26th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2022)

Locally Smoothed Gaussian Process Regression

Davit Gogolashvili[*a,1], Bogdan Kozyrskiy[*a], Maurizio Filippone[a]

[a]Department of Data Science, EURECOM, 450 Route des Chappes, 06410 Biot, France

Abstract

We develop a novel framework to accelerate Gaussian process regression (GPR). In particular, we consider localization kernels at each data point to down-weigh the contributions from other data points that are far away, and we derive the GPR model stemming from the application of such localization operation. Through a set of experiments, we demonstrate the competitive performance of the proposed approach compared to full GPR, other localized models, and deep Gaussian processes. Crucially, these performances are obtained with considerable speedups compared to standard global GPR due to the sparsification effect of the Gram matrix induced by the localization operation.

Keywords: Gaussian processes; kernel smoothing; local regression

1. Introduction

Function estimation is a fundamental problem in Machine Learning. In supervised learning tasks applied to a data set composed of observed input data and labels, the goal of function estimation is to establish a mapping between these two groups of observed quantities. Function estimation can be approached in various ways, and we can broadly divide algorithms in two categories, as *global* and *local*. Examples of global algorithms are Neural Networks [25] and kernel machines [33], which impose a functional form yielding a global representation of the function. The functional form is parameterized by a set of parameters which are optimized or inferred based on all the available data. The estimated model can later be used to query the function at any input points of interest. In local algorithms such as K-Nearest Neighbors (KNN), instead, the target point is fixed and the corresponding value of the function is estimated based on the closest data available.

---

[*] Equal contribution

[1] Corresponding author. Tel.: 04 93 00 81 79.

E-mail address: davit.gogolashvili@eurecom.fr

Obviously, any global algorithm can be made local by training it only for the few training points located in the vicinity of the target test point. While it may seem that the idea of localizing global algorithms is not a very profound one, empirical evidence shows that localization could improve the performance of the best global models [5]. The idea of localization was therefore applied to global models such as SVMs [3, 4]. In addition to performance gains, by operating on smaller sets of data points, these local approaches enjoy computational advantages, which are particularly attractive for kernel machines for which scalability with the number of data points is generally an issue [7, 31, 32].

In this work, we develop novel ideas to implement a localization of Gaussian processes (GPs) in order to obtain performance gains, as well as computational ones. GPs are great candidates to benefit from computational speedups given that a naïve implementation requires expensive algebraic computations with the covariance matrix; denoting by n the number of input data, such operations cost $O(n^3)$ operations and require storing $O(n^2)$ elements, hindering the applicability of GPs to data sets beyond a few thousand data points [28]. Another issue with GPs is how to choose a suitable kernel for the problem at hand so as to avoid problems of model misspecification. Both of these issues have been addressed in various ways, by proposing scalable approximations based on inducing points [15] and random features [29, 8], and by composing GPs to obtain a rich and flexible class of equivalent kernels [40].

In this work we explore an alternative way to address scalability and kernel design issues by localizing GPs. In particular, we show how the localization operation leads to a particular form for the localized GP, and what is the effect on the kernel of this model. Furthermore, the localization makes it apparent how to implement the model with considerable gains compared to other approaches to approximate GPs. We demonstrate such performance gains on regression tasks on standard UCI benchmarks [2].

1.1. Related work

Local learning algorithms were introduced by Bottou and Vapnik [5], with the main objective of estimating the optimal decision function for each single testing point. Examples of local learning algorithms include the well-known K-Nearest Neighbor regression [1] and local polynomial regression [12]. These methods provide simple means for solving regression problems for the cases where training data are nonstationary or their size is prohibitively large for building a global model. However, neither of these methods provides ways to quantify uncertainty in predictions, which is a highly desirable feature in cost-sensitive applications.

Gaussian Process Regression (GPR) [30] is a popular nonparametric regression method based on Bayesian principles which provides uncertainty estimates for its predictions. Similarly to other kernel methods (e.g., SVMs and KRR), GPR is a global method, meaning that it takes into account the whole dataset at prediction time. Thus, GPR inherits the computational complexity of global kernel methods, which is prohibitive for large datasets. Among the large class of scalable approximations for GPR, successful ones are based on Random Fourier Features [29] and on sparsification of the Gram matrix induced by the kernel [30].

Random feature approximation of the kernel proposed in [29] is based on Bochner theorem and allows representing the kernel function as a dot product of (possibly infinite) feature maps applied to the input data. In practice, infinite feature maps are replaced by a finite Monte Carlo approximation. The disadvantage of this approach is that it is necessary to construct a specific random feature mapping for each type of kernel. While random feature approximations are known for popular kernels such as RBF [29] and polynomial [26], there is no straightforward application of this method to approximate arbitrary kernels.

The Gram matrix sparsification approach is based on the idea of introducing so-called inducing points in order to approximate the full Gram matrix. One of the most popular methods in this family is the Nyström approximation [30]. The main drawback of this approach is that a low number of inducing points might lead to a poor approximation of the original model, which affects predictive performance. An important advancement within this family of approaches which provides a scalable variational formulation was proposed in [37].

