Journal of Machine Learning Research 23 (2022) 1-34 Submitted 7/19; Revised 9/20; Published 2/22

Sparse Additive Gaussian Process Regression

Hengrui Luo LUO.619@OSU.EDU
Department of Statistics
The Ohio State University
Columbus, OH 43210, USA

Giovanni Nattino NATTINO.1@OSU.EDU
Division of Biostatistics, College of Public Health
The Ohio State University
Columbus, OH 43210, USA

Matthew T. Pratola MPRATOLA@STAT.OSU.EDU
Department of Statistics
The Ohio State University
Columbus, OH 43210, USA

Editor: Robert McCulloch

Abstract

In this paper we introduce a novel model for Gaussian process (GP) regression in the fully Bayesian setting. Motivated by the ideas of sparsification, localization and Bayesian additive modeling, our model is built around a recursive partitioning (RP) scheme. Within each RP partition, a sparse GP (SGP) regression model is fitted. A Bayesian additive framework then combines multiple layers of partitioned SGPs, capturing both global trends and local refinements with efficient computations. The model addresses both the problem of efficiency in fitting a full Gaussian process regression model and the problem of prediction performance associated with a single SGP. Our approach mitigates the issue of pseudo-input selection and avoids the need for complex inter-block correlations in existing methods. The crucial trade-off becomes choosing between many simpler local model components or fewer complex global model components, which the practitioner can sensibly tune. Implementation is via a Metropolis-Hasting Markov chain Monte-Carlo algorithm with Bayesian back-fitting. We compare our model against popular alternatives on simulated and real datasets, and find the performance is competitive, while the fully Bayesian procedure enables the quantification of model uncertainties.

Keywords: Sparse Gaussian Process, Recursive Partition Scheme, Bayesian Additive Model, Nonparametric Regression.

1. Introduction

Gaussian process (GP) regression is a widely adopted regression model (Rasmussen and Williams, 2006). Taking a Bayesian approach, its posterior distribution provides a principled way to quantify uncertainties while having nice theoretical properties (Gelman et al., 2013). However, the computational cost of GP likelihood evaluations based on an observed dataset $\{y, \mathcal{X}\}$ of size n is of order $\mathcal{O}(n^3)$, which primarily results from the need to invert an $n \times n$ covariance matrix. Therefore, the computational cost could be prohibitively high in scenarios where large datasets need to be analyzed. It is a focus of much current research to solve this problem of high computational cost for GP regression (Banerjee et al., 2012; Liu et al., 2020).

Many approaches to circumvent this problem have been explored, such as low-rank covariance approximation (Titsias, 2009), model likelihood approximations (Kaufman et al., 2008) and local GP approximations (Snelson and Ghahramani, 2007; Gramacy and Apley, 2015). However, most of these approaches are not fully Bayesian.

We are inspired by the idea of low-rank sparse GP regression (Snelson and Ghahramani, 2006) and localization ideas (Chipman et al., 1998; Lee et al., 2017; Gramacy and Apley, 2015; Park and Huang, 2016; Nguyen-Tuong et al., 2009; Chipman et al., 2016; Lee et al., 2017), but we still want to incorporate these methods within a fully Bayesian framework. Borrowing the framework of Bayesian (generalized) additive modeling (Hastie and Tibshirani, 1990, 2000), we propose the Sparse Additive Gaussian Process (SAGP) model. SAGP combines sparse GP regression and a recursive partition (RP) scheme within a fully Bayesian model. It turns out that our approach can simultaneously handle both local and global features in large datasets while realizing gains in computational efficiency. Furthermore, it provides principled uncertainty quantification for parameters and posterior predictions. A key feature of the approach is a much simplified fixed partitioning scheme that avoids the added computational costs of stochastic tree-based partitioning models (e.g. (Chipman et al., 1998, 2016; Gramacy et al., 2007)). To the best of our knowledge, this kind of additive Bayesian model, combining both sparsification and localization, has never been explored.

The paper is organized as it follows. In section 2 we will briefly review the background knowledge for sparse GP, localization and Bayesian additive modeling as they are essential ingredients of SAGP modeling. In section 3 we will specify the SAGP model. Sections 4 and 5 are analyses of simulated and real-world datasets. Finally, we conclude our paper with a discussion in section 6.

2. Background

2.1 Gaussian Process Regression

We start with GP regression on the input domain X and use the notation $N_d(\boldsymbol{m}, \Sigma)$ to denote the d-dimensional Gaussian distribution with mean vector $\boldsymbol{m}$ and covariance matrix Σ, and the notation $N_d(\boldsymbol{y} \mid \boldsymbol{m}, \Sigma)$ to denote the d-dimensional Normal density evaluated at $\boldsymbol{y} \in \mathbb{R}^n$. The prior of the mean regression function is assumed to be a GP with known mean and covariance kernel function. Posterior estimation and prediction arise from combining the prior belief with the information contained in the likelihood of response variables $\boldsymbol{y} =$

$(y_1, \ldots, y_n)^T$, observed at known input locations $\mathcal{X} = \{\boldsymbol{x}_i \in \mathbb{R}^d, i = 1, \ldots, n\} \subset \mathsf{X}$, by using Bayes theorem. We also call f the target and the variable $\boldsymbol{x}_i$ the input, based on the model form

$$\begin{aligned} y(\boldsymbol{x}_i) &= f(\boldsymbol{x}_i) + \epsilon_i, i = 1, \ldots, n \\ \epsilon_i &\sim N_1(0, \sigma_\epsilon^2) \end{aligned} \tag{1}$$

which expresses the relationship between input $\boldsymbol{x}_i$ and the unknown response $f(\boldsymbol{x}_i)$ observed as y_i with observational error ϵ_i having variance σ_ϵ^2. Using vector notations we write $\boldsymbol{y} = (y_1, \ldots, y_n)^T = (y(\boldsymbol{x}_1), y(\boldsymbol{x}_2), \ldots, y(\boldsymbol{x}_n))^T$, $\boldsymbol{f} = (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n))^T$ and the noise $\boldsymbol{\epsilon} \sim N_n(\boldsymbol{0}_n, \sigma_\epsilon^2 \boldsymbol{I}_n)$ to yield $\boldsymbol{y} = \boldsymbol{f} + \boldsymbol{\epsilon}$.

Without loss of generality, it is often convenient to assume that the mean vector $\boldsymbol{f}$ is a realization of a zero mean Gaussian process, $\boldsymbol{f} \sim N_n(\boldsymbol{0}, \boldsymbol{K}_n)$, where $\boldsymbol{K}_n = [K(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^n$, with covariance kernel $K(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ encoding assumed properties of the unknown function f to satisfy the application of interest (Rasmussen and Williams, 2006).

