Kernel Monte Carlo Filter

Motonobu Kanagawa* Yu Nishiyama† Arthur Gretton‡
Kenji Fukumizu§

December 18, 2013

Abstract

Filtering methods for state-space models have been successfully applied to a wide range of applications. However, standard methods assume that the observation model is explicitly known, at least in a parametric form. This can be a problem for tasks where the observation model cannot be easily obtained. This paper proposes a filtering method for such situations based on the recent nonparametric framework of RKHS embeddings. The proposed kernel Monte Carlo filter combines Monte Carlo simulation with the state-transition model and RKHS embeddings for the unknown observation model. As a basis for the proposed method, we theoretically justify the use of Monte Carlo simulation with RKHS embeddings. This is the first work incorporating explicit probabilistic models into the RKHS embedding approach. Experimental results with artificial data and a real vision-based mobile robot localization problem confirm superior performance of the kernel Monte Carlo filter over existing approaches.

1 Introduction

Filtering methods for state-space models (SSMs) have been applied to wide variety of applications including econometrics, robotics, and computer vision. An SSM with hidden state $x_t \in \mathcal{X}$ and observation $z_t \in \mathcal{Z}$, where $t$ denotes time step, is specified by the state-transition model $p(x_t|x_{t-1})$ and the observation model $p(z_t|x_t)$. The task of filtering is to estimate the posterior distribution over states

---

*The Graduate University for Advanced Studies, kanagawa@ism.ac.jp
†The Institute of Statistical Mathematics, nishiyam@ism.ac.jp
‡Gatsby Computational Neuroscience Unit, University College London, arthur.gretton@gmail.com
§The Institute of Statistical Mathematics, fukumizu@ism.ac.jp
given a sequence of observations \( z_{1:t} := (z_1, \ldots, z_t) \) sequentially for any time \( t \). In the classical formulation, linear Gaussian models are used for the state-transition and observation, and simple and efficient algorithms (such as the Kalman filter) are well known. Nonlinear extensions such as the extended and unscented Kalman filters (Julier and Uhlmann, 1997) are also popular. If the statistical models are more complex but still explicitly known, particle filters (Doucet et al., 2001) provide a strong solution for filtering. Particle filters employ Monte Carlo simulation with the state-transition model while using the observation model to compute the likelihood of a given observation.

These standard filtering methods for SSMs, however, all make the assumption that the observation model is explicitly known, at least in a parametric form. This can be a limitation for real world problems for which the observation model is not easily given. Consider for example a vision-based mobile robot localization problem (Dellaert et al., 1999), where the state \( x_t \) is the position of a robot in a map and the observation \( z_t \) is its vision data. The observation model \( p(z_t|x_t) \) is the conditional probability of vision images given positions of the robot. In this case, it is not easy to introduce an appropriate observation model, since images are high-dimensional and their generative process is highly environment-dependent. Similar situations occur in the problem of location estimation based on signal-strength (Ladd et al., 2002), where the observation model \( p(z_t|x_t) \) is the conditional probability of signal-strengths given positions of a mobile device. In these examples, standard methods that require explicit knowledge of the observation models are not easily applicable.

**Part-Nonparametric SSMs.** In this paper, we deal with filtering problems where the observation model \( p(z_t|x_t) \) is not easily obtained, and therefore to be learned nonparametrically from training data \((X_1, Z_1), \ldots, (X_n, Z_n)\). We assume, on the other hand, that Monte Carlo simulation with the state-transition model \( p(x_t|x_{t-1}) \) is possible, as for standard particle filters. We call such problems or SSMs as part-nonparametric in this paper (Figure 1). In the above robot localization example, the complicated observation model is suitable for nonparametric learning with training data, since any parametric model for the problem may cause significant bias errors. On the other hand, simulation with the state-transition model is possible, as this represents robot motion (Thrun et al., 2005). The part-nonparametric approach is also effective for the signal-strength localization problem by similar reason.

Filtering methods for the part-nonparametric SSMs have been already proposed in the literature. Vlassis et al. (2002) consider the vision-based localization problem, and construct a particle filter combined with \(k\)-nearest neighbor approach. For the signal-strength-based location estimation task, Ferris et al. (2006) use Gaussian process regression for learning the observation model \( p(z_t|x_t) \), and
Figure 1: Part-nonparametric state-space model.

apply a particle filter. These methods, however, must introduce certain approximations for the observation model, as seen in Section 6, and thus may cause significant bias errors in posterior estimation.

**RKHS Embeddings.** Alternatively, a nonparametric statistical inference approach may be based on RKHS embeddings (Smola et al., 2007; Sriperumbudur et al., 2010), which represent probability distributions by elements in a reproducing kernel Hilbert space (RKHS). This approach has been successfully applied to diverse applications including hypothesis testing (Gretton et al., 2012, 2008), Bayesian inference (Song et al., 2009; Fukumizu et al., 2011), and reinforcement learning (Grünewälder et al., 2012; Nishiyama et al., 2012). The RKHS framework has the following desirable properties: 1) it is completely nonparametric, and therefore makes no strong assumptions on models, 2) it naturally incorporates information represented as similarity functions between data, i.e. kernels, and 3) it is effective for high-dimensional data. The implementation of RKHS embedding-based methods simply require basic linear algebra operations on kernel matrices computed from data. Thus far, however, these methods have only been applied to fully nonparametric situations, with models containing no parametric components.

**Contributions.** In this paper, we propose a filtering method for the part-nonparametric SSMs based on the above RKHS embedding approach. The proposed Kernel Monte Carlo Filter (KMCF) combines Monte Carlo simulation with the state-transition model and the matrix computation of RKHS embeddings for the unknown observation model. Our contributions are twofold.

First, we develop a strong solution for the part-nonparametric SSMs. Unlike the existing approaches for this setting, the proposed KMCF does not make strong assumptions on the observation model by virtue of RKHS embeddings. We apply Kernel Bayes’ rule (Fukumizu et al., 2011) for estimating the posterior distribu-
tions over states. The proposed method can be applied not only to $\mathbb{R}^d$, but to other domains such as graphs, images, or documents once appropriate kernels are defined. This significantly widens the range of potential applications.

Our second contribution is to enrich the RKHS embedding approach. While the approach has been used for diverse applications, there has not been any work incorporating parametric/known models into inference methods based on RKHS embeddings. The proposed KMCF is the first algorithm that combines an explicit probabilistic model (i.e. the state-transition model) with the nonparametric kernel method. The key ingredient is Monte Carlo simulation with a conditional probability, and we develop necessary theoretical results to justify its use with RKHS embeddings. Note that the theoretical justification is not obvious beforehand, since we cannot straightforwardly apply the standard theoretical arguments from particle methods to the RKHS setting, as we will explain later.

This paper is organized as follows. We first review the RKHS embedding approach in Section 2. We discuss the use of Monte Carlo simulation with RKHS embeddings in Section 3, and the kernel Monte Carlo filter in Section 4. Related work is shown in Section 5. The effectiveness of the proposed method is demonstrated with experiments in Section 6.