While providing good performance and scalability for large datasets, these approaches still require some design choices for the kernel. For stationary kernels, they assume that the same kernel is suitable for all the regions of input space and if data are nonstationary, this may harm the predictive performance. The literature has a wide range of proposals to address kernel design by incorporating ideas from deep learning [40, 9].

Recently, partitioning strategies have also gained some attention. The main idea is to divide the input space in regions where local estimators are defined [23, 36, 24, 6]. In partition-based methods, the main challenge is to define an effective partitioning of the space.

There are several approaches that use the idea of local learning for training GP models. The method proposed in [22] and extended in [21] mostly focuses on Bayesian parametric linear regression. The methods in these papers build an ensemble of local models centered at several fixed points, where each training point is weighted accordingly to the distance from the center of the model. Predictions are computed as a weighted sum of the local models. The authors claim that their approach extends to GPR, but in this case each local model considers the full training set. This means that these methods use localization to address nonstationarity, but poorly scale to large datasets. The method proposed in [35] proposes to build local GP models that use only subsets of the training data, but it lacks a mechanism that assigns importance weight for the training points for each local model according to the distance from the center of the model. That is why the model can make overconfident predictions for the points that lay far away from the centers of the local models. In [14] in order to obtain fast approximate prediction at a target point, the Authors propose a forward step-wise variable selection procedure to find the optimal sub-design.

The paper is organized as follows: Section 2 introduces GPs and the idea of localization, along with a discussion on the connection between our localized GPs and local kernel ridge regression. The experimental campaign in Section 3 reports results on a variety of benchmarks for regression, and Section 4 concludes the paper.

2. Gaussian Processes, Kernel Ridge Regression, and Localization

2.1. Gaussian Process Regression

A zero-mean Gaussian process (GP) $(f(\mathbf{x}) : \mathbf{x} \in X \subset \mathbb{R}^d)$ is a set of random variables $f(\mathbf{x})$ indexed by the input set X such that for each finite subset $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subset X$ the random vector $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_m))$ is a zero-mean multivariate normal. The finite-dimensional distribution of such a process is determined by the covariance function $K : X \times X \to \mathbb{R}$, defined by

$$K(\mathbf{x}, \mathbf{x}') = E[f(\mathbf{x})f(\mathbf{x}')]. \tag{1}$$

The fact that $f(\mathbf{x})$ is Gaussian process with covariance kernel K is commonly denoted by

$$f(\mathbf{x}) \sim \mathcal{GP}(0, K(\mathbf{x}, \mathbf{x}')).$$

Given a data set $\mathcal{D}$ comprising a set of input-label pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,n}$, GPs can be used as a prior over functions to model the relationship between inputs and labels. The likelihood function for GPs applied to regression tasks can be derived from assuming that

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{2}$$

The GP prior, together with the Gaussian assumption on the relationship between $f(\mathbf{x})$ and y_i induces a multivariate normal distribution over $\mathbf{y} = (y_1, \ldots, y_n)^\top$ as

$$\mathbf{y} \sim \mathcal{N}(0, K_{XX} + \sigma^2 I), \tag{3}$$

where $(K_{XX})_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.

In GP regression it is possible to compute the predictive distribution of the function values at any arbitrary input points $\mathbf{x} \in X$ given $\mathcal{D}$ by means of well-known formulas for conditional distributions of Gaussian random vectors. Given the data set $\mathcal{D}$, it is straightforward to show [30] that the posterior over the function is also a GP

$$f|\mathbf{y}, X \sim \mathcal{GP}(m(\mathbf{x}), \mathcal{K}(\mathbf{x}, \mathbf{x}'))$$

with mean

$$m(\mathbf{x}) = K_{\mathbf{x}X}\left(K_{XX} + \sigma^2 I_n\right)^{-1}\mathbf{y}, \quad \mathbf{x} \in X, \tag{4}$$

and covariance function

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}, \mathbf{x}') - K_{\mathbf{x}X}\left(K_{XX} + \sigma^2 I_n\right)^{-1} K_{X\mathbf{x}'}, \quad \mathbf{x}, \mathbf{x}' \in X, \tag{5}$$

where $K_{X\mathbf{x}} = K_{\mathbf{x}X}^\top = (K(\mathbf{x}_1, \mathbf{x}), \ldots, K(\mathbf{x}_n, \mathbf{x}))^\top$.
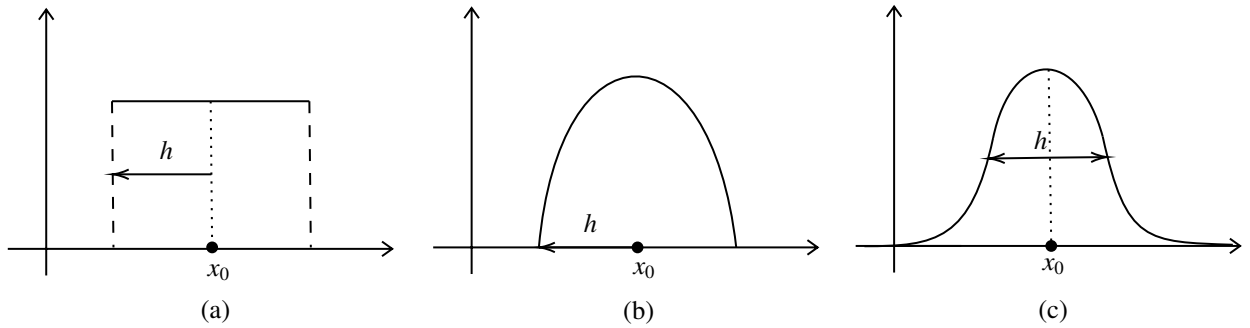
Fig. 1. Examples of local kernels: (a) Rectangular kernel $k(\mathbf{x}) = \mathbb{I}(\|\mathbf{x}\| \leq 1)$. (b) Epanechnikov kernel $k(\mathbf{x}) = \frac{d+2}{2V_d}(1 - \|\mathbf{x}\|^2)\mathbb{I}(\|\mathbf{x}\| \leq 1)$. (c) Gaussian kernel $k(\mathbf{x}) = \frac{1}{2\pi}e^{-\|\mathbf{x}\|^2}$.

