Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

A KNN Based Kalman Filter Gaussian Process Regression

Yali Wang and **Brahim Chaib-draa**
Department of Computer Science and Software Engineering, Laval University, Canada
wang@damas.ift.ulaval.ca brahim.chaib-draa@ift.ulaval.ca

Abstract

The standard Gaussian process (GP) regression is often intractable when a data set is large or spatially nonstationary. In this paper, we address these challenging data properties by designing a novel K nearest neighbor based Kalman filter Gaussian process (KNN-KFGP) regression. Based on a state space model established by the KNN driven data grouping, our KNN-KFGP recursively filters out the latent function values in a computationally efficient and accurate Kalman filtering framework. Moreover, KNN allows each test point to find its strongly correlated local training subset, so our KNN-KFGP provides a suitable way to deal with spatial nonstationary problems. We evaluate the performance of our KNN-KFGP on several synthetic and real data sets to show its validity.

1 Introduction

Over the past years, Bayesian parametric approaches have been well-developed for regression analysis [Bishop, 2006]. However, the data flexibility usually cannot be established by parametric models in all realistic details. Hence, Bayesian nonparametric methods have become popular due to the plausibly probabilistic framework. Gaussian Process (GP) [Rasmussen and Williams, 2006], a nonparametric model with an elegant probabilistic form, has been applied for regression problems such as robotics [Plagemann *et al.*, 2007; Ko and Fox, 2008; Deisenroth *et al.*, 2011; Wang and Chaib-draa, 2012], geophysics [Paciorek and Schervish, 2004], econometrics [Lázaro-Gredilla and Titsias, 2011] and so on. However, there exist several difficulties in GP due to the data properties in the real world [Plagemann, 2008].

First of all, the size of real data sets N is usually quite large, and it leads to the unsatisfactory computational load in GP $(O(N^3))$. A number of sparse forms have been proposed for computation reduction. Concretely, by introducing a small pseudo-input training data [Snelson and Ghahramani, 2006] or spectral representation of GP [Lázaro-Gredilla *et al.*, 2010], sparse GPs efficiently solve several large data sets in robotics. However, the approximation accuracy still needs to be further improved since sparse GPs may fall into the overfitting risk. Recently, a Kalman filter Gaussian process (KFGP) [Reece and Roberts, 2010] has been developed by correlating the latent function values of different data subsets in the well-known Kalman filter (KF) framework to reduce computation load. But KFGP still has to pay the price of accuracy due to the random training subset selection.

Secondly, many real data sets in geophysics or biology reflect the input-dependent smoothness (nonstationarity) where the correlation between function values $f(\mathbf{x})$ and $f(\mathbf{x}')$ does not only depend on $\mathbf{x}-\mathbf{x}'$ but it is also related to input locations $\mathbf{x}$ and $\mathbf{x}'$ themselves [Rasmussen and Williams, 2006]. Usually, it is hard for the standard GP to correctly interpret the nonstationarity. A direct method is to define a nonstationary covariance function of GP [Paciorek and Schervish, 2004]. However, the eigenvalue decomposition for the covariance matrix of each input density is complicated since it requires a great number of parameters even in the low dimensional cases. Several extensions have been made by using kernel adaptation [Lang *et al.*, 2007] and local smoothness [Plagemann *et al.*, 2008]. Moreover, there also exists an infinite mixture model to address computation burden and nonstationarity together [Rasmussen and Ghahramani, 2002], but its hierarchical structure is difficult for membership determination in the gating network [Paciorek and Schervish, 2004]. A more practical approach is a local mixture of Gaussian processes [Urtasun and Darrell, 2008] that contains offline training hyperparameters and online inference using local GP experts. However, its accuracy depends on the number of local GP experts.

In this paper, we propose a novel KNN based Kalman filter Gaussian process (KNN-KFGP) method for various regression tasks. We firstly apply a test-input driven KNN search to construct a small training subset for each test, and then establish a state space model by correlating GP prior for different training subsets. Finally, a KF is applied to infer the latent function values in a predict-update manner. Our approach reduces computational load since the estimation for each test is based on a KNN constructed training subset. Meanwhile, it preserves the accuracy since KF produces a statistically optimal solution for linear and Gaussian systems [Kalman, 1960]. Moreover, KNN provides a locally correlated training subset for each test point, which is a suitable way to deal with the nonstationarity.

The rest of this paper is organized as follows: In Section 2, we briefly review the standard GP regression. In Section

3, we firstly introduce a construction procedure for the KNN based state space model, then apply KF to infer latent function values. In Section 4, we evaluate our proposed KNN-KFGP by using several synthetic and real data sets. Finally, we conclude the paper in Section 5.

