_U]_{i,j}}{\partial\mathbf{W}_d}$ can be easily obtained by matrix calculus. $\frac{\partial\tilde{\boldsymbol{\alpha}}^T\mathbf{K}_U^2\tilde{\boldsymbol{\alpha}}}{\partial\mathbf{K}_{U\,i,j}} = \tilde{\boldsymbol{\alpha}}^T(\mathbf{K}_U S^{(i,j)} + S^{(i,j)}\mathbf{K}_U)\tilde{\boldsymbol{\alpha}}$ where $S^{(i,j)}$ is a matrix where $[S^{(i,j)}]_{p,q} = 1$, if $p = i$ and $q = j$ and $[S^{(i,j)}] = 0$, otherwise. $\frac{\partial[\mathbf{K}_U]_{i,j}}{\partial\mathbf{W}_d}$ can also be obtained by the chain rule of the matrix derivative. Part (b) of Equation (7) can be solved using a similar idea. Due to constraint $\mathbf{W}_d^T\mathbf{W}_d = I_d$, the solution lies in the Grassmannian manifold. We apply conjugate gradient descent

to solve the optimization problem. The alternating optimization is repeated until convergence.

Algorithm 1 State-Augmentation Algorithm

function Offline Initialization of the Non-linear Function in the Exogenous Subspace
  Collect observation transition trajectories and select randomly N instances denoted by t_n.
  Initialize projection $\mathbf{W}_d$ for extracting the exogenous subspace.
  repeat
    $\boldsymbol{\zeta}_t = \mathbf{W}_d \mathbf{o}_t$.
    Estimate kernel embeddings of conditional distribution $\hat{\mathcal{U}}^{(n)}$ for each t_n.
    Based on Eq. (6): (1) Optimize $\tilde{\boldsymbol{\alpha}}$ using eigen-decomposition; (2) Update $\mathbf{W}_d$ using conjugate gradient descent.
  until $\mathbf{W}_d$ and $\tilde{\boldsymbol{\alpha}}$ converge.
end function
function Online Estimation of Latent Variables
  Get a new sample $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1}) \overset{\mathbf{W}_d}{\rightarrow} (\boldsymbol{\zeta}_t, \mathbf{a}_t, \boldsymbol{\zeta}_{t+1})$.
  Update Π_t and P_t as in Eq. (9) and (10), and $\hat{\mathcal{U}}_t = P_t \Pi_t$.
  Update $\tilde{\boldsymbol{\alpha}}_t$ as in Eq. (11).
  Estimate $\hat{\mathbf{v}}_t = \tilde{\boldsymbol{\alpha}}_t^T \Phi_U^T \phi(\hat{\mathcal{U}}^{(t)})$.
  Get $\mathbf{o}_t^{aug} = [\mathbf{v}_t, \mathbf{o}_t]^T$ and feed $\mathbf{o}_t^{aug}$ in Q-learning.
end function

4.3 Online Update of the Exogenous Variable

In this section, we show how to estimate the latent variable $\mathbf{v}_t$ in an online fashion. The first difficulty lies in estimating the conditional mean embedding $\mathcal{U}_t$ in an online fashion when each new data point arrives. Now instead of relying on the implicit kernel mapping ϕ, which leads the curse of dimensionality, we explicitly map the data to a high dimensional feature space with Fourier Series. This idea, which employs Bochner's theorem on shift-invariant kernels, provides more scalability and feasibility of online learning.

Assume we have a shift-invariant kernel, i.e. $k(\mathbf{x}, \mathbf{y}) = k(\mathbf{x}-\mathbf{y}) \triangleq k(\delta)$, if $k(\delta)$ is properly scaled, Bochner's theorem [26] guarantees that its Fourier transform $p(w)$ is a proper probability distribution, i.e.,

$$\begin{aligned} k(\mathbf{x}-\mathbf{y}) &= \int_{\mathcal{R}^d} p(w) \exp(-j w^T(\mathbf{x}-\mathbf{y}))\, dw \\ &= \mathbb{E}_p[\langle(\cos(w^T\mathbf{x}), \sin(w^T\mathbf{x})), (\cos(w^T\mathbf{y}), \sin(w^T\mathbf{y}))\rangle]. \end{aligned}$$