The problem with the expressions above is that they require solving a linear system involving a matrix of size $n \times n$. Direct methods to solve these operations require $O(n^3)$ operations and storing $O(n^2)$. Iterative solvers, instead, can reduce these complexities by relying exclusively on matrix-vector products, which require $O(n^2)$ operation per iteration and do not need to store the Gram matrix [13, 10]. However, a quadratic time complexity may still prohibitive for large-scale problems.

There is a rich literature on approaches that recover tractability by introducing approximations. One popular line of work introduces m so-called inducing-points as a means to approximate the whole GP prior [28]. This treatment of GPs was later extended within a scalable variational framework [37, 19], making the complexity cubic in the number of inducing points m. Another approach, proposes ways to linearize GPs by obtaining an explicit set of features so as to obtain a close approximation to the original kernel-based model. Within this framework, a popular approach is based on random features [29]. Denoting by Φ the $n \times D$ matrix obtained by applying a set of D random basis functions to the inputs in $\mathbf{x}_1 \ldots, \mathbf{x}_n$, these approximations are so that $\Phi\Phi^\top$ approximates in an unbiased way the original kernel matrix K_{XX}, that is $E[\Phi\Phi^\top] = K_{XX}$. For the Gaussian kernel, for example, a Fourier analysis shows that the basis functions that satisfy this property are trigonometric functions with random frequencies [29]. This approach has been applied to GPs in [20] and later made scalable by operating on mini-batches in [8].

2.2. Locally Smoothed Gaussian Process Regression

GP regression as formulated above is an example of a *global learner*; for a fixed training data $\mathcal{D}$ it builds a posterior distribution over functions that can be used to calculate the predictive distribution for any test inputs. To construct a predictive distribution for *any* input point it is necessary to use all the information available from the data $\mathcal{D}$, resulting in the need to do algebraic operations with the matrix $(K_{XX} + \sigma^2 I_n)$. However, if we focus on the prediction problem locally, at a *given* target input $\mathbf{x}_0$, most of the information carried by the (potentially large) covariance matrix might be neglected with little loss of information. The main idea behind localized GPs is to down-weight the contribution of the data points far from $\mathbf{x}_0$, so that the structure of the covariance matrix is more adapted to the prediction task at a given point. To make this general idea work, we need to tackle two challenges. First, we need to specify what it means to be far or close to a given point; second, the change of the structure of the covariance matrix must give us a valid covariance matrix, i.e., the resulting covariance matrix should be symmetric and positive definite. Note that a simple truncation of the covariance function to obtain a compact-support covariance function may generally destroy positive definiteness [17].

We accomplish localization of GPs in a straightforward manner as follows. We localize the target and the prior in the model (2) by multiplying them by the square root of the weighting function

$$k_h(\mathbf{x}, \mathbf{x}_0) := \frac{1}{h}k\left(\frac{\|\mathbf{x} - \mathbf{x}_0\|}{h}\right),$$