2.2 Sparsification of Gaussian Processes

There are a variety of sparse approximation approaches to GP regression (e.g. Lawrence et al. 2003; Quinonero-Candela and Rasmussen 2005). A popular approach is the pseudo-input (or latent variable) approach. By replacing the exact covariance matrix in the likelihood computation with a low-rank approximation, one can greatly reduce computational cost. Snelson and Ghahramani (2006) propose the Sparse Gaussian Process (SGP) model by using a subset of the full inputs $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ as pseudo-inputs, denoted as $\bar{\mathcal{X}} = \{\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_m\} \subset \mathcal{X}$, for $m \ll n$. Then, $\bar{\boldsymbol{f}} = (f(\bar{\boldsymbol{x}}_1), f(\bar{\boldsymbol{x}}_2), \ldots, f(\bar{\boldsymbol{x}}_m))^T$ are called pseudo-targets, and

$$\begin{aligned} \boldsymbol{K}_n &\coloneqq [K(\boldsymbol{x}_k, \boldsymbol{x}_l)]_{k,l=1}^n, \\ \boldsymbol{K}_m &\coloneqq [K(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{x}}_l)]_{k,l=1}^m, \\ \boldsymbol{K}_{nm} &= [K(\boldsymbol{x}_i, \bar{\boldsymbol{x}}_j)]_{i,j=1}^{n,m} = \boldsymbol{K}_{mn}^T \end{aligned}$$

denote the (cross-)covariance matrices among and between the full targets $\boldsymbol{f}$ and pseudo-targets $\bar{\boldsymbol{f}}$. Their approach treats the pseudo-inputs as (hyper-)parameters, resulting in a likelihood function that only requires the inversion of the dense $m \times m$ matrix K_m, a significant computational savings. The posterior and posterior predictive distributions can then be written in closed form by Gaussian conjugacy (Snelson and Ghahramani, 2006).

For an SGP model with m pseudo-inputs, the full likelihood is $P(\boldsymbol{y} \mid \mathcal{X}, \bar{\mathcal{X}}, \bar{\boldsymbol{f}}, \sigma_\epsilon^2) = N_n(\boldsymbol{y} \mid \boldsymbol{K}_{nm}\boldsymbol{K}_m^{-1}\bar{\boldsymbol{f}}, \boldsymbol{\Lambda} + \sigma_\epsilon^2 \boldsymbol{I}_n)$ where $\boldsymbol{\Lambda} = \operatorname{diag}\left(K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \boldsymbol{k}_i^T \boldsymbol{K}_m^{-1} \boldsymbol{k}_i\right)_{i=1}^n$ and $\boldsymbol{k}_i = (K(\bar{\boldsymbol{x}}_1, \boldsymbol{x}_i), \ldots, K(\bar{\boldsymbol{x}}_m, \boldsymbol{x}_i))^T$. Using Bayes theorem, we can write the posterior distribution of pseudo-targets as

$$P(\bar{\boldsymbol{f}} \big| \mathcal{X}, \boldsymbol{y}, \bar{\mathcal{X}}, \sigma_\epsilon^2) = N_m(\bar{\boldsymbol{f}} \mid \boldsymbol{K}_m \boldsymbol{Q}_m^{-1} \boldsymbol{K}_{mn} \left(\boldsymbol{\Lambda} + \sigma_\epsilon^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y},\ \boldsymbol{K}_m \boldsymbol{Q}_m^{-1} \boldsymbol{K}_m)$$

where $\boldsymbol{Q}_m = \boldsymbol{K}_m + \boldsymbol{K}_{mn} \left(\boldsymbol{\Lambda} + \sigma_\epsilon^2 \boldsymbol{I}\right)^{-1} \boldsymbol{K}_{nm}$. The posterior predictive distribution for y^* at a new input $\boldsymbol{x}^*$, after integrating out the pseudo-target $\bar{\boldsymbol{f}}$, can be written as $P(y_* \mid \boldsymbol{x}_*, \mathcal{X}, \boldsymbol{y}, \sigma_\epsilon^2) = N_1(\boldsymbol{k}_*^T \boldsymbol{Q}_m^{-1} \boldsymbol{K}_{mn} \left(\boldsymbol{\Lambda}_n + \sigma_\epsilon^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}, \sigma_\epsilon^2 + K_{**} - \boldsymbol{k}_*^T \boldsymbol{K}_m^{-1} \boldsymbol{k}_* + \boldsymbol{k}_*^T \boldsymbol{Q}_{m_j}^{-1} \boldsymbol{k}_*)$, where

$\boldsymbol{k}_* = (K(\bar{\boldsymbol{x}}_1, \boldsymbol{x}_*), \ldots, K(\bar{\boldsymbol{x}}_m, \boldsymbol{x}_*))^T$. In particular, when $n = m$ we obtain the posterior distributions of the full Gaussian process model.

One central problem of the SGP approach is that the sparsification depends on the choice of the pseudo-inputs $\bar{\mathcal{X}}$, which are treated as hyperparameters to be (somehow) selected once and then held fixed. In the original work, Snelson and Ghahramani (2006) propose to choose the pseudo-inputs by optimizing the marginal likelihood. Others have suggested to minimize the KL divergence (Titsias, 2009; Damianou and Lawrence, 2013). In our Bayesian approach, instead of using a fixed choice of pseudo-inputs (Titsias, 2009; Lee et al., 2017), we draw the pseudo-inputs from a prior distribution.

2.3 Bayesian Additive Modeling and Back-fitting

Bayesian additive modeling (Hastie and Tibshirani, 1990; Chipman et al., 1998) is a flexible technique that is widely adopted. Such additive models are formed by taking the sum of many model components, where each component captures a portion of the overall response variability. In the Gaussian setting, fitting a Bayesian additive model can be accomplished by using partial residuals and updating each component sequentially in the so-called back-fitting scheme (Hastie and Tibshirani, 2000). Following this scheme, we can represent an additive model with N components without intercept term in vector form as $\boldsymbol{y} = \sum_{j=1}^{N} \boldsymbol{f}_j + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim N_n(\boldsymbol{0}_n, \sigma_\epsilon^2 \boldsymbol{I}_n)$.

Bayesian back-fitting proceeds by fitting each additive component, $\boldsymbol{f}_j$, by using the "j-th partial residuals", $\boldsymbol{r}_j = \boldsymbol{y} - \sum_{i \neq j} \boldsymbol{f}_i$. These residuals are used as "data" for the j-th component. Starting with a particular initial value, the back-fitting algorithm (Algorithm 3.1 in Hastie and Tibshirani (2000)) iterates until the joint distribution of all mean functions $\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_N$ stabilizes.

One insight into the usefulness of this algorithm is to recognize that it allows updating the f_j's one at a time rather than requiring an expensive joint update like the full GP regression on a large dataset. Therefore, it would be advantageous to make the f_j updates computationally cheap.

2.4 Localization via Partition Schemes

Partitioning the input space $\mathsf{X} = \bigcup_j \mathsf{X}_j$ has been another popular way of scaling-up regression models. In this line of research, pioneering works were performed by Breiman (1984), Denison et al. (1998) and Chipman et al. (1998, 2010, 2016). Furthermore, various choices of partition schemes of the input domain are discussed in the local GP regression literature (Nguyen-Tuong et al., 2009; Gramacy and Apley, 2015; Park and Huang, 2016).

