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

An Efficient, Expressive and Local Minima-Free Method for Learning Controlled Dynamical Systems

Ahmed Hefny
Carnegie Mellon University

Carlton Downey
Carnegie Mellon University

Geoffrey Gordon
Carnegie Mellon University

Abstract

We propose a framework for modeling and estimating the state of controlled dynamical systems, where an agent can affect the system through actions and receives partial observations. Based on this framework, we propose Predictive State Representation with Random Fourier Features (RFF-PSR). A key property in RFF-PSRs is that the state estimate is represented by a conditional distribution of future observations given future actions. RFFPSRs combine this representation with moment-matching, kernel embedding and local optimization to achieve a method that enjoys several favorable qualities: It can represent controlled environments which can be affected by actions, it has an efficient and theoretically justified learning algorithm, it uses a non-parametric representation that has expressive power to represent continuous non-linear dynamics. We provide a detailed formulation, a theoretical analysis and an experimental evaluation that demonstrates the effectiveness of our method.

1 Introduction

Controlled dynamical systems, where an agent can influence an environment through actions and receive partial observations, emerge in numerous applications in robotics and automatic control. Modeling and learning these systems from data is of great importance in these fields.

The general problem of learning dynamical systems from data (also known as system identification) has been extensively studied and several methods were proposed to tackle it. However, having an expressive, efficient and consistent method for non-linear controlled systems remains an open problem.

Many system identification methods rely on likelihood-based optimization or sampling using EM, MCMC or gradient descent. which makes them prone to poor local optima. There is another class of methods that alleviates the local optima problem and offers a tractable and statistically consistent approach to system identification. These methods, usually referred to as spectral algorithms, have two key properties in common: predictive representation and method of moments. Instead of the state being a latent variable, they represent the estimated state by the expectation of sufficient statistics (or features) of future observations and they use method of moments to learn model parameters from data. [1]

Initially introduced for linear-Gaussian systems (van Overschee and de Moor 1996), these algorithms have been extended to discrete systems (Hsu, Kakade, and Zhang 2009; Siddiqi, Boots, and Gordon 2010; Boots, Siddiqi, and Gordon 2011) and then to general smooth continuous systems (Boots, Gretton, and Gordon 2013). More recently, it has been shown that a wide class of spectral learning algorithms for uncontrolled systems is an instance of a two-stage regression framework (Hefny, Downey, and Gordon 2015), where system identification is reduced to solving a set of regression problems. This framework allows for seamless integration of compressing non-linearities, sparsity (Xia 2016) and online learning (Venkatraman et al. 2016) into system identification, and for establishing theoretical guarantees by leveraging the rich literature on supervised regression.

Unfortunately, the formulation in (Hefny, Downey, and Gordon 2015) is limited to uncontrolled systems. On the contrary, we are interested in controlled systems, where the user can affect the system through actions. This gives rise to a key issue: the policy that determines the actions can change at test time. For this reason, the representation of the predictive state must be independent of the training policy and therefore must encode a *conditional distribution* of future observations given future actions. To adopt such a representation into a practical method that retains the benefits of the two-stage regression formulation, there are a number of challenges that need to be tackled.

First, we need a suitable state representation and dynamics model that can be used to represent a wide class of controlled dynamical systems while ensuring the learning problem remains tractable. Second, we would like to benefit from the two-stage regression view of (Hefny, Downey, and Gordon 2015) to facilitate model formulation. However, a key assumption in that work is that future observations provide an unbiased estimate of the predictive state, which is not true when the state is a conditional distribution. A suitable state estimation method needs to be provided. Third, having a different state representation and having action policy playing a

[1] There is a class of spectral algorithms that maintains the latent variable view. This is exemplified by tensor decomposition methods (Anandkumar et al. 2014).

Method	Actions	Continuous	Non-linear	Partially observable	Scalable	Consistent
Non-linear ARX	✓	✓	✓	×	✓	✓
N4SID for Kalman Filter	✓	✓	×	✓	✓	✓
Non-convex optimization (e.g. EM)	✓	✓	✓	✓	✓	×
Gram-Matrix (e.g. HSE-PSR)	✓	✓	✓	✓	×	✓
Spectral PSR/POMDP	✓	×	✓	✓	✓	✓
Reduction to Supervised Learning	×	✓	✓	✓	✓	✓
RFF-PSR	✓	✓	✓	✓	✓	✓

Table 1: Comparison between proposed RFF-PSR and existing system identification methods in terms of the type of systems they can model as well as their computational efficiency and statistical consistency. The table should be interpreted as follows: for each method there exists an instantiation that *simultaneously* satisfies all properties marked with ✓ but there is no instantiation that is guaranteed to satisfy the properties marked with ×. A method is scalable if computational and memory costs scale at most linearly with the number of training examples. For RFF-based methods, consistency is up to an approximation error that is controllable by the number of features (Rahimi and Recht 2008).

key role on determining the training data require more theoretical analysis than the one in (Hefny, Downey, and Gordon 2015). Forth, because they are based on method of moments, two stage regression models are statistically inefficient. Having the ability to refine the model using local optimization can lead to significant gains in predictive performance.