where $k : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}$ is a non-negative, integrable function satisfying $\int K(\mathbf{x})d\mathbf{x} = 1$ and $\|\cdot\|$ is Euclidean norm on $\mathbb{R}^d$. Considering the square root of the weighting function will be convenient later when we discuss the link between
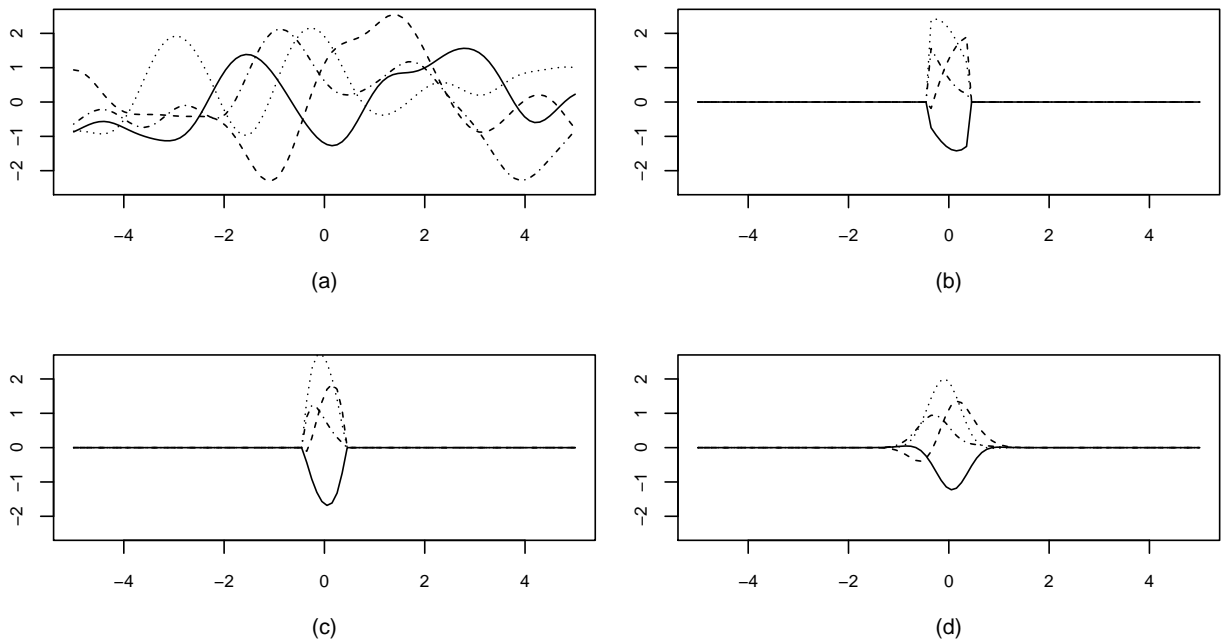
Fig. 2. (a) Samples from a global GP prior with exponential kernel. (b) Samples from a GP prior with exponential kernel localized by rectangular smoother centered at $x_0 = 0$. (c) Samples from a GP prior with exponential kernel localized by Epanechnikov smoother centered at $x_0 = 0$. (d) Samples from a GP prior with exponential kernel localized by Gaussian smoother centered at $x_0 = 0$.

local GPs and local Kernel Ridge Regression. Some classical examples of the weighting functions are given in Fig. 1. Because of the linearity of the weighting operation, the resulting model is another zero-mean Gaussian process $\tilde{f}(\mathbf{x})$ with covariance function given by

$$\tilde{K}(\mathbf{x}, \mathbf{x}'; \mathbf{x}_0) = k_h^{\frac{1}{2}}(\mathbf{x}, \mathbf{x}_0) K(\mathbf{x}, \mathbf{x}') k_h^{\frac{1}{2}}(\mathbf{x}', \mathbf{x}_0). \tag{6}$$

In this formulation, we have localized the relationship between noisy targets and function realizations as

$$\tilde{y}_i = \tilde{f}(\mathbf{x}_i) + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2), \tag{7}$$

with $\tilde{y}_i = \sqrt{k_h(\mathbf{x}_i, \mathbf{x}_0)} y_i$, and the prior is given by a zero-mean GP with the localized covariance kernel (6). The model (7) can be alternatively written as a model with heteroscedastic noise

$$y_i = f(\mathbf{x}_i) + \frac{1}{\sqrt{k_h(\mathbf{x}_i, \mathbf{x}_0)}} \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2).$$

Making the noise parameter location-dependent can significantly improve the performance for problems where the assumption of a homoscedastic noise is not satisfied.

Proposition 1. *Let $I = \{i : \|\mathbf{x}_i - \mathbf{x}_0\| \leq h\}$, $X_I = \{\mathbf{x}_i : i \in I\}$ and $\mathbf{y}_I = \{y_i\}_{i \in I} \in \mathbb{R}^{|I|}$. Assume that (7) holds for the fixed target point $\mathbf{x}_0$. Then $f(\mathbf{x}_0) \mid \mathbf{y}_I$ is a Gaussian random variable with mean and variance given by*

$$\tilde{m}(\mathbf{x}_0) = K_{\mathbf{x}_0 X_I} \left(K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right)^{-1} \mathbf{y}_I \tag{8}$$

$$\tilde{\mathcal{K}}(\mathbf{x}_0, \mathbf{x}_0) = K(\mathbf{x}_0, \mathbf{x}_0) - K_{\mathbf{x}_0 X_I} \left(K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right)^{-1} K_{X_I \mathbf{x}_0} \tag{9}$$

where $W_{\mathbf{x}_0}$ is the diagonal matrix with main diagonal entries $k_h(\mathbf{x}_i, \mathbf{x}_0)$, $\mathbf{x}_i \in X_I$.

Proof: Let $\mathbf{x}_0$ be any fixed target point. Then the observations $\mathbf{y}_I \in \mathbb{R}^{|I|}$ and GP-function value at target point $f_0 = f(\mathbf{x}_0) \in \mathbb{R}$ are jointly Gaussian such that

$$\begin{bmatrix} \mathbf{y} \\ f_0 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0}_I \\ 0 \end{bmatrix}, \begin{pmatrix} K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} & K_{X_I \mathbf{x}_0} \\ K_{\mathbf{x}_0 X_I} & K(\mathbf{x}_0, \mathbf{x}_0) \end{pmatrix} \right).$$

Then the proposition follows from the basic formula for conditional distributions of Gaussian random vectors (see, e.g., [30], Appendix A.2).

