Hilbert Space Embeddings of Conditional Distributions with Applications to Dynamical Systems

Le Song LESONG@CS.CMU.EDU
Jonathan Huang JCH1@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University

Alex Smola ALEX@SMOLA.ORG
Yahoo! Research, Santa Clara, CA, USA

Kenji Fukumizu FUKUMIZU@ISM.AC.JP
Institute of Statistical Mathematics, Japan

Abstract

In this paper, we extend the Hilbert space embedding approach to handle *conditional* distributions. We derive a kernel estimate for the conditional embedding, and show its connection to ordinary embeddings. Conditional embeddings largely extend our ability to manipulate distributions in Hibert spaces, and as an example, we derive a nonparametric method for modeling dynamical systems where the belief state of the system is maintained as a conditional embedding. Our method is very general in terms of both the domains and the types of distributions that it can handle, and we demonstrate the effectiveness of our method in various dynamical systems. We expect that conditional embeddings will have wider applications beyond modeling dynamical systems.

1. Introduction

Recent advances in kernel methods have yielded a method (Smola et al., 2007) for dealing with probability distributions by representing them as points in a suitable reproducing kernel Hilbert space (RKHS). This method of RKHS embeddings has proven itself to be a powerful and flexible tool for dealing with high-order statistics of random variables and has found wide application, ranging from two-sample tests (Gretton et al., 2007) to dimensionality reduction (Song et al., 2008).

In this paper, we introduce the idea of embedding conditional distributions into a Hilbert space. Hilbert space embeddings of conditional distributions are potentially useful in applications where conditional distributions are the key quantities of interest, such as regressing structured response variables. In particular, we use the embeddings obtained in this paper to design a novel nonparametric approach for maintaining the belief state in a dynamical system. By not committing to fixed parameterizations of the transition and observation models, our nonparametric method handles a wider variety of problems than most existing solutions. Moreover, while typical inference algorithms for dynamical systems are formulated for a particular domain (such as $\mathbb{R}^n$), our method can be applied to any domain admitting an appropriate kernel function and can thus be applied to not only Euclidean spaces, but more exotic spaces (both continuous and discrete) such as the rotation matrices, permutations, strings, graphs, etc.

In the following, we summarize some of the main contributions of this paper.

1. We introduce the concept of embedding conditional distributions in an RKHS and present a novel method for estimating such embeddings from training data.
2. We consider several useful probabilistic inference operations such as marginalization and conditioning, and show, using our theory, that these operations can be performed solely in the RKHS.
3. We apply our inference algorithms to learn nonparametric models and perform inference for dynamical systems. Our algorithms are

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009. Copyright 2009 by the author(s)/owner(s).

Table 1. Table of Notation

random variable	X	Y	Z
domain	$\mathcal{X}$	$\mathcal{Y}$	$\mathcal{Z}$
observation	x	y	z
kernel	$k(x, x')$	$l(y, y')$	$u(z, z')$
kernel matrix	K	L	U
feature map	$\varphi(x), k(x, \cdot)$	$\phi(y), l(y, \cdot)$	$\psi(z), u(z, \cdot)$
feature matrix	Υ	Φ	Ψ
RKHS	$\mathcal{F}$	$\mathcal{G}$	$\mathcal{H}$
dynamic system	s^t	s^{t+1}	o^{t+1}

general first because they handle a wide variety of possible nonlinear/nongaussian models and second because they apply in any setting in which an appropriate kernel function can be defined.

2. Hilbert Space Embedding

We begin by providing an overview of Hilbert space embeddings in which one represents probability distributions by elements in a Hilbert space. In our setting of dynamical systems, we will eventually think of representing the belief state at each timestep as a point in an Hilbert space. In the following we denote by X, a random variable with domain $\mathcal{X}$ and refer to instantiations of X by the lower case character, x. We endow $\mathcal{X}$ with some σ-algebra $\mathscr{A}$ and denote the space of all probability distributions (with respect to $\mathscr{A}$) on $\mathcal{X}$ by $\mathcal{P}$. Finally, we will consider a single distribution $P(X)$ on $\mathcal{X}$.

A *reproducing kernel Hilbert space (RKHS)* $\mathcal{F}$ on $\mathcal{X}$ with kernel k is a Hilbert space of functions $f : \mathcal{X} \mapsto \mathbb{R}$. Its dot product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$, satisfies the reproducing property:

$$\langle f(\cdot), k(x, \cdot) \rangle_{\mathcal{F}} = f(x), \text{ and consequently,} \quad (1a)$$

$$\langle k(x, \cdot), k(x', \cdot) \rangle_{\mathcal{F}} = k(x, x'), \quad (1b)$$

meaning that we can view the evaluation of a function f at any point $x \in \mathcal{X}$ as an inner product, and the linear evaluation operator is given by $k(x, \cdot)$, *i.e.* the kernel function. Alternatively, $k(x, \cdot)$ can also be viewed as a feature map $\varphi(x)$ where $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{F}}$. Popular kernel functions on $\mathbb{R}^n$ include the polynomial kernel $k(x, x') = \langle x, x' \rangle^d$ and the Gaussian RBF kernel $k(x, x') = \exp(-\lambda \|x - x'\|^2)$. Good kernel functions have also been defined on graphs, time series, dynamical systems, images, and structured objects (Schölkopf et al., 2004). Analogously, the above definitions and notational convention also apply to random variables Y and Z (See Table 1 for a summary).