In this work we address these challenges by combining ideas from two-stage regression, kernel embedding and approximation , and gradient descent with backpropagation through time to develop RFF-PSRs. Overall, RFF-PSRs enjoy a number of advantages that, to our knowledge, are not attained by existing system identification methods. We summarize these advantages in Table 1.

In summary, the contributions of this work are as follows: (1) We develop a two-stage regression framework for controlled dynamical systems that admits tractable learning (Sections 3-4). (2) Through the two-stage regression view, we provide theoretical guarantees on learning the parameters of a controlled system (Section 4.4). (3) We use the extended formulation to construct RFF-PSRs, an efficient approximation of kernel-based predictive state representations (HSE-PSRs) (Section 5). (4) We provide a means to refine the parameters of a controlled dynamical system and apply it to our proposed RFF-PSR model (Section 5.5). (5) We demonstrate the advantages of our proposed method through synthetic and robot simulation experiments (Section 6).

2 Related Work

Developing tractable and consistent algorithms for latent state dynamical systems dates back to spectral subspace identification algorithms for Kalman filters (van Overschee and de Moor 1996). At their heart, these algorithms represent the state as a prediction of the future observations conditioned on history and future actions and use matrix factorization to obtain a basis for the state.

This notion of the state as a prediction is the basis of *predictive state representations* (PSRs) (Singh, James, and Rudary 2004), where the state is represented by the success probabilities of a number of *tests*. A test succeeds if a specified sequence test observations is observed when administering a specified sequence of test actions.

Noting that the state and parameters of a PSR are defined up to a similarity transformation has led to a family of tractable and consistent spectral algorithms for learning PSRs (Rosencrantz and Gordon 2004). More recently, (Boots, Gretton, and Gordon 2013) proposed a generalization of PSRs in a reproducing kernel Hilbert space (RKHS). This Hilbert space embedding of PSRs (HSE-PSRs) is able to represent systems with continuous observations and actions while still offering a tractable and consistent learning algorithm. HSE-PSRs, however, use a Gram matrix formulation, whose computational and storage requirements can grow rapidly with the size of training data. A finite dimensional approximation for non-linear PSRs was proposed by (Boots and Gordon 2011). However, it can be thought of as an approximation of HSE-HMMs (Song et al. 2010) with actions, which has poor theoretical guarantees (Boots, Gretton, and Gordon 2013). In addition, (Boots and Gordon 2011) did not provide examples on how to apply the proposed model to controlled processes with continuous actions. In contrast, the model we propose is an approximation of HSE-PSRs, which is a more principled generalization of PSRs as it performs true Bayesian inference in the RKHS. In addition, our proposed learning algorithm incorporates a local optimization procedure that we demonstrate to be very effective.

We use a reduction of system identification to supervised regression. Similar reductions has been proposed in the literature (Langford, Salakhutdinov, and Zhang 2009; Hefny, Downey, and Gordon 2015; Boots and Gordon 2011; Venkatraman et al. 2016; Sun et al. 2016). These reductions, however, assume uncontrolled systems, where future observation statistics constitute an unbiased representation of the predictive state. [2] Modeling controlled systems is more subtle since the the state of the system is a *conditional distribution* of observations given actions.

Another related work is the spectral learning algorithm for POMDPs proposed by (Azizzadenesheli, Lazaric, and Anandkumar 2016). This method uses tensor factorization

[2] implicit reductions do exist in the system identification literature (van Overschee and de Moor 1996) but they assume linear systems.

to recover POMDP parameters from examples collected by a non-blind memoryless policy. However, this method is limited to discrete POMDPs Also, PSRs have more representational capacity than POMDPs and can compactly represent more sophisticated systems (Singh, James, and Rudary 2004). There are other classes of dynamical system learning algorithms that are based on local optimization or sampling approaches (Fox et al. 2009; Frigola et al. 2013) but they do not offer consistency guarantees.

3 Formulation

We define a class of models that extends predictive state models of (Hefny, Downey, and Gordon 2015) to controlled systems. We first introduce some notation: We denote by $\Pr[x \mid \mathbf{do}(Y = y)]$ the probability of x given that we *intervene* by setting Y to y. This is different from $\Pr[x \mid Y = y]$ which denotes conditioning on *observing* $Y = y$; in the former case, we ignore all effects on Y by other variables. We denote by $V_{A|B;c}$ the linear operator that satisfies

$$\mathbb{E}[A|B = b, C = c] = V_{A|B;c}b \quad \forall b, c$$

In other words for each c, $V_{A|B;c}$ is a conditional expectation operator from B to A (In the discrete case, $V_{A|B;c}$ is just a conditional probability table).

When dealing with multiple variables, we will use tensor notation e.g. $V_{A,B|C,D}$ is a 4-mode tensor. We will use

$$V_{A,B|C,D} \times_C c \times_D d$$

to denote multiplying $V_{A,B|C,D}$ by c along the mode corresponding to C and by d along the mode corresponding to D. If c is a matrix then the multiplication is performed along the first dimension of c.

We will also use $\| \cdot \|_F$ to denote Frobenius norm, $a \otimes b$ to denote Kronecker product of two vectors and $A \star B$ to denote the Khatri-Rao product of two matrices (columnwise Kronecker product).

3.1 Model Definition

We will consider k-observable systems, where the posterior belief state given all previous observations and actions is uniquely identified by the conditional distribution $\Pr[o_{t:t+k-1} \mid \mathbf{do}(a_{t:t+k-1})]$.