2 RKHS Embeddings

We first briefly review the RKHS embedding approach [Smola et al., 2007; Sriperumbudur et al., 2010], which provides a framework for nonparametric statistical inference with positive definite kernels.

Kernel Mean. Let $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ be a measurable space. A symmetric measurable kernel $k_\mathcal{X}: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is called positive definite, if $\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k_\mathcal{X}(X_i, X_j) \geq 0$ for any $n \in \mathbb{N}$, $c_1, \ldots, c_n \in \mathbb{R}$ and $X_1, \ldots, X_n \in \mathcal{X}$. Throughout this paper we will use the terminology kernel to refer to functions satisfying the positive definiteness. A kernel $k_\mathcal{X}$ uniquely defines a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_\mathcal{X}$ such that the reproducing property $f(x) = \langle f, k_\mathcal{X}(\cdot, x) \rangle_{\mathcal{H}_\mathcal{X}}$ holds for any $f \in \mathcal{H}_\mathcal{X}$ and $x \in \mathcal{X}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}_\mathcal{X}}$ denotes the inner-product of $\mathcal{H}_\mathcal{X}$. Let $\| \cdot \|_{\mathcal{H}_\mathcal{X}}$ denote the norm of $\mathcal{H}_\mathcal{X}$. $k_\mathcal{X}$ is said to be bounded if $\sup_{x \in \mathcal{X}} k_\mathcal{X}(x, x) < \infty$. Kernels that appear in this paper are assumed to be bounded.

In the RKHS embedding approach, we represent any probability distribution $P$ on $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ by an element in $\mathcal{H}_\mathcal{X}$ called kernel mean

$$m_P := \mathbb{E}_{X \sim P}[k_\mathcal{X}(\cdot, X)] := \int k_\mathcal{X}(\cdot, x) dP(x) \in \mathcal{H}_\mathcal{X}.$$ 

An important property of the kernel mean is the reproducing property of the expectation, that is $\mathbb{E}_{X \sim P}[f(X)] = \langle m_P, f \rangle_{\mathcal{H}_\mathcal{X}}$ for all $f \in \mathcal{H}_\mathcal{X}$.
If the kernel \( k_X \) is \textit{characteristic}, any probability distribution is uniquely determined by its kernel mean \cite{Fukumizu2004, Fukumizu2009, Sriperumbudur2010}. Examples of characteristic kernels include standard kernels such as Gaussian kernel \( k_\sigma(x, x') = \exp(-\|x-x'\|^2/2\sigma^2) \) over \( \mathcal{X} = \mathbb{R}^d \) \cite{Sriperumbudur2010}. Using the characteristic kernels, we aim to directly estimate the kernel mean \( m_P \) from samples, instead of estimating the distribution \( P \) itself.

**Kernel Mean Estimator.** If we have an i.i.d. sample \( X_1, \ldots, X_n \) from \( P \), the kernel mean \( m_P \) can be estimated by the empirical mean \( \hat{m}_P := \frac{1}{n} \sum_{i=1}^n k_X(\cdot, X_i) \) with the rate \( \| \hat{m}_P - m_P \|_{H_X} = O_p(n^{-\frac{1}{2}}) \) \cite{Smola2007}. In general the kernel mean of \( P \) may be estimated with samples \( X_i \) of a distribution different from \( P \), and therefore it may have a form of weighted sum of feature vectors

\[
\hat{m}_P := \sum_{i=1}^n \alpha_i k_X(\cdot, X_i)
\]

with weights \( \alpha_1, \ldots, \alpha_n \in \mathbb{R} \) \cite{Song2009, Fukumizu2011}. A typical example is the kernel mean estimator of a conditional probability \cite{Song2009}, in which the weights take the form

\[
\alpha_i = ((G_X + n\varepsilon_n I_n)^{-1} k_Y(y))_i,
\]

where \( G_X \) is a Gram matrix, \( I_n \) is an identity, \( \varepsilon_n > 0 \) is a regularization constant, and \( k_Y(y) = (k_Y(y, Y_1), \ldots, k_Y(y, Y_n))^T \in \mathbb{R}^n \).

The expression Eq. (1) can be regarded as the kernel mean of the (signed) measure \( \sum_{i=1}^n \alpha_i \delta_{X_i} \), where \( \delta_x \) is a Dirac measure at \( x \), and thus it shows a weighted sample expression of \( P \) in the form of \( \{(X_i, \alpha_i)\}_{i=1}^n \). Note, however, that the weights \( \alpha_i \) in general may include \textit{negative} values. It is not difficult to see, for example, that the above conditional kernel mean can have negative weights.

### 3 Monte Carlo with Kernel Means

Filtering methods for SSMs in general require the estimation of the predictive distribution \( P(x_{t+1}|z_{1:t}) \) from an estimated posterior distribution \( P(x_t|z_{1:t}) \) on the current state and the state-transition model \( p(x_{t+1}|x_t) \). In this section, we consider such operation in a general setting, and discuss an estimator of the predictive distribution in terms of kernel means.

Let \( (\mathcal{X}, \mathcal{B}_X) \) and \( (\mathcal{Y}, \mathcal{B}_Y) \) be measurable spaces, \( P_X \) be a distribution on \( (\mathcal{X}, \mathcal{B}_X) \), and \( p(y|x) \) be a conditional probability on \( (\mathcal{Y}, \mathcal{B}_Y) \) conditioned on \( X \). We also assume that sampling from \( p(y|x) \) is possible. Our objective here is to compute the

\footnote{In general the weights \( \alpha_i \) may depend on the number of samples \( n \).}
predictive distribution $P_Y := \int p(\cdot|x)dP_X(x)$ from some estimator of $P_X$ by generating samples from $p(y|x)$.

In particle methods, $P_Y$ is estimated from an empirical distribution $\hat{P}_X = \frac{1}{n} \sum_{i=1}^{n} \delta_{X_i}(\cdot)$ by generating samples $Y_i \sim p(\cdot|X_i)$ ($i = 1, \ldots, n$) and making an estimator by $\hat{P}_Y = \frac{1}{n} \sum_{i=1}^{n} \delta_{Y_i}(\cdot)$. Convergence results for such estimators have been obtained in the particle methods literature (e.g., Chopin, 2004).

In the following, we will show that the same sampling procedure for the estimator $\hat{m}_{P_X} = \sum_{i=1}^{n} \alpha_i k_X(\cdot, X_i)$ (Eq. (1)) gives a consistent estimator for the kernel mean $m_{P_Y} = \mathbb{E}_{Y \sim P_Y}[k_Y(\cdot, Y)]$ of the predictive distribution. One might think it analogous to particle methods. However, because of the existence of negative weights in $\{\alpha_i\}_{i=1}^{n}$, the standard theoretical arguments from particle methods are not straightforwardly applicable. We therefore need a theoretical analysis specialized for the RKHS embedding setting.