Embedding distribution. At the heart of the approach lie two simple embeddings — the expected feature map and its empirical estimate:

$$\text{(a) } \mu_X := \mathbb{E}_X[\varphi(X)], \quad \text{(b) } \hat{\mu}_X := \frac{1}{m} \sum_{i=1}^{m} \varphi(x_i), \quad (2)$$

where $\mathcal{D}_X = \{x_1, \ldots, x_m\}$ is a training set assumed to have been drawn *i.i.d.* from $P(X)$. If the condition $\mathbb{E}_X[k(X, X)] < \infty$ is satisfied, then the mapping μ_X is guaranteed to be an element of the RKHS. By virtue of the reproducing property of $\mathcal{F}$, both mappings satisfy $\langle \mu_X, f \rangle_{\mathcal{F}} = \mathbb{E}_X[f(X)]$ and $\langle \hat{\mu}_X, f \rangle_{\mathcal{F}} = \frac{1}{m} \sum_{i=1}^{m} f(x_i)$, thus showing that we can compute expectations and empirical means with respect to $P(X)$ and $\mathcal{D}_X$ respectively by taking inner products with the embeddings μ_X and $\hat{\mu}_X$. We will hence also refer to the two embeddings as *mean maps*. Mean maps are attractive for several reasons (Smola et al., 2007; Sriperumbudur et al., 2008). First, it turns out that for certain choices of kernel functions, we can guarantee that distinct distributions map to distinct points in an RKHS. We say that such a kernel function is *characteristic*. More precisely, we have:

Definition 1 *When the mean map $\mu_X : \mathcal{P} \to \mathcal{F}$ is injective, the kernel function k is called* characteristic.

Secondly, while we rarely have access to the true underlying distribution, a finite sample of size m from a distribution P suffices to (with high probability) compute an approximation within an error of $O_p(m^{-\frac{1}{2}})$:

Theorem 2 *The empirical mean $\hat{\mu}_X$ converges to μ_X in the RKHS norm at a rate of $O_p(m^{-\frac{1}{2}})$.*

Cross-covariance operator. To incorporate the transition and observation models that arise in Markovian dynamics, we will present in the next section, as one of our main contributions, a method for embedding conditional distributions. Our technique relies on a generalization of the covariance matrix known as the *cross-covariance operator* $\mathcal{C}_{XY} : \mathcal{G} \to \mathcal{F}$, which is defined as (Baker, 1973):

$$\mathcal{C}_{XY} = \mathbb{E}_{XY}[\varphi(X) \otimes \phi(Y)] - \mu_X \otimes \mu_Y, \quad (3)$$

where $\otimes$ is the tensor product. Alternatively, $\mathcal{C}_{XY}$ can also be viewed as an element in the tensor product space $\mathcal{G} \otimes \mathcal{F}$. Given two functions, $f \in \mathcal{F}$ and $g \in \mathcal{G}$, their cross-covariance, $Cov_{XY}[f(X), g(Y)] := \mathbb{E}_{XY}[f(X)g(Y)] - \mathbb{E}_X[f(X)]\mathbb{E}_Y[g(Y)]$, is computed as:

$$\langle f, \mathcal{C}_{XY} g \rangle_{\mathcal{F}} \text{ or equivalently } \langle f \otimes g, \mathcal{C}_{XY} \rangle_{\mathcal{F} \otimes \mathcal{G}}. \quad (4)$$

Given m pairs of training examples $\mathcal{D}_{XY} = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ drawn *i.i.d.* from $P(X, Y)$, we denote by $\Upsilon = (\varphi(x_1), \ldots, \varphi(x_m))$ and $\Phi = (\phi(y_1), \ldots, \phi(y_m))$ the feature matrices. The covariance operator $\mathcal{C}_{XY}$ can then be estimated as

$\hat{\mathcal{C}}_{XY} = \frac{1}{m}(\Upsilon - \hat{\mu}_X \mathbf{1}^\top)(\Phi - \hat{\mu}_Y \mathbf{1}^\top)^\top = \frac{1}{m}\Upsilon H \Phi^\top$ where H is an idempotent *centering matrix* defined by $H = I - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$.

3. Embedding Conditional Distributions

In this section, we discuss our first main contribution: embedding conditional distributions of the form $P(Y|X)$ into an RKHS. Unlike the embeddings discussed in the previous section, the conditional embeddings which we will define shortly will not be single elements in the RKHS, but will instead sweep out a family of points in the RKHS, each one indexed by a fixed value of the conditioning variable X. It is only by fixing X to a particular value x, that we will be able to obtain a single RKHS element, $\mu_{Y|x} \in \mathcal{G}$. In other words, $\mathcal{U}_{Y|X}$ will be viewed as an operator mapping from $\mathcal{F}$ to $\mathcal{G}$. In order to define the conditional embedding, $\mathcal{U}_{Y|X}$, we will want it to satisfy two properties:

1. $\mu_{Y|x} := \mathbb{E}_{Y|x}[\phi(Y)|x] = \mathcal{U}_{Y|X} k(x, \cdot)$, and
2. $\mathbb{E}_{Y|x}[g(Y)|x] = \langle g, \mu_{Y|x} \rangle_{\mathcal{G}}$.

Assume now that for all $g \in \mathcal{G}$, the conditional expectation $\mathbb{E}_{Y|X}[g(Y)|X]$ is an element of $\mathcal{F}$. We now show that the conditional embedding can be defined in terms of cross-covariance operators using a relation provided by Fukumizu et al. (2004),

$$\mathcal{C}_{XX} \mathbb{E}_{Y|X}[g(Y)|X] = \mathcal{C}_{XY} g. \tag{5}$$

Definition 3 *The operator $\mathcal{U}_{Y|X}$ is defined as: $\mathcal{U}_{Y|X} := \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}$.*

We can now show that that under our definition, $\mathcal{U}_{Y|X}$ satisfies the properties that we wanted.

