Prediction under Uncertainty in Sparse Spectrum Gaussian Processes with Applications to Filtering and Control

Yunpeng Pan [1 2] **Xinyan Yan** [1 3] **Evangelos A. Theodorou** [1 2] **Byron Boots** [1 3]

Abstract

Sparse Spectrum Gaussian Processes (SSGPs) are a powerful tool for scaling Gaussian processes (GPs) to large datasets. Existing SSGP algorithms for regression assume deterministic inputs, precluding their use in many real-world robotics and engineering applications where accounting for input uncertainty is crucial. We address this problem by proposing two analytic moment-based approaches with closed-form expressions for SSGP regression with uncertain inputs. Our methods are more general and scalable than their standard GP counterparts, and are naturally applicable to multi-step prediction or uncertainty propagation. We show that efficient algorithms for Bayesian filtering and stochastic model predictive control can use these methods, and we evaluate our algorithms with comparative analyses and both real-world and simulated experiments.

1. Introduction

The problem of *prediction under uncertainty*, appears in many fields of science and engineering that involve sequential prediction including state estimation (Ko & Fox, 2009; Deisenroth et al., 2012), time series prediction (Girard et al., 2003), stochastic process approximation (Archambeau et al., 2007), and planning and control (Deisenroth et al., 2015). In these problems, uncertainty can be found in both the predictive models and the model's inputs. Formally, we are often interested in finding the probability density of a prediction y, given a distribution $p(x)$ and a probabilistic model $p(y|x)$. By marginalization,

$$p(y) = \int p(y|x)p(x)\,\mathrm{d}x. \tag{1}$$

Unfortunately, computing this integral exactly is often intractable. In this paper, we tackle a subfamily of (1) where: 1) the probabilistic model is learned from data and specified by a sparse spectrum representation of a Gaussian process (SSGP); and 2) the input x is normally distributed. We show that analytic expressions of the moments of $p(y)$ can be derived and that these are directly applicable to sequential prediction problems like filtering and control.

1.1. Related work

Gaussian Process (GP) regression with uncertain inputs has been addressed by Candela et al. (2003); Girard et al. (2003), and extended to the multivariate outputs by Kuss (2006). These methods have led to the development of many algorithms in reinforcement learning (Rasmussen & Kuss, 2004; Deisenroth et al., 2015), Bayesian filtering (Ko & Fox, 2009; Deisenroth et al., 2009), and smoothing (Deisenroth et al., 2012). However, these approaches have two major limitations: 1) they are not directly applicable to large datasets, due to the polynomial time complexity for exact inference (Williams & Rasmussen, 2006); and 2) analytic moment expressions, when used, are restricted to squared exponential (SE) kernels (Kuss, 2006) and cannot be generalized to other kernels in a straightforward way.

A common method for approximating large-scale kernel machines is through random Fourier features (Rahimi & Recht, 2007). The key idea is to map the input to a low-dimensional feature space yielding fast linear methods. In the context of GP regression (GPR), this idea leads to the sparse spectrum GPR (SSGPR) algorithm (Lázaro-Gredilla et al., 2010). SSGP has been extended in a number of ways for, e.g. incremental model learning (Gijsberts & Metta, 2013), and large-scale GPR (Dai et al., 2014; Yan et al., 2015). However, to the best of our knowledge, prediction under uncertainty for SSGPs has not been explored. Although there are several alternative approximations to exact GP inference including approximating the posterior distribution using inducing points, *e.g.*, (Snelson & Ghahramani, 2006; Titsias, 2009; Cheng & Boots, 2016), comparing different GP approximations is not the focus of this paper.

[1]Georgia Institute of Technology, Atlanta, Georgia, USA [2]School of Aerospace Engineering [3]School of Interactive Computing. Correspondence to: Yunpeng Pan <ypan37@gatech.edu>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

1.2. Applications

We consider two key problems that are widely encountered in robotics and engineering: Bayesian filtering and stochastic model predictive control.

The goal of Bayesian filtering is to infer a hidden system state through the recursive application of Bayes' rule. Well-known frameworks for Bayesian filtering include unscented Kalman Filtering (UKF), particle filtering (PF), extended Kalman filtering (EKF), and assumed density filtering (ADF). GP-based Bayesian filtering with SE kernels has been developed for these frameworks by (Ko & Fox, 2009; Deisenroth et al., 2009). We extend this work with highly efficient SSGP-based EKF and ADF algorithms.

The goal of stochastic model predictive control (MPC) is to find finite horizon optimal control at each time instant. Due to the high computational cost of GP inference and real-time optimization requirements in MPC, most GP-based control methods (Deisenroth et al., 2015; Pan & Theodorou, 2014; Kupcsik et al., 2014) are restricted to episodic reinforcement learning tasks. To cope with this challenge, we present an SSGP-based MPC algorithm that is fast enough to perform probabilistic trajectory optimization and model adaptation on-the-fly.

1.3. Our contributions

- We propose two approaches to prediction under uncertainty in SSGPs with closed-form expressions for the predictive distribution. Compared to previous GP counterparts, our methods: 1) are more scalable, and 2) can be generalized to any continuous shift-invariant kernels with a Fourier feature representation.
- We demonstrate successful applications of the proposed approaches by presenting scalable algorithms for 1) recursive Bayesian filtering and 2) stochastic model predictive control via probabilistic trajectory optimization.

The rest of the paper is organized as follows. In §2, we give an introduction to SSGPs, which serves as our probabilistic model. Derivation and expressions of the two proposed prediction methods are detailed in §3. Applications to filtering and control, and experimental results are presented in §4 and §5 respectively. Finally §6 concludes the paper.

2. Sparse Spectral Representation of GPs

Consider the task of learning the function $f : \mathbf{R}^d \to \mathbf{R}$, given IID data $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$, with each pair related by

$$y = f(x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_n^2), \tag{2}$$

where ϵ is IID additive Gaussian noise. Gaussian process regression (GPR) is a principled way of performing Bayesian inference in function space, assuming that function f has a prior distribution $f \sim \mathcal{GP}(m, k)$, with mean function $m : \mathbf{R}^d \to \mathbf{R}$ and kernel $k : \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$. Without loss of generality, we assume $m(x) = 0$. Exact GPR is challenging for large datasets due to its $O(n^3)$ time and $O(n^2)$ space complexity (Williams & Rasmussen, 2006), which is a direct consequence of having to store and invert an $n \times n$ Gram matrix.