2 Gaussian Process Regression

The goal of the nonlinear regression is to learn a function $f(\mathbf{x})$ in the noisy relationship between an input $\mathbf{x} \in R^d$ and its corresponding output $y \in R$

$$y = f(\mathbf{x}) + \epsilon \tag{1}$$

where the noise ϵ is typically assumed as a Gaussian distribution $\epsilon \sim N(0, \sigma^2)$. Since parametric approaches restrict the richness of the assumed function family when solving the regression problem, Bayesian nonparametric methods have recently become more attractive by giving a prior probability to every possible function.

Gaussian process (GP) is a popular nonparametric model due to the fact that it provides a flexible manner to represent a distribution over functions. Formally, it is a collection of random variables, any finite number of which have a joint Gaussian distribution [Rasmussen and Williams, 2006]. Similar to a Gaussian distribution parameterizing by a mean vector and covariance matrix, a GP is fully specified as $f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ with its mean function

$$m(\mathbf{x}) = E[f(\mathbf{x})] \tag{2}$$

and covariance function

$$k(\mathbf{x}, \mathbf{x}') = E[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \tag{3}$$

In this paper, we follow a commonly used setting [Rasmussen and Williams, 2006]. The mean function in (2) is set to be zero and the covariance function in (3) is chosen as a stationary squared exponential covariance function $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 exp[-0.5(\mathbf{x} - \mathbf{x}')^T L^{-2}(\mathbf{x} - \mathbf{x}')]$ where $L = diag([l_1 \cdots l_d])$ is the length-scale matrix. For convenience, all the hyperparameters are denoted into a vector $\theta = [\sigma \ \ \sigma_f \ \ l_1 \cdots l_d]^T$.

GP solves a regression problem in two steps. Firstly, learning the θ given the training set $(X, \mathbf{y}) = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. To do that, we apply a gradient based optimizer as presented in [Rasmussen and Williams, 2006] by maximizing the marginal likelihood $p(\mathbf{y}|X, \theta)$. Secondly, inferring the predictive distribution of the function value at test inputs $X_\star = \{\mathbf{x}_\star^i\}_{i=1}^M$. In the context of GP, the probability $p(\mathbf{y}, f(X_\star)|X, X_\star, \theta)$ is a jointly Gaussian distribution because any finite set of function values is jointly Gaussian distributed and the noise ϵ in (1) is independent identically distributed (iid) Gaussian noise. Hence, based on the conditional property of Gaussian distribution, the predictive distribution $p(f(X_\star)|X, \mathbf{y}, X_\star, \theta)$ is also Gaussian [Rasmussen and Williams, 2006]:

$$\bar{f}(X_\star) = K_\theta(X_\star, X)[K_\theta(X, X) + \sigma^2 I]^{-1}\mathbf{y} \tag{4}$$

$$\begin{aligned} P(X_\star, X_\star) = K_\theta(X_\star, X_\star) - K_\theta(X_\star, X)\times \\ [K_\theta(X, X) + \sigma^2 I]^{-1} K_\theta(X_\star, X)^T \end{aligned} \tag{5}$$

where $K_\theta(X_\star, X)$ denotes an $M \times N$ covariance matrix in which each entry is computed by the covariance function $k(\mathbf{x}, \mathbf{x}')$ with the learned θ. The $K_\theta(X, X)$ and $K_\theta(X_\star, X_\star)$ are constructed in a similar way.

3 KNN Based Kalman Filter Gaussian Process Regression

In practice, GP suffers from an unsatisfactory computational burden ($O(N^3)$) if the size of data set N is beyond a few thousand. By constructing a state space model with several small data collections, the recent KFGP [Reece and Roberts, 2010] provided an elegantly predict-update KF framework for computation reduction ($O(MK^3)$, where M is the number of KF iterations and K is the maximum size of all the training subsets). However, the blindly random training data selection for each test in KFGP largely deteriorates the estimation performance. Hence, we firstly need to explore an efficient mechanism of training subset selection for the test points in order to improve the accuracy of Bayesian inference in the state space model.

3.1 KNN Constructed State Space Model

Since the goal is to accurately estimate latent function values at the test inputs $\{\mathbf{x}_\star^i\}_{i=1}^M$, we propose a test-input-driven KNN method to explore the strongly correlated local structure of each test point. Concretely, for $\mathbf{x}_\star^t$, we find its K_t nearest training inputs X_t according to the Euclidian distance, then use X_t and the outputs $\mathbf{y}_t$ of X_t as a small training set $(X_t, \mathbf{y}_t) = \{(\mathbf{x}_i, y_i)\}_{i=1}^{K_t}$ for $\mathbf{x}_\star^t$.