Theorem 4 *Assuming that $\mathbb{E}_{Y|X}[g(Y)|X] \in \mathcal{F}$, the embedding of conditional distributions in Definition 3 satisfies properties 1 and 2.*

Proof By virtue of the reproducing property, given an x, we have $\mathbb{E}_{Y|x}[g(Y)|x] = \langle \mathbb{E}_{Y|X}[g(Y)|X], k(x, \cdot) \rangle_{\mathcal{F}}$. Using relation (5) and taking the conjugate transpose of $\mathcal{C}_{XX}^{-1}\mathcal{C}_{XY}$, we have $\mathbb{E}_{Y|x}[g(Y)|x] = \langle g, \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1} k(x, \cdot) \rangle_{\mathcal{G}}$ which proves the theorem. ■

We remark that while the assumption $\mathbb{E}_{Y|X}[g(Y)|X] \in \mathcal{F}$ always holds for finite domains with characteristic kernels, it is not necessarily true for continuous domains. In the cases where the assumption does not hold, We use the expression $\mathcal{C}_{YX}\mathcal{C}_{XX}^{-1} k(x, \cdot)$ as an approximation of the conditional mean $\mu_{Y|x}$ and propose a kernel estimate based on the expression for practical use. Given a dataset $\mathcal{D}_{XY}$ of size m, the conditional embedding $\hat{\mu}_{Y|x}$ can be estimated using the following theorem (proof omitted).

Theorem 5 *Let $k_x := \Upsilon^\top \varphi(x)$. Then $\hat{\mu}_{Y|x}$ can be estimated as: $\hat{\mu}_{Y|x} = \Phi(HK + \lambda m I)^{-1} H k_x = \sum_{i=1}^m \beta_i(x)\phi(y_i)$, where each β_i is a real-valued weight.*

Theorem 5 shows that the empirical estimator of the conditional embedding bears a remarkable similarity to the estimator of the ordinary embedding from Equation (2). The difference is that, instead of applying uniform weights $\frac{1}{m}$, the former applies *non-uniform* weights, β_i, on observations which are, in turn, determined by the conditioning variable. These non-uniform weights reflect the effects of conditioning on Hilbert space embeddings.

A special case of this conditional embedding was employed in Quadrianto et al. (2008) for discrete conditioning variables x (multiclass labels), where a delta kernel is applied on x and the space of observations (y) is partitioned according to the value of x. In this case, the conditional embedding is given by a $\beta_i(x) = \frac{1}{n}$ if $x_i = x$ otherwise $\beta_i(x) = \frac{1}{n-m}$, where $n \leq m$ is the total number of x_i with label x.[1] Our new definition of conditional embedding does not require an explicit partitioning of the observation space and is thus applicable to larger, more general spaces such as images and strings.

We can further show that our estimator is consistent (proof omitted).

Theorem 6 *Assume $k(x, \cdot)$ is in the range of $\mathcal{C}_{XX}$. The empirical conditional embedding $\hat{\mu}_{Y|x}$ converges to $\mu_{Y|x}$ in the RKHS norm at a rate of $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

Theorem 6 suggests that conditional embeddings are harder to estimate than ordinary embeddings. If we seek an estimator with a bias level of ϵ, we can fix λ to the order of $O(\epsilon^2)$ and obtain an overall convergence rate of $O(m^{-\frac{1}{2}})$.

4. Operations on RKHS Embeddings

Conditional embeddings and cross-covariance operators largely extend our ability to manipulate distributions embedded in an RKHS. In this section, we discuss several examples of this new vocabulary of operations.

Sum rule: Given a joint distribution over X and Y, we would like to compute the marginal distribution over a single variable X by summing out Y. $P(X) = \int_Y P(X, Y) = \int_Y P(X|Y)P(Y)$, We would like to now derive the Hilbert space counterpart of this operation. Using the law of total expectation, we

[1] Negative weights $\frac{1}{n-m}$ are due to centering matrix H.

have $\mu_X = \mathbb{E}_{XY}[\varphi(X)] = \mathbb{E}_Y \mathbb{E}_{X|Y}[\varphi(X)|Y]$. Plugging in the conditional embedding, we have

$$\mu_X = \mathbb{E}_Y[\mathcal{U}_{X|Y}\phi(Y)] = \mathcal{U}_{X|Y}\mathbb{E}_Y[\phi(Y)] = \mathcal{U}_{X|Y}\mu_Y. \quad (6)$$

Chain rule: The chain rule is given by $P(X,Y) = P(X|Y)P(Y) = P(Y|X)P(X)$, where we factorize the joint distribution into a product of two distributions. This rule builds the connection between the two ways of factorization which form the basis for Bayes rule. In this case, if we use a tensor product joint feature map $\varphi(X) \otimes \phi(Y)$, there are two equivalent ways of factoring $\mu_{XY} = \mathbb{E}_{XY}[\varphi(X) \otimes \phi(Y)]$ according to the law of total expectation:

$$\mathbb{E}_Y[\mathbb{E}_{X|Y}[\varphi(X)|Y] \otimes \phi(Y)] = \mathcal{U}_{X|Y}\mathbb{E}_Y[\phi(Y) \otimes \phi(Y)]$$
$$\mathbb{E}_X[\mathbb{E}_{Y|X}[\phi(Y)|X] \otimes \phi(X)] = \mathcal{U}_{Y|X}\mathbb{E}_X[\varphi(X) \otimes \varphi(X)].$$

Let $\mu_X^{\otimes} := \mathbb{E}_X[\varphi(X) \otimes \varphi(X)]$ and $\mu_Y^{\otimes} := \mathbb{E}_Y[\phi(Y) \otimes \phi(Y)]$, we have:

$$\mu_{XY} = \mathcal{U}_{X|Y}\mu_Y^{\otimes} = \mathcal{U}_{Y|X}\mu_X^{\otimes} \quad (7)$$