Random features can be used to form an unbiased approximation of continuous shift-invariant kernel functions, and are proposed as a general mechanism to accelerate large-scale kernel machines (Rahimi & Recht, 2007), via explicitly mapping inputs to low-dimensional feature space. Based on Bochner's theorem, the Fourier transform of a continuous shift-invariant positive definite kernel $k(x, x')$ is a proper probability distribution $p(\omega)$, assuming $k(x, x')$ is properly scaled (Rahimi & Recht, 2007):

$$\begin{aligned} k(x, x') &= \int p(\omega) e^{j\omega^T (x - x')} \, \mathrm{d}\omega \\ &= \mathbf{E}(\phi_\omega(x) \phi_\omega(x')^*), \quad \omega \sim p(\omega), \end{aligned} \tag{3}$$

where $\phi_\omega(x) = e^{j\omega^T x}$, and we can see that $k(x, x')$ only depends on the lag vector separating x and x': $x - x'$. Equation (3) leads to an unbiased finite sample approximation of k: $k(x, x') \approx \frac{1}{m} \sum \phi_{\omega_i}(x) \phi_{\omega_i}(x')^*$, where random frequencies $\{\omega_i\}_{i=1}^m$ are drawn IID from $p(\omega)$. Utilizing the fact that ϕ_ω can be replaced by sinusoidal functions since both $p(\omega)$ and $k(x, x')$ are reals, and concatenating features $\{\phi_{\omega_i}\}_{i=1}^m$ into a succinct vector form, an approximation for $k(x, x')$ is expressed as

$$k(x, x') \approx \phi(x)^T \phi(x'), \ \phi(x) = \begin{bmatrix} \phi^c(x) \\ \phi^s(x) \end{bmatrix}, \tag{4}$$

$$\phi_i^c(x) = \sigma_k \cos(\omega_i^T x), \ \phi_i^s(x) = \sigma_k \sin(\omega_i^T x), \ \omega_i \sim p(\omega),$$

where σ_k is a scaling coefficient. For the commonly used Squared Exponential (SE) kernel: $k(x, x') = \sigma_f^2 \exp(-\frac{1}{2} \|x - x'\|_{\Lambda^{-1}}^2)$, $p(\omega) = \mathcal{N}(0, \Lambda^{-1})$ and $\sigma_k = \frac{\sigma_f}{\sqrt{m}}$, where the coefficient σ_f and the diagonal matrix Λ are the hyperparameters, examples of kernels and corresponding spectral densities can be found in Table 1.

In accordance with this feature map (4), Sparse Spectrum GPs are defined as follows

Definition 1. *Sparse Spectrum GPs (SSGPs) are GPs with kernels defined on the finite-dimensional and randomized feature map ϕ (4):*

$$k(x, x') = \phi(x)^T \phi(x') + \sigma_n^2 \delta(x - x'), \tag{5}$$

where the function δ is the Kronecker delta function.

The second term in (5) accounts for the additive zero mean Gaussian noise in (2), if the goal is to learn the correlation between x and y directly as in our case of learning the probabilistic model $p(y|x)$, instead of learning the latent function f.

Because of the explicit finite-dimensional feature map (4), each SSGP is equivalent to a Gaussian distribution over the weights of features $w \in \mathbf{R}^{2m}$. Assuming that prior distribution of weights w is $\mathcal{N}(0, I)$ [1] and the feature map is fixed, after conditioning on the data $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$, the posterior distribution of w is [2]

$$w \sim \mathcal{N}(\alpha,\ \sigma_n^2 A^{-1}), \qquad (6)$$
$$\alpha = A^{-1}\Phi Y, \quad A = \Phi\Phi^T + \sigma_n^2 I,$$

which can be derived through Bayesian linear regression. In (6), the column vector Y and the matrix Φ are specified by the data $\mathcal{D}$: $Y = \begin{bmatrix} y_1 & \dots & y_n \end{bmatrix}^T$, $\Phi = \begin{bmatrix} \phi(x_1) & \dots & \phi(x_n) \end{bmatrix}$. Consequently, the posterior distribution over the output y in (2) at a test point x is *exactly Gaussian*, in which the posterior variance explicitly captures the model uncertainty in prediction with input x:

$$p(y|x) = \mathcal{N}(\alpha^T\phi(x),\ \sigma_n^2 + \sigma_n^2\|\phi(x)\|^2_{A^{-1}}). \qquad (7)$$

This Bayesian linear regression method for SSGP is proposed in Lázaro-Gredilla et al. (2010). Its time complexity is $O(nm^2 + m^3)$, which is significantly more efficient than standard GPR's $O(n^3)$ when $m \ll n$.

Remark It's worth noting that the methods proposed in this paper are not tied to specific algorithms for SSGP regression such as Bayesian linear regression (Lázaro-Gredilla et al., 2010), but able to account for any SSGP with specified feature weights distribution (6), where posterior α and A can be computed by any means. Variations on A include sparse approximations by a low rank plus diagonal matrix, or iterative solutions by optimization methods like doubly stochastic gradient descent (Dai et al., 2014).

3. Prediction under Uncertainty

Two methods for prediction under uncertainty are presented under two conditions: 1) the uncertain input is normally distributed: $x \sim \mathcal{N}(\mu, \Sigma)$, and 2) probabilistic models are in the form of (7) specified by SSGPs. Despite these conditions, evaluating the integral in (1) is still intractable. In this work, we approximate the true predictive distribution $p(y)$ by a Gaussian distribution with moments that are analytically computed through: 1) exact moment matching, and 2) linearization of posterior mean function. Closed-form expressions for predictive mean, variance, covariance, and input-prediction cross-covariance are derived. We consider multivariate outputs by utilizing conditionally independent scalar models for each output dimension, *i.e.*, assuming for outputs in different dimension y_a and y_b, $p(y_a, y_b|x) = p(y_a|x)p(y_b|x)$. Discussions on this assumption can be found in Appendix §6.1. For notational simplicity, we suppress the dependency of $\phi(x)$ on x, and treat y as a scalar by default.

3.1. Exact moment matching (SSGP-EMM)

We derive the closed-form expressions for exact moments: 1) the predictive mean $\mathbf{E}\, y$, 2) the predictive variance $\mathbf{Var}\, y$ and covariance $\mathbf{Cov}(y_a, y_b)$, which in the multivariate case correspond to the diagonal and off-diagonal entries of the predictive covariance matrix, and 3) the cross-covariance between input and prediction $\mathbf{Cov}(x, y)$.

Using the expressions for SSGP (4), (7), and the law of total expectation, the predictive mean becomes

$$\mathbf{E}\, y = \mathbf{E}\,\mathbf{E}(y|x) = \mathbf{E}\left(\alpha^T\phi\right) = \alpha^T\,\mathbf{E}\begin{bmatrix}\phi^c \\ \phi^s\end{bmatrix}, \qquad (8)$$
$$\mathbf{E}\,\phi_i^c = \sigma_k\,\mathbf{E}\cos(\omega_i^T x), \quad \mathbf{E}\,\phi_i^s = \sigma_k\,\mathbf{E}\sin(\omega_i^T x),$$

where $i = 1, \dots, m$, and in the nested expectation $\mathbf{E}\,\mathbf{E}(y|x)$, the outer expectation is over the input distribution $p(x) = \mathcal{N}(\mu, \Sigma)$, and the inner expectation is over the conditional distribution $p(y|x)$ (7).