This means that if we draw a random vector w according to $p(w)$ and form two vectors $\phi(\mathbf{x}) = (\cos(w^T\mathbf{x}), \sin(w^T\mathbf{x}))$ and $\phi(\mathbf{y}) = (\cos(w^T\mathbf{y}), \sin(w^T\mathbf{y}))$, then the expected value of $\langle\phi(\mathbf{x}), \phi(\mathbf{y})\rangle$ is $k(\mathbf{x}, \mathbf{y})$. Therefore, for $\mathbf{x} \in \mathcal{R}^d$ and a large k, if we choose the transformation

$$\phi(\mathbf{x}) = \frac{1}{\sqrt{k}}[\cos(w_1^T\mathbf{x}), \sin(w_1^T\mathbf{x}), \ldots, \cos(w_k^T\mathbf{x}), \sin(w_k^T\mathbf{x})]. \quad (8)$$

with $w_1, \ldots, w_k$ drawn according to $p(w)$, linear inner products in this transformed space will approximate $k(\cdot, \cdot)$. More details can be found in [26]. Thus, as each new sample $(\boldsymbol{\zeta}_t, \mathbf{a}_t, \boldsymbol{\zeta}_{t+1})$ arrives, we can explicitly represent $\phi(\boldsymbol{\zeta}_t, \mathbf{a}_t)$ and $\phi(\boldsymbol{\zeta}_{t+1})$ using Equation (8).

Now, we show how to estimate $\mathcal{U}_t$. According to the definition of the conditional mean embedding $\mathcal{U}_{Y|X} := C_{YX} C_{XX}^{-1}$, we have $\hat{C}_{YX} = \frac{1}{t}\sum_i^t \lambda^{t-i} \phi(\boldsymbol{\zeta}_{i+1}) \phi(\boldsymbol{\zeta}_i, \mathbf{a}_i)^T$, i.e. $\hat{C}_{YX} = \frac{1}{t}\Gamma_t \Lambda_t \Phi_t^T$, where $\Gamma_t = [\phi(\boldsymbol{\zeta}_1), \ldots, \phi(\boldsymbol{\zeta}_{t+1})]$, $\Phi_t = [\phi(\boldsymbol{\zeta}_0, \mathbf{a}_0), \ldots, \phi(\boldsymbol{\zeta}_t, \mathbf{a}_t)]$ and Λ_t is a diagonal matrix with $\lambda_{ii} = \lambda^{t-i}$ and $\lambda \in (0, 1)$. We also have $\hat{C}_{XX} = \frac{1}{t}\Phi_t \Lambda_t \Phi_t^T$. Therefore, at any time t,

$$\hat{\mathcal{U}}_t = \Gamma_t \Lambda_t \Phi_t^T (\Phi_t \Lambda_t \Phi_t^T + \lambda^t \epsilon I)^{-1}.$$

Suppose we now have a new sample $(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}, \boldsymbol{\zeta}_{t+2})$, we want to estimate $\hat{\mathcal{U}}_{t+1}$. Note that

$$\begin{aligned} \Phi_{t+1}\Lambda_{t+1}\Phi_{t+1}^T &= [\Phi_t, \phi(\boldsymbol{\zeta}_{t+1})] \begin{bmatrix} \lambda\Lambda_t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Phi_t^T \\ \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1})^T \end{bmatrix}, \\ &= \lambda \Phi_t \Lambda_t \Phi_t^T + \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}) \phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}). \end{aligned}$$

Thus,

$$\begin{aligned} &(\Phi_{t+1}\Lambda_{t+1}\Phi_{t+1}^T + \lambda^{t+1}\epsilon I)^{-1} = \\ &\lambda^{-1}(\Phi_t \Lambda_t \Phi_t^T + \lambda^t \epsilon I + 1/\lambda \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}) \phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}))^{-1}. \end{aligned}$$

Denoting $\Pi_t = (\Phi_t \Lambda_t \Phi_t^T + \lambda^t \epsilon I)^{-1}$ and utilizing the Woodbury identity [36], we obtain

$$\Pi_{t+1} = \lambda^{-1}\Pi_t - \frac{\lambda^{-2}\Pi_t \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}) \phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}) \Pi_t}{1 + \lambda^{-1}\phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}) \Pi_t \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1})}. \quad (9)$$

Similarly,

$$\begin{aligned} \Gamma_{t+1}\Lambda_{t+1}\Phi_{t+1}^T &= [\Gamma_t, \phi(\boldsymbol{\zeta}_{t+2})] \begin{bmatrix} \lambda\Lambda_t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Phi_t^T \\ \phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1})^T \end{bmatrix}, \\ &= \lambda \Gamma_t \Lambda_t \Phi_t^T + \phi(\boldsymbol{\zeta}_{t+2}) \phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}). \end{aligned}$$

Denote by $P_t = \Gamma_t \Lambda_t \Phi_t$, we have

$$P_{t+1} = \lambda P_t + \phi(\boldsymbol{\zeta}_{t+2}) \phi^T(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_{t+1}). \quad (10)$$

$\phi(\boldsymbol{\zeta}_{t+1}, \mathbf{a}_t)$ and $\phi(\boldsymbol{\zeta}_{t+1})$ can be calculated based on Equation (8). The update of Π_{t+1} and P_{t+1} is similar to the idea of recursive least-squares algorithms, which avoid calculating the inverse Π_t in each update. In this case, we can compute the conditional mean embedding online,

$$\hat{\mathcal{U}}_{t+1} = P_{t+1}\Pi_{t+1}.$$

In our experiments, P_0 and Π_0 are initialized by utilizing $\Gamma^{(n)}$ and $\Phi^{(n)}$ from the last instance t_{N-1} in the offline learning phase.