Conditional cross-covariance. Sometimes, we would like to measure the strength of dependence between two random variables Y and Z conditioned on a given value $X = x$. To handle these situations, it will be convenient to first introduce a new operator, *conditional cross-covariance operator*[2]. In particular, just as the cross-covariance operator $\mathcal{C}_{XY}$ enabled us to compute $Cov_{XY}[f(X), g(Y)]$ using Equation (4), we would like to have an operator $\mathcal{C}_{YZ|x}$ which would enable us to compute $Cov_{YZ|x}[g(Y), r(Z)|x] := \mathbb{E}_{YZ|x}[g(Y)r(Z)|x] - \mathbb{E}_{Y|x}[g(Y)|x]\mathbb{E}_{Z|x}[g(Z)|x]$ via:

$$\langle g, \mathcal{C}_{YZ|x} r \rangle_{\mathcal{G}} \text{ or equivalently } \langle g \otimes r, \mathcal{C}_{YZ|x} \rangle_{\mathcal{G}\otimes\mathcal{H}}. \quad (8)$$

We show defining the conditional cross-covariance by

$$\mathcal{C}_{YZ|x} := \mu_{YZ|x} - \mu_{Y|x} \otimes \mu_{Z|x} \quad (9)$$

works: $Cov_{YZ|x}[g(Y), r(Z)|x]$ can be written as an inner product, $\langle g \otimes r, \mathbb{E}_{YZ|x}[\phi(Y) \otimes \psi(Z)|x] - \mathbb{E}_{Y|x}[\phi(Y)|x] \otimes \mathbb{E}_{Z|x}[\psi(Z)|x] \rangle_{\mathcal{G}\otimes\mathcal{H}}$; Plugging in conditional embeddings leads to the expression.

The conditional dependence between Y and Z given x can be quantified by, *e.g.* the Hilbert-Schmidt norm of $\mathcal{C}_{YZ|x}$. This new quantity measures the dependence between Y and Z locally at x; it is different from the dependence measure designed by Fukumizu et al. (2008) where the whole space of X is integrated out.

[2] An operator termed the conditional cross-covariance operator is also mentioned in the appendix of (Fukumizu et al., 2004), however it is defined differently from the operator that we use. In particular, here, we do not take the expectation with respect to the conditioning variable.

5. Application: Dynamical Systems

Having now developed the necessary machinery for manipulating conditional embeddings, we turn our attention to learning and inference in a dynamical system. In this section, we formulate a simple nonparametric algorithm for learning models and performing probabilistic inference in dynamical systems.

Setting. We model uncertainty in a dynamic system using a partially observable Markov model, which is a joint distribution $P(s^1, \ldots, s^T, o^1, \ldots, o^T)$ where s^t is the hidden state at timestep t and o^t is the corresponding observation. We assume Markovian dynamics, so that the joint distribution factorizes as $P(o^1|s^1) \prod_t P(o^t|s^t) P(s^t|s^{t-1})$. The conditional distribution $P(s^t|s^{t-1})$ is called the *transition model*, and describes the evolution of the system from one timestep to the next; the distribution $P(o^t|s^t)$ is called the *observation model*, and captures the uncertainty of a noisy measurement process.

We focus on *filtering*, in which one queries the model for the posterior at some timestep conditioned on all past observations. Denote the history of the dynamic system as $h^t := (o^1, \ldots, o^t)$. In filtering, one recursively maintains a belief state, $P(s^{t+1}|h^{t+1})$, in two steps: a *prediction* step and a *conditioning* step. The first updates the distribution by multiplying by the transition model and marginalizing out the previous timestep: $P(s^{t+1}|h^t) = \mathbb{E}_{s^t|h^t}[P(s^{t+1}|s^t)|h^t]$. The second conditions the distribution on a new observation o^{t+1} using Bayes rule: $P(s^{t+1}|h^t o^{t+1}) \propto P(o^{t+1}|s^{t+1})P(s^{t+1}|h^t)$.

Prediction in Hilbert space. The prediction and conditioning steps which we have discussed can be *exactly* reformulated with respect to the Hilbert space embeddings. We begin by discussing how the prediction step of inference can be implemented with respect to the estimated Hilbert space embedding of the distribution $P(s^{t+1}|h^t) = \mathbb{E}_{s^t|h^t}[P(s^{t+1}|s^t)|h^t]$. Due to the fact that we maintain the belief state recursively, we can assume that $\mu_{s^t|h^t}$ is known. Thus our goal is to express the Hilbert space embedding, $\mu_{s^{t+1}|h^t}$, in terms of $\mu_{s^t|h^t}$ and the conditional embedding $\mathcal{U}_{s^{t+1}|s^t}$, for the transition model $P(s^{t+1}|s^t)$. We have the following (using the notation from section 2):

Theorem 7 *Hilbert space prediction step is given by:* $\mu_{s^{t+1}|h} = \mathcal{U}_{s^{t+1}|s^t}\mu_{s^t|h}$.

Proof Using Markov property $s^{t+1} \perp h^t|s^t$, we have $\mathbb{E}_{s^{t+1}|h}[\phi(s^{t+1})|h] = \mathbb{E}_{s^t|h}[\mathbb{E}_{s^{t+1}|s^t}[\phi(s^{t+1})|s^t]|h]$. Furthermore, $\mathbb{E}_{s^t|h}[\mathbb{E}_{s^{t+1}|s^t}[\phi(s^{t+1})|s^t]|h] = \mathcal{U}_{s^{t+1}|s^t}\mu_{s^t|h}$ using the sum rule in Section 4. ■

Conditioning in Hilbert space. We now discuss how the conditioning step of inference can be implemented with respect to the Hilbert space embeddings, in which we obtain the embedding of the posterior distribution, $\mu_{s^{t+1}|h^t o^{t+1}}$, given the prior and likelihood. Note that the prior term, $\mu_{s^{t+1}|h^t}$, is supplied by the prediction step in the recursion. Here the goal is to obtain $\mu_{s^{t+1}|h^t o^{t+1}}$ by conditioning further on variable o^{t+1}, and we accomplish this iterated conditioning by using the conditional cross-covariance operator from Section 4. We have the following:

Theorem 8 *Hilbert space conditioning step is given by:* $\mu_{s^{t+1}|h^t o^{t+1}} = \mathcal{C}_{s^{t+1}o^{t+1}|h^t} \mathcal{C}^{-1}_{o^{t+1}o^{t+1}|h^t} \psi(o^{t+1})$.