By observing (8), we see that the expectation of sinusoids under the Gaussian distribution is the key to computing the predictive mean. Thus, we state the following proposition:

Proposition 1. *The expectation of sinusoids over multivariate Gaussian distributions: $x \sim \mathcal{N}(\mu, \Sigma)$, $x \in \mathbf{R}^d$, i.e., $p(x) = (2\pi)^{-\frac{d}{2}}(\det\Sigma)^{-\frac{1}{2}}\exp(-\frac{1}{2}\|x-\mu\|^2_{\Sigma^{-1}})$, can be computed analytically:*

$$\mathbf{E}\cos(\omega^T x) = \exp(-\frac{1}{2}\|\omega\|^2_\Sigma)\cos(\omega^T\mu),$$
$$\mathbf{E}\sin(\omega^T x) = \exp(-\frac{1}{2}\|\omega\|^2_\Sigma)\sin(\omega^T\mu).$$

To prove it, we invoke Euler's formula to transform the left-hand-side to complex domain, apply identities involving quadratic exponentials, and then convert back to real numbers (see Appendix §3.2 for details). In Proposition 1, the expectations depend on the mean and variance of the input Gaussian distribution. Intuitively, after passing a Gaussian distributed input through a sinusoidal function, the expectation of the output is equal to passing the mean of the input through the sinusoid, and then scaling it by a constant $\exp(-\frac{1}{2}\|\omega\|^2_\Sigma)$, which depends on the variance of the input. Expectations are smaller with larger input variance due to the periodicity of sinusoids.

The exact moments are then derived using Proposition 1. By the law of total variance, the predictive variance is

$$\begin{aligned}\mathbf{Var}\, y &= \mathbf{E}\,\mathbf{Var}(y|x) + \mathbf{Var}\,\mathbf{E}(y|x) \\ &= \sigma_n^2 + \sigma_n^2\,\mathbf{Tr}\left(A^{-1}\Psi\right) + \alpha^T\Psi\alpha - (\mathbf{E}\, y)^2,\end{aligned} \qquad (9)$$

[1] I is the identity matrix with proper size. The prior covariance is identity since $\mathbf{E}\,(f(x)f(x)) = \mathbf{E}\left(\phi(x)^T w w^T \phi(x')\right) = \phi(x)^T\,\mathbf{E}(ww^T)\phi(x')$, and $\mathbf{E}\,(f(x)f(x')) = \phi(x)^T\phi(x')$ (see §2.2 in Rasmussen & Kuss (2004) for details.)

[2] Conditioning on data $\mathcal{D}$ is omitted, *e.g.*, in $w|\mathcal{D}$, for simplicity in notation.

where Ψ is defined as the expectation of the outer product of feature vectors over input distribution $p(x)$. Specifically, we compute Ψ by applying the product-to-sum trigonometric identities:

$$\mathbf{E}\left(\phi\phi^T\right) = \Psi = \begin{bmatrix} \Psi^{cc} & \Psi^{cs} \\ \Psi^{sc} & \Psi^{ss} \end{bmatrix},$$

$$\Psi^{cc}_{ij} = \frac{\sigma_k^2}{2}\left(\mathbf{E}\left(\cos(\omega_i+\omega_j)^T x\right) + \mathbf{E}\left(\cos(\omega_i-\omega_j)^T x\right)\right),$$

$$\Psi^{ss}_{ij} = \frac{\sigma_k^2}{2}\left(\mathbf{E}\left(\cos(\omega_i-\omega_j)^T x\right) - \mathbf{E}\left(\cos(\omega_i+\omega_j)^T x\right)\right),$$

$$\Psi^{cs}_{ij} = \frac{\sigma_k^2}{2}\left(\mathbf{E}\left(\sin(\omega_i+\omega_j)^T x\right) - \mathbf{E}\left(\sin(\omega_i-\omega_j)^T x\right)\right),$$

where $\Psi^{cc}, \Psi^{ss}, \Psi^{cs}$ are $m \times m$ matrices, and $i, j = 1, \ldots, m$, on whose terms Proposition 1 can be directly applied.

Next, we derive the covariance for different output dimensions for multivariate prediction. These correspond to the off-diagonal entries of the predictive covariance matrix. We show that, despite the conditional independence assumption for different outputs given a deterministic input, outputs become coupled with uncertain inputs. Using the law of total covariance, the covariance is

$$\begin{aligned} \mathbf{Cov}(y_a, y_b) &= \mathbf{Cov}\left(\mathbf{E}(y_a|x), \mathbf{E}(y_b|x)\right) \\ &= \mathbf{E}\left(\mathbf{E}(y_a|x), \mathbf{E}(y_b|x)\right) - (\mathbf{E}\, y_a)(\mathbf{E}\, y_b) \\ &= \alpha_a^T \Psi_{ab} \alpha_b - (\alpha_a^T\, \mathbf{E}\, \phi_a)(\alpha_b^T\, \mathbf{E}\, \phi_b), \end{aligned} \quad (10)$$

where matrix Ψ_{ab} is the expectation of the outer product of feature vectors corresponding to different feature maps ϕ_a, ϕ_b for outputs y_a, y_b, computed similarly as in (3.1) with corresponding random frequencies $\{\omega_i\}$, and the scaling coefficient σ_k (4). Vectors α_a and α_b are the corresponding weight vectors for y_a and y_b (7). Compared to the expression for the variance of a single output in (9), the term $\mathbf{E}\left(\mathbf{Cov}(y_a|x)\,\mathbf{Cov}(y_b|x)\right)$ that is included in the law of total covariance is neglected due to the assumption of conditional independence of different outputs (§2), so (10) does not have the corresponding first two terms in (9).