We now use KNN based multiple data collections to establish a state space model including the state and observation equation. Firstly, the state equation represents the Markovian transition of latent function values. Hence it is necessary to explore the relation between the $(t-1)_{th}$ and t_{th} data subset. We denoted $X_t^c = X_t \cup \mathbf{x}_\star^t$ and $f_t^c = f(X_t^c)$. If $f(\mathbf{x})$ follows a GP, the prior $p(f_t^c, f_{t-1}^c|X_{t-1}^c, X_t^c, \theta)$ is Gaussian distributed:

$$N(\mathbf{0}, \begin{bmatrix} K_\theta(X_t^c, X_t^c) & K_\theta(X_t^c, X_{t-1}^c) \\ K_\theta(X_t^c, X_{t-1}^c)^T & K_\theta(X_{t-1}^c, X_{t-1}^c) \end{bmatrix})$$

Then the conditional distribution of f_t^c is also Gaussian:

$$p(f_t^c|f_{t-1}^c, X_{t-1}^c, X_t^c, \theta) = N(G(\theta)f_{t-1}^c, Q(\theta)) \tag{6}$$

where

$$G(\theta) = K_\theta(X_t^c, X_{t-1}^c)K_\theta^{-1}(X_{t-1}^c, X_{t-1}^c) \tag{7}$$

$$\begin{aligned} Q(\theta) = K_\theta(X_t^c, X_t^c) - K_\theta(X_t^c, X_{t-1}^c)\times \\ K_\theta^{-1}(X_{t-1}^c, X_{t-1}^c)K_\theta(X_t^c, X_{t-1}^c)^T \end{aligned} \tag{8}$$

This conditional probability (6) actually reflects that the state equation is a linear evolution between f_t^c and f_{t-1}^c with an additive Gaussian noise $v_t^f \sim N(\mathbf{0}, Q(\theta))$:

$$f_t^c = G(\theta)f_{t-1}^c + v_t^f \tag{9}$$

Secondly, the observation equation, which is a relationship between hidden function values and outputs, is directly obtained from (1):

$$\mathbf{y}_t = H_t f_t^c + v_t^y \tag{10}$$

where $H_t = [I_{K_t} \ \ \mathbf{0}]$ is an index matrix so that the observation $\mathbf{y}_t$ is only related to its corresponding X_t by $H_t f_t^c = f(X_t)$. The covariance $R(\theta) = \sigma^2 I$ of the noise v_t^y is originally from ϵ in (1). To sum up, our state space model is fully specified by (9-10). The whole construction is shown in Figure 1.

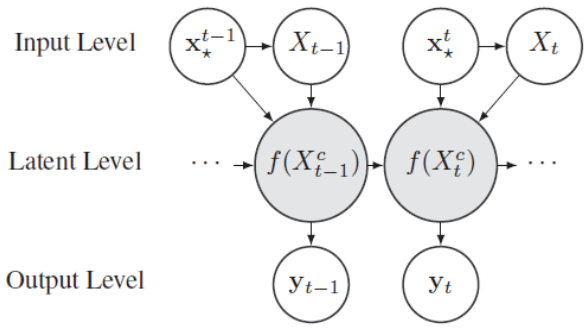


Figure 1: State space model that is established by a test-input-driven KNN data collection procedure. The arrows between $\mathbf{x}_\star^{t-1}$ and X_{t-1}, $\mathbf{x}_\star^{t}$ and X_t represent the KNN mechanism.

3.2 Bayesian Inference by Kalman Filter

After constructing the state space model in Subsection 3.1, we can now consider a regression task as a state estimation problem using recursive Bayesian filtering techniques. Given the learned θ, Equation (9) and (10) in the state space model are linear with additive Gaussian noises. The well-known Kalman Filter (KF) provides an optimal solution according to maximum-a-posteriori (MAP) [Kalman, 1960].

The detailed Bayesian inference procedure is described as follows. Suppose that the $(t-1)^{th}$ posterior distribution $p(f_{t-1}^c|\mathbf{y}_{1:t-1}, X_{1:t-1}, \mathbf{x}_\star^{1:t-1}, \theta)$ is a Gaussian distribution $N(f_{t-1|t-1}^c, P_{t-1|t-1}^c)$. The first step is to incorporate the transition distribution $p(f_t^c|f_{t-1}^c, X_{t-1:t}, \mathbf{x}_\star^{t-1:t}, \theta)$ (directly from (6)) with the $(t-1)^{th}$ posterior to make the prediction for the t^{th} hidden function values:

$$\begin{aligned}
&p(f_t^c|\mathbf{y}_{1:t-1}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta) \\
&= \int p(f_t^c|f_{t-1}^c, X_{t-1:t}, \mathbf{x}_\star^{t-1:t}, \theta) \times \\
&\quad p(f_{t-1}^c|\mathbf{y}_{1:t-1}, X_{1:t-1}, \mathbf{x}_\star^{1:t-1}, \theta) df_{t-1}^c \\
&= \int N(G(\theta)f_{t-1}^c, Q(\theta)) N(f_{t-1|t-1}^c, P_{t-1|t-1}^c) df_{t-1}^c \\
&= N(G(\theta)f_{t-1|t-1}^c, \ G(\theta)P_{t-1|t-1}^c G(\theta)^T + Q(\theta)) \quad (11)
\end{aligned}$$