Proof This is a straightforward application of the conditional cross-covariance operator in section 4 and conditional embedding from Theorem 4. As an intermediate step, the embeddings for $\mu_{o^{t+1}o^{t+1}|h^t}$ and $\mu_{o^{t+1}|h^t}$ are needed. They can be derived using the sum rule and the Markov property. ■

While having the exact updates for prediction (Theorem 7) and conditioning (Theorem 8) is theoretically satisfying, it does not lead to practical estimation algorithms due to the fact that the conditional cross-covariance operators need to be estimated in each conditioning step, which is both statistically difficult and computationally costly. In the following, we will make simplifying assumptions to the dynamical systems and derive an efficient approximate algorithm.

Approximate inference. Our goal still remains to recursively maintain the Hilbert space embedding $\mu_{s^t|h^t}$. However, in our approximation, we directly construct the operator $\mathcal{U}_{s^{t+1}|s^t o^{t+1}} = \mathcal{C}_{s^{t+1}(s^t o^{t+1})} \mathcal{C}^{-1}_{(s^t o^{t+1})(s^t o^{t+1})}$, which takes both the previous belief state and an observation as inputs, and outputs the predictive embedding for $P(s^{t+1}|s^t o^{t+1})$. That is $\mu_{s^{t+1}|s^t o^{t+1}} = \mathcal{U}_{s^{t+1}|s^t o^{t+1}} \omega((s^t, o^{t+1}))$, where $\omega(\cdot)$ is the joint feature map for s^t and o^{t+1}.

In the same spirit as the decomposition of sufficient statistics in exponential families (Altun et al., 2004; Zhang et al., 2009), we use the concatenation of the feature map $\varphi(s^t)$ and $\psi(o^{t+1})$ as $\omega((s^t, o^{t+1}))$; that is $\omega((s^t, o^{t+1})) = (\varphi(s^t)^\top, \psi(o^{t+1})^\top)^\top$ whose kernel decomposes into the sum of two kernels: $\langle \omega((s,o)), \omega((s',o')) \rangle = k(s,s') + u(o,o')$. With this feature map, the operator $\mathcal{U}_{s^{t+1}|s^t o^{t+1}}$ can be equivalently viewed as the concatenation of two operators, $\mathcal{U}_{s^{t+1}|s^t o^{t+1}} = (\mathcal{T}_1, \mathcal{T}_2)$. More specifically,

$$\mu_{s^{t+1}|s^t o^{t+1}} = \mathcal{T}_1 \varphi(s^t) + \mathcal{T}_2 \psi(o^{t+1}) \tag{10}$$

Next we want to marginalize out variable s^t and condition on the history h^t. We follow the same logic from the sum rule in Section 4, and take the expectation of $\mu_{s^{t+1}|s^t o^{t+1}}$ with respect to $P(s^t|h^t o^{t+1})$:

$$\begin{aligned}
\mu_{s^{t+1}|h^t o^{t+1}} &= \mathbb{E}_{s^t|h^t o^{t+1}} \left[\mu_{s^{t+1}|s^t o^{t+1}} \right] \\
&= \mathcal{T}_1 \mathbb{E}_{s^t|h^t o^{t+1}} \left[\varphi(s^t) | h^t o^{t+1} \right] + \mathcal{T}_2 \psi(o^{t+1}) \\
&\approx \mathcal{T}_1 \mathbb{E}_{s^t|h^t} \left[\varphi(s^t) | h^t \right] + \mathcal{T}_2 \psi(o^{t+1}) \\
&= \mathcal{T}_1 \mu_{s^t|h^t} + \mathcal{T}_2 \psi(o^{t+1}),
\end{aligned} \tag{11}$$

where we have approximated the distribution $P(s^t|h^t, o^{t+1})$ by a less confident one $P(s^t|h^t)$.

Updating the belief state with respect to RKHS embeddings thus conveniently decomposes into two simple operations: first, propagate from time t to time $t+1$ via $\mathcal{T}_1 \mu_{s^t|h}$; and second, account for the most current current observation via $\mathcal{T}_2 \psi(o^{t+1})$. The main point which distinguishes our approximation from the exact inference operations is that the approximate approach carries out the prediction and conditioning operations in parallel while the exact method performs the operations sequentially.

Computationally, our approximate algorithm is also quite efficient. The operators $\mathcal{T}_1$ and $\mathcal{T}_2$ only needs to be estimated once (during the training stage), and at each filtering iteration, we only need to perform a matrix-vector mulitiplication. More specifically:

Theorem 9 *Let* $G^\Phi_\Upsilon := \Upsilon^\top \Phi$, *and* $u_{o^{t+1}} := \Psi^\top \psi(o^{t+1})$. *The update rule for the belief state is given by:* $\hat{\mu}_{s^{t+1}|ho^{t+1}} \leftarrow \Phi(HK + HU + \lambda m I)^{-1} H (G^\Phi_\Upsilon \beta + u_{o^{t+1}})$. *Learning can be accomplished in* $O(m^3)$ *time and each filtering iteration requires* $O(m^2)$ *time.*