Finally, we compute the cross-covariance between input and each output dimension. Invoking the law of total covariance:

$$\begin{aligned} \mathbf{Cov}(x, y) &= \mathbf{Cov}(x, \mathbf{E}(y|x)) \\ &= \mathbf{E}\left(x\,\mathbf{E}(y|x)\right) - (\mathbf{E}\, x)(\mathbf{E}\, y) \\ &= \Upsilon\alpha - (\mathbf{E}\, y)\mu, \end{aligned} \quad (11)$$

where matrix Υ is the expectation of the outer product of the input x and the feature vector $\phi(x)$ over input distribution $x \sim \mathcal{N}(\mu, \Sigma)$:

$$\mathbf{E}(x\phi^T) = \Upsilon = \begin{bmatrix} \Upsilon_1^c & \ldots & \Upsilon_m^c & \Upsilon_1^s & \ldots & \Upsilon_m^s \end{bmatrix},$$

$$\Upsilon_i^c = \sigma_k\, \mathbf{E}\left(\cos(\omega_i^T x)x\right), \quad \Upsilon_i^s = \sigma_k\, \mathbf{E}\left(\cos(\omega_i^T x)x\right),$$

Kernel	$k(x, x')$	$p(\omega)$
Gaussian	$\exp(-\frac{1}{2}\|x-x'\|^2_{\Lambda^{-1}})$	$\mathcal{N}(0, \Lambda^{-1})$
Laplacian	$\exp(-\|x-x'\|_1)$	$\prod_{i=1}^d \frac{1}{\pi(1+\omega_i)}$
Matérn	$\frac{2^{1-\nu}}{\Gamma(\nu)} r^\nu K_\nu(r)$	$h(\frac{2\nu}{\ell^2} + 4\pi^2\|\omega\|_2^2)^{\nu+\frac{d}{2}}$

Table 1: Examples of continuous shift-invariant positive-definite kernels and their corresponding spectral densities, where $r = \frac{\sqrt{2\nu}\|x-x'\|_2}{\ell}$, K_ν is a modified Bessel function, and $h = \frac{2^d \pi^{\frac{d}{2}} \Gamma(\nu+\frac{d}{2})(2\nu)^\nu}{\Gamma(\nu)\ell^{2\nu}}$.

where $i = 1, \ldots, m$. We state the following proposition to compute each column in Υ consisting of expectations of the product sinusoidal functions and inputs.

Proposition 2. *The expectation of the multiplication of sinusoids and linear functions over multivariate Gaussian distributions: $x \sim \mathcal{N}(\mu, \Sigma)$, can be computed analytically:*

$$\mathbf{E}\left(\cos(\omega^T x)x\right) = \left(\mathbf{E}\cos(\omega^T x)\right)\mu - (\mathbf{E}(\sin(\omega^T x))\Sigma\omega,$$
$$\mathbf{E}\left(\sin(\omega^T x)x\right) = \left(\mathbf{E}\sin(\omega^T x)\right)\mu + \left(\mathbf{E}\cos(\omega^T x)\right)\Sigma\omega,$$

where the right-hand-side expectations have analytical expressions (Proposition 1).

To prove it, we find an expression for $\mathbf{E}\left(a^T x \cos(\omega^T x)\right)$, for any a, through the complex domain trick used to prove Proposition 1. Next, the result is extended to $\mathbf{E}\left(x\cos(\omega^T x)\right)$, by setting a to consist of indicator vectors (see Appendix §3.3 for details). Applying Proposition 1 and 2, we complete the derivation of $\mathbf{Cov}(x, y)$ in (11).

Remark In summary, SSGP-EMM computes the exact posterior moments. This is equivalent to expectation propagation (Minka, 2001) by minimizing the Kullback-Leibler divergence between the true distribution and its Gaussian approximation with respect to the natural parameters. SSGP-EMM's computation complexity is $O\left(m^2k^2d^2\right)$, where m is the number of features, k is the output dimension, and d is the input dimension. The most computationally demanding part is constructing matrices Ψ_{ab} (10) for each output pair, where each requires $O\left(m^2d^2\right)$.

Compared to the multivariate moment-matching approach for GPs (GP-EMM) (Girard et al., 2003; Kuss, 2006) with $O\left(n^2k^2d^2\right)$ time complexity, SSGP-EMM is more efficient when $m \ll n$. Moreover, our approach is applicable to any positive-definite continuous shift-invariant kernel with different spectral densities (see examples in Table 1), while previous approaches like GP-EMM (Kuss, 2006) are only derived for squared exponential (SE) or polynomial kernels. Next we introduce a more computationally efficient but less accurate approach that avoids the computation of Ψ_{ab}'s.

3.2. Linearization (SSGP-Lin)

An alternative approach to computing the exact moments of the predictive distribution is based on the linearization of the posterior mean function in (7) at the input mean μ:

$$m(x) = \alpha^T \phi(x) \approx m(\mu) + \alpha^T \underbrace{D\phi(\mu)}_{M}(x-\mu), \quad (12)$$

where $D\phi(\mu)$ denotes taking the derivative of function ϕ at μ. Given the definition of ϕ in (4), $D\phi$ can be found by chain rule: $D\phi_i^c(x) = -\sigma_k \sin(\omega_i^T x)\omega_i^T$, $D\phi_i^s(x) = \sigma_k \cos(\omega_i^T x)\omega_i^T$.

Utilizing the linearized posterior mean function (12), the predictive moments can be approximated. The predictive mean approximation is

$$\mathbf{E}\, y = \mathbf{E}\,\mathbf{E}(y|x) \approx m(\mu), \quad (13)$$

and the predictive variance approximation is

$$\begin{aligned}\mathbf{Var}\, y &= \mathbf{E}\,\mathbf{Var}(y|x) + \mathbf{Var}\,\mathbf{E}(y|x) \\ &\approx \mathbf{Var}(y|\mu) + \mathbf{Var}(\alpha^T M x) \\ &= \sigma_n^2 + \sigma_n^2 \|\phi(\mu)\|^2_{A^{-1}} + \alpha^T M \Sigma M^T \alpha.\end{aligned} \quad (14)$$

and the approximate covariance between output dimension a and b is

$$\begin{aligned}\mathbf{Cov}(y_a, y_b) &= \mathbf{Cov}\left(\mathbf{E}(y_a|x), \mathbf{E}(y_b|x)\right) \\ &= \mathbf{E}\left(\alpha_a^T M_a (x-\mu)(x-\mu)^T M_b^T \alpha_b\right) \\ &\approx \alpha_a^T M_a \Sigma M_b^T \alpha_b,\end{aligned} \quad (15)$$

where M_a and M_b are defined as M in (12), except that they correspond to feature maps ϕ_a and ϕ_b. Notice that the assumption of conditional independence between different outputs is invoked here again, *cf.*, (10).

Finally, the cross-covariance between the input and output can be approximated as

$$\begin{aligned}\mathbf{Cov}(x, y) &= \mathbf{Cov}(x, \mathbf{E}(y|x)) \\ &\approx \mathbf{E}\left((x-\mu)(\alpha^T M(x-\mu))\right) \\ &= \alpha^T M \Sigma\end{aligned} \quad (16)$$

Unlike SSGP-EMM, which computes exact moments (§3.1), this linearization-based approach SSGP-Lin computes an *approximation* of the predictive moments. In contrast to SSGP-EMM's $O\big(m^2k^2d\big)$ computational complexity, the computation time of SSGP-Lin is reduced to $O\big(m^2kd\big)$, as a direct consequence of avoiding the construction of Ψ (3.1) in SSGP-EMM (10), which makes SSGP-Lin more efficient than SSGP-EMM, especially when the output dimension is high.