We denote $p(f_t^c|\mathbf{y}_{1:t-1}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta)$ as $N(f_{t|t-1}^c, P_{t|t-1}^c)$, then the prediction step is fully expressed as follows:

$$f_{t|t-1}^c = G(\theta) f_{t-1|t-1}^c \quad (12)$$

$$P_{t|t-1}^c = G(\theta) P_{t-1|t-1}^c G(\theta)^T + Q(\theta) \quad (13)$$

The second step is to update the predicted estimation with the t^{th} observed output $\mathbf{y}_t$:

$$\begin{aligned}
&p(f_t^c|\mathbf{y}_{1:t}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta) \\
&= \frac{p(\mathbf{y}_t|f_t^c, \theta, X_t, \mathbf{x}_\star^t) p(f_t^c|\mathbf{y}_{1:t-1}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta)}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \theta, X_{1:t}, \mathbf{x}_\star^{1:t})} \quad (14)
\end{aligned}$$

where the likelihood $p(\mathbf{y}_t|f_t^c, \theta, X_t, \mathbf{x}_\star^t)$ is directly from (10), the prediction $p(f_t^c|\mathbf{y}_{1:t-1}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta)$ is from (11), and the marginal distribution is an analytical integral:

$$\begin{aligned}
&p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \theta, X_{1:t}, \mathbf{x}_\star^{1:t}) \\
&= \int p(\mathbf{y}_t|f_t^c, \theta, X_t, \mathbf{x}_\star^t) p(f_t^c|\mathbf{y}_{1:t-1}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta) df_t^c \\
&= \int N(H_t f_t^c, R(\theta)) N(f_{t|t-1}^c, P_{t|t-1}^c) df_t^c \\
&= N(H_t f_{t|t-1}^c, H_t P_{t|t-1}^c H_t^T + R(\theta)) \quad (15)
\end{aligned}$$

Finally we denote $p(f_t^c|\mathbf{y}_{1:t}, X_{1:t}, \mathbf{x}_\star^{1:t}, \theta) = N(f_{t|t}^c, P_{t|t}^c)$, then put (10), (11) and (15) into (14), we obtain the recursive form for the update step with the Kalman Gain Γ_t :

$$\Gamma_t = P_{t|t-1}^c H_t^T (H_t P_{t|t-1}^c H_t^T + R(\theta))^{-1} \quad (16)$$

$$f_{t|t}^c = f_{t|t-1}^c + \Gamma_t(\mathbf{y}_t - H_t f_{t|t-1}^c) \quad (17)$$

$$P_{t|t}^c = P_{t|t-1}^c - \Gamma_t H_t P_{t|t-1}^c \quad (18)$$

Algorithm 1 KNN-KFGP Regression

1: Training the hyperparameter vector θ using gradient optimization with an extra data set $\{(\mathbf{x}_i, y_i)\}_{i=1}^S$
2: **for** $t = 1$ **to** M **do**
3: Applying KNN based data construction to find the training data $(X_t, \mathbf{y}_t) = \{(\mathbf{x}_i, y_i)\}_{i=1}^{K_t}$ for $\mathbf{x}_\star^t$
4: Using θ to specify $G(\theta)$, $Q(\theta)$, $R(\theta)$ in (7-8) and (10) for state space model construction
5: Kalman Predict Step: Equation (12-13)
6: Kalman Update Step: Equation (16-18)
7: **end for**

The whole algorithm is summarized in Algorithm 1. We notice that, if the standard GP with stationary covariance function (we call it as the stationary GP) can suitably learn the model of the data set, there is a tradeoff of accuracy vs computation load. The stationary GP establishes the whole correlation between the full training set and each test point, so the accuracy is high. However, its computational load is also high. Our KNN Kalman filter Gaussian process (KNN-KFGP) is computationally efficient, compared with the stationary GP. The main computation reduction is the estimation step that is mainly governed by Kalman filter $O(MK^3)$ and KNN search $O(MN)$ after various precomputations [Urtasun and Darrell, 2008]. The K is the maximum value of K_t, M is the number of test inputs and N is the total number of training points. But the accuracy of our KNN-KFGP is not as good as the stationary GP in this case due to the fact that our KNN-KFGP selects a small training subset for each test point. However, its accuracy will not decrease much. The reason is that the small training subset in our KNN-KFGP is strongly-correlated to the corresponding test point, and the estimation of f_{t-1}^c can be used to improve the estimation of f_t^c in the predict-update Kalman filter framework.

Moreover, if the data set reflects the nonstationary phenomenon, the stationary GP cannot correctly model it. On the contrary, our KNN-KFGP allows each test point to find its strongly correlated small training subset, which is a suitable way to capture the local data property to deal with nonstationarity problems.