Following (Hefny, Downey, and Gordon 2015), we denote by ψ_t^o, ψ_t^a, ξ_t^o and ξ_t^a sufficient features of future observations $o_{t:t+k-1}$, future actions $a_{t:t+k-1}$, extended future observations $o_{t:t+k}$ and extended future actions $a_{t:t+k}$ at time t respectively.

We also use $h_t^\infty \equiv o_{1:t-1}, a_{1:t-1}$ to denote the entire history of observations and actions at time t and use $\psi_t^h \equiv \psi^h(o_{1:t-1}, a_{1:t-1})$ to denote finite features of previous observations and actions before time t[3].

We are now ready to define the class of systems we are interested in.

Definition 1. *A dynamical system is said to conform to a* ***predictive state controlled model (PSCM)*** *if it satisfies the following properties:*

- *For each time t, there exists a linear operator $Q_t = V_{\psi_t^o|\mathbf{do}(\psi_t^a);h_t^\infty}$ (referred to as predictive state) such that $\mathbb{E}[\psi_t^o \mid \mathbf{do}(a_{t:t+k-1}), h_t^\infty] = Q_t\psi_t^a$*
- *For each time t, there exists a linear operator $P_t = V_{\xi_t^o|\mathbf{do}(\xi_t^a);h_t^\infty}$ (referred to as extended state) such that $\mathbb{E}[\xi_t^o \mid \mathbf{do}(a_{t:t+k}), h_t^\infty] = P_t\xi_t^a$*
- *There exists a linear map W_{sys} (referred to as system parameter map), such that, for each time t,*

$$P_t = W_{\text{sys}}(Q_t) \tag{1}$$

- *There exists a filtering function f_{filter} such that, for each time t, $Q_{t+1} = f_{\text{filter}}(P_t, o_t, a_t)$. f_{filter} is typically non-linear but known in advance.*

It follows that a PSCM is specified by the tuple $(Q_0, W_{\text{sys}}, f_{\text{filter}})$, where Q_0 denotes the initial belief state.

There are a number of aspects of PSCMs that warrant discussion. First, unlike latent state models, the state Q_t is represented by a conditional distribution of observed quantities. Second, Q_t is a deterministic function of the history h_t^∞. It represents the *belief* state that one should maintain after observing the history to make optimal predictions. Third, a PSCM specifies a recursive filter where given an action a_t and an observation o_t, the state update equation is given by

$$Q_{t+1} = f_{\text{filter}}(W_{\text{sys}}(Q_t), o_t, a_t) \tag{2}$$

This construction allows us to learn a linear map W_{sys} and use it to build models with non-linear state updates, including IO-HMMs (Bengio and Frasconi 1995), Kalman filters with inputs (van Overschee and de Moor 1996) and HSE-PSRs (Boots, Gretton, and Gordon 2013). As we see in Section 4, avoiding latent variables and having a linear W_{sys} enable the formulation of a consistent learning algorithm.

4 Learning A Predictive State Controlled Model

We assume that the extended features ξ_t^o and ξ_t^a are chosen such that f_{filter} is known. The parameters to learn are thus W_{sys} and Q_0. We also assume that a fixed blind (open-loop) policy is used to collect training data and so, we can treat causal conditioning on action $\mathbf{do}(a_t)$ as ordinary conditioning on a_t. [4] It is possible, however, that a different (possibly non-blind) policy is used at test time.

To learn model parameters, we will adapt the two-stage regression method of (Hefny, Downey, and Gordon 2015). Let $\bar{Q}_t \equiv \mathbb{E}[Q_t \mid \psi_t^h]$ (resp. $\bar{P}_t \equiv \mathbb{E}[P_t \mid \psi_t^h]$) be the expected state (resp. expected extended state) conditioned on finite history features ψ_t^h. For brevity, we might refer to $\bar{Q}_t$ simply as the (predictive) state when the distinction from Q_t is clear. It follows from linearity of expectation that

[3] Often but not always, ψ_t^h is a computed from fixed-size window of previous observations and actions ending at $t-1$.

[4] One way to deal with non-blind training policies is to assign importance weight to training examples to correct the bias resulting from non-blindness (Bowling et al. 2006; Boots, Siddiqi, and Gordon 2011). This, however, requires knowledge of the data collection policy and can result in a high variance of the estimated parameters. We defer the case of unknown non-blind policy to future work.

$\mathbb{E}[\psi_t^o \mid \psi_t^a, \psi_t^h] = \bar{Q}_t \psi_t^a$ and $\mathbb{E}[\xi_t^o \mid \xi_t^a, \psi_t^h] = \bar{P}_t \xi_t^a$; and it follows from the linearity of W_{sys} that

$$\bar{P}_t = W_{\text{sys}}(\bar{Q}_t)$$

So, we train regression models (referred to S1 regression models) to estimate $\bar{Q}_t$ and $\bar{P}_t$ from ψ_t^h. Then, we train another (S2) regression model to estimate W_{sys} from $\bar{Q}_t$ and $\bar{P}_t$. Being conditional distributions, estimating $\bar{Q}_t$ and $\bar{P}_t$ from ψ_t^h is more subtle compared to uncontrolled systems, since we cannot use observation features as estimates of the state. We describe two methods to construct an S1 regression model to estimate $\bar{Q}_t$. The same methods apply to $\bar{P}_t$. As we show below, instances of both methods exist in the literature of system identification.