Both SSGP-EMM and SSGP-Lin are applicable to a general family of kernels. See Table 2 for a comparison between our methods and GP-EMM (Girard et al., 2003; Kuss, 2006). In the next section, we compare these approaches in applications of filtering and control.

Method	SSGP-EMM	SSGP-Lin	GP-EMM
Time	$O(m^2k^2d^2)$	$O(m^2k + mk^2d)$	$O(n^2k^2d^2)$
Applicable kernels	continuous shift-invariant kernels	continuous shift-invariant kernels	SE or polynomial kernels

Table 2: Comparison of our proposed methods and GP-EMM (Girard et al., 2003; Kuss, 2006) in terms of computational complexity and generalizability.

4. Applications

We focus on the application of the proposed methods to Bayesian filtering and predictive control. We begin by introducing Gauss-Markov models, which can be expressed by the following discrete-time nonlinear dynamical system:

$$x_{t+1} = f(x_t, u_t) + \epsilon_t^x, \qquad \epsilon_t^x \sim \mathcal{N}(0, \Sigma_{\epsilon^x}), \quad (17)$$

$$y_t = g(x_t) + \epsilon_t^y, \qquad \epsilon_t^y \sim \mathcal{N}(0, \Sigma_{\epsilon^y}), \quad (18)$$

where $x_t \in \mathbf{R}^d$ is state, $u_t \in \mathbf{R}^r$ is control, $y_t \in \mathbf{R}^k$ is observation or measurement, $\epsilon_t^x \in \mathbf{R}^d$ is IID process noise, $\epsilon_t^y \in \mathbf{R}^k$ is IID measurement noise, and subscript t denotes discrete time index. We call the probabilistic models (17) and (18) the dynamics and observation models, and the corresponding deterministic functions f and g the dynamics and observation functions.

We consider scenarios where f and g are unknown but a dataset $\mathcal{D} = \left(\{(x_t, u_t), x_{t+1}\}_{t=1}^{n-1}, \{x_t, y_t\}_{t=1}^{n}\right)$ is provided. The probabilistic models specified by SSGPs can be learned from the dataset, and then used to model the dynamics and observation (17) (18). More concretely, the dynamics model $p(x_{t+1}|x_t, u_t)$ is learned using state transition pairs $\{(x_t, u_t), x_{t+1}\}_{t=1}^{n-1}$, and the observation model $p(y_t|x_t)$ is learned separately from state-observation pairs $\{x_t, y_t\}_{t=1}^{n}$.

4.1. Bayesian filtering

The task of Bayesian filtering is to infer the posterior distribution of the current state of a dynamical system based on the current and past noisy observations, *i.e.*, finding $p(x_{t|t})$, where the notation $x_{t|s}$ denotes the random variable $x_t|y_0, \ldots, y_s$. Due to the Markov property of the process x, *i.e.*, $x_t|x_0, \ldots, x_{t-1} = x_t|x_{t-1}$, in Gauss-Markov models, $p(x_{t|t})$ can be computed *recursively* through alternating *prediction step* and *correction step*.

4.1.1. Prediction step ($x_{t-1|t-1} \to x_{t|t-1}$)

In the prediction step, $x_{t-1|t-1}$ is propagated through the dynamics model $p(x_t|x_{t-1}, u_{t-1})$:

$$p(x_{t|t-1}) = \int p(x_t|x_{t-1}, u_{t-1}) p(x_{t-1|t-1})\, \mathrm{d}x_{t-1},$$

which can be viewed as prediction under uncertainty (1). Suppose that $p(x_{t-1|t-1}) = \mathcal{N}(\hat{\mu}_{t-1|t-1}, \hat{\Sigma}_{t-1|t-1})$, with learned SSGP representation for the dynamics, Gaussian approximations of the output: $p(x_{t|t-1}) \approx \mathcal{N}(\hat{\mu}_{t|t-1}, \hat{\Sigma}_{t|t-1})$ can be obtained by either SSGP-EMM

(§3.1) using (8), (9) and (10), or SSGP-Lin (§3.2) using (13), (14) and (15).

4.1.2. CORRECTION STEP ($x_{t|t-1} \to x_{t|t}$)

The correction step conditions $x_{t|t-1}$ on the current observation y_t using Bayes' rule:

$$p(x_{t|t}) = \frac{p(y_t|x_{t|t-1})p(x_{t|t-1})}{\int p(y_t|x_{t|t-1})p(x_{t|t-1})\,\mathrm{d}x_t}. \tag{19}$$

In the preceding prediction step, we obtain $p(x_{t|t-1}) \approx \mathcal{N}(\hat{\mu}_{t|t-1}, \hat{\Sigma}_{t-1|t-1})$, which serves as a prior on x_t in this correction step. Due to the intractability of the integral in the denominator, to apply Bayes' rule we first seek Gaussian approximations for the joint distribution, as in the previous work on Bayesian filtering relying on GPs (Deisenroth et al., 2009; Ko & Fox, 2009):

$$\begin{bmatrix} x_{t|t-1} \\ y_{t|t-1} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \hat{\mu}_{t|t-1} \\ \hat{\mu}_y \end{bmatrix}, \begin{bmatrix} \hat{\Sigma}_{t|t-1} & \hat{\Sigma}_{xy} \\ \hat{\Sigma}_{xy}^T & \hat{\Sigma}_y \end{bmatrix} \right), \tag{20}$$

Invoking $p(y_{t|t-1}) = \int p(y_t|x_{t|t-1})p(x_{t|t-1})\,\mathrm{d}x_t$, the moments $\hat{\mu}_y$, $\hat{\Sigma}_y$, and $\hat{\Sigma}_{xy}$ in the joint Gaussian approximation can be computed as the predictive mean, predictive covariance, and input-prediction cross-covariance, for the observation model $p(y_t|x_t)$ with input $p(x_{t|t-1})$, using SSGP-EMM or SSGP-Lin. Having all terms in (20) determined, we condition $x_{t|t-1}$ exactly on current observation y_t:

$$\begin{aligned} \hat{\mu}_{t|t} &= \hat{\mu}_{t|t-1} + \hat{\Sigma}_{xy}\hat{\Sigma}_y^{-1}(y - \hat{\mu}_y), \\ \hat{\Sigma}_{t|t} &= \hat{\Sigma}_{t|t-1} - \hat{\Sigma}_{xy}\hat{\Sigma}_y^{-1}\hat{\Sigma}_{xy}. \end{aligned} \tag{21}$$

This Gaussian approximation $p(x_{t|t}) \approx \mathcal{N}(\hat{\mu}_{t|t}, \hat{\Sigma}_{t|t})$ is then used as input to the prediction step. Thus, we have shown that starting from $p(x_0) = \mathcal{N}(\mu_0, \Sigma_0)$, by consecutively applying prediction and correction steps presented above, we recursively obtain state estimates for $x_{t|t-1}$ and $x_{t|t}$. Rather than using a finite sample-based approximation such as in the GP-UKF (Ko & Fox, 2009), the Gaussian approximations of the full densities $p(x_{t|t})$ and $p(x_{t|t-1})$ are propagated.