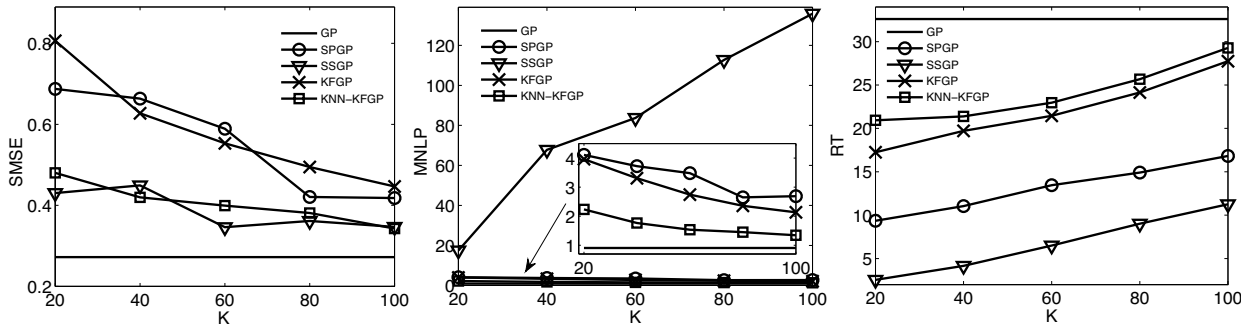


Figure 2: Prediction Performance for Pendulum Data

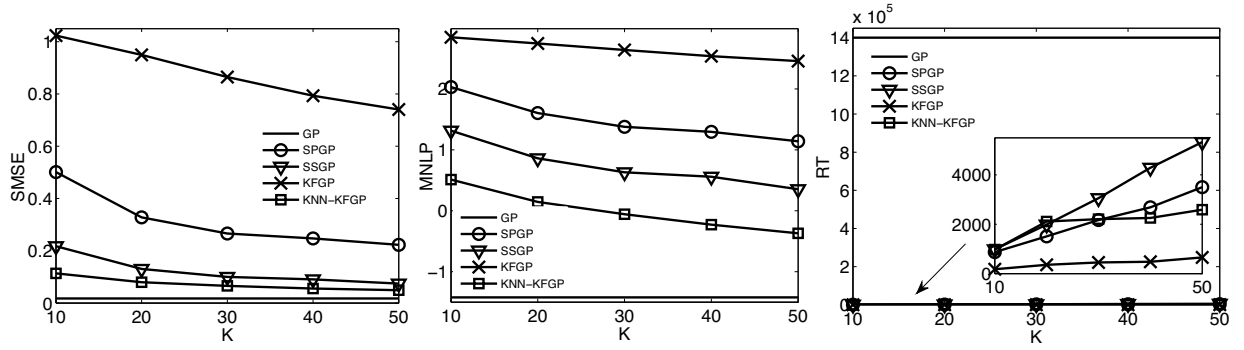


Figure 3: Prediction Performance for Kin-40k Data

4 Experiments

In this section, we evaluate our KNN-KFGP on several synthetic and real data sets. We aims to achieve two objectives in these experiments. Firstly, if the data sets can be successfully modeled by the stationary GP, we will evaluate whether our KNN-KFGP is a computationally efficient and accurate approximation of the stationary GP. The pendulum and kin-40k data sets[1] are used to validate this objective. Secondly, if the data sets cannot be suitably modeled by the stationary GP, we will test whether our KNN-KFGP is able to deal with the nonstationarity. The mobile robot perception and global land-surface precipitation data sets[2] are chosen for this objective. For all the data sets, the accuracy is evaluated by the test Standardized Mean Square Error (SMSE)

$$\frac{1}{M}\sum_{i=1}^{M}\frac{(y_\star^i-\mu_\star^i)^2}{variance(\mathbf{y}_\star)}$$

and the test Mean Negative Log Probability (MNLP)

$$\frac{1}{M}\sum_{i=1}^{M}(\frac{(y_\star^i-\mu_\star^i)^2}{(\sigma_\star^i)^2}+2\log(\sigma_\star^i)+\log 2\pi)$$

The computational efficiency is evaluated by Running Time (RT). In the SMSE and MNLP, the predictive mean $\mu_\star^i$ for the i^{th} test output is directly obtained from (17) and the predictive variance $\sigma_\star^i$ is the sum of the learned noise variance σ^2 and the function variance from (18) .