**Estimator.** We explicitly define here the estimator of the kernel mean $\hat{m}_{P_Y}$ of the predictive distribution. Let $k_X$ and $k_Y$ be kernels on $X$ and $Y$, respectively, and $\mathcal{H}_X$ and $\mathcal{H}_Y$ be the respective RKHSs. Assume that we have a consistent estimator of the kernel mean $m_{P_X} = \mathbb{E}_{X \sim P_X}[k_X(\cdot, X)]$ given by

$$\hat{m}_{P_X} = \sum_{i=1}^{n} \alpha_i k_X(\cdot, X_i), \quad \alpha_1, \ldots, \alpha_n \in \mathbb{R}. \tag{2}$$

Then by generating samples $Y_i$ from the conditional probability

$$Y_i \sim p(\cdot|X_i) \quad i = 1, \ldots, n,$$

the kernel mean $m_{P_Y} = \mathbb{E}_{Y \sim P_Y}[k_Y(\cdot, Y)]$ of the predictive distribution is estimated by

$$\hat{m}_{P_Y} = \sum_{i=1}^{n} \alpha_i k_Y(\cdot, Y_i). \tag{3}$$

Theorem 1 below shows that this is in fact a consistent estimator of the kernel mean $m_{P_Y}$ and does not worsen the convergence rate over that of $\hat{m}_{P_X}$ (the proof is in Appendix A). In the following $\mathcal{H}_X \otimes \mathcal{H}_X$ denotes the RKHS of the product kernel $k_Xk_X: (X \times X) \times (X \times X) \to \mathbb{R}$. Note that the weights $\{\alpha_i\}_{i=1}^{n}$ are random variables since they depend on the random variables $\{X_i\}_{i=1}^{n}$.

**Theorem 1.** Let $k_X$ and $k_Y$ be bounded characteristic kernels. Let $\hat{m}_{P_X} = \sum_{i=1}^{n} \alpha_i k_X(\cdot, X_i)$ be a consistent estimator of $m_{P_X}$ such that $\mathbb{E}[\|\hat{m}_{P_X} - m_{P_X}\|_{\mathcal{H}_X}] = O(n^{-2b})$ and $\mathbb{E}[\sum_{i=1}^{n} \alpha_i^2] = O(n^{-2c})$ for some $c \geq b > 0$ as $n \to \infty$. Assume that $\mathbb{E}_{Y \sim p(\cdot|X)}[k_Y(\cdot, Y)]$ is an element of $\mathcal{H}_X \otimes \mathcal{H}_X$ as a function of $(x, \tilde{x})$. Then $m_{P_Y}$ given by Eq. (3) satisfies

$$\mathbb{E}[\|\hat{m}_{P_Y} - m_{P_Y}\|_{\mathcal{H}_Y}^2] = O(n^{-2b}), \quad (n \to \infty).$$
The function \( E_{Y \sim p(\cdot|z), \tilde{Y} \sim p(\cdot|\tilde{z})} [k_Y(Y, \tilde{Y})] \) can be interpreted as a real-valued regression function over \( \mathbb{R}^{2d} \). The assumption in Theorem 1 that \( E_{Y \sim p(\cdot|z), \tilde{Y} \sim p(\cdot|\tilde{z})} [k_Y(Y, \tilde{Y})] \in \mathcal{H}_X \otimes \mathcal{H}_X \) is standard in the learning theory literature (e.g. Caponnetto and Vito, 2007). Note, however, that the assumption may be weakened by assuming that \( E_{Y \sim p(\cdot|z), \tilde{Y} \sim p(\cdot|\tilde{z})} [k_Y(Y, \tilde{Y})] \) lies in a certain smoothness space such as a Sobolev or Besov space, and considering the approximation errors that arise when we instead use functions in the RKHS \( \mathcal{H}_X \otimes \mathcal{H}_X \), as discussed by Eberts and Steinwart (2013). We will not delve further into the assumption in this paper.

The convergence of the sum of squared weights \( \mathbb{E}[\sum_{i=1}^{n} \alpha_i^2] \) of a kernel mean estimator, which appears in Theorem 1, has not been considered in the literature. Obviously we have \( \mathbb{E}[\sum_{i=1}^{n} \alpha_i^2] = O(n^{-1}) \), i.e. \( c = 1/2 \), when the weights are uniformly distributed \( \alpha_i = 1/n \). We can show as in Proposition 1 that the convergence of \( \mathbb{E}[\sum_{i=1}^{n} \alpha_i^2] \) is also guaranteed when the weights are given by the kernel mean estimator of a conditional probability (Song et al., 2009), which is a basis for the kernel Bayes’ rule (Fukumizu et al., 2011). For details, see Appendix B.

**Proposition 1.** Let \((X_1, Y_1), \ldots, (X_n, Y_n)\) be an i.i.d. sample, and \( \alpha \in \mathbb{R}^n \) be the weights of the kernel mean estimator of a conditional probability (Eq. (2)) with regularization constant \( \varepsilon_n = n^{-(1-2c)} \) for some \( 0 < c < 1/2 \). Then we have

\[
\mathbb{E}[\sum_{i=1}^{n} \alpha_i^2] = O(n^{-2c}), \quad (n \to \infty).
\]

The computation of Eq. (3) incorporates the model expressed in the conditional probability \( p(y|x) \) into the setting of RKHS embeddings, and to our knowledge this is the first such hybrid approach, enabling us to combine explicit probabilistic models with other nonparametric RKHS embedding methods. As an application, \( p(y|x) \) may be the state-transition model of an SSM for the robot localization problem, and \( p(y|x) \) then encodes the underlying dynamics, as derived from robot motion models (Thrun et al., 2005).

### 4 Kernel Monte Carlo Filter

The main objective in this paper is to propose a filtering method for part-nonparametric SSMs (Figure 1). In making a filter for this setting, there are training and test phases. In the training phase, we assume that training data \( \{(X_i, Z_i)\}_{i=1}^{n} \) for the unknown observation model is available, where \((X_i, Z_i)\) is a pair of state and observation, while in the test phase the posterior distributions are estimated based solely on test observation sequences \((z_1, z_2, \ldots)\). At first sight, the assumption
that the training sample is available may appear to be somewhat strong. For the part-nonparametric SSM, however, the observation model lacks by definition an explicit parametric form, and training data are a necessary requirement. In the examples of vision-based mobile robot localization and location estimation based on signal-strengths, which are mentioned in Section 1, it is possible to measure the location but very expensive. Therefore, one wishes to estimate the location from other sensor information with a limited amount of training data.

Let $p(x_t|x_{t-1}, u_t)$ be the state-transition model from which sampling is possible. We include here the control $u_t$ into the state-transition model to explicitly describe the overall algorithm, but omit it from the expressions for probabilities and kernel means for notational brevity. Let $(z_1, z_2, \ldots)$ denote the observation sequence in the test phase, with the abbreviation $z_{1:t} := (z_1, \ldots, z_t)$. The objective of filtering is to estimate the posterior distribution $p(x_t|z_{1:t})$ over states given the sequence of observations.

Let $k_X$ and $k_Z$ be kernels on the space of state $X$ and observation $Z$, respectively, and $\mathcal{H}_X$ and $\mathcal{H}_Z$ be the respective RKHSs. Our aim is to directly estimate the corresponding kernel mean of the posterior distribution

$$m_{x_t|z_{1:t}} := \int k_X(\cdot, x_t)p(x_t|z_{1:t})dx_t.$$ 

As in general filtering methods for SSMs, the proposed KMCF algorithm iterates prediction and correction steps.