4.1 Joint S1 Approach

Let ψ_t^{oa} denote a sufficient statistic of the joint observation/action distribution $\Pr(\psi_t^o, \psi_t^a \mid \psi_t^h)$. This distribution is fixed for each value of ψ_t^h since we assume a fixed model and policy. We use an S1 regression model to learn the map $f : \psi_t^h \mapsto \mathbb{E}[\psi_t^{ao} \mid \psi^h]$ by solving the optimization problem

$$\arg\min_{f \in \mathcal{F}} \sum_{t=1}^{T} l(f(\psi_t^h), \psi_t^{oa}) + R(f)$$

for some suitable Bregman divergence loss l (e.g. square loss) and regularization R.

Once we learn f, we can estimate $\bar{Q}_t$ by first estimating the joint distribution $\Pr(\psi_t^o, \psi_t^a \mid \psi_t^h)$ and then deriving the conditional operator $\bar{Q}_t$. By the continuous mapping theorem, a consistent estimator of f results in a consistent estimator of $\bar{Q}_t$. An example of applying this method is using kernel Bayes rule (Fukumizu, Song, and Gretton 2013) to estimate states in HSE-PSR (Boots, Gretton, and Gordon 2013).

4.2 Conditional S1 Approach

In this method, instead of estimating the joint distribution represented by $\mathbb{E}[\psi_t^{oa} \mid \psi_t^h]$, we directly estimate the conditional distribution $\bar{Q}_t$. We exploit the fact that each training example ψ_t^o is an unbiased estimate of $\bar{Q}_t \psi_t^a = \mathbb{E}[\psi_t^o \mid \psi_t^a, \psi_t^h]$. We can formulate the S1 regression problem as learning a function $f : \psi_t^h \mapsto \bar{Q}_t$ that best matches the training examples. i.e. we solve the problem

$$\arg\min_{f \in \mathcal{F}} \sum_{t=1}^{T} l(f(\psi_t^h)\psi_t^a, \psi_t^o) + R(f) \tag{3}$$

for some suitable Bregman divergence loss l (e.g. square loss) and regularization R. An example of applying this method is the oblique projection method used in spectral system identification (van Overschee and de Moor 1996). It is worth emphasizing that both the joint and conditional S1 approaches assume the state to be a *conditional* distribution. They only differ in the way to estimate that distribution.

4.3 S2 Regression and Learning Algorithm

Given S1 regression models to estimate $\bar{Q}_t$ and $\bar{P}_t$, learning a controlled dynamical system proceeds as shown in Algorithm 1.

Algorithm 1 Two-stage regression for predictive state controlled models

Input: $\psi_{n,t}^h, \psi_{n,t}^o, \psi_{n,t}^a, \xi_{n,t}^o, \xi_{n,t}^a$ for $1 \le n \le N, 1 \le t \le T_n$ (N is the number of trajectories, T_n is the length of n^{th} trajectory)

Output: Dynamics matrix $\hat{W}_{\text{sys}}$ and initial state $\hat{Q}_0$

Use S1A regression to estimate $\bar{Q}_{n,t}$.

Use S1B regression to estimate $\bar{P}_{n,t}$.

Let $\hat{W}_{\text{sys}}$ be the (regularized) least squares solution to the system of equations

$$\bar{P}_{n,t} \approx W_{\text{sys}}(\bar{Q}_{n,t}) \quad \forall n, t$$

if N is sufficiently large **then**

  Let $\bar{Q}_0$ be the (regularized) least square solution to the system of equations $\psi_{n,1}^o \approx Q_0 \psi_{n,1}^a \quad \forall n$

else

  Set $\hat{Q}_0$ to the average of $\bar{Q}_{n,t}$

end if

4.4 Theoretical Guarantees

It is worth noting that Algorithm 1 is still an instance of the two stage regression framework described in (Hefny, Downey, and Gordon 2015) and hence retains its theoretical guarantees: mainly that we can bound the error in estimating the dynamics matrix W_{sys} in terms of S1 regression error bounds, assuming that we collect examples from the stationary distribution of a blind policy with sufficient exploration.

A blind policy provides sufficient exploration if it has a stationary distribution that (1) visits a sufficient history set such that the set of equations $\mathbb{E}[P_t|\psi_t^h] = W_{sys}(\mathbb{E}[Q_t|\psi_t^h])$ are sufficient for estimating W_{sys} ,and (2) provides training data to estimate $\mathbb{E}[Q_t|\psi_t^h]$ and $\mathbb{E}[P_t|\psi_t^h]$ with increasing accuracy.