4.1 The Pendulum Data

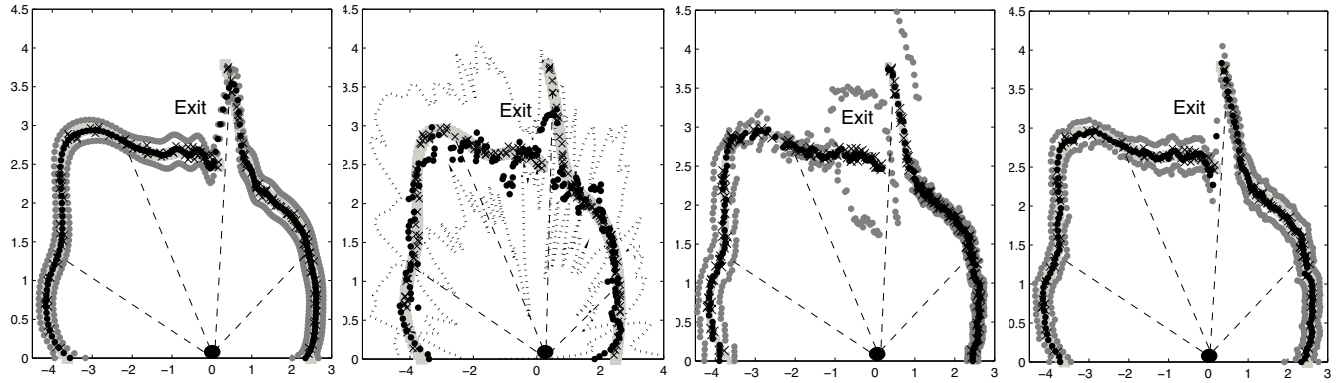
The first data set is a realistic simulation of a mechanical pendulum. There are 315 training and 315 test input-output pairs in this data set. Both the training and test sets are randomly ordered. In each pair, the input is a 9-dimensional pendulum parameter vector, and its corresponding output is an angular velocity of the pendulum.

In this experiment, we compare our KNN-KFGP with GP, Kalman Filter Gaussian Process (KFGP) [Reece and Roberts, 2010], Sparse Pseudo-input Gaussian Process (SPGP) [Snelson and Ghahramani, 2006] and Sparse Spectrum Gaussian Process (SSGP) [Lázaro-Gredilla *et al.*, 2010]. We then report the prediction performance as a function of K in Figure 2, where K respectively represents the size of training set for each test point in KFGP and KNN-KFGP, the size of pseudo-input set for each test point in SPGP, the number of the basis function pairs for each test point in SSGP. Moreover, we learn the hyperparameters using gradient optimization with the full training set. For each K, we run all the methods three times and calculate the average SMSE, MNLP and RT (seconds).

Since the prediction for each test point in the stationary GP is made with the whole training set, we use its results as the

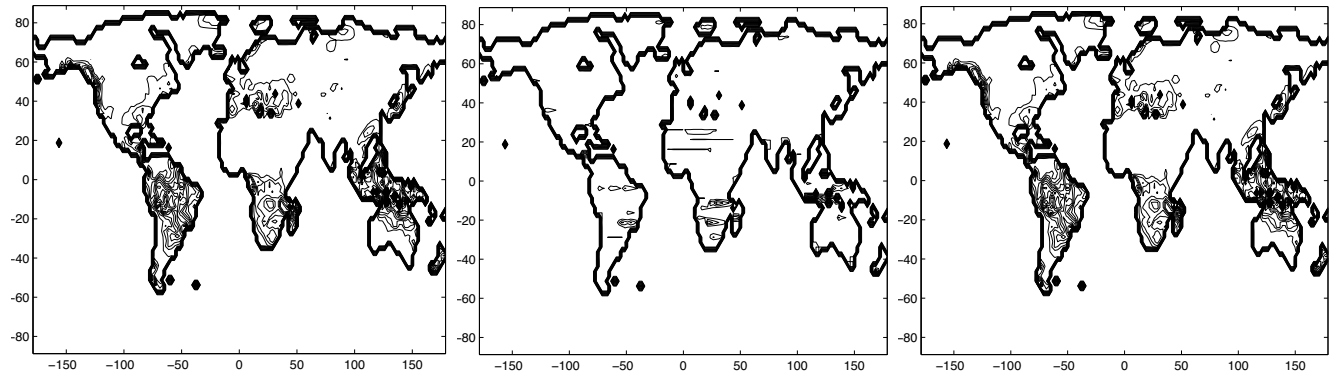
[1]The pendulum set and the kin-40k data set are available at: http://www.tsc.uc3m.es/ miguel/downloads.php

[2]The global land-surface precipitation data set is available at: http://www.cgd.ucar.edu/cas/catalog/surface/precip/gpcc.html



(a) GP (b) KFGP (c) LGP (d) KNN-KFGP

Figure 4: Mobile Robot Perception. A mobile robot (black ellipse at (0,0)) uses its range sensors (four black dashed lines) to obtain distance measurements (black crosses) for obstacle (grey squares) detection. The prediction mean for all the methods are represented by black dots. The 95% confidence interval for KFGP are represented by grey dotted line, for other methods it is represented by grey dots.



(a) Observation (b) KFGP (c) KNN-KFGP

Figure 5: Global Land-Surface Precipitation. The contour in all three subplots represents the level of precipitation value. When the contours are concentrated, the precipitation value is high. Otherwise, the the precipitation value is low. The subplots of KFGP and KNN-KFGP show the predicted mean of the global land-surface precipitation.

base line. We found that it successfully captures the data flexibility with the best SMSE and MNLP. However, the tradeoff is the highest computation load. On the contrary, SSGP is more computationally efficient than stationary GP. The SMSE of SSGP is also good when K is small. But its MNLP becomes poorly deteriorated as K increases. The reason is that SSGP falls into the overfitting trap [Lázaro-Gredilla *et al.*, 2010]. Compared to SSGP, SPGP is more acceptable for this data set. Moreover, the accuracy of KFGP is slightly better than SPGP, but the computation load is higher than SPGP.