**Prediction Step.** In the prediction step, the kernel mean of the predictive distribution $m_{x_t|z_{1:t-1}} := \int k_X(\cdot, x_t)p(x_t|z_{1:t-1})dx_t$ is estimated based on the kernel mean of the previous posterior distribution and the state-transition model. We will use the estimator for the kernel mean of a predictive distribution described in Section 3. Assume that the kernel mean of the previous posterior distribution has been already estimated as\(^2\)

$$\hat{m}_{x_{t-1}|z_{1:t-1}} = \sum_{i=1}^{n} \alpha_i^{(t-1)} k_X(\cdot, X_i),$$

where $\alpha_1^{(t-1)}, \ldots, \alpha_n^{(t-1)} \in \mathbb{R}$. Then with a given control input $u_t$ we generate samples using the state-transition model

$$X_{i}^{(t)} \sim p(\cdot|X_i, u_t) \quad i = 1, \ldots, n,$$

\(^2\)The sample points $X_1, \ldots, X_n$ appearing in the posterior distribution denote the training data for kernel Bayes’ rule, which is used in the correction step.
and estimate \( m_{x_t|z_{1:t-1}} \) by

\[
\hat{m}_{x_t|z_{1:t-1}}(x_i) = \sum_{i=1}^{n} \alpha_i^{(t)} k(\cdot, x_i),
\]

(4)

The consistency of the estimated kernel mean follows from Theorem 1, assuming that \( \hat{m}_{x_{t-1}|z_{1:t-1}} \) satisfies the conditions in Theorem 1.

One may consider resampling before computing the predictive distribution, as for standard particle methods (Doucet et al., 2001). However, we cannot straightforwardly employ such a strategy since the weights \( \alpha_i^{(t-1)} \) may include negative values. We have also observed in a preliminary experiment that resampling after truncating the negative weights gives worse performance. Note that resampling is done in particle methods for avoiding degeneracy of weights that may occur after iterating many filtering steps. In the RKHS embedding approach, however, such degeneracy of weights does not occur and therefore we do not need a resampling step.

**Correction Step.** Let \( z_t \) be a new observation. In the correction step, the kernel mean of the posterior distribution \( m_{x_t|z_{1:t}} := \int k(\cdot, x_t)p(x_t|z_{1:t})dx_t \) is estimated by Kernel Bayes’ rule (KBR) (Fukumizu et al., 2011). Given an observation, training data representing the likelihood, and a consistent estimator of the kernel mean of prior, KBR gives a consistent estimator of the kernel mean of the posterior distribution. Here we regard the kernel mean estimated in the prediction step (Eq. (4)) as the prior, and the training data \( \{(X_i, Z_i)\}_{i=1}^{n} \) as representing the likelihood.

Let \( G_X := (k_X(X_i, X_j)) \) and \( G_Z := (k_Z(Z_i, Z_j)) \in \mathbb{R}^{n \times n} \) denote the Gram matrices, and

\[
\mathbf{m}_{x_t|z_{1:t-1}} := (\hat{m}_{x_t|z_{1:t-1}}(X_i))_{i=1}^{n} \in \mathbb{R}^{n},
\]

(5)

\[
k_Z(z_t) := (k(z_t, Z_i))_{i=1}^{n} \in \mathbb{R}^{n}.
\]

(6)

Then using KBR the kernel mean of the posterior distribution \( m_{x_t|z_{1:t}} \) is estimated by

\[
\hat{m}_{x_t|z_{1:t}} = \sum_{i=1}^{n} \alpha_i^{(t)} k(\cdot, x_i),
\]

(7)

\[
\alpha^{(t)} = \Lambda G_Z((\Lambda G_Z)^2 + \delta_n I_n)^{-1} \Lambda k_Z(z_t),
\]

(8)

\[
\Lambda = \text{diag}((G_X + n\varepsilon_n I_n)^{-1} \mathbf{m}_{x_t|z_{1:t-1}}),
\]

(9)

where \( \text{diag}(\cdot) \) is a diagonal matrix and \( \varepsilon_n, \delta_n > 0 \) are regularization coefficients of KBR.
Algorithm Overview. The procedure of the proposed filter is summarized in Algorithm 1 where \( p_{\text{init}} \) denotes a probability distribution on the initial state. Note that the matrix inverse \( (G_X + \varepsilon_n I_n)^{-1} \) (Eq. (9)) is computed only once before the test phase, since it involves only the training data \( X_1, \ldots, X_n \). The most computationally demanding step is the matrix inverse \( ((\Lambda G_Z)^2 + \delta_n I_n)^{-1} \) (Eq. (8)) of KBR, whose computational cost is of \( O(n^3) \) if naively computed. The cost, however, can be reduced to \( O(nr^2) \) by approximating \( G_Z \) with a low rank matrix of rank \( r \ll n \) using a low rank approximation method such as incomplete Cholesky decomposition (Fine and Scheinberg, 2001) and then applying the Woodbury identity.

The performance of KMCF depends on the choice of kernels \( k_X \) and \( k_Z \) (or parameters such as bandwidth parameters in Gaussian kernels), and the regularization coefficients \( \varepsilon_n \) and \( \delta_n \) of KBR. These parameters can be selected by cross-validation with training data. We can employ the heuristics for reducing the search space in (Fukumizu et al., 2011). For example, consider the case where Gaussian kernels are used for states and observations. We can set the regularization constants to \( \varepsilon_n = c_1 n^{-1/3} \) and \( \delta_n = c_1 n^{-4/27} \), and the Gaussian kernel parameters to \( c_2 \sigma_X \) and \( c_2 \sigma_Z \), where \( \sigma_X \) and \( \sigma_Z \) are the median of the pairwise distances among training data. In this way, only two parameters \( c_1 \) and \( c_2 \) remain to be tuned. The decay rate of the regularization constants is based on the convergence theorem of KBR (Fukumizu et al., 2011).

State Estimation. We discuss here point estimation of the state from the estimated kernel mean \( \hat{m}_{x,t|z_{1:t}} = \sum_{i=1}^{n} \alpha_i(t) k_X(\cdot, X_i) \). A natural way of point estimation is to compute the posterior mean of the state. Let \( \mathcal{X} = \mathbb{R}^d \) and assume that the functions \( f_k : \mathbb{R}^d \rightarrow \mathbb{R} \) defined by \( f_k(x) = x_k \) satisfy \( f_k \in \mathcal{H}_X \) for \( k = 1, \ldots, d \). By the reproducing property of the expectation, we can then estimate the posterior mean of the \( k \)-th coordinate by \( \langle f_k, \hat{m}_{x,t|z_{1:t}} \rangle_{\mathcal{H}_X} = \sum_{i=1}^{n} \alpha_i(t) f_k(X_i) \).