Proof Applying Theorem 5, we have $\hat{\mu}_{s^{t+1}|ho^{t+1}}$ as $\Phi(HK+HU+\lambda m I)^{-1} H(\Upsilon^\top \hat{\mu}_{s^t|h} + \Psi^\top \psi(o^{t+1}))$, where we have used the fact that $\langle \omega((s,o)), \omega((s',o')) \rangle = k(s,s') + u(o,o')$. Since the belief state $\hat{\mu}_{s^t|h}$ is maintained by Φ and a set of weights β, replacing $\hat{\mu}_{s^{t+1}|h}$ by $\Phi\beta$, we have the update rule in the theorem. Let $\tilde{\mathcal{T}}_2 := (HK + HU + \lambda m I)^{-1} H$ and $\tilde{\mathcal{T}}_1 := \tilde{\mathcal{T}}_2 G^\Phi_\Upsilon$ (note that since s^t and s^{t+1} are in the same space, G^Φ_Υ is a valid inner product matrix). Therefore, updating $\hat{\mu}_{s^{t+1}|ho^{t+1}}$ is also equivalent to updating β via $\beta \leftarrow \tilde{\mathcal{T}}_1 \beta + \tilde{\mathcal{T}}_2 u_{o^{t+1}}$. Computing $\tilde{\mathcal{T}}_1$ and $\tilde{\mathcal{T}}_2$ in the learning stage is dominated by a matrix inversion which is $O(m^3)$ in general. In the filtering stage, applying the update rule for β involves $O(m)$ kernel evaluations and two $O(m^2)$ matrix-vector multiplications. Each filtering ieration is $O(m^2)$ in term of the number of kernel evaluations. ■

MAP inference. We have now shown how to maintain a posterior distribution over the hidden state conditioned on the current and all past observations. Sometimes, however, we would like to

Algorithm 1 Learning

Input: $\Upsilon = (\varphi(s^t))$, $\Phi = (\phi(s^{t+1}))$ and $\Psi = (\psi(o^{t+1}))$ where $t = 1 \dots m$.

1: Compute $K = \Upsilon^\top \Upsilon$, $U = \Psi^\top \Psi$, $G_\Upsilon^\Phi = \Upsilon^\top \Phi$.
2: Compute $\tilde{\mathcal{T}}_2 := (HK + HU + \lambda m I)^{-1} H$,
3: Compute $\tilde{\mathcal{T}}_1 := \tilde{\mathcal{T}}_2 G_\Upsilon^\Phi$.

Algorithm 2 Filtering

Input: Ψ, $\tilde{\mathcal{T}}_1$, $\tilde{\mathcal{T}}_2$ from learning, and let $\beta = \frac{1}{m}\mathbf{1}$.

1: **for** each new observation o^{t+1} **do**
2: Compute $u_{o^{t+1}} = \Psi^\top \psi(o^{t+1})$.
3: Belief update: $\beta \leftarrow \tilde{\mathcal{T}}_1 \beta + \tilde{\mathcal{T}}_2 u_{o^{t+1}}$.
4: Inference: $\hat{s}^{t+1} = \operatorname{argmax}_s 2\beta^\top l_s - l(s,s)$.
5: **end for**

determine the MAP (maximum a posteriori) assignments of hidden states. We accomplish MAP inference by performing the optimization $\hat{s}^{t+1} = \operatorname{argmin}_s \left\| \phi(s) - \hat{\mu}_{s^{t+1}|ho^{t+1}} \right\|_{\mathcal{G}}^2$. Since the belief state $\hat{\mu}_{s^{t+1}|ho^{t+1}}$ is maintained as the feature matrix Φ and a set of weights β, the optimization can be expressed using kernels as:

$$\hat{s}^{t+1} = \operatorname{argmax}_s 2\beta^\top l_s - l(s,s), \tag{12}$$

where $l_s := \Phi^\top \phi(s)$. Note that this optimization may be a hard problem in general, and is analogous to the argmax operation in structured SVMs (Tsochantaridis et al., 2005). In many cases, however, it is possible to define the feature map $\phi(s)$ in such a way that an efficient algorithm for solving the optimization (Equation (12)) can be obtained. For instance, consider the *identity management* problem arising in multiobject tracking, where the hidden states are permutations (one-to-one matchings between tracks and identities). Using the kernel $l(\pi, \pi') := \operatorname{tr}(\pi^\top \pi')$ defined over permutation matrices, we obtain a linear assignment problem which can be solved efficiently using a variety of algorithms.

Algorithm. Our method maintains a representation of the state distribution as a weighted sum of feature functions evaluated at the set of training examples. To propagate the state distribution with respect to Markovian dynamics and to condition, our algorithms can simply be implemented using kernel evaluations and matrix operations on the weight vector β.

We summarize our method using the pseudocode in Algorithms 1 and 2. In the learning phase, we use the training data to form the appropriate kernel matrices and operator estimates. During the filtering phase, we are presented with an online input sequence of test observations and we use the update rule to maintain state distributions at each timestep. The user-specified parameters are kernel choices for the state and observation spaces, and the regularization parameter.