Theorem 1. *Let π be a blind data collection policy with a stationary distribution. If history, action and observation features are bounded, π provides sufficient exploration , and ridge regression is used with λ_1 and λ_2 regularization parameter for S1 and S2 regression respectively, then for all valid states Q the following is satisfied with probability at least $1 - \delta$.*

$$\begin{aligned}
&\|(\hat{W}_{\text{sys}} - W_{\text{sys}})(Q)\| \le \\
&O\left(\eta_{\delta,N}\left((1/\lambda_2) + (1/\lambda_2^{\frac{3}{2}})\sqrt{1 + \sqrt{\frac{\log(1/\delta)}{N}}}\right)\right) \\
&+ O\left(\frac{\log(1/\delta)}{\sqrt{N}}\left(\frac{1}{\lambda_2} + \frac{1}{\lambda_2^{\frac{3}{2}}}\right)\right) + O\left(\sqrt{\lambda_2}\right),
\end{aligned}$$

where

$$\eta_{\delta,N} = O_p\left(\frac{1/\sqrt{N} + \lambda_1}{c + \lambda_1}\right),$$

where $c > 0$ is a problem-dependent constant.

We omit the proof due to space constraints.

5 Predictive State Controlled Models With Random Fourier Features

Having a general framework for learning controlled dynamical systems, we now focus on HSE-PSR (Boots, Gretton, and Gordon 2013) as a non-parametric instance of that framework using Hilbert space embedding of distributions. We first describe HSE-PSR learning as a two-stage regression method. Then we demonstrate how to obtain a finite dimensional approximation using random Fourier features (RFF) (Rahimi and Recht 2008). Before describing HSE-PSR we give some necessary background on Hilbert space embedding and random Fourier features.

5.1 Hilbert Space Embedding of Distributions

We will briefly describe the concept of Hilbert space embedding of distributions. We refer the reader to (Smola et al. 2007) for more details on this topic. Hilbert space embedding of distributions provide a non-parametric generalizations of marginal, joint and conditional probability tables of discrete variables to continuous domains: namely, mean maps, covariance operators and conditional operators.

Let k be a kernel associated with a feature map $\phi(x)$ such that $k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$. A special case for discrete variables is the delta kernel where $\phi(x)$ maps x to an indicator vector. For a random variable X, the *mean map* μ_X is defined as $\mathbb{E}[\phi_{\mathcal{X}}(X)]$. Note that μ_X is an element of the reproducing kernel Hilbert space (RKHS) associated with k.

The uncentered covariance operator of two variables X and Y is $C_{XY} = \mathbb{E}[\phi_{\mathcal{X}}(X) \otimes \phi_{\mathcal{Y}}(Y)]$. For universal kernels $k_{\mathcal{X}}$ and $k_{\mathcal{Y}}$, C_{XY} is a sufficient representation of the joint distribution $\Pr(X, Y)$. In this paper, we will use $C_{XY|z}$ to denote the covariance of X and Y given that $Z = z$.

Under smoothness assumptions, (Song et al. 2009) show that $V_{\phi_{\mathcal{X}}(X)|\phi_{\mathcal{Y}}(Y)} = C_{XY} C_{XX}^{-1}$, where the conditional operator V is as defined in Section 3. More generally, $V_{\phi_{\mathcal{X}}(X)|\phi_{\mathcal{Y}}(Y);z} = C_{XY|z} C_{XX|z}^{-1}$.

5.2 HSE-PSR as a predictive state controlled model

HSE-PSR is a generalization of IO-HMM that has proven to be successful in practice (Boots, Gretton, and Gordon 2013; Boots and Fox 2013). It is suitable for high dimensional and continuous observations and/or actions. HSE-PSR uses kernel feature maps as sufficient statistics of observations and actions. We define four kernels k_O, k_A, k_o, k_a over future observation features, future action features, individual observations and individual actions respectively.

We can then define $\psi_t^o = \phi_O(o_{t:t+k-1})$ and similarly $\psi_t^a = \phi_A(a_{t:t+k-1})$. We will also use ϕ_t^o and ϕ_t^a as shorthands for $\phi_o(o_t)$ and $\phi_a(a_t)$. The extended future is then defined as $\xi_t^o = \psi_t^o \otimes \phi_t^o$ and $\xi_t^a = \psi_t^a \otimes \phi_t^a$

Under the assumption of a blind learning policy, the operators Q_t and P_t are defined to be

$$Q_t = V_{\psi_t^o|\psi_t^a;h_t^\infty} \tag{4}$$

$$P_t = (P_t^\xi, P_t^o) = (V_{\psi_{t+1}^o \otimes \phi_t^o|\psi_{t+1}^a \otimes \phi_t^a;h_t^\infty}, V_{\phi_t^o \otimes \phi_t^o|\phi_t^a;h_t^\infty}) \tag{5}$$

Therefore, Q_t specifies the state of the system as a conditional distribution of future observations given future actions while P_t is a tuple of two operators that allow us to condition on the pair (a_t, o_t) to obtain Q_{t+1}. In more detail, filtering in an HSE-PSR is carried out as follows

- From o_t and a_t, obtain ϕ_t^o and ϕ_t^a.
- Compute $C_{o_t o_t|h_t^\infty,a_t} = V_{\phi_t^o \otimes \phi_t^o|\phi_t^a;h_t^\infty} \phi_t^a$
- Multiply by inverse observation covariance to change "predicting ϕ_t^o" into "conditioning on ϕ_t^o":

$$V_{\psi_{t+1}^o|\psi_{t+1}^a,\phi_t^o,\phi_t^a;h_t^\infty} = V_{\psi_{t+1}^o \otimes \phi_t^o|\psi_{t+1}^a,\phi_t^a;h_t^\infty} \times_{\phi_t^o} (C_{o_t o_t|h_t^\infty,a_t} + \lambda I)^{-1}$$