Therefore the posterior mean itself can be estimated by \( \sum_{i=1}^{n} \alpha_i(t) X_i \). Another option for point estimation is to compute the pre-image of the kernel mean, that is arg min\( x \in \mathcal{X} \) \( \| \hat{m}_{x,t|z_{1:t}} - k_X(\cdot, x) \|_{\mathcal{H}_X} \) (Song et al., 2009; Fukumizu et al., 2011). However, if the posterior distribution is highly multimodal, the above estimation strategies may fail. In this case, we can instead use another heuristic to use the sample point with maximum weight as a state estimate.

1Note that since polynomial functions are not included in the RKHS defined by a Gaussian kernel, this posterior mean estimation can be considered heuristics when we use Gaussian kernels.
Algorithm 1 Kernel Monte Carlo Filter

1: **Input:** training data \( \{(X_i, Z_i)\}_{i=1}^n \), test observations \( \{z_j\}_{j=1}^T \), control inputs \( \{u_j\}_{j=1}^T \).
2: set \( \alpha_i^{(0)} = 1/n \), \( i = 1, \ldots, n \).
3: for \( t = 1 \) to \( T \) do
4:  if \( t = 1 \) then
5:  generate \( X_i^{(1)} \sim p_{\text{init}}, \ i = 1, \ldots, n \).
6:  else
7:  generate \( X_i^{(t)} \sim p(\cdot|X_i, u_t), \ i = 1, \ldots, n \).
8:  end if
9:  calculate \( m_{x_t|z_{1:t-1}} \) (Eqs. (4)-(5))
10: observe \( z_t \) and calculate \( k_{X} (z_t) \) (Eq. (5))
11: calculate \( \alpha(t) \in \mathbb{R}^n \) (Eqs. (8)-(9)).
12: end for
13: **Output:** kernel means of the posterior distributions \( \hat{m}_{x_t|z_{1:t}} = \sum_{i=1}^n \alpha_i^{(t)} k_{X}(\cdot, X_i), \ t = 1, \ldots, T \).

5 Related Work

Fully nonparametric SSMs also have been considered, where not only the observation model but the state-transition model are learned nonparametrically with data. Song et al. (2009) and Fukumizu et al. (2011) proposed filtering methods based on RKHS embeddings. Ko and Fox (2009) and Deisenroth et al. (2009) use Gaussian Processes for learning the state-transition model as well as the observation model. These methods are applied to situations where the dynamics of hidden states are not easily given in explicit forms but data is available. Recently Frigola et al. (2013) used Gaussian processes for learning a SSM based only on a sequence of observations, but they assumed a parametric form of the observation model, hence considered a different setting from the part-nonparametric SSMs (see also the references therein).

Note also that while there are particle filters designed for “intractable” observation models (Rossi and Vila [2009], Jasra et al. [2012]), the settings considered in these works are also different from the part-nonparametric SSMs, since they assume that the observation model is known and sampling is possible. In principle, filtering methods for part-nonparametric SSMs including ours are all applicable for such SSMs, by generating training data using the observation model. However, comparison under this setting is out the scope of the present paper.
6 Experiments

In this section we compare the proposed kernel Monte Carlo filter with the following existing methods.

**k-NN Particle Filter** \cite{Vlassis02}. This combines the \( k \)-nearest neighbor (\( k \)-NN) approach with the particle filter. The authors applied this method to the vision-based mobile robot localization problem. To avoid the curse of dimensionality of high-dimensional vision data, they estimate the observation model \( p(z_t|x_t) \) by alternatively estimating its inverse conditional probability \( p(x_t|z_t) \) with a mixture computed by the \( k \)-NN approach. Due to this approximation, however, this method may cause significant bias errors in posterior estimation.

**GP Particle Filter** \cite{Ferris06}. This is a particle filter that uses Gaussian process regression (GP) for learning the observation model from data. The GP particle filter was proposed to deal with location estimation problems with signal-strength observations. Because of the use of GP regression, the additive-Gaussian noise assumption for observation noise is needed for good performance. Additionally, Gaussian processes are learned for each dimension of observation, and thus application to a high-dimensional observation is computationally intractable.

The above \( k \)-NN and GP particle filters are tailored for part-nonparametric SSMs: they assume that simulation with the state-transition model is possible and training data for the observation model are available.

**KBR Filter** \cite{Fukumizu11}. The KBR filter is a filtering algorithm for fully-nonparametric SSMs, where the state-transition model is also to be learned from data. It combines KBR with Kernel sum rule \cite{Song09}, which learns the state-transition model from state-transition examples. Note that existing papers \cite{Song09, Fukumizu11} already experimentally compared the kernel-based nonparametric filters with linear and nonlinear Kalman filters (extended and unscented). We therefore use the KBR filter as a baseline for our experiments instead of the Kalman filters.

In the following we fix the number of particles in the GP and \( k \)-NN particle filters to 5000, since in preliminary experiments we found this is large enough compared to the number of training data for the observation models, and did not observe any improvement with a larger number of particles. We also fix the number of training data for the state-transition models needed for the KBR filter to 1000. We use an open-source code for Gaussian process regression\footnote{http://www.gaussianprocess.org/gpml/code/matlab/doc/}, and therefore omit comparisons on computational time with the GP particle filter.

We evaluate the performance by the root mean square errors (RMSE) of point estimation for states. We also use the RMSE values as evaluation criteria for cross-
validation in the following experiments.

6.1 Synthetic Data

We use the following SSMs defined over $X = Z = \mathbb{R}$ to generate synthetic data, where $x_t, z_t,$ and $u_t$ denote states, observations, and control signals, respectively.

SSM1.

$$x_1 = v_1, \quad v_1 \sim \mathcal{N}(0, 1/(1 - 0.9^2)).$$
$$x_t = 0.9x_{t-1} + 0.5u_t + 0.5v_t, \quad v_t \sim \mathcal{N}(0, 1).$$
$$z_t = x_t + w_t, \quad w_t \sim \mathcal{N}(0, 1).$$

SSM2.

$$x_1 = v_1, \quad v_1 \sim \mathcal{N}(0, 1/(1 - 0.9^2)).$$
$$x_t = 0.9x_{t-1} + 0.5u_t + 0.5v_t, \quad v_t \sim \mathcal{N}(0, 1).$$
$$z_t = 0.5 \exp(x_t/2)w_t, \quad w_t \sim \mathcal{N}(0, 1).$$

SSM3.

$$x_1 = v_1, \quad v_1 \sim \text{uniform}([-3, 3]),$$
$$a_t = x_{t-1} + u_t + 0.3v_t, \quad v_t \sim \mathcal{N}(0, 1),$$
if $|a_t| \leq 3$ : $x_t = a_t$, else : $x_t = -3$.
$$b_t = x_t + 0.5w_t, \quad w_t \sim \mathcal{N}(0, 1),$$
if $|b_t| \leq 3$ : $z_t = b_t$, else : $z_t = b_t - 6b_t/|b_t|.$

SSM1 is a linear Gaussian model. SSM2 uses the same state-transition model as SSM1, while the observation model is nonlinear and the noise is multiplicative. In SSM3 the states and the observations jump near the edges of the interval $[-3, 3]$, hence there is strong nonlinearity in the state and observation processes. Note that SSM2 and SSM3 violate the additive Gaussian noise assumption needed for the GP particle filter.