6. Related work

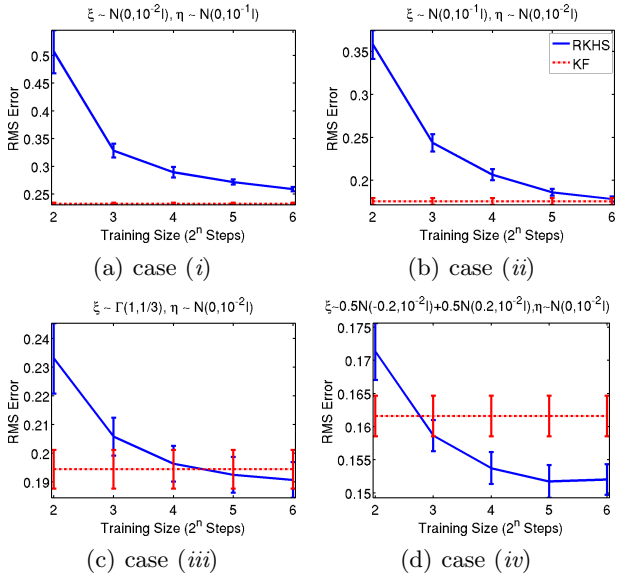
Gaussian process regression uses $k_x^\top (K + \lambda I)^{-1} \mathbf{y}$ as the predictive mean (Rasmussen & Williams, 2006). As it turns out, this is equivalent to the conditional embedding of the distribution $P(Y|x)$ restricted to a linear feature map on Y. Cortes et al. (2005) studied kernel dependency estimation for transduction. There, a function $\mathbf{W}\varphi(x)$ is learned for mapping from x to $\phi(y)$. By solving an optimization, they derive the form, $\Phi(K + \lambda I)^{-1}\Upsilon$, for $\mathbf{W}$. This is exactly the conditional embedding operator $\mathcal{U}_{Y|X}$ we discussed in this paper. Therefore, our work immediately provides theoretical and statstical support for their approach. Finally, Vishwanathan et al. (2007) considered linear dynamical systems in an RKHS, where a transition operator A and an observation operator B are used to describe the dynamics: $\varphi(s^{t+1}) = A\varphi(s^t)$ and $\psi(o^{t+1}) = B\varphi(s^{t+1})$. Under our Hilbert space embedding view of dynamical systems, the operators A and B correspond to $\mathcal{U}_{s^{t+1}|s^t}$ and $\mathcal{U}_{o^{t+1}|s^{t+1}}$ respectively, which enables the entire distribution for the hidden state s^t to propagate through the dynamics.

7. Experiments

We evaluate our algorithm on two dynamical systems: a synthetic dataset generated from a linear dynamical systems with known model parameters, and a challenging camera tracking problem.

Synthetic Data. We simulate a particle rotating around the origin in $\mathbb{R}^2$ according to the following dynamics: $s^{t+1} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} s^t + \xi$, $o^{t+1} = s^{t+1} + \eta$ where $\theta = 0.02$ radians, ξ and η are process noise and observation noise respectively. To evaluate our algorithm, we compare the performace of our method to the performance of a Kalman filter in estimating the position of the particle. The Kalman filter requires the exact knowledge of the dynamics in order to perform filtering, while our method only requires a sequence of training pairs of hidden states and observations. We study how the different methods respond to various noise models. In particular, we use 4 noise configurations arranged in increasing nonlinearity (Let $Q_1 := \mathcal{N}(-0.2, 10^{-2} I)$ and $Q_2 := \mathcal{N}(0.2, 10^{-2} I)$):

	Process noise ξ	Observation noise η
(i)	$\mathcal{N}(0, 10^{-2} I)$	$\mathcal{N}(0, 10^{-1} I)$
(ii)	$\mathcal{N}(0, 10^{-1} I)$	$\mathcal{N}(0, 10^{-2} I)$
(iii)	Gamma $\Gamma(1, \frac{1}{3})$	$\mathcal{N}(0, 10^{-1} I)$
(iv)	Gaussian Mixture $\frac{1}{2}Q_1 + \frac{1}{2}Q_2$	$\mathcal{N}(0, 10^{-2} I)$



(a) case (*i*) (b) case (*ii*)

(c) case (*iii*) (d) case (*iv*)

Figure 1. Averge RMS and standard error over 50 runs of the experiment for noise configurations *i,ii,iii and iv. KF: Kalman filter; RKHS: our method.*

We also study how our method scales in terms of filtering performance as more training data are available. We generate training data by observing the particle rotate for 2^n steps where $n = 1 \dots 6$. The test data always contains observations from a single cycle. For each training set size, we randomly instantiate the experiment 50 times to obtain the standard errors. We use an RBF kernel for both the hidden state and the observations, and we set the regularization parameter to $\lambda = 10^{-3}$. We perform MAP inference by searching over a predefined grid over $[-3 : 0.03 : 3]^2$ (alternatively, one can also use a gradient descent for MAP). The average RMS (root mean square error) over 50 runs and the standard errors are shown in Figure 1.

The general trend is that, as more training data are available, both the average RMS and the standard error of our method decrease monotonically. In cases when the Kalman filter is optimal (case *i, ii*), the performance of our method approaches that of the Kalman filter as more training data become available. In nonlinear dynamical systems (case *iii, iv*), our method outperforms the Kalman filter when training datasets are sufficiently large. This is expected since the Kalman update is specifically designed for Gaussian noise while our nonparametric method is more versatile in handling general noise distributions.

Camera Rotation. Here we apply our dynamical system tools to a computer vision problem. We try to determine the camera orientation based on the images it observes. In our setting, the camera focal point is fixed at a position and traces out a smooth path of rotations while making observations. To evaluate our algorithm, we use POVRAY (www.povray.org) to render images observed by the camera. The virtual scene is a rectangular-shaped room with a ceiling light and two pieces of furniture. The images exhibit complex lighting effects such as shadows, interreflections, and global illumination, all of which make determining the camera rotation difficult. As an illustration, we display sample images in Figure 2. The sizes of the color images are $100 \times 100 \times 3$. We downsample them to $20 \times 20 \times 3$ in order for our competitor, the Kalman filter, to learn its model efficiently.

We generate a sequence of image observations as follows: first we uniformly sample 180 random rotation matrices. Then we interpolate between these anchor positions using 20 midpoints. For each generated rotation matrix, we render an image using the rotation matrix as camera parameters. In total we obtain 3600 frames, and we use the first 1800 frames for training and the remaing 1800 frames for testing. The dynamics governing the camera rotation is a piece-wise smooth random walk. This is an unconventional dynamical system in that the hidden state is a rotation matrix R from $SO(3)$; and the observations are images which are high dimensional spaces with correlation between pixel values.