Compared to global GPs, in the local formulation, in order to compute the posterior mean and variance, we need to invert $\left(K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right)$. This might give a key advantage when dealing with large data sets, as the localization by compactly supported kernel (local) could significantly sparsify the Gram matrix corresponding to K_{XX}. Denoting by s_0 the number of inputs for which the localizing weights are nonzero for a test point $\mathbf{x}_0$, the complexity of performing such an inversion is $O(s_0^3)$. Another interesting observation is that the kernel function $\tilde{K}(\mathbf{x}, \mathbf{x}')$ is potentially more flexible than the original kernel function; this is due to the multiplication by the localizing weighting function, which may introduce some interesting nonstationarity even for kernel functions which are stationary, depending on the choice of the weighting function.

The calculation of the predictive distribution in with Locally Smoothed Gaussian Process Regression (LSGPR) is described in Algorithm 1 The parameter selection in probabilistic models given by GPs is based on the *marginal log-likelihood* maximization, which, in our local formulation, can be defined as follows

$$\log p(\mathbf{y}_I | X_I) = -\frac{1}{2} \mathbf{y}_I^\top \left(K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right)^{-1} \mathbf{y}_I - \frac{1}{2} \log \left| K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right| - \frac{n}{2} \log(2\pi)$$

Unfortunately, gradient-based optimization cannot be used to find the optimal localization parameter h, as the marginal log-likelihood is not continuously differentiable w.r.t. this parameter when compactly supported local kernels are used. The simplest way to resolve this problem is by using grid search for the localization parameter h, while kernel parameters can be optimized by gradient-based methods for any given h.

Algorithm 1 LSGPR

1: **Input:** X, $\mathbf{y}$, σ^2, h, $\mathbf{x}_0$
2: **Output:** $\tilde{m}(\mathbf{x}_0)$, $\tilde{\mathcal{K}}(\mathbf{x}_0, \mathbf{x}_0)$
3: $I := \{i : \|\mathbf{x}_i - \mathbf{x}_0\| \le h\}$
4: $X_I := \{\mathbf{x}_i : i \in I\}$
5: $\mathbf{y}_I := \{y_i : i \in I\}$
6: $W_{\mathbf{x}_0} := \mathrm{diag}(\{k_h(\mathbf{x}_i, \mathbf{x}_0) : i \in I\})$
7: L:=Cholesky$\left(K_{X_I X_I} + \sigma^2 W_{\mathbf{x}_0}^{-1} \right)$
8: $\boldsymbol{\alpha} := (L^{-1})^\top L^{-1} \mathbf{y}$
9: $\tilde{m}(\mathbf{x}_0) := K_{\mathbf{x}_0, X_I} \boldsymbol{\alpha}$
10: $\mathbf{v} := L^{-1} K_{\mathbf{x}_0, X_I}$
11: $\tilde{\mathcal{K}}(\mathbf{x}_0, \mathbf{x}_0) := K(\mathbf{x}_0, \mathbf{x}_0) - \mathbf{v}^\top \mathbf{v}$

2.3. Local Kernel Ridge Regression

For every Gaussian process $f(\mathbf{x})$ with covariance function $K(\mathbf{x}, \mathbf{x}')$ there is a unique corresponding Hilbert space $\mathcal{H}_K$. This is commonly referred to as a *reproducing kernel Hilbert space* (RKHS) and constructed as a completion of the linear space of all functions

$$\mathbf{x} \mapsto \sum_{i=1}^{k} \alpha_i K(\mathbf{a}_i, \mathbf{x}), \quad \alpha_1, \dots, \alpha_k \in \mathbb{R}, \mathbf{a}_1, \dots, \mathbf{a}_k \in \mathcal{X}, k \in \mathbb{N}$$

relative to the norm induced by the inner product

$$
\left\langle \sum_{i=1}^{k} \alpha_i K\left(\mathbf{s}_i, \cdot\right), \sum_{j=1}^{l} \beta_j K\left(\mathbf{t}_j, \cdot\right) \right\rangle_{\mathcal{H}_K} = \sum_{i=1}^{k} \sum_{j=1}^{l} \alpha_i \beta_j K\left(\mathbf{s}_i, \mathbf{t}_j\right).
$$

It is well know that the posterior mean of Gaussian process regression can be alternatively derived by minimizing the regularized empirical risk over the RKHS [18]; see, e.g., [16] for a recent review. For local GPs, this corresponds to a weighted least square minimization over the RKHS with the weights given by $k_h(\mathbf{x}, \mathbf{x}_0)$, that is

$$
\tilde{m}(\mathbf{x}) = \arg\min_{f \in \mathcal{H}_k} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 \, k_h\left(\mathbf{x}_i, \mathbf{x}_0\right) + \frac{\sigma^2}{n} \|f\|_{\mathcal{H}_k}^2. \tag{10}
$$

Note that in the local formulation, for a given point $\mathbf{x}_0$ one has to estimate both the parameters of the reproducing kernel and the width of the local kernel *h*. Here are two examples of well-known classical local methods which are the solution of the empirical risk minimization problem (10).

K-nearest neighbors. This model corresponds to the noise-free case ($\sigma = 0$) with a positive constant reproducing kernel and a rectangular local kernel whose width is adjusted to contain exactly *k* data points. The solution of the minimization problem (10) is the mean of the outputs corresponding to the *k* closest to $\mathbf{x}_0$ input points.