Algorithm 1 SSGP-ADF and SSGP-EKF

1: Model learning: collect dataset $\mathcal{D}$, and learn SSGP dynamics and observations models (§2.)
2: Initialization: set prior $p(x_0)$.
3: **for** $t = 1, \ldots$ **do**
4: Prediction: compute $\hat{\mu}_{t|t-1}$ and $\hat{\Sigma}_{t|t-1}$. by either SSGP-EMM (§3.1) or SSGP-Lin (§3.2).
5: Measurement: make an observation y_t.
6: Correction: compute $\hat{\mu}_{t|t}$ and $\hat{\Sigma}_{t|t}$ according to (21) by either SSGP-EMM (§3.1) or SSGP-Lin (§3.2).
7: **end for**

We summarize the resulting filtering algorithm SSGP-ADF (assumed density filtering) and SSGP-EKF (extended Kalman filtering), based on SSGP-EMM and SSGP-Lin, respectively, in Algorithm 1. These are analogs of GP-ADF (Deisenroth et al., 2009) and GP-EKF (Ko & Fox, 2009).

4.2. Stochastic Model Predictive Control

The stochastic model predictive control (MPC) problem is to choose a control sequence that minimizes the expected cost, provided $p(x_t)$:

$$u^\star_{t+1:t+T} = \underset{u_{t+1:t+T}}{\operatorname{argmin}}\, \mathbf{E}\left(h(x_{t+T}) + \sum_i^{i+T} l(x_{t+i}, u_{t+i})\right),$$

at each time step, subject to stochastic system dynamics (17), where function $h : \mathbf{R}^d \to \mathbf{R}$ and $l : \mathbf{R}^d \times \mathbf{R}^r \to \mathbf{R}$ are the final and running cost respectively. There are two main challenges to applying MPC in practice: 1) MPC requires an accurate dynamics model for multi-step prediction, and 2) online optimization is very computationally expensive. For clarity in presentation, we will assume that the state is fully observable henceforth.

Algorithm 2 MPC via probabilistic trajectory optimization (1-3: offline optimization, 4-8: online optimization)

1: Model learning: collect dataset $\mathcal{D}$, and learn SSGP dynamics model (§2).
2: Initialization: set $t = 0$, and estimate $p(x_0)$.
3: Trajectory optimization: perform trajectory optimization in belief space, obtain $u^\star_{t+1:t+T}$.
4: **repeat**
5: Policy execution: apply one-step control $u^\star_{t+1}$ to the system and move one step forward, update $t = t+1$.
6: Model adaptation: incorporate new data and update SSGP dynamics model.
7: Trajectory optimization: perform re-optimization with the updated model. Initialize with the previously optimized trajectory and obtain new $u^\star_{t+1:t+T}$.
8: **until** Task terminated

4.2.1. MPC VIA PROBABILISTIC TRAJECTORY OPTIMIZATION

We address the aforementioned challenges by employing a combination of prediction under uncertainty and trajectory optimization. More precisely, we use SSGP-EMM or SSGP-Lin to efficiently obtain approximate Gaussian distribution over trajectory of states and perform trajectory optimization in the resultant *Gaussian belief space* based on differential dynamic programming (DDP) (Abbeel et al., 2007; Tassa et al., 2007). Note that DDP-related methods require computation of first and second order derivatives of the dynamics and cost. Our analytic moment expressions provide a robust and efficient way to compute these derivatives. Details are omitted due to space limit, but they can be found in Appendix §4.

Within the SSGP framework, we may incrementally *update* the posterior distribution over the feature weights w (6) given a new sample without storing or inverting the matrix A explicitly, Instead we keep track of its upper triangular Cholesky factor $A = R^T R$ (Gijsberts & Metta, 2013). Given a new sample, a rank-1 update is applied to

Method	SSGP-ADF	SSGP-EKF	GP-ADF	GP-EKF
NL_x	2.5003	3.415467	2.489385	3.396343
$RMSE$	4.6822	5.1451	4.6854	5.1012

Table 3: Comparison of our methods with GP-ADF (Deisenroth et al., 2009) and GP-EKF (Ko & Fox, 2009) in terms of average NL_x (negative log-likelihood) of the ground truth states given estimates and RMSE (root-mean-square error). Lower values are better. The results correspond to the filtering task in sec 5.1.1.

the Cholesky factor R, which requires $O(m^2)$ time. To cope with time-varying systems and to make the method more adaptive, we employ a forgetting factor $\lambda \in (0, 1)$, such that the impact of the previous samples decays exponentially in time (Ljung, 1998).

Our proposed MPC algorithm, summarized in Algorithm 2, is related to several algorithms and differs in both model and controller learning. First, SSGPs are more robust to modeling error than Locally Weighted Projection Regression (LWPR) used in iLQG-LD (Mitrovic et al., 2010). See a numerical comparison in (Gijsberts & Metta, 2013). Second, we efficiently propagate uncertainty in multi-step prediction which is crucial in MPC. In contrast, AGP-iLQR (Boedecker et al., 2014) drops the input uncertainty and uses subset of regressors (SoR-GP) which lacks a principled way to select reference points. In addition, PDDP (Pan & Theodorou, 2014) uses GPs which are computationally expensive for online optimization. Two deep neural networks are used for modeling in (Yamaguchi & Atkeson, 2016), which make it difficult to perform online incremental learning, as we do here.

5. Experimental Results

5.1. Bayesian filtering

5.1.1. 1D One-step filtering

We consider a synthetic dynamical system with ground-truth dynamics $f(x) = \frac{1}{2}x + \frac{25x}{1+x^2}$ and observation $g(x) = 6\sin(2x)$ with $\Sigma_{\epsilon^x} = 1.5^2$ and $\Sigma_{\epsilon^y} = 1$ in (17,18), in a similar setting to Deisenroth et al. (2009). We compare the performance of four filters, SSGP-ADF, SSGP-EKF, GP-ADF (Deisenroth et al., 2009) and GP-EKF (Ko & Fox, 2009). All models are trained using 800 samples. However, for SSGP models, only 10 random Fourier features of a SE kernel are used. Figure 1 illustrates the comparison of filtered state distribution of a typical realization. We evaluate the methods by computing NL_x (the negative log-likelihood of the ground truth samples in the filtered distribution) and RMSE (root-mean-square error between filtered mean and ground truth samples). See Table 3 for a detailed comparison. Our methods SSGP-ADF and SSGP-EKF are able to offer close performance with their full GP counterparts but with greatly reduced computational cost. See Appendix §6.2 for further discussions on the comparison between SSGP-ADF and SSGP-EKF.