To sum up, there is a tradeoff between computation load and accuracy for all the methods above. It is worth mentioning that this pendulum data is small, so all the methods are competitively fast. However, in terms of accuracy, our KNN-KFGP is the most suitable choice among all the approximate GPs. In order to convincingly show our KNN-KFGP outperforms other methods in terms of the tradeoff between accuracy and computational efficiency, it is necessary to further evaluate all the methods in the following intensive data set.

4.2 The Kin-40k Data

The second data set is a realistic simulation of a robot arm. The randomly ordered training (10000 instances) and test (30000 instances) set are massive. In each instance, the input is a 8-dimensional parameter vector, and the output is scalar. In Figure 3, we show the prediction performance for all the methods (stationary GP, SPGP, SSGP, KFGP and our KNN-KFGP) as a function of K. The notation of K is the same as the pendulum experiment. Because this data set is massive, we learn the hyperparameters using gradient optimization with a randomly selected training subset including 1000 points. For each K, we run all the methods three times and calculate the average SMSE, MNLP and RT(seconds).

The stationary GP makes the prediction for each test point with the whole training set, so its accuracy results (SMSE and MNLP) are the best. However, the computation burden is unsatisfactorily high. In contrast, KFGP is computationally efficient while its accuracy is bad. Furthermore, the computation

Accuracy Evaluation	SMSE	MNLP
GP	0.0130	-2.1775
LGP	0.0093	-2.3216
KFGP	0.2218	0.0791
KNN-KFGP	0.0057	-2.5060

Table 1: Robot Perception

Accuracy Evaluation	SMSE	MNLP
GP	0.0128	7.6863
LGP	0.0036	7.5979
KFGP	1.2582	12.1196
KNN-KFGP	0.0027	7.1255

Table 2: Land Precipitation

load of SPGP, SSGP, our KNN-KFGP is similar and much better than the stationary GP. But the best accuracy among all the approximation methods is obtained by our KNN-KFGP. Hence, compared to other approximate GPs, our KNN-KFGP is a more accurate and efficient approach for large data sets.

4.3 Mobile Robot Perception

Environment perception is a fundamental task for mobile robot systems. It is essential for a mobile robot to correctly interpret the sensor information in spatially nonstationary environments in order to find its precise location. Here we consider a perception task to evaluate how well our KNN-KFGP can deal with the nonstationarity. Concretely, a mobile robot is located at $(0, 0)$ with $\pi/2$ heading. To find an exit, it uses its range sensors for obstacle detection. We simulate 200 training points where the input is a bearing angle in $[0, \pi]$ and the output is the corresponding distance measurement. The test inputs are from 0 to π with an interval $\pi/180$. The observations at different sides of the exit show the discontinuity of the measured data, which is a case of spatial nonstationarity.

In this experiment, we compare our KNN-KFGP to the stationary GP, KFGP and local GP (LGP) which is a related work using local GP experts to capture the nonstationarity [Urtasun and Darrell, 2008]. The notation of K in KFGP and KNN-KFGP is the same as the first two experiments. In LGP, K is the number of local GP experts. It is worth mentioning that, for our KNN-KFGP, the accuracy will not be monotonically increasing in the nonstationary cases when K is increasing. The underlying reason is that, if the K is too large, some disturbed, even wrongly correlated training points will be selected for each test point. That is why the stationary GP cannot correctly model the nonstationarity. Moreover, the random training data selection in KFGP leads to its low accuracy. In order to improve the accuracy, KFGP has to use more training data for each test. Hence, its performance is prone to stationary GP which is poor for nonstationary cases.

In this experiment, We learn the hyperparameters using gradient optimization with the full training set, then run all the methods three times from $K = 1$ to 5. We found that KNN-KFGP shows the best performance when $K = 2$, KFGP and LGP represents the best performance when $K = 5$. Notice that, we choose a range that contains the best K since we mainly focus on finding the best K for our KNN-KFGP. Of course, the accuracy of LGP is better when K is larger, but the computation load is also higher. The results of all the methods are illustrated in Figure 4 and Table 1. In Figure 4(a), the standard GP mistakenly predicts that there is no exit even though the robot has detected the data on both sides of the exit. In Figure 4(b), the prediction of KFGP fails since the randomly selected training points in this case are disturbed. In Figure 4(c), LGP correctly finds the exit, but the uncertainty at the exit is high. Finally, in Figure 4(d) our KNN-KFGP successfully models the exit since the training subset for each test is strongly correlated to that test. From Table 1, it is also shown that LGP and our KNN-KFGP are suitable approaches for nonstationarity modeling, but our KNN-KFGP performs better with higher accuracy than LGP.