Updating Nonlinear Projection. The latent variable is defined as in Equation (4), $\mathbf{v}_t = \boldsymbol{\alpha}^T \phi(\mathcal{U}_t)$. $\boldsymbol{\alpha}$ can be updated incrementally by computing the top eigenvector of unknown covariance [19]

$$\begin{aligned} \boldsymbol{\alpha}_{t+1} =& \boldsymbol{\alpha}_t + \frac{1}{t+1}(\phi(\hat{\mathcal{U}}_{t+1})(\phi^T(\hat{\mathcal{U}}_{t+1}) + \gamma\phi^T(\hat{\mathcal{U}}_t)) \\ &+ \frac{\boldsymbol{\alpha}_t^T \phi(\hat{\mathcal{U}}_{t+1})(\phi^T(\hat{\mathcal{U}}_{t+1}) + \gamma\phi^T(\hat{\mathcal{U}}_t))\boldsymbol{\alpha}_t}{||\boldsymbol{\alpha}_t||^2} I_d)\boldsymbol{\alpha}_t. \end{aligned}$$

As $\boldsymbol{\alpha} = \Phi_U \tilde{\boldsymbol{\alpha}}$, where $\Phi_U = \left[\phi(\mathcal{U}^{(0)}), \ldots, \phi(\mathcal{U}^{(N-1)})\right]$, using the kernel trick we get,

$$\begin{aligned}\tilde{\boldsymbol{\alpha}}_{t+1} =& \tilde{\boldsymbol{\alpha}}_t + \frac{1}{t+1}\mathbf{K}_U^{-1}\mathbf{K}_{t+1}(\mathbf{K}_{t+1}^T + \gamma\mathbf{K}_t^T)\tilde{\boldsymbol{\alpha}}_t \\ &- \frac{1}{t+1}\frac{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_{t+1}(\mathbf{K}_{t+1}^T + \gamma\mathbf{K}_t^T)\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_U\tilde{\boldsymbol{\alpha}}_t}\tilde{\boldsymbol{\alpha}}_t,\end{aligned} \tag{11}$$

where $\mathbf{K}_{t+1} = \Phi_U^T\phi(\hat{\mathcal{U}}_{t+1})$, which is a column vector where $\mathbf{K}_{t+1,i} = k(\mathcal{U}^{(i)}, \hat{\mathcal{U}}_{t+1})$, and $[\mathbf{K}_U]_{m,n} = k(\hat{\mathcal{U}}^{(m)}, \hat{\mathcal{U}}^{(n)})$. The detailed derivation of Equation (11) can also be found in Appendix A. By adding a regularization term $\frac{\gamma}{t+1}\mathbf{K}_U^{-1}\mathbf{K}_t\mathbf{K}_t^T\tilde{\boldsymbol{\alpha}}_t - \frac{\gamma}{t+1}\frac{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_t\mathbf{K}_t^T\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_U\tilde{\boldsymbol{\alpha}}_t}\tilde{\boldsymbol{\alpha}}_t$ based on maximizing the autocorrelation of any two consecutive time steps, we can guarantee the predictability in this online updating, where γ is a regularization parameter. We obtain the latent variable by $\hat{\mathbf{v}}_{t+1} = \tilde{\boldsymbol{\alpha}_{t+1}}^T\Phi_U^T\phi(\hat{\mathcal{U}}^{(t+1)})$. This estimated latent variable is then concatenated with the original observed vector to form a new state vector: $\mathbf{o}_{t+1}^{aug} = [\mathbf{v}_{t+1}, \mathbf{o}_{t+1}]^T$.