In terms of Bayesian additive modeling, Chipman et al. (1998) model the data in each partition X_j using an independent model component, conditional on the partitioning defined by a binary tree. This associates the fitted mean function (or target) $\boldsymbol{f}_j$ with the data lying in the specific partition X_j. Subsequent works (Gramacy and Apley, 2015; Chipman et al., 2016; Pratola et al., 2020) demonstrate that assembling many simpler models over such partitioning schemes can usually out-perform a single complex model fitted to the entire modeling domain.

As pointed out in Gramacy and Apley (2015) and Park and Apley (2018), such localization of the input-domain will fit and predict non-stationary datasets better. Also,

multi-scale features of a dataset can usually be well captured by introducing a hierarchical structure on the input domain (Fox and Dunson, 2012; Lee et al., 2017).

In our approach, capturing global and local features is accomplished through a fixed partition scheme informed by the data $\mathcal{X}$. We will show how this can be done so that the partition scheme is well suited to the Bayesian backfitting algorithm, and use sparse model components to further enhance the scalability of the model.

3. Sparse Additive Gaussian Process Regression (SAGP)

The proposed SAGP model combines the three key ingredients of sparsification, Bayesian additive modeling (via backfitting), and localization in a clever way. In particular, our model has the usual additive form,

$$\boldsymbol{y} = \sum_{j=1}^{N} \boldsymbol{f}_j + \boldsymbol{\epsilon}, \tag{2}$$

for some error component $\boldsymbol{\epsilon}$ with variance σ_ϵ^2. Much effort in statistical modeling focuses on the $\boldsymbol{f}_j$. For instance, in linear regression, $\boldsymbol{f}_j = \boldsymbol{X}_j\beta_j$ for the jth column of some design matrix $\boldsymbol{X}$ and vector parameter $\boldsymbol{\beta}$. In our approach, each $\boldsymbol{f}_j$ has entries which are formed by weighted linear combinations of the pseudo-targets, $\boldsymbol{W}^T\bar{\boldsymbol{f}}_j$, and each vector of pseudo-targets $\bar{\boldsymbol{f}}_j$ arise from the pseudo-inputs $\bar{\mathcal{X}}^{(j)}$ belonging to the jth subdomain of the input domain X. Additional parameters $\boldsymbol{\kappa}$ will be involved in each component in forming the weights $\boldsymbol{W}$. Finally, the subdomains are defined by a partitioning scheme, $\mathcal{B}_N$. Let the collection of pseudo-inputs belonging to each partition be $\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}$. Then, our model takes a hierarchical form involving the likelihood function $L(\boldsymbol{y}|\bar{f}_1, \ldots, \bar{f}_N, \boldsymbol{\kappa}, \sigma_\epsilon^2, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X})$ as well as the prior distributions of the various additive model components in the overall model, $P(\bar{f}_1, \ldots, \bar{f}_N|\boldsymbol{\kappa}, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X})$, $P(\boldsymbol{\kappa} \mid \mathcal{B}_N, \mathcal{X})$, $P(\sigma_\epsilon^2)$, and $P(\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}|\mathcal{B}_N, \mathcal{X})$.

To perform inference and prediction, we will be interested in the marginal posterior distribution $P(\bar{f}_1, \ldots, \bar{f}_N, \boldsymbol{\kappa}, \sigma_\epsilon^2|\boldsymbol{y}, \mathcal{B}_N, \mathcal{X})$. Note that the posterior is dependent on the partitioning scheme $\mathcal{B}_N$, since it is held fixed in our modeling approach. Therefore, we will start by describing the proposed partitioning scheme. The partitioning scheme reduces the computational cost by limiting the sample size in each model component by exploiting localization. Second, conditional on this localization scheme, the model components (i.e. the f_j's) themselves leverage the sparse Gaussian Process. This sparsification reduces the computational cost as described earlier. Finally, our overall model combines all of these sparse localized components into a Bayesian additive model as defined by the likelihood function, and the overall model can be efficiently fit using Bayesian back-fitting.

3.1 A Recursive Partitioning Scheme

We consider a recursive partitioning of the domain X that can be represented as a 2^d-ary tree. Each node of the tree corresponds to a subregion B_j of $\mathsf{X} \subset \mathbb{R}^d$ called a *block*. Only the node at the first level, i.e., the root of the tree, corresponds to the whole domain ($B_1 = \mathsf{X}$). The collection of blocks corresponding to nodes at the same depth of the tree is referred to as a *layer*. The collection of all blocks across all layers of the tree comprises the partitioning

of X. More formally, we define a *Recursive Partitioning* (RP) scheme as a collection of blocks $\{B_1, ..., B_N\}$ and layers $\mathcal{L}_1, \ldots, \mathcal{L}_L$ of these blocks satisfying the following properties:

P1. (Nestedness) For a block $B_i \subset \mathbb{R}^d$ in the j-th layer $\mathcal{L}_j$, there exists a unique block $B_k \in \mathcal{L}_{j-1} \subset \mathbb{R}^d$ in the $(j-1)$-layer $\mathcal{L}_{j-1}$ such that $B_i \subset B_k$.

P2. (Disjointedness, or non-overlapping) For two blocks B_i, B_k in the j-th layer $\mathcal{L}_j$ such that $B_i \neq B_k$, their interiors do not intersect.

To facilitate manipulating and storing the RP scheme on a computer, we encode each block by its centroid $\boldsymbol{c}_j = (c_j^1, \cdots, c_j^d)$ and half-width $\boldsymbol{w}_j = (w_j^1, \ldots, w_j^d)$ where for simplicity we take the half-widths to be the same in each dimension given a layer l, $w_j^k = R_l, k = 1, \ldots, d$. The j-th block is then defined as

$$B_j \coloneqq B(\boldsymbol{c}_j, \boldsymbol{w}_j) = \left\{ \boldsymbol{x} = (x^1, \ldots, x^d) \in \mathbb{R}^d \middle| \, |x^k - c_j^k| \leq w_j^k, k = 1, \ldots, d \right\}.$$

We require each block to have a minimum of m_j observations, allowing us to later define an SGP with m_j pseudo-inputs in each B_j. For simplicity of exposition, we will assume $m_j = m$ for all $j = 1, \ldots, N$. We also require pseudo-inputs $\bar{\mathcal{X}}^{(j)}$ to be mutually disjoint, so that each input setting in X is chosen as a pseudo-input at most once.

An example RP scheme construction with $L = 3$ layers and $m = 3$ pseudo-inputs is shown in Figure 1. The construction starts with a complete 2^d-ary tree consisting of layers $\mathcal{L}_1 = \{B_1\}, \mathcal{L}_2 = \{B_2, B_3\}$ and $\mathcal{L}_3 = \{B_4, B_5, B_6, B_7\}$, and a dataset of $n = 15$ observations, shown as black dots. Then, the complete tree is pruned according to Algorithm 1, which ensures that each block B_j will have at least m pseudo-inputs available while also satisfying the required properties P1 and P2. Finally, given an RP scheme, one possible random selection of pseudo-inputs is shown.