5.1.2. Recursive filtering

We next consider a state estimation task in high-speed autonomous driving on a dirt track (Figure 2a). The goal is to recursively estimate the state of an autonomous rallycar given noisy measurements. The vehicle state consists of linear velocities (x and y), heading rate, and roll angle, in body frame. Controls are steering and throttle. Measurements are collected by wheel speed sensors. This filtering task is challenging because of the complex nonlinear dynamics and the amount of noise in the measurements. We do not use any prior model of the car, but learn the model from ground truth estimates of vehicle state generated by integrating GPS and IMU data via iSAM2 (Kaess et al., 2012). 50,000 samples are collected from wheel speed sensors and ground truth state estimates from iSAM2 for training. The platform for data collection is described in details in (Williams et al., 2016). Because of the sample size, it is too computationally expensive to use GP-based filter such as GP-ADF (Deisenroth et al., 2009). Instead, we use SSGP-ADF to perform 1,200 recursive filtering steps which correspond to 30 seconds of high-speed driving. Filtered distributions using 80 features are shown in Figure 2b, and Figure 2c shows the mean and twice the standard deviation of NL_x over six 30 seconds driving with different number of features. Surprisingly, only need a small number of features is necessary for satisfactory results.

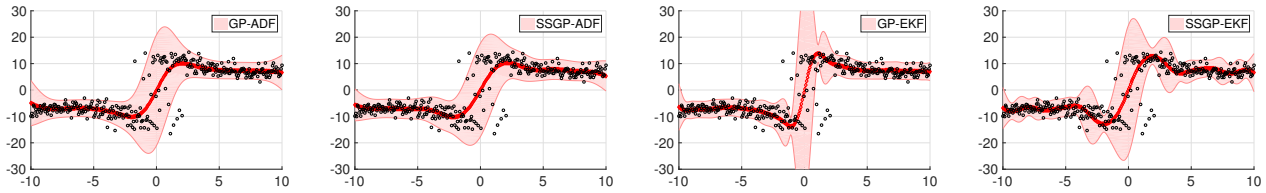
5.2. Model Predictive Control

5.2.1. Tracking a moving target

We consider the Puma-560 robotic arm and quadrotor systems with dynamics model specified by SSGPs. For both tasks the goal is to track a moving target. In addition, the true system dynamics vary online, which necessitates both online optimization and model update, as we do here. See Appendix §5.2 for detailed task descriptions. Results in terms of cost $l(x_t, u_t)$ are shown in Figure 4. Figure 4a shows that our methods outperform iLQG-LD (Mitrovic et al., 2010) and AGP-iLQR (Boedecker et al., 2014). The similarities and differences between these methods have been discussed in §4.2. Figure 4b shows that model update is necessary and more features could improve performance.

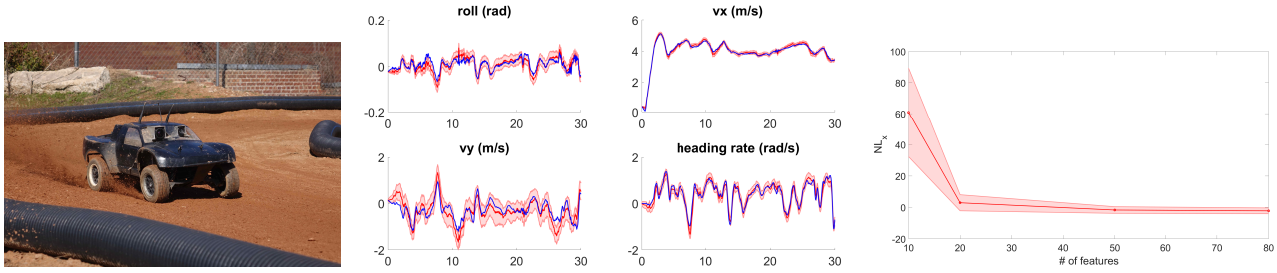
5.2.2. Autonomous drifting

We study the control of an autonomous car during extreme operating conditions (powerslide). The task is to stabilize the vehicle to a specified steady-state using purely longitudinal control during high-speed cornering. This problem has been studied in Velenis et al. (2010) where the authors developed a LQR control scheme based on a physics-based dynamics model. We apply our MPC algorithm to this task without any prior model knowledge and 2,500 data points generated by the model in Velenis et al. (2010). SSGP-Lin is used for multi-step prediction. Results and comparison to Velenis et al. (2010) are illustrated in Figure 3.



(a) GP-ADF (800 data points) (b) SSGP-ADF (10 features) (c) GP-EKF (800 data points) (d) SSGP-EKF (10 features)

Figure 1: Black points are ground truth states, red areas are filter distributions for (a) GP-ADF (Deisenroth et al., 2009), (c) GP-EKF (Ko & Fox, 2009), our proposed methods (b) SSGP-ADF and (d) SSGP-EKF. The x-axis is the mean of initial belief $p(x_0)$, which is randomly distributed in $[-10, 10]$ and y-axis shows the mean and twice the standard deviation of filtered distribution $p(x_1|y_1)$ after observing y_1.



(a) Autonomous driving task (b) Filtered distribution vs. ground truth (c) NL_x vs. # of features

Figure 2: Recursive filtering task for high-speed autonomous driving. Figure (b) shows trajectories of all the states of a 30 seconds continuous driving (1,200 steps), where blue lines are the ground truth, and red lines and red areas are the mean and twice the standard deviation of the filtered distributions respectively. In (c), the red line and area are the mean and twice the standard deviation of NL_x over six 30 seconds driving with varying number of features.

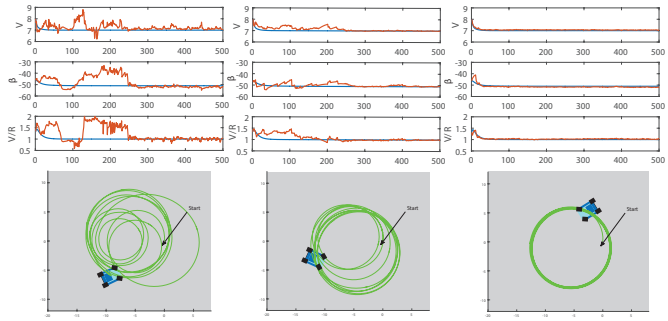
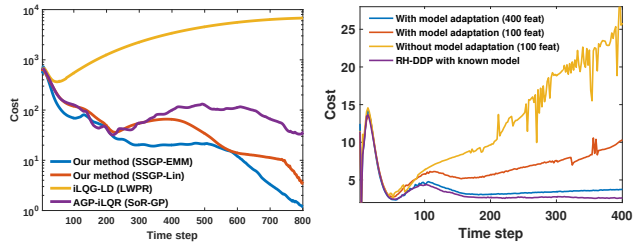


Figure 3: Comparison of the drifting performance using 50 (left), 150 (middle) and 400 (right) random features. Blue lines are the solution provided in Velenis et al. (2010). Performance improves with a larger number of features, and with a moderate number of features, MPC with SSGP-Lin behaves very closely to the ground truth solution.