- Condition on ϕ_t^o and ϕ_t^a to obtain shifted state

$$Q_{t+1} \equiv V_{\psi_{t+1}^o|\psi_{t+1}^a;\phi_t^o,\phi_t^a,h_t^\infty} = V_{\psi_{t+1}^o|\psi_{t+1}^a,\phi_t^o,\phi_t^a;h_t^\infty} \times_{\phi_t^o} \phi_t^o \times_{\phi_t^a} \phi_t^a$$

Thus, in HSE-PSR, the parameter W_{sys} is composed of two linear maps; f_o and f_ξ such that $P_t^\xi = f_\xi(Q_t)$ and $P_t^o = f_o(Q_t)$. In the following section we show how to estimate $\bar{Q}_t$ and $\bar{P}_t$ from data. Estimation of f_ξ, f_o can then be carried out using kernel regression.

Learning and filtering in an HSE-PSR can be implicitly carried out in RKHS using a Gram matrix formulation. We will describe learning in terms of RKHS elements and refer the reader to (Boots, Gretton, and Gordon 2013) for details on the Gram matrix formulation.

5.3 S1 Regression for HSE-PSR

As discussed in section 4 we can use a joint or conditional approach for S1 regression. We now demonstrate how these two approaches apply to HSE-PSR.

Joint S1 Regression for HSE-PSR This is the method used in (Boots, Gretton, and Gordon 2013). In this approach we exploit the fact that

$$\bar{Q}_t = W_{\psi_t^o|\psi_t^a;\psi_t^h} = C_{\psi_t^o \psi_t^a|\psi_t^h} (C_{\psi_t^a \psi_t^a|\psi_t^h} + \lambda I)^{-1}$$

So, we learn two linear maps T_{oa} and T_a such that $T_{oa}(\psi_t^h) \approx C_{\psi_t^o \psi_t^a|\psi_t^h}$ and $T_a(\psi_t^h) \approx C_{\psi_t^a \psi_t^a|\psi_t^h}$. The training examples for T_{oa} and T_a consist of pairs $(\psi_t^h, \psi_t^o \otimes \psi_t^a)$ and $(\psi_t^h, \psi_t^a \otimes \psi_t^a)$ respectively.

Once we learn this map, we can estimate $C_{\psi_t^o \psi_t^a|\psi_t^h}$ and $C_{\psi_t^a \psi_t^a|\psi_t^h}$ and consequently estimate $\bar{Q}_t$.

Conditional S1 Regression for HSE-PSR It is also possible to apply the conditional S1 regression formulation in Section 4.2. Specifically, let $\mathcal{F}$ be the set of 3-mode tensors, with modes corresponding to ψ_t^o, ψ_t^o and ψ_t^h. We estimate a tensor T^* by optimizing

$$T^* = \arg\min_{T \in \mathcal{F}} \|(T \times_{\psi_t^h} \psi_t^h \times_{\psi_a^t} \psi_a^t) - \psi_o^t\|^2 + \lambda \|T\|_{HS}^2,$$

where $\|.\|_{HS}^2$ is the Hilbert-Schmidt norm, which translates to Frobenius norm in finite-dimensional Euclidan spaces. We can then use

$$\bar{Q}_t = T^* \times_{\psi_t^h} \psi_t^h$$

For both regression approaches, the same procedure can be used to estimate the extended state $\bar{P}_t$ by replacing features ψ_t^o and ψ_t^a with their extended counterparts ξ_t^o and ξ_t^a.

5.4 Approximating HSE-PSR with Random Fourier Features

A Gram matrix formulation of the HSE-PSR has computational and memory requirements that grow rapidly with the number of training examples. To alleviate this problem, we resort to kernel approximation–that is, we replace RKHS vectors such as ψ_t^o and ψ_t^a with finite dimensional vectors that approximately preserve inner products. We use random Fourier features (RFF) (Rahimi and Recht 2008) as an approximation but it is possible to use other approximation methods. Unfortunately RFF approximation can typically require D to be prohibitively large. Therefore, we apply principal component analysis (PCA) to the feature maps to reduce their dimension to $p \ll D$. We apply PCA again to quantities that require p^2 space such as extended features ξ_t^o, ξ_t^a and states $\bar{Q}_t$, reducing them to p-dimensions. We map them back to p^2 dimensions when needed (e.g. for filtering). We also employ randomized SVD (Halko, Martinsson, and Tropp 2011) for fast computation of PCA, resulting in an algorithm that scales linearly with N and D.

5.5 Model refinement by local optimization

A common practice is to use the output of a moment-based algorithm to initialize a non-convex optimization algorithm such as EM (Belanger and Kakade 2015) or gradient descent (Jiang, Kulesza, and Singh 2016). Since EM is not directly applicable to RFF-PSR, we propose a gradient descent approach. We can observe that filtering in an RFF-PSR defines a recurrent structure given by.

$$q_{t+1} = f_{\text{filter}}(W_{\text{sys}} q_t, o_t, a_t),$$
$$\mathbb{E}[o_t | q_t] = W_{\text{pred}}(q_t \otimes \phi(a_t)),$$

where W_{pred} is a linear operator that predicts the next observation [5]. If f_{filter} is differentiable, we can improve our estimates of W_{sys} and W_{pred} using backpropagation through time (BPTT) (Werbos 1990). It is possible to optimize the error in predicting (features of) a window of observations. In our experiments, we learn to predict $o_{t:t+k-1}$ given $a_{t:t+k-1}$.