For each model we randomly generated the control signals from the normal distribution $u_t \overset{i.i.d.}{\sim} \mathcal{N}(0, 1)$. Training data of the observation models $(X_i, Z_i)_{i=1}^n$ (and of the state-transition models for the KBR filter) were generated by running the models sequentially. We set the length of a test data sequence to 100. We used Gaussian kernels for each of $X$ and $Z$, and also for control inputs for the KBR filter. By cross-validation we selected the parameters of each filter, dividing the
training data into two sequences. The hyper-parameters of the Gaussian processes of the GP particle filter were learned by maximizing the marginal likelihood on training data. We ran the experiment 10 times for each of different size of training data. Each filter estimated ground-truth states by posterior means.

**Results.** The results are shown in Figure 2. For SSM1 the GP particle filter performs best, but this is not surprising since the observation noise is additive Gaussian. The proposed KMCF outperforms other methods for SSM2 and SSM3, where nonlinearity of the models is stronger than for SSM1. We also conducted a paired sign-test for the result (see Appendix C). KMCF performs better than competing approaches at 5% significance level for SSM2 at sample size 400. For SSM3, KMCF outperforms competing methods at 5% significance level in all cases except when compared to the $k$-NN particle filter at sample size 100. Computation time of KMCF is competitive with that of the KBR filter (results for SSM1 and SSM3 are omitted since they were almost identical).

### 6.2 Vision-based mobile robot localization

As a real application we conduct an experiment on the vision-based mobile robot localization problem in indoor environment. In this experiment, both training and test data are taken from the COLD database (Pronobis and Caputo, 2009). We used three similar sequences consisting of position-image pairs, two of them for training and the rest for test. The state-space is $\mathcal{X} = \mathbb{R}^2 \times [0, 2\pi]$, consisting of two-dimensional location and orientation of the robot. The space of observations $\mathcal{Z}$ consists of vision images. As will be explained later, the dimension of the feature vectors of images in this experiment is 4200. Since it is computationally hard to apply GP regression to such high-dimensional output vectors, we will not compare the performance with the GP particle filter in this experiment.

**SSM.** The unknown observation model is to be learned nonparametrically from training data consisting of position-image pairs. We use the *odometry motion model* as a state-transition model, which is standard in robotics (Thrun et al., 2005). Briefly, to estimate the next state, the model adds the relative difference of positions computed by odometry data to the current state and then Gaussian noise. We fix the variance of the Gaussian noise to the small value of 0.1. We use the uniform distribution over training samples as an initial distribution on states.

**Kernels.** As a kernel on the images, we used the spatial pyramid match kernel (Lazebnik et al., 2006), which constructs feature vectors based on the SIFT descriptors. In this experiment, we used the parametrization that gives 4200-dimensional feature vectors as suggested by Lazebnik et al. (2006). Gaussian RBF kernels are

---

5 Freiburg, Part A Path 1, cloudy. The time resolution is set to 0.44 images per second in this experiment.
Figure 2: Results of synthetic data experiments. (KMC): proposed method. (KBR): KBR filter. (NN): $k$-NN particle filter. (GP): GP particle filter.

defined on the state-space $\mathcal{X}$ for the kernel-based filters,\footnote{We projected the orientation of a robot in $[0, 2\pi]$ onto the unit circle in $\mathbb{R}^2$ and defined Gaussian kernel on it.} and also on control inputs (relative change of positions computed by odometry) for the KBR filter.

The parameters for each filter were determined by two-fold cross-validation using the two sequences of training data. As a baseline, we also performed a naive method (NAI) that estimates a ground-truth state by the state in training data that has the closest observation to the test observation. We ran the experiment 10 times for each of different size of training data. In preliminary experiments, we found that state estimation with posterior means did not work for any filters because of highly multimodal nature of the posteriors in this problem. Therefore we used the sample point with maximum weight as a state estimate for the kernel-based filters.\footnote{We found that the pre-image computation \cite{song2009,fukumizu2011} also did not}
$k$-NN particle filter estimated the ground-truth as a particle with maximum weight.

Results. The results are shown in Figure 3. The proposed KMCF significantly outperforms other methods. The computation time of KMCF is competitive with that of the KBR filter. The result shows the strength of the proposed method in estimation accuracy in comparison with the $k$-NN particle filter, which was originally proposed for this vision-based mobile robot localization problem. We can also see by comparing with the KBR filter that KMCF effectively incorporates the knowledge expressed in the form of the state-transition model into the RKHS embedding approach.

7 Conclusion

We have presented a filtering method for part-nonparametric SSMs, where the observation model is to be learned nonparametrically with training data. The proposed method is based on RKHS embeddings, and we theoretically justify the use of Monte Carlo simulation to compute predictive distributions within the RKHS framework. We evaluated the performance of KMCF on synthetic and real problems, where we confirmed that it outperforms competing approaches where nonlinear dependencies are present. In this paper, we have assumed that there is no parameter in the state-transition model. One interesting direction for future work is to consider parameter learning for the state-transition model with RKHS embeddings.
Appendix A. Proof of Theorem 1.

Proof. Let $X_n = (X_1, \ldots, X_n)$ and $A_n = (\alpha_1, \ldots, \alpha_n)$. Recall $\hat{m}_{P_Y} = \sum_{i=1}^n \alpha_i k_Y(\cdot, Y_i), Y_i \sim p(\cdot|X_i), i = 1, \ldots, n$, and $m_{P_Y} = \mathbb{E}_{Y \sim P_Y}[k_Y(\cdot, Y)]$. Then

\begin{align*}
\mathbb{E}[\|\hat{m}_{P_Y} - m_{P_Y}\|^2_{H_Y}|X_n, A_n] &= \mathbb{E}[\langle \hat{m}_{P_Y}, \hat{m}_{P_Y}\rangle_{\mathcal{H}_{Y'}} - 2 \langle \hat{m}_{P_Y}, m_{P_Y}\rangle_{\mathcal{H}_{Y'}} + \langle m_{P_Y}, m_{P_Y}\rangle_{\mathcal{H}_{Y'}}|X_n, A_n] \\
&= \sum_{i,j=1}^n \alpha_i \alpha_j \mathbb{E}_{Y_i \sim p(\cdot|X_i), Y_j \sim p(\cdot|X_j)}[k_Y(Y_i, Y_j)] \\
&\quad - 2 \sum_{i=1}^n \alpha_i \mathbb{E}_{\tilde{Y} \sim p(\cdot|X_i)} \langle k_Y(\cdot, Y_i), m_{P_Y} \rangle + \mathbb{E}_{Y, \tilde{Y} \sim P_Y}[k_Y(Y, \tilde{Y})] \\
&= \sum_{i,j=1}^n \alpha_i \alpha_j \mathbb{E}_{Y_i \sim p(\cdot|X_i), Y_j \sim p(\cdot|X_j)}[k_Y(Y_i, Y_j)] \\
&\quad - 2 \sum_{i=1}^n \alpha_i \mathbb{E}_{Y \sim P_Y, Y_i \sim p(\cdot|X_i)}[k_Y(Y, Y_i)] + \mathbb{E}_{Y, \tilde{Y} \sim P_Y}[k_Y(Y, \tilde{Y})] \\
&= \sum_{i \neq j} \alpha_i \alpha_j \mathbb{E}_{Y \sim p(\cdot|X_i), \tilde{Y} \sim p(\cdot|X_j)}[k_Y(Y, \tilde{Y})] + \sum_{i=1}^n \alpha_i^2 \mathbb{E}_{\tilde{Y} \sim p(\cdot|X_i)}[k_Y(Y, \tilde{Y})] \\
&\quad - 2 \sum_{i=1}^n \alpha_i \mathbb{E}_{Y \sim P_Y, \tilde{Y} \sim p(\cdot|X_i)}[k_Y(Y, \tilde{Y})] + \mathbb{E}_{Y, \tilde{Y} \sim P_Y}[k_Y(Y, \tilde{Y})]. \quad (A1)
\end{align*}
Define the function $\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ by $\theta(x, \tilde{x}) := \mathbb{E}_{Y \sim p(\cdot | x), \tilde{Y} \sim p(\cdot | \tilde{x})}[k_Y(Y, \tilde{Y})] = \int \int k_Y(y, \tilde{y})p(y|x)p(\tilde{y}|\tilde{x})dyd\tilde{y}$. Recalling $P_Y \equiv \int p(\cdot | x)dP_X(x)$, we have