(a) Robotic arm task cost (b) Quadrotor task cost

Figure 4: Cost comparison for arm and quadrotor tasks.

6. Discussion and Conclusion

We introduced two analytic moment-based approaches to *prediction under uncertainty* in sparse spectrum Gaussian processes (SSGPs). Compared to their full GP counterparts, our methods are more general: they are applicable to any continuous shift-invariant kernel. They also scale to larger datasets by leveraging random features with frequencies sampled from the spectral density of a given kernel (see Table 1, 2). Although we adopt the name SSGP, our proposed methods are not tied to specific model learning methods such as linear Bayesian regression (Lázaro-Gredilla et al., 2010). They can be applied to any SSGP with a specified feature weight distribution (6), and α and A can be computed via different approaches. For example, A can be iteratively computed by methods like doubly stochastic gradient descent (Dai et al., 2014). We studied the application of the proposed methods to Bayesian filtering and model predictive control. Our methods directly address the challenging aspects of these problems: model uncertainty and real-time execution constraints. We evaluated our algorithms on real-world and simulated examples and showed that SSGP-EMM (§3.1) and SSGP-Lin (§3.2) are accurate alternatives to their full GP counterparts when learning from large amounts of data.

Acknowledgements

This work was supported by NSF NRI awards 1637758 and 1426945.

References

Abbeel, P., Coates, A., Quigley, M., and Ng, A. Y. An application of reinforcement learning to aerobatic helicopter flight. *NIPS*, 19:1, 2007.

Archambeau, Cedric, Cornford, Dan, Opper, Manfred, and Shawe-Taylor, John. Gaussian process approximations of stochastic differential equations. *Gaussian Processes in Practice*, 1:1–16, 2007.

Boedecker, J., Springenberg, JT., Wulfing, J., and Riedmiller, M. Approximate real-time optimal control based on sparse Gaussian process models. In *ADPRL 2014*, pp. 1–8. IEEE, 2014.

Candela, J. Quinonero, Girard, A., Larsen, J., and Rasmussen, C. E. Propagation of uncertainty in Bayesian kernel models-application to multiple-step ahead forecasting. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2003.

Cheng, Ching-An and Boots, Byron. Incremental variational sparse Gaussian process regression. In *Proceedings of Advances in Neural Information Processing Systems 30 (NIPS)*, 2016.

Dai, Bo, Xie, Bo, He, Niao, Liang, Yingyu, Raj, Anant, Balcan, Maria-Florina F, and Song, Le. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pp. 3041–3049, 2014.

Deisenroth, M., Fox, D., and Rasmussen, C. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transsactions on Pattern Analysis and Machine Intelligence*, 27:75–90, 2015.

Deisenroth, Marc Peter, Huber, Marco F, and Hanebeck, Uwe D. Analytic moment-based Gaussian process filtering. In *Proceedings of the 26th annual international conference on machine learning*, pp. 225–232. ACM, 2009.

Deisenroth, Marc Peter, Turner, Ryan Darby, Huber, Marco F, Hanebeck, Uwe D, and Rasmussen, Carl Edward. Robust filtering and smoothing with Gaussian processes. *IEEE Transactions on Automatic Control*, 57(7):1865–1871, 2012.

Gijsberts, A. and Metta, G. Real-time model learning using incremental sparse spectrum Gaussian process regression. *Neural Networks*, 41:59–69, 2013.

Girard, A., Rasmussen, C.E., Quinonero-Candela, J., and Murray-Smith, R. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In *NIPS*, 2003.

Kaess, Michael, Johannsson, Hordur, Roberts, Richard, Ila, Viorela, Leonard, John J, and Dellaert, Frank. isam2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

Ko, J. and Fox, D. Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. *Autonomous Robots*, 27(1):75–90, 2009.

Kupcsik, A., Deisenroth, M.P., Peters, J., Loh, AP, Vadakkepat, P., and Neumann, G. Model-based contextual policy search for data-efficient generalization of robot skills. *Artificial Intelligence*, 2014.

Kuss, Malte. *Gaussian process models for robust regression, classification, and reinforcement learning*. PhD thesis, Technische Universität, 2006.

Lázaro-Gredilla, M., Quiñonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. Sparse spectrum Gaussian process regression. *The Journal of Machine Learning Research*, 99: 1865–1881, 2010.

Ljung, Lennart. *System identification*. Springer, 1998.

Minka, Thomas P. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

Mitrovic, D., Klanke, S., and Vijayakumar, S. Adaptive optimal feedback control with learned internal dynamics models. In *From Motor Learning to Interaction Learning in Robots*, pp. 65–84. Springer, 2010.

Pan, Y. and Theodorou, E. Probabilistic differential dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1907–1915, 2014.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2007.

Rasmussen, C. and Kuss, M. Gaussian processes in reinforcement learning. In *NIPS*, volume 4, pp. 1, 2004.

Snelson, E. and Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. *NIPS*, 18:1257, 2006.

Tassa, Y., Erez, T., and Smart, W. D. Receding horizon differential dynamic programming. In *NIPS*, 2007.

Titsias, Michalis K. Variational learning of inducing variables in sparse gaussian processes. In *AISTATS*, volume 5, pp. 567–574, 2009.

Velenis, E., Frazzoli, E., and Tsiotras, P. Steady-state cornering equilibria and stabilisation for a vehicle during extreme operating conditions. *International Journal of Vehicle Autonomous Systems*, 8(2-4):217–241, 2010.

Williams, C.K.I and Rasmussen, C.E. *Gaussian processes for machine learning*. MIT Press, 2006.

Williams, Grady, Drews, Paul, Goldfain, Brian, Rehg, James M, and Theodorou, Evangelos A. Aggressive driving with model predictive path integral control. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 1433–1440. IEEE, 2016.

Yamaguchi, Akihiko and Atkeson, Christopher G. Neural networks and differential dynamic programming for reinforcement learning problems. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5434–5441. IEEE, 2016.

Yan, Xinyan, Xie, Bo, Song, Le, and Boots, Byron. Large-scale Gaussian process regression via doubly stochastic gradient descent. *The ICML Workshop on Large-Scale Kernel Learning*, 2015.