Local polynomial regression. If we use the polynomial kernel $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}\mathbf{x}')^k$ for the space $\mathcal{H}_K$, and use any smooth local kernel (i.e. exponential), then in the noise-free case the solution of the minimization problem (10) is so called local polynomial regression [38]. In this special case, when the degree of the polynomial is 0, we have Nadaraya-Watson regression, which is the minimizer of the local squared loss over the constant function.

3. Experiments

3.1. Toy dataset

In order to illustrate the behavior of the proposed Locally Smoothed Gaussian Process (LSGPs), we start from a toy dataset generated from the Doppler function (Fig. 3).

$$
y(x) = \sqrt{x(1-x)} \sin\left(\frac{2.1\pi}{x + 0.05}\right) + \varepsilon, \quad 0 \le x \le 1, \quad \varepsilon \sim \mathcal{N}(0, 0.1) \tag{11}
$$

For this experiment, we used the RBF kernel and the Epanechnikov localizing kernel

$$
k(x) = \frac{3}{4}(1 - |x|^2)\mathbb{I}(|x| \le 1). \tag{12}
$$

We tuned the lengthscale parameter of the RBF kernel by optimizing the marginal log-likelihood of the model. We used the L-BFGS algorithm for gradient optimization [27]. We chose the value of the parameter *h* of the localizing kernel that gave the best Mean Squared Error (MSE) on a validation set. The LSGP model used on average 7 training points to make a prediction. We compared the predictions of LSGP with the predictions of standard GP regression.

As we can see from Fig. (3), the Gaussian Process with RBF kernel is unable to make reasonable predictions in the region where the target function contains high frequency components. While some nonstationary covariance functions might be appropriate for this example, the combination of a standard stationary covariance function with the localization approach offers substantial modeling improvements.

3.2. UCI datasets

We evaluated the performance of the LSGP method on several problems from the UCI datasets collection and compared it against standard GPR, Deep GPs approximated with random features [8], and k-nearest neighbors (KNN) regression. In particular, we aim to compare the predictive performance offered by the localization against the baseline
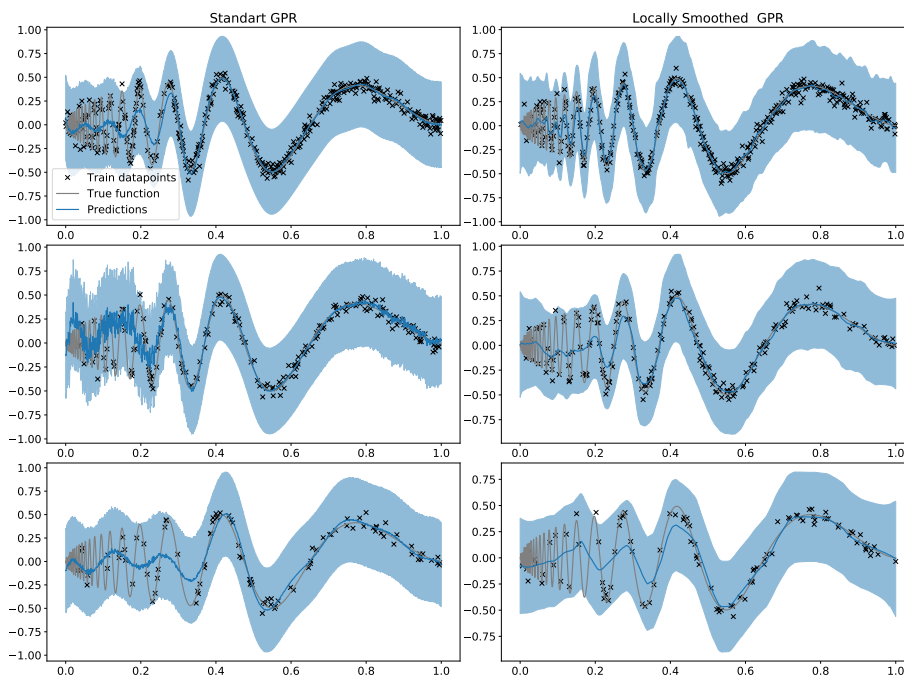
Fig. 3. Illustration of the predictive distribution of GPs (left) and LSGPs (right) applied to data sampled from the Doppler function for 400 training points (top), 200 training points (middle) and 100 training points.

Table 1. UCI datasets used for evaluation.

Dataset	Training instances	Dimensionality
Yacht	308	6
Boston	506	13
Concrete	1030	8
Kin8nm	8192	8
Powerplant	9568	4
Protein	45730	9

of exact GPR, and to verify that any performance gains are not just due to localization, meaning that we expect to outperform KNN. Because our model is more flexible than standard GPR, we also added to the comparison Deep GP models based on random features expansion. The size and dimensionality of these problems is outlined in Table 1. Since the Euclidean norm used in the local kernels depends on the units in each coordinate, all the datasets except the Protein were scaled within the [0, 1] range. We used a standardization procedure for the Protein dataset because the baseline model worked much better with this type of preprocessing. For the Deep GP model we used the hyperparameters and the data preprocessing described in the original paper.