5 EXPERIMENTS

We demonstrate the effectiveness of our method with a stochastic toy task, a traditional pendulum control benchmark from the RL literature [3, 5, 30], and a network resource allocation problem with real data. Our state-augmentation algorithm is directly combined with Q-learning (henceforth "State-Aug" or SA). We compare our approach with traditional Q-learning [35] (QL) and Q-learning augmented with the true latent exogenous variable (which we refer to loosely as "ground truth", or TRUE). We also compare it with two state-of-the-art approaches for reinforcement learning in dynamic environments based on extensions of Q-learning: Frequency Adjusted Q-learning (FAQL) [18] and Repeated Update Q-learning (RUQL) [1].[1] In addition, we compare our method with a nonparametric approach for policy learning for POMDPs which utilizes Hilbert Space Embeddings (HSE-POMDP or HSE) [23].[2]

For all our experiments, we define the following parameters: C is the parameter that controls the speed of environment changes; $t \in [0, t_{tr}]$, is the offline learning trajectory; $t \in [0, ..., t_{le}]$, is the online Q-learning period. For simplicity, throughout all experiments, we assume the dimension of the latent exogenous subspace $d = 1$. We choose hyperparameters $k = 50$ (the dimensionality of ϕ from Equation (8)) and decay factor $\lambda = 0.99$ for all our experiments (our results are not sensitive to varying k and λ). Q-values are parameterized by a Q-net (neural network) and the actions are discretized in all experiments.

5.1 Toy Task

We first consider a toy model that demonstrates the intuition of our state-augmentation algorithm. In this toy model, the observed vector is a mixture of the endogenous component $\eta_{t+1} = \eta_t + 0.1a_t + \epsilon_\eta$, and the semi-endogenous component $\zeta_{t+1} = |\sin(t/C)|\zeta_t + a_t + \epsilon_\zeta$ which is influenced by the exogenous variable $|\sin(t/C)|$, where $\epsilon_\eta, \epsilon_\zeta$ are noisy parameters generated from $\mathcal{N}(0, 0.1)$, and $C = 100$. The underlying state vector at time t is defined as $\boldsymbol{\chi}_t$. However, the observed vector $\mathbf{o}_t = \mathbf{M}\boldsymbol{\chi}_t$ is mixed according to matrix $\mathbf{M}$. The reward function is simply defined by $r(\mathbf{o}_t, \mathbf{a}_t) = \exp(-|\zeta_t - 3.0|)$. Since the state space is a mixture of both endogenous and exogenous components, the estimation accuracy of the latent exogenous variable is not satisfactory when we skip the projection step and treat the whole observed vector as exogenous. This is because the exogenous component(s) or subspace(s) is overwhelmed by the endogenous counterparts (see Figure 3-(a)). Figure 3-(b) demonstrates that our approach successfully recovers the latent variable (i.e., sin function) by first finding the exogenous subspace, and then extracting a latent variable based on the projection in that subspace. This result corroborates our intuition that a projection $\mathbf{W}_d$ is essential for good performance.

5.2 Pendulum Experiment

We compare the performance of different algorithms on a type of pendulum tasks commonly used in the POMDP literature. We simulate a pendulum with length $l = 1$m and mass $m = 1$kg distributed evenly along its length. The dynamic of the pendulum is modeled by $\ddot{\theta} = (glm\sin\theta + a - k\dot{\theta})/(ml^2)$, where $k = 0.1$Ns is a friction coefficient and $g = 9.81$ is the gravitational constant. The maximum torque is 5Nm. The state of the system is the angle and angular velocity of the pendulum $(\theta, \dot{\theta})$, however the agent only observes the angle. We also apply additive noise to the controls. The reward function in this experiment is $r(s, a) = \exp(-\theta^2/20 - \dot{\theta}^2/4000)$, where θ is mapped to $[-0.5\pi, 1.5\pi)$ to constrain the range of motion.

HSE-POMDP requires a training phase in which the true states are available, but we do not make this additional information available to the other approaches. The training phase of HSE-POMDP is collected by applying actions uniformly at random from states $(\theta, \dot{\theta})$ where $\theta, \dot{\theta}$ are drawn independently and uniformly at random from their ranges. The offline learning phase of our approach State-Aug begins at a random state, and applies actions uniformly at random at each step until t_{tr}. We vary the size of training/offline learning phase t_{tr} and examine the average reward of $t_{le} = 100$ learning steps. The maximum of the average rewards is 1. Table 1 shows rewards averaged over 100 experiments. Note that t_{tr} is the number of training samples for HSE-POMDP, but to the length of offline learning (which does not require true state) for State-Aug. State-Aug is efficient and learns well with a small set of offline examples, but continues to improve in performance as the length of the offline learning phase increases. HSE-POMDP, on the other hand, requires many more training examples (as well as needing the ground truth in these examples), and its performance still does not compare well with the other approaches. Since HSE requires computing the inverse of a $t_{tr} \times t_{tr}$ matrix each iteration, processing time increases significantly, quickly becoming a major constraint, as the number of training samples goes up.

5.3 Pendulum with Wind

In this experiment, we investigate the performance of the algorithms on a more complicated pendulum task. We simulate a pendulum exposed to horizontal wind, where the strength of the wind changes with time according to a sin function, $v = \sin(t/C), C = 100$. Hence, the non-stationarity or exogeneity is caused by the wind. In this experiment, both angle and angular velocity of the

[1] We explore several different settings of the parameter β in FAQL and report the best results.