Algorithm 1 is able to perform the required pruning in general. Essentially, the algorithm works by requiring that the total number of observations in block B_j and all of B_j's children satisfies the total required number of pseudo-inputs for these components. Starting from the bottom layer, Algorithm 1 recursively works up the tree, pruning sub-trees that do not satisfy this constraint on total number of observations. Once the pruning is complete, the random selection of pseudo-inputs to blocks can be drawn by starting with blocks in layer L and working back to B_1, thereby guaranteeing that all blocks meet the minimum of m pseudo-inputs per block.

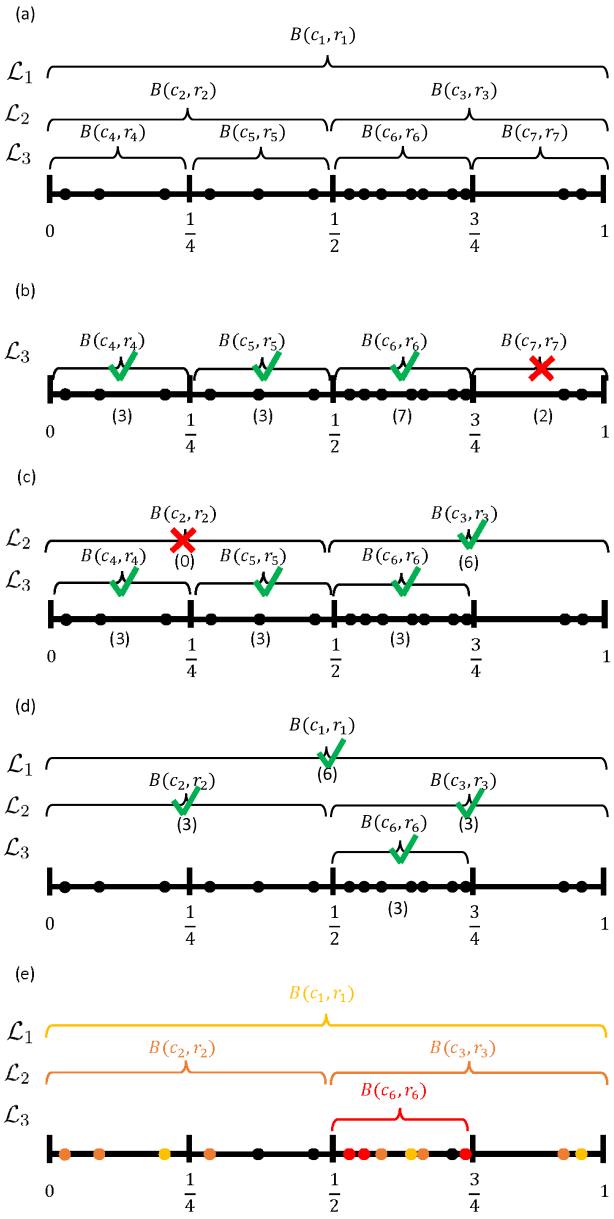
3.2 SAGP Model

Given a (pruned) RP scheme $\mathcal{B}_N$, we propose the additive model (2) for the response $\boldsymbol{y}$, where each component $f_j = (f_j(\boldsymbol{x}_1), \ldots, f_j(\boldsymbol{x}_n))^T$ is described by an SGP model on the domain B_j and $\boldsymbol{\epsilon} \sim N_n(\boldsymbol{0}_n, \sigma_\epsilon^2 \boldsymbol{I}_n)$. For block $B(\boldsymbol{c}_j, \boldsymbol{w}_j)$, we use $\bar{\mathcal{X}}^{(j)}$ to denote the pseudo-inputs for that block,

$$\bar{\mathcal{X}}^{(j)} = \{\bar{\boldsymbol{x}}_1^{(j)}, \ldots, \bar{\boldsymbol{x}}_{m_j}^{(j)}\} \subset B(\boldsymbol{c}_j, \boldsymbol{w}_j) \text{ s.t. } \bar{\boldsymbol{x}}_k^{(j)} \in \mathsf{X} \, \forall k.$$

The SGP associated with $\boldsymbol{f}_j$ and pseudo-inputs $\bar{\mathcal{X}}^{(j)}$ has corresponding pseudo-targets,

$$\bar{\boldsymbol{f}}_j = \left(f_j(\bar{\boldsymbol{x}}_1^{(j)}), f_j(\bar{\boldsymbol{x}}_2^{(j)}), \ldots, f_j(\bar{\boldsymbol{x}}_{m_j}^{(j)}) \right)^T \in \mathbb{R}^{m_j}, j = 1, \ldots, N. \tag{3}$$



The initial complete RP scheme as a binary tree with 3-layers on $[0, 1]$.

Starting from layer 3, we prune $B(\boldsymbol{c}_7, \boldsymbol{w}_7)$ since there are only $2 < 3$ observations available. We keep $B(\boldsymbol{c}_6, \boldsymbol{w}_6), B(\boldsymbol{c}_5, \boldsymbol{w}_5), B(\boldsymbol{c}_4, \boldsymbol{w}_4)$ as they all contain at least $m = 3$ observations.

Moving to layer 2, $B(\boldsymbol{c}_3, \boldsymbol{w}_3)$ has the 6 observations required by itself and its child $B(\boldsymbol{c}_6, \boldsymbol{w}_6)$. Checking $B(\boldsymbol{c}_2, \boldsymbol{w}_2)$, it contains only 6 observations so we prune its children $B(\boldsymbol{c}_4, \boldsymbol{w}_4), B(\boldsymbol{c}_5, \boldsymbol{w}_5)$.

Moving to layer 1, $B(\boldsymbol{c}_1, \boldsymbol{w}_1)$ contains 15 observations, therefore there are sufficient observations for $B(\boldsymbol{c}_1, \boldsymbol{w}_1)$ and its children $B(\boldsymbol{c}_2, \boldsymbol{w}_2)$, $B(\boldsymbol{c}_3, \boldsymbol{w}_3)$ and $B(\boldsymbol{c}_6, \boldsymbol{w}_6)$. We keep $B(\boldsymbol{c}_1, \boldsymbol{w}_1)$ and its children, completing the partitioning.

Given the final RP scheme $B(\boldsymbol{c}_1, \boldsymbol{w}_1)$, $B(\boldsymbol{c}_2, \boldsymbol{w}_2)$, $B(\boldsymbol{c}_3, \boldsymbol{w}_3)$ and $B(\boldsymbol{c}_6, \boldsymbol{w}_6)$, one possible random selection of the pseudo-inputs $\bar{\mathcal{X}}_j$ for the $j = 1, \ldots, N$ different additive components (here $N = 4$) conditional on the RP scheme is shown as colored dots. Points with the same color belong to blocks on the same layer.

Figure 1: RP scheme on the domain $\mathsf{X} = [0, 1]$ as a 2^1-ary tree with 3 layers and $m = 3$ pseudo-inputs per block. The $n = 15$ data points $\mathcal{X}$ are represented as dots. The right panels describe the application of the RP pruning Algorithm 1 (a)-(d), and the selection of pseudo-inputs given the RP scheme in (e). The left panels provide the analogous graphical construction of the RP scheme.