6 Experiments

6.1 Synthetic Data

We use the benchmark synthetic non-linear system used by (Boots, Gretton, and Gordon 2013) :

$$\dot{x}_1(t) = x_2(t) - 0.1\cos(x_1(t))(5x_1(t) - 4x_1^3(t) + x_1^5(t)) - 0.5\cos(x_1(t))a(t)$$
$$\dot{x}_2(t) = -65x_1(t) + 50x_1^3(t) - 15x_1^5(t) - x_2(t) - 100a(t)$$
$$o(t) = x_1(t)$$

The input a is generated as zero-order hold white noise, uniformly distributed between -0.5 and 0.5. We collected 20 trajectories of 100 observations and actions at 20Hz and we split them into 10 training, 5 validation and 5 test trajectories. The prediction target for this experiment is $o(t)$.

[4]Code is available at: https://github.com/ahefnycmu/rffpsr

[5]The linearity of W_{pred} is a valid assumption for a universal kernel.

6.2 Predicting windshield view

In this experiment we used TORCS car simulation server, which outputs 64x64 images (see Figure 2). The observations are produced by converting the images to greyscale and projecting them to 200 dimensions via PCA. The car is controlled by a built-in controller that controls acceleration while the external actions control steering. We collected 50 trajectories by applying a sine wave with random starting phase to the steering control and letting the simulator run until the car gets off the track. We used 40 trajectories for training, 5 for validation and 5 for testing. The prediction target is the projected image.

6.3 Predicting the nose position of a simulated swimmer robot

We consider the 3-link simulated swimmer robot from the open-source package RLPy (Geramifard et al. 2013). The 2-d action consists of torques applied on the two joints of the links. The observation model returns the angles of the joints and the position of the nose (in body coordinates). The measurements are contaminated with Gaussian noise whose standard deviation is 5% of the true signal standard deviation. To collect the data, we use an open-loop policy that selects actions uniformly at random. We collected 25 trajectories of length 100 each and use 24 for training and 1 for validation. We generate test trajectories using a mixed policy: with probability p_{blind}, we sample a uniformly random action, while with probability $1 - p_{\text{blind}}$, we sample an action from a pre-specified deterministic policy that seeks a goal point. We generate two sets of 10 test trajectories each, one with $p_{\text{blind}} = 0.8$ and another with $p_{\text{blind}} = 0.2$. The prediction target is the position of the nose.

6.4 Tested Methods and Evaluation Procedure

We tested three different initializations of RFF-PSR (with RBF kernel): random initialization, two-stage regression with joint S1, and two-stage regression with conditional S1 (Section 5.3). For each initialization, we tested the model before and after refinement. For refinement we used BPTT with a decreasing step size: the step size is reduced by half if validation error increases. Early stopping occurs if the step size becomes too small (10^{-5}) or the relative change in validation is insignificant (10^{-3}). We also test the following baselines.

HSE-PSR: We implemented the Gram matrix HSE-PSR as described in (Boots, Gretton, and Gordon 2013).

N4SID: We used MATLAB's implementation of subspace identification of linear dynamical systems.

Non-linear Auto Regression (RFF-ARX): We implemented a version of auto regression where the predictor variable is the RFF representation of future actions together with a finite history of previous observations and actions, and the target variable is future observations.

Models were trained with future length of 10 and history length of 20. For RFF-PSR and RFF-ARX we used 10000 RFF features and applied PCA to project features onto 20 dimensions. Kernel bandwidths were set to the median of the

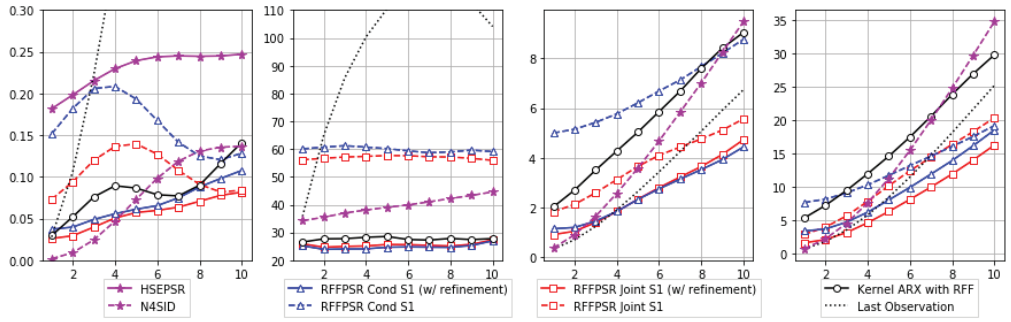

Figure 1: Mean square error for 10-step prediction on (from left to right) synthetic model, TORCS car simulator, swimming robot simulation with 80% blind test-policy, and swimming robot with 20% blind test policy. Baselines with very high MSE are not shown for clarity. A comparison with HSE-PSR on TORCS and swimmer datasets was not possible as it required prohibitively large memory.