[2] The parameter d in HSE-POMDP is set to 1 throughout all the experiments.

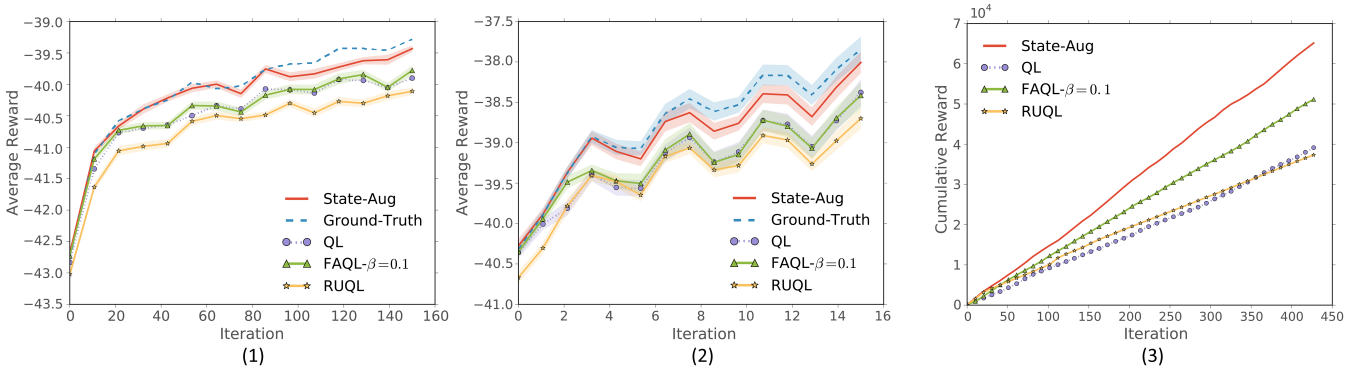

Figure 2: Performance comparisons in the (1) pendulum with wind strength $v = \sin(t/C), C = 100$, (2) the pendulum with wind strength $v = \sin(t/C)\sin(0.01t/C), C = 100$ and (3) pilot power control.

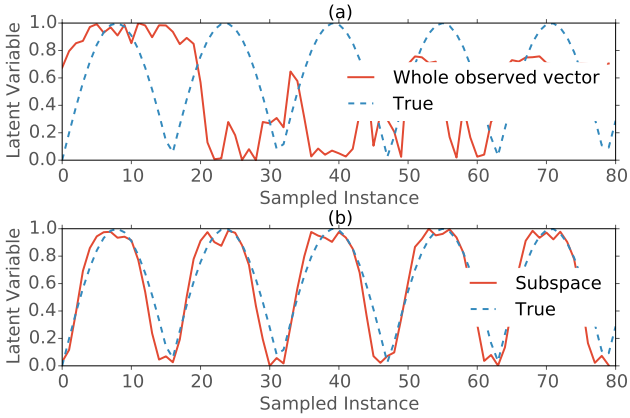

Figure 3: Inferred latent exogenous variable based on the entire observed vector (a) and using the exogenous subspace (b).

t_{tr}	SA	TRUE	QL	FAQL	RUQL	HSE
200	0.800	0.83	0.801	0.802	0.80	0.53
500	0.803	0.83	0.801	0.802	0.80	0.62
1000	0.808	0.83	0.801	0.802	0.80	0.68
2000	0.811	0.83	0.801	0.802	0.80	0.69

Table 1: Performance comparisons in POMDP-pendulum experiment.

pendulum can be observed by the agent. But the non-stationarity caused by the wind cannot be observed. The reward function $r(s,a) = -10\theta^2 - 10^{-1}\dot{\theta}^2 - 10^{-3}a^2$, where θ is again mapped to $[-0.5\pi, 1.5\pi)$. In this experiment, $t_{tr} = 1000, t_{le} = 3000000$ for all methods except HSE-POMDP, and the results are averaged over 100 simulations. For expositional clarity, we report the average reward every 20000 time steps.

The performance comparison of State-Aug, ground-truth, QL, FAQL and RUQL is shown in Figure 2-(1). In this experiment, sophisticated Q-learning methods like RUQL and FAQL do not achieve better convergence time or higher reward due to the complexity of the scenario. By augmenting the observable process with the latent variable, we can see that the results of our framework (red solid curve) are very close to the ground truth (blue dash curve). For HSE-POMDP, $t_{tr} = 1000$, due to the computational costs of repeated inversion of a $t_{tr} \times t_{tr}$ matrix, it would be prohibitive to allow $t_{le} = 3000000$. Instead, we find the value of $t_{le}(20000)$ which allows the algorithm to run in ten times the amount of time needed by State-Aug. The results for HSE-POMDP are significantly worse than all the other methods (the average reward of 20000 iterations is around -50), so we do not show these results in Figure 2-(1).