Algorithm 1 Pruning algorithm for RP scheme.

Input : RP partition scheme $\mathcal{A}$ consisting of N components, Observed dataset $\{\mathcal{X}, \boldsymbol{y}\}$.
Output: RP partition scheme $\mathcal{A}^{'}$ consisting of $N^{'} \leq N$ components,

```
for l in L : 1 do
    for each component j in the l-th layer L_l do
        for s in L : l do
            m_req ← Sum of the numbers of pesudo-inputs required for all components contained in B(c_j, w_j) in A'.
            if | X ∩ B(c_j, w_j) | ≥ m_req then
                break
            else
                Remove all the children components of component j from the model in s-th layer.
            end
        end
    end
end
```

Conditional on the RP scheme and pseudo-inputs, the joint posterior of pseudo-targets and other parameters in (2) can be written as:

$$P(\bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2 \mid \mathcal{B}_N, \boldsymbol{y}, \mathcal{X}, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}) \propto$$
$$\underbrace{P(\boldsymbol{y} \mid \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X})}_{\text{Likelihood Function}} \underbrace{P(\bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N \mid \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X}, \boldsymbol{\kappa})}_{\text{Pseudo-target Prior}} \times$$
$$\underbrace{P(\boldsymbol{\kappa} \mid \mathcal{B}_N)}_{\text{Kernel Prior}} \underbrace{P(\boldsymbol{\sigma}_\epsilon^2)}_{\text{Error Prior}} . \tag{4}$$

In effect, we view the choice of pseudo-inputs $\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}$ as nuisance parameters, and ultimately will integrate them out with respect to the prior $P(\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)} \mid \mathcal{B}_N)$, which gives the marginal posterior of interest,

$$P(\bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2 \mid \mathcal{B}_N, \boldsymbol{y}, \mathcal{X}) =$$
$$\int P(\bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2 \mid \mathcal{B}_N, \boldsymbol{y}, \mathcal{X}, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}) \underbrace{P(\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)} \mid \mathcal{B}_N)}_{\text{Pseudo-input Prior}} d\bar{\boldsymbol{x}}^{m_1} \ldots d\bar{\boldsymbol{x}}^{m_N},$$

where $d\bar{\boldsymbol{x}}^{m_j} = d\bar{\boldsymbol{x}}_1^{(j)} \times \ldots \times d\bar{\boldsymbol{x}}_{m_j}^{(j)}$.

In subsection 3.2.7 we will show a Gibbs sampler algorithm for SAGP fitting and for calculating predictions, but first we describe in greater detail the likelihood function and various prior distributions involved in the SAGP model.

3.2.1 Likelihood Function, $P(y \mid \bar{f}_1, \ldots, \bar{f}_N, \boldsymbol{\kappa}, \sigma_\epsilon^2, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X})$

Let us denote the covariance kernel for the SGP in the j-th block by $K^{(j)}, j = 1, \ldots, N$. We use the Gaussian covariance kernel supported inside B_j with parameters $\boldsymbol{\kappa}^{(j)} = (\rho^{(j)}, \eta^{(j)})$.

We have

$$K^{(j)}(\boldsymbol{x}, \boldsymbol{x}') \coloneqq \frac{1}{\eta^{(j)}} \cdot \left(\rho^{(j)}\right)^{\left[(\boldsymbol{x}-\boldsymbol{x}')^T(\boldsymbol{x}-\boldsymbol{x}')\right]}, \forall \boldsymbol{x}, \boldsymbol{x}' \in B_j. \tag{5}$$

Using (5), we can write down the (cross-)covariance matrices among and between inputs in $\mathcal{X}$ and $\bar{\mathcal{X}}^{(j)}$ as:

$$\begin{aligned}
\boldsymbol{K}_n^{(j)} &\coloneqq \left[K^{(j)}(\boldsymbol{x}_k, \boldsymbol{x}_l)\right]_{k,l=1}^{n}, \\
\boldsymbol{K}_{m_j}^{(j)} &\coloneqq \left[K^{(j)}(\bar{\boldsymbol{x}}_k^{(j)}, \bar{\boldsymbol{x}}_l^{(j)})\right]_{k,l=1}^{m_j}, \\
\boldsymbol{K}_{nm_j}^{(j)} &\coloneqq \left[K^{(j)}(\boldsymbol{x}_k, \bar{\boldsymbol{x}}_l^{(j)})\right]_{k,l=1}^{n,m_j} = \left(\boldsymbol{K}_{m_j n}^{(j)}\right)^T.
\end{aligned}$$

For a general $\boldsymbol{x} \in \mathbb{R}^d$, we also have

$$\boldsymbol{k}_{\boldsymbol{x}}^{(j)} \coloneqq \left(\; K^{(j)}(\bar{\boldsymbol{x}}_1^{(j)}, \boldsymbol{x}), \quad \ldots \quad , K^{(j)}(\bar{\boldsymbol{x}}_{m_j}^{(j)}, \boldsymbol{x}) \; \right)^T. \tag{6}$$

Assuming the additive components are conditionally independent, the likelihood is (see Lemma 1 in Appendix D)

$$P(\boldsymbol{y} \mid \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}) = N_n\left(\boldsymbol{y} \middle| \sum_{j=1}^{N} \boldsymbol{K}_{nm_j}^{(j)} \left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} \bar{\boldsymbol{f}}_j, \sigma_\epsilon^2 \boldsymbol{I}_n + \sum_{j=1}^{N} \boldsymbol{\Lambda}_n^{(j)}\right) \tag{7}$$

where the matrix $\boldsymbol{\Lambda}_n^{(j)} \coloneqq \operatorname{diag}\left(K_{ii}^{(j)} - \boldsymbol{k}_i^{(j)T}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{k}_i^{(j)}\right)_{n\times n}$ takes the diagonal form, with $\boldsymbol{k}_i^{(j)}$ as defined in (6) (with subscript i being shorthand for $\boldsymbol{x}_i$).

3.2.2 Pseudo-target Prior, $P(\bar{f}_1, \ldots, \bar{f}_N \mid \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X}, \kappa)$

The prior distribution of pseudo-targets given pseudo-inputs and covariance function parameters is straight-forward. Following Snelson and Ghahramani (2006), the pseudo-targets are assumed to be a priori conditionally independent, and so have Gaussian distributions with prescribed kernels:

$$\begin{aligned}
P(\bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N \mid \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X}, \boldsymbol{\kappa}) &= \prod_{j=1}^{N} P(\bar{\boldsymbol{f}}_j \mid \mathcal{B}_N, \bar{\mathcal{X}}^{(j)}, \mathcal{X}, \boldsymbol{\kappa}^{(j)}) \\
&= \prod_{j=1}^{N} N_{m_j}\left(\bar{\boldsymbol{f}}_j \middle| \boldsymbol{0}_{m_j}, \boldsymbol{K}_{m_j}^{(j)}\right).
\end{aligned} \tag{8}$$

3.2.3 Pseudo-input Prior, $P(\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)} \mid \mathcal{B}_N, \mathcal{X})$

The idea of the proposed pseudo-input prior is to sample pseudo-inputs uniformly within each block B_j while satisfying properties P1–P3 required for the RP scheme, $\mathcal{B}_N$. Algorithm 2 implements such a sampling scheme, which we now motivate. Let the index set $\mathcal{I}_j$

Algorithm 2 Sampling pseudo-inputs given RP scheme $\mathcal{B}_N$.