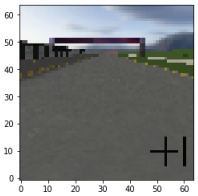

Figure 2: An example of windshield view output by TORCS.

distance between training points (median trick). For evaluation, we perform filtering on the data and estimate the prediction target of the experiment at test time t given the history $o_{1:t-H}, a_{1:t}$, where H is the prediction horizon. We report the mean square error across all times t for each value of $H \in \{1, 2, \ldots, 10\}$.

6.5 Results and Discussion

The results are shown in Figure 1. [6] There are a number of important observations.

- In general, joint S1 training closely matches or outperforms conditional S1 training, with and without refinement.
- Local refinement significantly improves predictive performance for all initialization methods.
- Local refinement, on its own, is not sufficient to produce a good model. The two stage regression provides a good initialization of the refinement procedure.
- Even without refinement, RFF-PSR outperforms HSE-PSR. This could be attributed to the dimensionality reduction step, which adds appropriate inductive bias.
- Compared to other methods, RFF-PSR has better performance with non-blind test policies.

[6] We omit the results for randomly initialized RFF-PSR as they were significantly worse. A comparison with HSE-PSR on the swimmer dataset was not possible as it required prohibitively large memory.

7 Conclusion

We proposed a framework to learn controlled dynamical systems using two-stage regression. We then applied this framework to develop a scalable method for controlled non-linear system identification: using RFF approximation of HSE-PSR together with a refinement procedure to enhance the model after a two-stage regression initialization. We have demonstrated promising results for the proposed method in terms of predictive performance. As future work, we would like to use this framework for further tasks such as imitation learning and reinforcement learning.

Acknowledgements The authors gratefully acknowledge support from ONR (grant number N000141512365) and DARPA (grant number FA87501720152). The authors would like to thank Wen Sun and Yuxiang Wang for the helpful discussions.

References

Anandkumar, A.; Ge, R.; Hsu, D.; Kakade, S. M.; and Telgarsky, M. 2014. Tensor decompositions for learning latent variable models. *J. Mach. Learn. Res.*

Azizzadenesheli, K.; Lazaric, A.; and Anandkumar, A. 2016. Reinforcement learning of pomdp's using spectral methods. *CoRR* abs/1602.07764.

Belanger, D., and Kakade, S. M. 2015. A linear dynamical system model for text. In *ICML*.

Bengio, Y., and Frasconi, P. 1995. An input output HMM architecture. In *NIPS*.

Boots, B., and Fox, D. 2013. Learning dynamic policies from demonstration. *NIPS Workshop on Advances in Machine Learning for Sensorimotor Control*.

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

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

Boots, B.; Siddiqi, S.; and Gordon, G. 2011. Closing the learning planning loop with predictive state representations. In *I. J. Robotic Research*, volume 30, 954–956.

Bowling, M.; McCracken, P.; James, M.; Neufeld, J.; and Wilkinson, D. 2006. Learning predictive state representations using non-blind policies. In *ICML*.

Fox, E.; Sudderth, E. B.; Jordan, M. I.; and Willsky, A. S. 2009. Nonparametric bayesian learning of switching linear dynamical systems. In *NIPS*.

Frigola, R.; Lindsten, F.; Schön, T. B.; and Rasmussen, C. E. 2013. Bayesian inference and learning in gaussian process state-space models with particle mcmc. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 3156–3164.

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

Geramifard, A.; Klein, R. H.; Dann, C.; Dabney, W.; and How, J. P. 2013. RLPy: The Reinforcement Learning Library for Education and Research. http://acl.mit.edu/RLPy.

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

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

Hsu, D.; Kakade, S. M.; and Zhang, T. 2009. A spectral algorithm for learning hidden markov models. In *COLT*.

Jiang, N.; Kulesza, A.; and Singh, S. P. 2016. Improving predictive state representations via gradient descent. In *AAAI*.

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

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

Rosencrantz, M., and Gordon, G. 2004. Learning low dimensional predictive representations. In *ICML*, 695–702.

Siddiqi, S.; Boots, B.; and Gordon, G. J. 2010. Reduced-rank hidden Markov models. In *AISTATS*.

Singh, S.; James, M. R.; and Rudary, M. R. 2004. Predictive state representations: A new theory for modeling dynamical systems. In *UAI*.

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

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

Song, L.; Boots, B.; Siddiqi, S. M.; Gordon, G. J.; and Smola, A. J. 2010. Hilbert space embeddings of hidden Markov models. In *ICML*.

Sun, W.; Venkatraman, A.; Boots, B.; and Bagnell, J. A. 2016. Learning to filter with predictive state inference machines. In *ICML*.

van Overschee, P., and de Moor, L. 1996. *Subspace identification for linear systems: theory, implementation, applications*. Kluwer Academic Publishers.

Venkatraman, A.; Sun, W.; Hebert, M.; Bagnell, J. A.; and Boots, B. 2016. Online instrumental variable regression with applications to online linear system identification. In *AAAI*.

Werbos, P. J. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*.

Xia, G. G. 2016. *Expressive Collaborative Music Performance via Machine Learning*. Ph.D. Dissertation, Carnegie Mellon University.