We also investigate a case where the envelope of the wind also changes, i.e. $v = \sin(t/C)\sin(0.01t/C), C = 100$. Results are shown in Figure 2-(2). For expositional clarity, we report the average reward every 200000 time steps. State-Aug (red solid curve) still achieves rewards very close to the ground truth (blue dash curve). For HSE-POMDP, the average reward of 20000 iterations is approximately -47.

5.4 Antenna Pilot Power Control in Wireless Network

In the final experiment, we consider the optimization of the pilot power of an antenna, an important problem for wireless service providers. The core issue is that setting the pilot power too high or too low eventually affects the total throughput of a cell. Maximizing the throughput by fine tuning the pilot power is challenging due to many factors including user mobility, time-evolving user demands, as well as interference between neighboring cells. Consequently, designing a policy to select the appropriate level of pilot power is by no means an easy task especially in a time-evolving environment. We apply our proposed framework to this application to find a policy for the pilot power configuration. The experiment is conducted using a simulator developed by a well-known network equipment manufacturer for wireless communication. Due to confidentiality, we are not able to disclose the details of the simulator. The state vector that we use in the reinforcement learning algorithms consists

of (1) current pilot power in the cell; (2) the number of users in the cell; (3) the current traffic load in the cell; and (4) the differences of traffic loads between the cell and its neighboring cells. To clarify the comparison for this task, which has very noisy rewards, we show the cumulative rewards instead of instant rewards. Figure 2-(3) demonstrates that State-Aug clearly outperforms the other three methods. As true states are never available, we do not report on HSE-POMDP in this experiment.

6 CONCLUSION AND DISCUSSION

In this paper, we proposed a state augmentation framework for reinforcement learning in environments with exogeneity. The main idea is to infer a latent exogenous vector by RKHS conditional mean embeddings that characterize the evolution of the observable process. Experimental results on both synthetic toy problems as well as real world tasks demonstrate the effectiveness of our state augmentation framework. In the future, this work could be extended in several directions; important directions include determining the dimensionality of the hidden exogenous subspace, and combining the technique with other RL algorithms.

ACKNOWLEDGMENTS

Zhuoshu Li and Sanmay Das are grateful for support from the US National Science Foundation under Award #1527037.

A DETAILED DERIVATION OF EQ. (11)

$\hat{\mathbf{v}}_{t+1} = \boldsymbol{\alpha}_{t+1}\phi(\hat{\mathcal{U}}_{t+1})$.

$$\begin{aligned}\boldsymbol{\alpha}_{t+1} &= \boldsymbol{\alpha}_t + \frac{1}{t+1}(\phi(\hat{\mathcal{U}}_{t+1})\phi(\hat{\mathcal{U}}_{t+1})^T \\ &\quad - \frac{\boldsymbol{\alpha}_t^T\phi(\hat{\mathcal{U}}_{t+1})\phi(\hat{\mathcal{U}}_{t+1})^T\boldsymbol{\alpha}_t}{||\boldsymbol{\alpha}_t||^2}I_d)\boldsymbol{\alpha}_t,\\ \mathbf{K}_U^{-1}\Phi_U^T\Phi_U\tilde{\boldsymbol{\alpha}}_{t+1} &= \mathbf{K}_U^{-1}\Phi_U^T\Phi_U\tilde{\boldsymbol{\alpha}}_t \\ &\quad + \frac{1}{t+1}(\mathbf{K}_U^{-1}\Phi_U^T\phi(\hat{\mathcal{U}}_{t+1})\phi(\hat{\mathcal{U}}_{t+1})^T \\ &\quad - \frac{\mathbf{K}_U^{-1}\Phi_U^T\tilde{\boldsymbol{\alpha}}_t^T\Phi_U^T\phi(\hat{\mathcal{U}}_{t+1})\phi(\hat{\mathcal{U}}_{t+1})^T\Phi_U\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\Phi_U^T\Phi_U\tilde{\boldsymbol{\alpha}}_t}I_d)\Phi_U\tilde{\boldsymbol{\alpha}}_t.\end{aligned}$$