The LSGPR method requires creating a new local model with its own set of hyperparameters for each input point where the prediction has to be made. During the optimization, the kernel parameters of each model were constrained to be equal among all local models. Considering the hyperparameter h, we found that it is hard to find values of h which perform well across all regions of the input space. Thus, for each input point of interest, we chose values of

Table 2. Comparison in terms of test set MSE between LSGPR a standard GPR.

Dataset	LSGPR Hilbert	LSGPR Epanechnikov	GP	DeepGP	KNN
Yacht	**0.63±0.12**	2.02±0.58	1.09±0.05	0.93±0.13	57.80±16.65
Boston	14.78±0.88	15.30±1.28	17.94±0.71	**7.92±0.14**	23.30±2.58
Concrete	**34.79±1.07**	40.43±3.16	37.81±0.61	130.94±3.93	94.23±7.89
Kin8nm	0.01±0.000	0.01±0.00	0.01±0.000	0.06±0.00	0.01±0.00
Powerplant	14.65±0.34	**14.40±0.67**	16.85±0.69	14.57±0.15	15.35±0.49
Protein	36.85±2.15	**12.50±0.26**	17.03±0.57	16.94 ±0.16	19.89±4.26

h that ensured that the localizer considers at least m neighboring training points. In this experiment, we used 3-fold cross-validation to choose the noise variance σ^2, the lengthscale of the GP kernel and the parameter m of the localizing kernel. We report the results on the held-out test set.

In this experiment, we also used the Hilbert localizing kernel [34, 11]

$$k(\mathbf{x}) = \|\mathbf{x}\|^{-1} \mathbb{I}(\|\mathbf{x}\| \leq 1), \tag{13}$$

which showed good performance for most of the datasets. In Table 2 locally smoothed Gaussian Process Regression based on Hilbert kernel is referred to as LSGPR Hilbert, while the same model based on Epanechnikov kernel is referred as LSGPR Epanechnikov. GP regression, Deep GP regression and KNN regression are referred to as GP, DeepGP and KNN, respectively.

The results indicate that LSGP offers competitive performance with respect to GPR and Deep GP baselines. The results also clearly show that LSGP offers superior performance to KNN, suggesting that the localization alone is not enough to obtain good performance, and that this works well in combination with the GP model. To make a comparison with the baselines, we used the one-sided Wilcoxon test [39]. For each method, we measured its performance on 10 data splits, and we used exactly the same splits for testing performance of each method, so we had matched samples of the MSEs. Then we used the test to compare methods in pairs, where the alternative hypothesis was that the MSE of any given method is smaller than the MSE of a competitors. We used confidence level $\alpha = 0.05$. In Table 2 the results that are statistically better than the competitors are marked in bold.

4. Conclusions

In this work we developed a novel framework to localize Gaussian processes (GPs). We focused in particular on Gaussian Process Regression (GPR), and we derived the GP model after applying the localization operation through the down-weighting of contributions from input points which are far away from a given test point. The form of the localized GP maintains positive definiteness of the covariance, and it allows for considerable speedups compared to standard global GPR due to the sparsification effect of the Gram matrix.

The proposed method requires cross-validation to tune the scale parameter of localizing kernel, while others GP-based techniques use a less expensive marginal log-likelihood (MLL) gradient optimization to tune these types of parameters. We found MLL gradient optimization problematic because of the discontinuity of local kernel with respect to the scale parameter, which in turn makes MLL function discontinuous with respect to this parameter. It would be interesting to investigate ways to extend the idea of localization for GPR to other tasks, such as classification.

References

[1] Altman, N.S., 1992. An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician 46, 175–185.
[2] Asuncion, A., Newman, D.J., 2007. UCI machine learning repository.
[3] Blanzieri, E., Melgani, F., 2006. An adaptive SVM nearest neighbor classifier for remotely sensed imagery, in: 2006 IEEE International Symposium on Geoscience and Remote Sensing, IEEE. pp. 3931–3934.
[4] Blanzieri, E., Melgani, F., 2008. Nearest neighbor classification of remote sensing images with the maximal margin principle. IEEE Transactions on geoscience and remote sensing 46, 1804–1811.
[5] Bottou, L., Vapnik, V., 1992. Local learning algorithms. Neural computation 4, 888–900.

[6] Carratino, L., Vigogna, S., Calandriello, D., Rosasco, L., 2021. Park: Sound and efficient kernel ridge regression by feature space partitions. Advances in Neural Information Processing Systems 34.

[7] Cheng, H., Tan, P.N., Jin, R., 2007. Localized support vector machine and its efficient algorithm, in: Proceedings of the 2007 SIAM International Conference on Data Mining, SIAM. pp. 461–466.

[8] Cutajar, K., Bonilla, E.V., Michiardi, P., Filippone, M., 2017a. Random feature expansions for deep Gaussian processes, in: Proceedings of the 34th International Conference on Machine Learning, PMLR, International Convention Centre, Sydney, Australia. pp. 884–893.

[9] Cutajar, K., Bonilla, E.V., Michiardi, P., Filippone, M., 2017b. Random feature expansions for deep Gaussian processes, in: International Conference on Machine Learning, PMLR. pp. 884–893.