Input : RP scheme $\mathcal{B}_N$ consisting of N blocks, Observed inputs $\mathcal{X}$.
Output: Sample of $\bar{\mathcal{X}}^{(j)}, j = 1, \ldots, N$ conditional on RP scheme $\mathcal{B}_N$.

```
Initialize X_A = X as available inputs.
for j in N : 1 do
    Sample a random sample X̄^(j) ⊂ X ⊂ R^d of size m_j from X ∩ B_j ∩ X_A.
    X_A ← X_A \ X̄^(j) // Remove X̄^(j) sampled in the previous step from X_A.
end
```

representing the indices of children blocks of block B_j, which is defined as

$$\mathcal{I}_j \coloneqq \{k \neq j \text{ such that } B_k \subset B_j\},$$

and also define the collection of already selected pseudo-inputs of these child blocks as $\mathcal{C}(B_j) \coloneqq \cup_{k \in \mathcal{I}_j} \bar{\mathcal{X}}^{(k)}$. Then,

$$P(\bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)} \mid \mathcal{B}_N) = \prod_{\ell=L}^{1} \prod_{j: B_j \in \mathcal{L}_\ell} P(\bar{\mathcal{X}}^{(j)} \mid \mathcal{C}(B_j))$$

where

$$P(\bar{\mathcal{X}}^{(j)} \mid \mathcal{C}(B_j)) = \prod_{i=1}^{m_j} P(\bar{\boldsymbol{x}}_i^{(j)} \mid \mathcal{C}(B_j))$$

and

$$P(\bar{\boldsymbol{x}}_i^{(j)} \mid \mathcal{C}(B_j)) = \text{Discrete Uniform}\left(\{\boldsymbol{x} \in \mathcal{X} \subset B_j \backslash \mathcal{C}(B_j)\}\right).$$

In the expression above, we essentially draw a random sample from all those observed locations that have not been selected as pseudo-inputs of any children components in the lower layers of the RP scheme.

Unlike the standard SGP approach, this allows us to capture the uncertainty of pseudo-input selection by sampling the pseudo-inputs using Algorithm 2 and propagating this uncertainty to the posterior. Alternatives such as a continuous uniform prior over each component domain B_j, or sampling accordingly to design-theoretic considerations (Pratola et al., 2019), are possible.

3.2.4 Additional Prior Distributions, $P(\boldsymbol{\kappa} \mid \mathcal{B}_N)$ and $P(\sigma_\epsilon^2)$

We place a conjugate inverse gamma prior on the noise variance, σ_ϵ^2,

$$\sigma_\epsilon^2 \sim \text{InverseGamma}(\alpha_\epsilon, \beta_\epsilon).$$

The hyper-parameters α_ϵ and β_ϵ may be chosen as the hyper-parameters of the noise variance in traditional Bayesian GP regression.

We assume independent priors on the scale and correlation parameters of the kernel, $\eta^{(j)}$ and $\rho^{(j)}$,

$$P(\boldsymbol{\kappa}) = P(\boldsymbol{\kappa}^{(1)}, \ldots, \boldsymbol{\kappa}^{(j)}) = \prod_{\ell=1}^{L} \prod_{B_j \in \mathcal{L}_\ell} P(\eta^{(j)} \mid \alpha_\eta^l, \beta_\eta^l) P(\rho^{(j)}).$$

The precision parameters $\eta^{(j)}$ are assumed to have gamma priors,

$$\eta^{(j)} \sim \text{Gamma}(\alpha_\eta^l, \beta_\eta^l),$$

with $\alpha_\eta^l, \beta_\eta^l > 0, l = 1, \ldots, L$. The hyper-parameters $\alpha_\eta^l, \beta_\eta^l$ are the same for components within the same layer. We set up these hyper-parameters so that the variance of the response explained by the SAGP model is unequally partitioned across the L layers, with components in higher layers of the partitioning scheme explaining larger portions of the variance. To facilitate the set-up of the hyper-parameters, we first normalize the observed responses $y_1, \ldots, y_n$, re-centering and re-scaling so that they have mean 0 and variance 1. For all the components in layer l, we set

$$\begin{aligned} \alpha_\eta^l =& \mathsf{c}_{1\eta} + 1, \\ \beta_\eta^l =& \mathsf{c}_{1\eta}(1 - \mathsf{c}_{2\eta})\mathsf{c}_{2\eta}^{l-1}, \end{aligned}$$

with $\mathsf{c}_{1\eta} > 0$ and $0 < \mathsf{c}_{2\eta} < 1$. For each component j in layer l, this choice implies that $1/\eta^{(j)}$, the marginal variance of the component, has prior mean

$$E[1/\eta^{(j)}] = \frac{\beta_\eta^l}{\alpha_\eta^l - 1} = (1 - \mathsf{c}_{2\eta})\mathsf{c}_{2\eta}^{l-1}.$$

For example, if $\mathsf{c}_{2\eta} = .1$, components on layer $l = 1$ are expected to have variance $1 - \mathsf{c}_{2\eta} = .9$, which is 90% of the variance of the response because of the normalization. Components on layer $l = 2$ are expected to have $(1 - \mathsf{c}_{2\eta})\mathsf{c}_{2\eta} = 0.09$, 9% of the variance of the response. Similarly, as l increases, components are expected to explain smaller portions of the variance of the response. In particular, the geometric decay of the prior mean of $1/\eta^{(j)}$ is chosen so that the expected layer-specific variances add up to approximately the total response variance, which is guaranteed because, if L is sufficiently big, $\sum_{l=1}^{L}(1 - \mathsf{c}_{2\eta})\mathsf{c}_{2\eta}^{l-1} \approx 1$. The other hyper-parameter $\mathsf{c}_{1\eta}$ controls the spread of the prior distributions of $1/\eta^{(j)}$, with larger values of $\mathsf{c}_{1\eta}$ imposing a tighter constraint to the prior mean. In our experience, values of $\mathsf{c}_{2\eta} = .1$ and $\mathsf{c}_{1\eta}$ between 10 and 50 appear to provide the best results in our applications.