If define $\Phi_U^T\phi(\hat{\mathcal{U}}_{t+1}) = \mathbf{K}_{t+1}$, which is a column vector where $\mathbf{K}_{\phi,i} = \mathbf{K}(U^{(i)}, \hat{\mathcal{U}}_{t+1})$, we have

$$\begin{aligned}I_d\tilde{\boldsymbol{\alpha}}_{t+1} &= I_d\tilde{\boldsymbol{\alpha}}_t + \frac{\mathbf{K}_U^{-1}\mathbf{K}_{t+1}\mathbf{K}_{t+1}^T\tilde{\boldsymbol{\alpha}}_t}{t+1} \\ &\quad - \frac{1}{t+1}\frac{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_{t+1}\mathbf{K}_{t+1}^T\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}\tilde{\boldsymbol{\alpha}}_t}\mathbf{K}_U^{-1}\Phi_U^T I_d\Phi_U\tilde{\boldsymbol{\alpha}}_t,\\ \tilde{\boldsymbol{\alpha}}_{t+1} &= \tilde{\boldsymbol{\alpha}}_t + \frac{\mathbf{K}_U^{-1}\mathbf{K}_{t+1}\mathbf{K}_{t+1}^T\tilde{\boldsymbol{\alpha}}_t}{t+1} - \frac{1}{t+1}\frac{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_{t+1}\mathbf{K}_{t+1}^T\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}\tilde{\boldsymbol{\alpha}}_t}\tilde{\boldsymbol{\alpha}}_t.\end{aligned}$$

By adding a regularization term, we have,

$$\begin{aligned}\tilde{\boldsymbol{\alpha}}_{t+1} &= \tilde{\boldsymbol{\alpha}}_t + \frac{1}{t+1}\mathbf{K}_U^{-1}\mathbf{K}_{t+1}\left(\mathbf{K}_{t+1}^T + \gamma\mathbf{K}_t^T\right)\tilde{\boldsymbol{\alpha}}_t \\ &\quad - \frac{1}{t+1}\frac{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_{t+1}\left(\mathbf{K}_{t+1}^T + \gamma\mathbf{K}_t^T\right)\tilde{\boldsymbol{\alpha}}_t}{\tilde{\boldsymbol{\alpha}}_t^T\mathbf{K}_U\tilde{\boldsymbol{\alpha}}_t}\tilde{\boldsymbol{\alpha}}_t.\end{aligned}$$

Thus,

$$\hat{\mathbf{v}}_{t+1} = \tilde{\boldsymbol{\alpha}}_{t+1}^T\Phi_U^T\phi(\hat{\mathcal{U}}_{t+1}) = \tilde{\boldsymbol{\alpha}}_{t+1}^T\mathbf{K}_{t+1}.$$

REFERENCES

[1] Sherief Abdallah and Michael Kaisers. 2016. Addressing Environment Non-Stationarity by Repeating Q-learning Updates. *Journal of Machine Learning Research* 17, 46 (2016), 1–31. http://jmlr.org/papers/v17/14-037.html

[2] Douglas Aberdeen and Jonathan Baxter. 2002. Scalable Internal-State Policy-Gradient Methods for POMDPs. In *Machine Learning, Proceedings of the Nineteenth International Conference*. 3–10.

[3] Eduardo W. Basso and Paulo M. Engel. 2009. Reinforcement learning in non-stationary continuous time and space scenarios. In *Anais do VII Brazilian Meeting on Artificial Intelligence ENIA*.

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

[5] Zhitang Chen, Pascal Poupart, and Yanhui Geng. 2016. Online Relative Entropy Policy Search using Reproducing Kernel Hilbert Space Embeddings. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*. 573–581.

[6] Zhitang Chen, Kun Zhang, Laiwan Chan, and Bernhard Schölkopf. 2014. Causal Discovery via Reproducing Kernel Hilbert Space Embeddings. *Neural Computation* 26, 7 (2014), 1484–1517.

[7] Samuel P. M. Choi, Dit-Yan Yeung, and Nevin Lianwen Zhang. 1999. An Environment Model for Nonstationary Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 987–993.

[8] Finale Doshi-Velez, David Pfau, Frank Wood, and Nicholas Roy. 2015. Bayesian nonparametric methods for partially-observable reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence* 37, 2 (2015), 394–407.

[9] Kenji Doya, Kazuyuki Samejima, Ken-ichi Katagiri, and Mitsuo Kawato. 2002. Multiple Model-Based Reinforcement Learning. *Neural Computation* 14, 6 (2002), 1347–1369.

[10] Kenji Fukumizu, Francis R. Bach, and Michael I. Jordan. 2004. Dimensionality Reduction for Supervised Learning with Reproducing Kernel Hilbert Spaces. *Journal of Machine Learning Research* 5 (2004), 73–99.

[11] Kenji Fukumizu, Arthur Gretton, Xiaohai Sun, and Bernhard Schölkopf. 2007. Kernel Measures of Conditional Dependence. In *Advances in Neural Information Processing Systems*. 489–496.

[12] Mohammad Ghavamzadeh, Shie Mannor, Joelle Pineau, and Aviv Tamar. 2015. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning* 8, 5-6 (2015), 359–483.

[13] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A Kernel Two-Sample Test. *Journal of Machine Learning Research* 13 (2012), 723–773.