We flatten each image to a vector of $\mathbb{R}^{1200}$, and apply a Gaussian RBF kernel. The bandwidth parameter of the kernel is fixed using the median distance between image vectors, and $\lambda = 10^{-6}$. We use an inner product kernel between two rotations R and $\tilde{R}$, *i.e.* $k(R, \tilde{R}) := \mathrm{tr}(R^\top \tilde{R})$. Using this kernel, we can perform efficient MAP inference for the rotation matrix. The optimization problem then becomes a linear program over $SO(3)$: $\mathrm{argmax}_R \, \mathrm{tr}(A^\top R)$ s.t. $R \in SO(3)$, where A is obtained from $\Phi\beta$ and can be efficiently solved using a steepest descent algorithm over Stiefel manifold (Abrudan et al., 2008).

We compare our method to a Kalman filter and random guessing. For the Kalman filter, we used the quaternion corresponding to a rotation matix R as the hidden state and the image vectors as the observations. We learn the model parameters of the linear dynamical system using linear regression. We consider two error metrics. The first is $\mathrm{tr}(R^\top \hat{R})$ which is the equal to $1 + 2\cos(\theta)$ where θ is the angle between the true rotation matrix R and its estimate $\hat{R}$. This metric ranges between $[-1, 3]$. As a second metric, we use the Frobenius norm $\|R - \hat{R}\|_{Fro}$. Note that these two measures are related, *i.e.* $\|R - \hat{R}\|_{Fro}^2 = 6 - 2\,\mathrm{tr}(R^\top \hat{R})$.

(a) (b) (c)

Figure 2. (a)(b) Example rendered images of the well known Cornell box with a rotating camera. (c) RKHS filter compared to simple regression in estimating camera rotation.

Table 2. Errors in estimating the camera rotation matrices by three methods. Note that for the trace measure, the *larger* the number the better the performance; but for the Frobenius norm, the *smaller* the number the better.

Error Metric	RKHS	Kalman	Random
Trace	2.91 ± 0.005	2.18 ± 0.016	0.01 ± 0.023
Frobenius	0.17 ± 0.005	1.18 ± 0.012	2.40 ± 0.023

The results on the test image sequence are shown in Table 7. We can see that in terms of both error metrics, our method performs significantly better than the Kalman filter and random guessing.

To investigate whether incorporating dynamics really helps to better estimate the camera rotations, we compare our RKHS filter to a simple regression which completely ignores the dynamics (camera orientations are close for adjacent timepoints). We add zero mean Gaussian white noise to the images and study the performance scaling of the two methods as we increase the noise variance. We observe that the performance of our RKHS filter which does take into account the dynamics degrades more gracefully for increasing noise levels (Figure 2(c)).

8. Conclusion

We have presented a general framework for embedding conditional distributions into a reproducing kernel Hilbert space. The resulting embedding method greatly improves our ability to manipulate distributions in RKHS, and in particular, we showed how conditional embeddings are useful for modeling and performing nonparametric inference on dynamical systems. The advantage using the Hilbert space embedding lies in its generality and ease of implementation. Our method suffers from several limitations at the moment. For example, we require training data for the hidden states. We believe that extending our embedding approach along these directions will be both fruitful and interesting for future research, and we expect that conditional embedding will be a useful tool for learning problems beyond dynamical systems.

Acknowledgments

J. Huang is grateful for support by the ONR under MURI N000140710747, and the NSF under grants DGE-0333420, EEEC-540865, NeTS-NOSS 0626151, TF-0634803.

References

Abrudan, T., Eriksson, J., & Koivunen, V. (2008). Steepest descent algorithms for optimization under unitary matrix constraint. *IEEE Transactions on Signal Processing*, *56*(3).

Altun, Y., Smola, A. J., & Hofmann, T. (2004). Exponential families for conditional random fields. In *Uncertainty in Artificial Intelligence*.

Baker, C. (1973). Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, *186*, 273–289.

Cortes, C., Mohri, M., & Weston, J. (2005). A general regression technique for learning transductions. In *Proceedings of the 22nd International Conference on Machine Learning*.

Fukumizu, K., Bach, F. R., & Jordan, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *JMLR.*, *5*, 73–99.

Fukumizu, K., Gretton, A., Sun, X., & Schölkopf, B. (2008). Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems 20*.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., & Smola, A. (2007). A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems 19*.

Quadrianto, N., Smola, A., Caetano, T., & Le, Q. (2008). Estimating labels from label proportions. In W. Cohen, A. McCallum, & S. Roweis, eds., *Proceedings of the 25th Annual International Conference on Machine Learning*.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press.

Schölkopf, B., Tsuda, K., & Vert, J.-P. (2004). *Kernel Methods in Computational Biology*. Cambridge, MA: MIT Press.

Smola, A., Gretton, A., Song, L., & Schölkopf, B. (2007). A hilbert space embedding for distributions. In E. Takimoto, ed., *Algorithmic Learning Theory*, Lecture Notes on Computer Science. Springer.

Song, L., Smola, A., Borgwardt, K., & Gretton, A. (2008). Colored maximum variance unfolding. In *Advances in Neural Information Processing Systems 20*.

Sriperumbudur, B., Gretton, A., Fukumizu, K., Lanckriet, G., & Schölkopf, B. (2008). Injective hilbert space embeddings of probability measures. In *Proc. Annual Conf. Computational Learning Theory*.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *JMLR*, *6*, 1453–1484.

Vishwanathan, S. V. N., Smola, A. J., & Vidal, R. (2007). Binet-Cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes. *International Journal of Computer Vision*, *73*(1), 95–119.

Zhang, X., Song, L., Gretton, A., & Smola, A. (2009). Kernel measures of independence for non-iid data. In *Advances in Neural Information Processing Systems 21*.