We set the prior distributions on the parameters $\rho^{(j)}$ in the following way. First of all, we assume that the inputs $\boldsymbol{x}_i$'s have been appropriately scaled, so that the domain X is mapped into the unit cube $[0, 1]^d$. This facilitates the definition of priors for $\rho^{(j)}$. Second, as for the $\eta^{(j)}$, we assume the same prior distribution for parameters corresponding to components in the same layer l. Third, we adopt a structure of priors imposing smoother behaviors for components in the top layers of the partitioning scheme. In other words, we impose a structure of priors where $\rho^{(j)}$ is expected to be greater than $\rho^{(j')}$ if component j belongs to a layer on a higher level than the layer of component j'. Despite a family of beta priors on the $\rho^{(j)}$ may be tuned to satisfy these properties, we empirically observed that setting the values

of these parameters to fixed layer-specific constants ρ_l (i.e., $P(\rho^{(j)}) = \delta_{\rho_l}$, the Dirac delta function on ρ_l) worked as well but was computationally less expensive. To get the sense on how the values of ρ_l affect the layer-specific correlations, one may plot the correlation for two responses as a function of the distance of their inputs, as specified by Equation (5). We provide this plot in Appendix A, in the case $L = 5$ and using the values $\rho_1 = 10^{-1}$, $\rho_L = 10^{-50}$ and the intermediate values $\rho_l, l = 2, \ldots, L-1$ to be equally spaced between ρ_1 and ρ_L on the logarithm (base 10) scale. Even though these values of ρ_l may appear to quickly become excessively small, the sizes of the subdomains where the components are defined (i.e., the blocks B_j) shrink as l increases. In our numerical example, if we consider a one-dimensional case with two inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$ at distance 0.0625 (i.e., the largest distance between two points in one block on the fifth layer), the assumed correlations on components on layer $l = 1$ to 5 are 0.99, 0.89, 0.80, 0.71 and 0.64, respectively. Notably, the decay of such values depends on the number of layers L, which can be tuned using prior beliefs and the information in the data. In our applications, trading the conventional estimation of the parameters $\rho^{(j)}$ with a set of fixed ρ_l and a data-driven selection of L via cross-validation (see Section 3.2.8) resulted in sufficiently flexible models.

3.2.5 Full Conditional Distribution of Pseudo-targets

In order to implement an MCMC algorithm for SAGP, we apply Bayes theorem on the pseudo-inputs $\bar{\boldsymbol{f}}_j$ in order to yield its full conditional distribution from (7) and (8) and the conditional independence assumption,

$$\begin{aligned}
&P(\bar{\boldsymbol{f}}_j \mid \boldsymbol{y}, \mathcal{X}, \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_{j-1}, \bar{\boldsymbol{f}}_{j+1}, \ldots \bar{\boldsymbol{f}}_N, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2) \\
&\propto P(\boldsymbol{y} \mid \bar{\boldsymbol{f}}_1, \ldots \bar{\boldsymbol{f}}_N, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2) \times \\
&P(\bar{\boldsymbol{f}}_j \mid \mathcal{X}, \mathcal{B}_N, \bar{\mathcal{X}}^{(j)}, \boldsymbol{\kappa}, \boldsymbol{\sigma}_\epsilon^2) \\
&= N_n\left(\boldsymbol{r}_j \middle| \, \boldsymbol{K}_{nm_j}^{(j)} \left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} \bar{\boldsymbol{f}}_j, \boldsymbol{\Lambda}_n^{(j)} + \sigma_\epsilon^2 \boldsymbol{I}_n\right) \times N_{m_j}\left(\bar{\boldsymbol{f}}_j \middle| \, \boldsymbol{0}_{m_j}, \boldsymbol{K}_{m_j}^{(j)}\right),
\end{aligned} \tag{9}$$

where $\boldsymbol{r}_j = \boldsymbol{y} - \sum_{l \neq j} \boldsymbol{K}_{nm_l}^{(l)} \left(\boldsymbol{K}_{m_l}^{(l)}\right)^{-1} \bar{\boldsymbol{f}}_l$. Using normal-normal conjugacy, we can identify the mean and variance of this normal distribution, $\bar{\boldsymbol{f}}_j \mid \boldsymbol{Mean}_j, \boldsymbol{Var}_j$, where (see Appendix D)

$$\begin{aligned}
\boldsymbol{Mean}_j &= \boldsymbol{K}_{m_j}^{(j)} \boldsymbol{Q}_{m_j}^{(j)-1} \boldsymbol{K}_{m_j n}^{(j)} \left(\boldsymbol{\Lambda}_n^{(j)} + \sigma_\epsilon^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{r}_j, \\
\boldsymbol{Var}_j &= \boldsymbol{K}_{m_j}^{(j)} \boldsymbol{Q}_{m_j}^{(j)-1} \boldsymbol{K}_{m_j}^{(j)}, \\
\text{and } \boldsymbol{Q}_{m_j}^{(j)} &= \boldsymbol{K}_{m_j}^{(j)} + \boldsymbol{K}_{m_j n}^{(j)} \left(\boldsymbol{\Lambda}_n^{(j)} + \sigma_\epsilon^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{K}_{nm_j}^{(j)}.
\end{aligned} \tag{10}$$

Although we still need to invert an $n \times n$ matrix $\boldsymbol{\Lambda}_n^{(j)} + \sigma_\epsilon^2 \boldsymbol{I}_n$, it is a diagonal matrix and hence its computational cost will be $\mathcal{O}(n)$.

3.2.6 Full Conditional Distribution of Noise Variance

As we mentioned in the previous section, we want to make use of the Gaussian-inverse gamma conjugacy. For the observation of sample size n, by conjugacy, the distribution σ_ϵ^2

is again inverse gamma,

$$P(\sigma_\epsilon^2 \mid \boldsymbol{y}, \mathcal{X}, \bar{\boldsymbol{f}}_1, \dots \bar{\boldsymbol{f}}_N, \mathcal{B}_N, \bar{\mathcal{X}}^{(1)}, \dots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}) =$$
$$\text{InverseGamma}\left(\alpha_\epsilon + \frac{n}{2}, \beta_\epsilon + \frac{1}{2}(\boldsymbol{y} - \hat{\boldsymbol{y}})^T(\boldsymbol{y} - \hat{\boldsymbol{y}})\right),$$

where $\hat{\boldsymbol{y}} \coloneqq \sum_{j=1}^{N} \boldsymbol{K}_{nm_j}^{(j)} \left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} \bar{\boldsymbol{f}}_j$ is the "fitted value" from the SAGP model. We can directly sample this parameter using a Gibbs step.

3.2.7 Sampling Algorithm

SAGP is fitted by a Metropolis-Hastings Markov chain Monte Carlo (MCMC) algorithm (Gelfand et al., 1990). For each additive component of the model, we have to use the partial residuals $\boldsymbol{r}_j$ defined in (9) as data. This step is from the back-fitting scheme designed for fitting additive Bayesian models (Hastie and Tibshirani, 2000).

From the likelihood derivations presented in section 3.2.5, we know that $\bar{\boldsymbol{f}}_j$ can be directly sampled from their conditional distributions for each $j = 1, \dots, N$ components. The difficulty in this step is to compute the $\boldsymbol{Mean}_j, \boldsymbol{Var}_j$ in (10). As mentioned earlier, the main computational cost occurs in the inversion of the covariance matrices in section 3.2.1, which has been reduced compared to a full GP covariance matrix. Numerical instability in inversion of these matrices may cause additional problems, so we adopt the Cholesky decomposition method with diagonal perturbation to solve this instability problem as in Rasmussen and Williams (2006). For each η_j we do not have normal conjugacy, therefore an adaptive Metropolis-Hasting step is used for sampling η_j (Banerjee et al., 2012).