[10] Cutajar, K., Osborne, M., Cunningham, J., Filippone, M., 2016. Preconditioning Kernel Matrices, in: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016, JMLR.org. pp. 2529–2538.

[11] Devroye, L., Györfi, L., Krzyżak, A., 1998. The Hilbert kernel regression estimate. Journal of Multivariate Analysis 65, 209–227.

[12] Fan, J., Gijbels, I., 2018. Local polynomial modelling and its applications. Routledge.

[13] Filippone, M., Engler, R., 2015. Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE), in: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, July 6-11, 2015, pp. 1015–1024.

[14] Gramacy, R.B., Apley, D.W., 2015. Local Gaussian process approximation for large computer experiments. Journal of Computational and Graphical Statistics 24, 561–578.

[15] Hensman, J., Fusi, N., Lawrence, N.D., 2013. Gaussian processes for big data, in: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, AUAI Press. p. 282–290.

[16] Kanagawa, M., Hennig, P., Sejdinovic, D., Sriperumbudur, B.K., 2018. Gaussian processes and kernel methods: A review on connections and equivalences. arXiv preprint arXiv:1807.02582 .

[17] Kaufman, C.G., Schervish, M.J., Nychka, D.W., 2008. Covariance tapering for likelihood-based estimation in large spatial data sets. Journal of the American Statistical Association 103, 1545–1555.

[18] Kimeldorf, G.S., Wahba, G., 1970. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. The Annals of Mathematical Statistics 41, 495–502.

[19] Krauth, K., Bonilla, E.V., Cutajar, K., Filippone, M., 2017. AutoGP: Exploring the capabilities and limitations of Gaussian process models, in: Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, August 11-15, 2017, Sydney, Australia.

[20] Lázaro-Gredilla, M., Quinonero-Candela, J., Rasmussen, C.E., Figueiras-Vidal, A.R., 2010. Sparse Spectrum Gaussian Process Regression. Journal of Machine Learning Research 11, 1865–1881.

[21] Meier, F., Hennig, P., Schaal, S., 2014a. Incremental local Gaussian regression. Advances in Neural Information Processing Systems 27.

[22] Meier, F., Hennig, P., Schaal, S., 2014b. Local Gaussian regression. arXiv preprint arXiv:1402.0645 .

[23] Meister, M., Steinwart, I., 2016. Optimal learning rates for localized SVMs. The Journal of Machine Learning Research 17, 6722–6765.

[24] Müecke, N., 2019. Reducing training time by efficient localized kernel regression, in: The 22nd International Conference on Artificial Intelligence and Statistics, PMLR. pp. 2603–2610.

[25] Neal, R.M., 1996. Bayesian Learning for Neural Networks (Lecture Notes in Statistics). 1 ed., Springer.

[26] Pennington, J., Yu, F.X.X., Kumar, S., 2015. Spherical random features for polynomial kernels, in: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.

[27] Pytlak, R., 2008. Conjugate gradient algorithms in nonconvex optimization. volume 89. Springer Science & Business Media.

[28] Quinonero-Candela, J., Rasmussen, C.E., 2005. A unifying view of sparse approximate Gaussian process regression. The Journal of Machine Learning Research 6, 1939–1959.

[29] Rahimi, A., Recht, B., 2008. Random Features for Large-Scale Kernel Machines, in: Platt, J.C., Koller, D., Singer, Y., Roweis, S.T. (Eds.), Advances in Neural Information Processing Systems 20. Curran Associates, Inc., pp. 1177–1184.

[30] Rasmussen, C.E., Williams, C., 2006. Gaussian Processes for Machine Learning. MIT Press.

[31] Segata, N., Blanzieri, E., 2010. Fast and scalable local kernel machines. Journal of Machine Learning Research 11.

[32] Segata, N., Pasolli, E., Melgani, F., Blanzieri, E., 2012. Local SVM approaches for fast and accurate classification of remote-sensing images. International Journal of Remote Sensing 33, 6186–6201.

[33] Shawe-Taylor, J., Cristianini, N., 2004. Kernel Methods for Pattern Analysis. Cambridge University Press, New York, NY, USA.

[34] Shepard, D., 1968. A two-dimensional interpolation function for irregularly-spaced data, in: Proceedings of the 1968 23rd ACM national conference, pp. 517–524.

[35] Snelson, E., Ghahramani, Z., 2007. Local and global sparse Gaussian process approximations, in: Artificial Intelligence and Statistics, PMLR. pp. 524–531.

[36] Tandon, R., Si, S., Ravikumar, P., Dhillon, I., 2016. Kernel ridge regression via partitioning. arXiv preprint arXiv:1608.01976 .

[37] Titsias, M.K., 2009. Variational learning of inducing variables in sparse Gaussian processes, in: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009, JMLR.org. pp. 567–574.

[38] Tsybakov, A., 2009. Introduction to Nonparametric Estimation. Springer, New York.

[39] Wilcoxon, F., 1945. Individual Comparisons by Ranking Methods. Biometrics Bulletin 1, 80–83.

[40] Wilson, A.G., Hu, Z., Salakhutdinov, R., Xing, E.P., 2016. Deep Kernel Learning, in: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, PMLR, Cadiz, Spain. pp. 370–378.