4.4 Global Land-Surface Precipitation

In practice, the environmental data sets usually exhibit the spatial nonstationarity. Hence we choose the global land-surface precipitation (January 2010) to evaluate whether our KNN-KFGP can handle the real-life nonstationarity. There are 3306 points in the training set. For each point, the input is the location and the output is the precipitation value. The test set is based on a regular grid with a spatial resolution of $2.5^o \times 2.5^o$ latitude by longitude. As before, we compare our KNN-KFGP to the stationary GP, KFGP and LGP. The notation of K is the same as the robot perception experiment. We train the hyperparameters using gradient optimization with a randomly selected training subset including 1000 points, then run all the methods three times from $K = 1$ to 10. We found that KNN-KFGP shows the best performance when $K = 5$, KFGP and LGP represents the best performance when $K = 10$. The results for all the methods are illustrated in Figure 5 and Table 2. In Figure 5, the prediction of KFGP almost fails due to the random training set selection. In contrast, our KNN-KFGP flexibly learns the land-surface precipitation model. The underlying reason is that each test point is learned by using its strongly correlated training set and the predict-update KF mechanism preserves the accuracy. From Table 2, it is also shown that LGP and our KNN-KFGP are suitable approaches for this spatial nonstationary case, but our KNN-KFGP performs better than LGP. At last, it is worth mentioning that the RT of our KNN-KFGP is 475s which is much more efficient than GP (15374s).

5 Conclusions

In this paper, we have proposed a novel KNN-KFGP for regression. Firstly, small training subsets, which are constructed by the test-driven KNN search, are used to establish a GP prior based state space model. Then a KF is applied to infer the latent function values in a predict-update manner. Compared to other sparse GPs, our KNN-KFGP finds a more accurate solution with higher computational efficiency for large data sets. Moreover, the KNN search provides each test with its strongly correlated training set, so our KNN-KFGP can more suitably solve the nonstationarity than GP. In the future, it will be interesting to explore an adaptive manner to learn K for each test to further improve the flexibility.

References

[Bishop, 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[Deisenroth *et al.*, 2011] Marc Peter Deisenroth, Carl Edward Rasmussen, and Dieter Fox. Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *RSS*, 2011.

[Kalman, 1960] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of basic Engineering*, 82(Series D):35–45, 1960.

[Ko and Fox, 2008] Jonathan Ko and Dieter Fox. Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. In *IROS*, 2008.

[Lang *et al.*, 2007] Tobias Lang, Christian Plagemann, and Wolfram Burgard. Adaptive non-stationary kernel regression for terrain modeling. In *RSS*, 2007.

[Lázaro-Gredilla and Titsias, 2011] Miguel Lázaro-Gredilla and Michalis K. Titsias. Variational heteroscedastic gaussian process regression. In *ICML*, 2011.

[Lázaro-Gredilla *et al.*, 2010] Miguel Lázaro-Gredilla, Joaquin Quinonero-Candela, Carl Edward Rasmussen, and Aníbal R. Figueiras-Vidal. Sparse spectrum gaussian process regression. *Journal of Machine Learning Research*, 11:1865–1881, 2010.

[Paciorek and Schervish, 2004] Christopher J. Paciorek and Mark J. Schervish. Nonstationary covariance functions for gaussian process regression. In *NIPS*, 2004.

[Plagemann *et al.*, 2007] Christian Plagemann, Dieter Fox, and Wolfram Burgard. Efficient failure detection on mobile robots using particle filters with gaussian process proposals. In *IJCAI*, pages 2185–2190, 2007.

[Plagemann *et al.*, 2008] Christian Plagemann, Kristian Kersting, and Wolfram Burgard. Nonstationary gaussian process regression using point estimates of local smoothness. In *ECML*, 2008.

[Plagemann, 2008] Christian Plagemann. *Gaussian processes for flexible robot learning*. PhD thesis, Albert-Ludwigs-Universität Freiburg, 2008.

[Rasmussen and Ghahramani, 2002] Carl Edward Rasmussen and Zoubin Ghahramani. Infinite mixture of gaussian process experts. In *NIPS*, 2002.

[Rasmussen and Williams, 2006] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Process for Machine Learning*. MIT Press, 2006.

[Reece and Roberts, 2010] Steven Reece and Stephen Roberts. An introduction to gaussian processes for the kalman filter expert. In *FUSION*, pages 1–9, 2010.

[Snelson and Ghahramani, 2006] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *NIPS*, pages 1257–1264, 2006.

[Urtasun and Darrell, 2008] Raquel Urtasun and Trevor Darrell. Sparse probabilistic regression for activity-independent human pose inference. In *CVPR*, pages 1–8, 2008.

[Wang and Chaib-draa, 2012] Yali Wang and Brahim Chaib-draa. An adaptive nonparametric particle filter for state estimation. In *ICRA*, pages 4355–4360, 2012.