$$
\mathbb{E}_{Y \sim P_Y, \tilde{Y} \sim p(\cdot | X_i)}[k_Y(Y, \tilde{Y})] = \int \int k_Y(y, \tilde{y})dP_Y(y)p(\tilde{y}|X_i)d\tilde{y} = \int \int \int k_Y(y, \tilde{y})p(y|x)p(\tilde{y}|X_i)dyd\tilde{y}dP_X(x) = \int \theta(x, X_i)dP_X(x) = \mathbb{E}_{X \sim P_X}[\theta(X, X_i)]
$$

$$
\mathbb{E}_{Y, \tilde{Y} \sim P_Y}[k_Y(Y, \tilde{Y})] = \int \int k_Y(y, \tilde{y})dP_Y(y)dP_Y(\tilde{y}) = \int \int \int k_Y(y, \tilde{y})p(y|x)p(\tilde{y}|\tilde{x})dyd\tilde{y}dP_X(x)dP_X(\tilde{x}) = \int \int \theta(x, \tilde{x})dP_X(x)dP_X(\tilde{x}) = \mathbb{E}_{X, \tilde{X} \sim P_X}[\theta(X, \tilde{X})]
$$

Then the above expression \((A1)-(A2)\) is further equal to

$$
\sum_{i=1}^{n} \alpha_i^2(\mathbb{E}_{Y \sim p(\cdot | X_i)}[k_Y(Y, Y)] - \mathbb{E}_{Y, \tilde{Y} \sim p(\cdot | X_i)}[k_Y(Y, \tilde{Y})]) + \sum_{i,j=1}^{n} \alpha_i \alpha_j \theta(X_i, X_j) - 2 \sum_{i=1}^{n} \alpha_i \mathbb{E}_{X \sim P_X}[\theta(X, X_i)] + \mathbb{E}_{X, \tilde{X} \sim P_X}[\theta(X, \tilde{X})]
$$

$$
= \sum_{i=1}^{n} \alpha_i^2(\mathbb{E}_{Y \sim p(\cdot | X_i)}[k_Y(Y, Y)] - \mathbb{E}_{Y, \tilde{Y} \sim p(\cdot | X_i)}[k_Y(Y, \tilde{Y})]) + \langle \hat{m}_{P_X} \otimes \hat{m}_{P_X}, \theta \rangle_{\mathcal{H}_X \otimes \mathcal{H}_X} - 2 \langle \hat{m}_{P_X} \otimes m_{P_X}, \theta \rangle_{\mathcal{H}_X \otimes \mathcal{H}_X} + \langle m_{P_X} \otimes m_{P_X}, \theta \rangle_{\mathcal{H}_X \otimes \mathcal{H}_X}
$$

$$
= \sum_{i=1}^{n} \alpha_i^2(\mathbb{E}_{Y \sim p(\cdot | X_i)}[k_Y(Y, Y)] - \mathbb{E}_{Y, \tilde{Y} \sim p(\cdot | X_i)}[k_Y(Y, \tilde{Y})]) + \langle \hat{m}_{P_X} - m_{P_X} \otimes \hat{m}_{P_X}, \theta \rangle_{\mathcal{H}_X \otimes \mathcal{H}_X},
$$

where we used the assumption $\theta \in \mathcal{H}_X \otimes \mathcal{H}_X$ in the second equality. Here $f \otimes g$ for $f, g \in \mathcal{H}_X$ denotes the tensor product of $f$ and $g$. By the boundedness of $k_Y$, the expectation of the first term is $O(n^{-2\epsilon})$. By the Cauchy-Schwartz inequality, the expectation of the second term is upper bounded.
by
\[ E \left[ \langle (\hat{m}_{P_X} - m_{P_X}) \otimes (\hat{m}_{P_X} - m_{P_X}), \theta \rangle_{H_X \otimes H_X} \right] \leq E \left[ \| \hat{m}_{P_X} - m_{P_X} \|_{H_X}^2 \right] \| \theta \|_{H_X \otimes H_X} \]
which is of \( O(n^{-2b}) \). This completes the proof. 

Appendix B. Proof of Proposition 1.

We first review the kernel mean estimator of a conditional probability (Song et al., 2009). Let \( (X, \mathcal{B}_X) \) and \( (Y, \mathcal{B}_Y) \) be measurable spaces, and \( P \) be a joint distribution on \( X \times Y \) with density \( p(x|y)p(y) \) for \( x \in X, y \in Y \). Let \( k_X \) and \( k_Y \) be bounded kernels on \( X \) and \( Y \), respectively, and \( H_X \) and \( H_Y \) be the respective RKHSs. Let \( (X_1, Y_1), \ldots, (X_n, Y_n) \) be an i.i.d. sample from \( P \) and let \( y \in Y \) be fixed. Then the following is a consistent estimator for the kernel mean of the conditional probability \( m_{P_X|y} := \int k_X(\cdot, x)p(x|y)dx \)

\[ \hat{m}_{P_X|y} = \sum_{i=1}^n \alpha_i k_X(\cdot, X_i), \quad \alpha = (G_Y + n\varepsilon_n I_n)^{-1}k_Y(y) \in \mathbb{R}^n, \quad (B3) \]

where \( G_Y = (k_Y(Y_i, Y_j)) \in \mathbb{R}^{n\times n} \) is a Gram matrix, \( \varepsilon_n > 0 \) is a regularization constant, and \( k_Y(y) = (k_Y(y, Y_i))_{i=1}^n \in \mathbb{R}^n \). It is known that for sufficiently slowly decaying regularization constants \( \varepsilon_n \to 0 \) it holds that

\[ \| \hat{m}_{P_X} - m_{P_X} \|_{H_X} \to 0 \quad (n \to \infty) \]

under some smoothness assumptions (Song et al., 2009).

To prove Proposition 1, we need the following lemma.