[14] Matthew Hausknecht and Peter Stone. 2015. Deep recurrent q-learning for partially observable MDPs. *arXiv preprint arXiv:1507.06527* (2015).

[15] Pablo Hernandez-Leal, Matthew E. Taylor, Benjamin Rosman, Luis Enrique Sucar, and Enrique Munoz de Cote. 2016. Identifying and Tracking Switching, Non-Stationary Opponents: A Bayesian Approach. In *Multiagent Interaction without Prior Coordination, Papers from the 2016 AAAI Workshop, Phoenix, Arizona, USA, February 13, 2016.*

[16] Tommi Jaakkola, Satinder P Singh, and Michael I Jordan. 1995. Reinforcement learning algorithm for partially observable Markov decision problems. *Advances in neural information processing systems* (1995), 345–352.

[17] Michael R James and Satinder Singh. 2004. Learning and discovery of predictive state representations in dynamical systems with reset. In *Proceedings of the twenty-first international conference on Machine learning*. ACM, 53.

[18] Michael Kaisers and Karl Tuyls. 2010. Frequency Adjusted Multi-agent Q-learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*. 309–316.

[19] TP Krasulina. 1969. The method of stochastic approximation for the determination of the least eigenvalue of a symmetrical matrix. *U. S. S. R. Comput. Math. and Math. Phys.* 9, 6 (1969), 189–195.

[20] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. 2017. Multi-agent Reinforcement Learning in Sequential Social Dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 464–473. http://dl.acm.org/citation.cfm?id=3091125.3091194

[21] Yitao Liang, Marlos C. Machado, Erik Talvitie, and Michael Bowling. 2016. State of the Art Control of Atari Games Using Shallow Reinforcement Learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems (AAMAS '16)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 485–493. http://dl.acm.org/citation.cfm?id=2936924.2936996

[22] Nicolas Meuleau, Leonid Peshkin, Kee-Eung Kim, and Leslie Pack Kaelbling. 1999. Learning finite-state controllers for partially observable environments. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 427–436.

[23] Yu Nishiyama, Abdeslam Boularias, Arthur Gretton, and Kenji Fukumizu. 2012. Hilbert Space Embeddings of POMDPs. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*. 644–653.

[24] Mathijs Pieters and Marco A. Wiering. 2016. Q-learning with Experience Replay in a Dynamic Environment. In *Adaptive Dynamic Programming and Reinrforcement Learning*.

[25] Pascal Poupart and Nikos A. Vlassis. 2008. Model-based Bayesian Reinforcement Learning in Partially Observable Domains. In *International Symposium on Artificial Intelligence and Mathematics*.

[26] Walter Rudin. 1994. *Fourier Analysis on Groups*. John Wiley & Sons.

[27] Bruno Castro da Silva, Eduardo W. Basso, Ana L. C. Bazzan, and Paulo Martins Engel. 2006. Dealing with non-stationary environments using context detection. In *Machine Learning, Proceedings of the Twenty-Third International Conference*. 217–224.

[28] Satinder P. Singh, Michael L. Littman, Nicholas K. Jong, David Pardoe, and Peter Stone. 2003. Learning Predictive State Representations. In *Machine Learning, Proceedings of the Twentieth International Conference*. 712–719.

[29] Le Song, Jonathan Huang, Alexander J. Smola, and Kenji Fukumizu. 2009. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 961–968.

[30] H. Van Hoof, J. Peters, and G. Neumann. 2015. Learning of Non-Parametric Control Policies with High-Dimensional State Features. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, Vol. 38. JMLR, 995âĂŞ1003.

[31] Vivek Veeriah, Harm van Seijen, and Richard S. Sutton. 2017. Forward Actor-Critic for Nonlinear Function Approximation in Reinforcement Learning. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 556–564. http://dl.acm.org/citation.cfm?id=3091125.3091207

[32] Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. 2011. A monte-carlo aixi approximation. *Journal of Artificial Intelligence Research* 40, 1 (2011), 95–142.

[33] Ngo Anh Vien, Peter Englert, and Marc Toussaint. 2016. Policy Search in Reproducing Kernel Hilbert Space. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 2089–2096.

[34] D.V. Voiculescu, K.J. Dykema, and A. Nica. 1992. *Free Random Variables*. American Mathematical Soc.

[35] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. In *Machine Learning*. 279–292.

[36] Max A. Woodbury. 1950. *Inverting modified matrices*. Princeton, NJ : Department of Statistics, Princeton University.

[37] Kun Zhang, Biwei Huang, Bernhard Schölkopf, Michel Besserve, Masataka Watanabe, and Dajiang Zhu. 2015. Towards Robust and Specific Causal Discovery from fMRI. *CoRR* abs/1509.08056 (2015).