**Lemma B1.** The solution to the minimization problem

\[ \min_{\gamma \in \mathbb{R}^n} \| k_Y(\cdot, y) - \sum_{i=1}^n \gamma_i k_Y(\cdot, Y_i) \|_{H_Y}^2 \]

is given by \( G_Y\gamma_* = k_Y(y) \), and the minimum value is

\[ \min_{\gamma \in \mathbb{R}^n} \| k_Y(\cdot, y) - \sum_{i=1}^n \gamma_i k_Y(\cdot, Y_i) \|_{H_Y}^2 = k_Y(y, y) - \gamma_*^T G_Y \gamma_* \]

**Proof.** Straightforward calculation. 

Proposition 1 in the main text is a direct consequence of the following proposition.
**Proposition B1.** Let $\alpha \in \mathbb{R}^n$ be given by Eq. (B3). Then we have

$$n\alpha^T \alpha \leq \frac{k_Y(y, y)}{\varepsilon_n}$$

**Proof.** From Lemma B1 there is $\gamma \in \mathbb{R}^n$ such that $k_Y(y) = G_Y \gamma$ and $\gamma^T G_Y \gamma \leq k_Y(y, y)$. We have

$$\alpha = (G_Y + n\varepsilon_n I_n)^{-1} G_Y \gamma,$$

and thus

$$n\alpha^{(n)T} \alpha = n\gamma^T (G_Y + n\varepsilon_n I_n)^{-2} G_Y \gamma$$

$$\leq n\gamma^T (G_Y + n\varepsilon_n I_n)^{-1} G_Y \gamma \leq \frac{1}{\varepsilon_n} \gamma^T G_Y \gamma \leq \frac{k_Y(y, y)}{\varepsilon_n},$$

where the first inequality in the second line uses $(G_Y + n\varepsilon_n I_n)^{-1} G_Y \leq I_n$ and the second one uses $n(G_Y + n\varepsilon_n)^{-1} \leq (1/\varepsilon_n) I_n$.

### Appendix C. Results of the Synthetic Data Experiments (SSM2 and SSM3)

We conducted two-sided paired sign tests for the results of the synthetic data experiments for SSM2 and SSM3 in Section 6. The p-values are shown in Table 1 and Table 2. Each test was done for a fixed training data size.

From Table 1 (SSM2) it can be seen that the proposed kernel Monte Carlo filter performed best when the sample size is $n = 400$ at 5% significance level. KMCF outperformed the KBR filter at 5% significance level in all cases except when $n = 100$. Table 2 (SSM3) shows that KMCF performed best at 5% significance level in all cases except when compared to the $k$-NN particle filter at sample size $n = 100$.

**Table 1: P-values by Two-Sided Paired Sign Test (SSM2)**

| Number of Training Data | 100  | 200  | 300  | 400  |
|-------------------------|------|------|------|------|
| KMCF v.s. KBR filter    | 0.109| 0.021| 0.021| 0.001|
| KMCF v.s. $k$-NN particle filter | 1    | 0.109| 0.109| 0.001|
| KMCF v.s. GP particle filter | 0.021| 0.109| 0.021| 0.021|
Table 2: P-values by Two-Sided Paired Sign Test (SSM3)

| Number of Training Data | 100 | 200 | 300 | 400 |
|-------------------------|-----|-----|-----|-----|
| KMCF v.s. KBR filter    | 0.01| 0.021| 0.001| 0.001|
| KMCF v.s. k-NN particle filter | 0.343| 0.021| 0.001| 0.001|
| KMCF v.s. GP particle filter | 0.021| 0.021| 0.021| 0.021|

References

Caponnetto, A. and Vito, E. D. (2007). Optimal rates for regularized least-squares algorithm. *Found. Comput. Math. J.*, 7(4):331–368.

Chopin, N. (2004). Central limit theorem for sequential monte carlo methods and its application to bayesian inference. *Ann. Statist.*, 32(6):2385–2411.

Deisenroth, M., Huber, M., and Hanebeck, U. (2009). Analytic moment-based Gaussian process filtering. In *ICML*.

Dellaert, F., Burgard, W., Fox, D., and Thrun, S. (1999). Using the condensation algorithm for robust, vision-based mobile robot localization. In *CVPR*.

Doucet, A., Freitas, N. D., and Gordon, N. J., editors (2001). *Sequential Monte Carlo Methods in Practice*. Springer.

Eberts, M. and Steinwart, I. (2013). Optimal regression rates for SVMs using Gaussian kernels. *Electron. J. Stat.*, 7:1–42.

Ferris, B., Hähnel, D., and Fox, D. (2006). Gaussian processes for signal strength-based location estimation. In *Robotics Science and Systems*.

Fine, S. and Scheinberg, K. (2001). Efficient SVM training using low-rank kernel representations. *JMLR*, 2:243–264.

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 *NIPS*.

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

Fukumizu, K., Bach, F., and Jordan, M. I. (2009). Kernel dimension reduction in regression. *Ann. Statist.*, 37(4):1871–1905.
Fukumizu, K., Song, L., and Gretton, A. (2011). Kernel Bayes’ rule. In NIPS, pages 1737–1745.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. JMLR, 13:723–773.

Gretton, A., Fukumizu, K., Teo, C., Song, L., Schoelkopf, B., and Smola, A. (2008). A kernel statistical test of independence. In NIPS.

Grünewälder, S., Lever, G., Baldassarre, L., Pontil, M., and Gretton, A. (2012). Modeling transition dynamics in MDPs with RKHS embeddings. In ICML.

Jasra, A., Singh, S. S., Martin, J. S., and McCoy, E. (2012). Filtering via approximate Bayesian computation. Statistics and Computing, 22:1223–1237.

Julier, S. J. and Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. In Proc. AeroSense: The 11th Intern. Symp. Aerospace/Defence Sensing, Simulation and Controls.

Ko, J. and Fox, D. (2009). GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. Autonomous Robots, 72(1):75–90.

Ladd, A. M., Bekris, K. E., Rudys, A., Marceau, G., Kavraki, L. E., and Wallach, D. S. (2002). Robotics-based location sensing using wireless ethernet. In MOBICOM.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In CVPR.

Nishiyama, Y., Boularias, A., Gretton, A., and Fukumizu, K. (2012). Hilbert space embeddings of POMDPs. In UAI.

Pronobis, A. and Caputo, B. (2009). COLD: COsy Localization Database. IJRR, 28(5):588–594.

Rossi, V. and Vila, J.-P. (2009). Convolutional particle filter for parameter estimation in general state-space models. IEEE Transactions on Aerospace and Electronic Systems, 45:1063–1072.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. (2007). A Hilbert space embedding for distributions. In ALT.

Song, L., Huang, J., Smola, A., and Fukumizu, K. (2009). Hilbert space embeddings of conditional distributions with applications to dynamical systems. In ICML, pages 961–968.
Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010). Hilbert space embeddings and metrics on probability measures. *JMLR*, 11:1517–1561.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press.

Vlassis, N., Terwijn, B., and Kröse, B. (2002). Auxiliary particle filter robot localization from high-dimensional sensor observations. In *ICRA*, pages 7–12.