The advantage of using such a fully Bayesian model is that the uncertainty quantification comes naturally with the posterior samples from the sampler. Our posterior inference below can be based on all these posterior samples. The algorithm for overall sampling is presented in Algorithm 3 in the Appendix C.

3.2.8 Tuning Parameters and Complexity

The trade-off between the number of layers L in the RP scheme and the number of pseudo-inputs m is central to the SAGP model. On one hand, in SGP modeling (Snelson and Ghahramani, 2006, 2007; Lee et al., 2017), we need to increase the number of pseudo-inputs m to get a better fit of the SGP model. On the other hand, for regression tree partitionining models (Chipman et al., 1998, 2016; Pratola et al., 2020), the more additive components a model has, the better fit we can expect. In the SAGP model, increasing both factors (number of pseudo-inputs, m, and number of layers, L,) would certainly improve the overall fit, but the interesting observation is that there exists a trade-off between these two tuning parameters. Increasing the number of layers L may counter-act the effect of decreasing the number of pseudo-inputs m, and vice versa. Theoretically, we can tune the choice of the number of layers using cross-validation (see Figure 4). Practically, we can usually choose reasonable m and L depending on the desired granularity of the RP scheme.

This trade-off between m and L can also be observed by considering the model's computational complexity. We already mentioned above that for a full GP model based on $\mathcal{X}$ the complexity is of order $\mathcal{O}(n^3)$; for an SGP model with $m \ll n$ pseudo-inputs selected

from $\mathcal{X}$, the complexity is of order $\mathcal{O}(nm^2)$ (Snelson and Ghahramani, 2006). Since each additive component in our SAGP model is essentially an SGP model, the overall complexity is given by the following proposition.

Proposition 1 *For an RP scheme on input-domain $[0,1]^d$, with b_i-ary tree (Storer, 2012) in the i-th dimension, the complexity of fitting an L-layer SAGP model with m pseudo-inputs for each block and an overall sample of size n is at most $\mathcal{O}\left(\sum_{\ell=1}^{L}\prod_{i=1}^{d} b_i^{\ell-1}\cdot n\cdot m^2\right)$.*

Proof See Appendix B. ■

For $N = 1$, where there is only one layer and one component, this complexity reduces to SGP complexity with m pseudo-inputs. We will revisit this *component number pseudo-input trade-off* in our data analyses.

4. Simulation Study

4.1 Design

To evaluate the performance of our methodology and compare it to competing approaches, we run a family of simulations. We focus on the one-dimensional case ($d = 1$) and we simulate from a GP with a mean function

$$f(x) = -5 - 6x^3 + 30(x - .5)^2 + 3\exp(2x - 1) + 3x^2\sin(12\pi x) + \cos(6\pi x), \tag{11}$$

which is represented in Figure 2 in the interval [0,1]. We generate a sample of $n = 200$ locations from a uniform distribution on $[0, 1]$ and we define the observed responses as $y_i = f(x_i) + \epsilon_i$ for all $x_i \in \mathcal{X}$, with $\epsilon_i \sim N_1(0, 0.1)$. The data are split into training and testing sets, with sizes 150 and 50, respectively. We consider two scenarios. In the first scenario, the testing set is selected at random. In the second scenario, the testing set is chosen as the subset with 50 data points with x_i closest to a point randomly chosen in $[0.25, 0.75]$. Figure 2 shows an example for each of these scenarios.

We generate 1000 datasets for each scenario (random and interval testing set). In each dataset, we fit the SAGP model with three configurations of m, L: (i) $m = 5$, $L = 4$; (ii) $m = 10$, $L = 3$; (iii) $m = 15$, $L = 3$. We compare the SAGP models to the following methods:

- Full GP regression. We use the implementation of GP regression model in the R package `DiceKriging` by Roustant et al. (2012).

- SGP regression. We consider the choices $m = 5$, $m = 10$ and $m = 15$ and use the implementation of SGP in the Matlab package implementation `SPGP` at `http://www.gaussianprocess.org` accompanying the paper by Snelson and Ghahramani (2006).

- Bayesian Additive Regression Trees (BART) Chipman et al. (1998, 2010); Pratola et al. (2020). We used the default number of trees as specified in Chipman et al. (2010) and the implementation at `http://bitbucket.org/mpratola/openbt`.

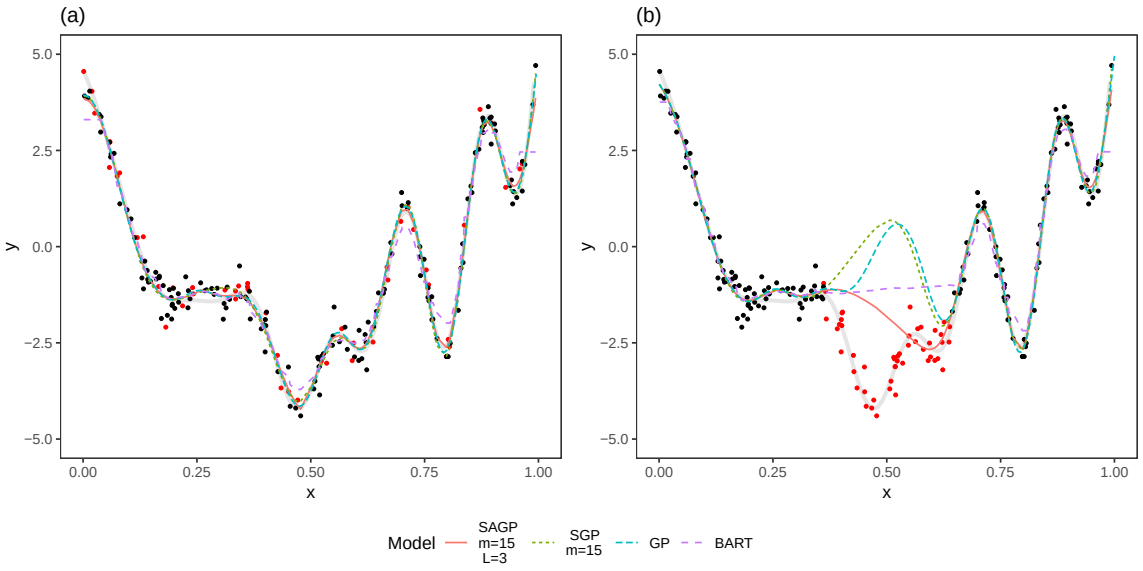


Figure 2: Example of data generated in the simulation study. The gray, bold, curve represents the true mean function $f(x)$. Training and testing sets are represented as black and red points. Panels (a) and (b) show the scenarios where the 50 data points of the testing set are chosen at random or as the input location that is closest to a randomly chosen point (0.5 in the example), respectively. The posterior predictive functions of four models, fit on the training portion of the data, are provided in both panels.

For each generated dataset, the models are fit on the training data and used to predict the response on the testing data. For each point in the testing set, we compute the estimated mean function $\hat{y}(x_i)$ (see Section 3.2.6) and the 95% prediction interval (PI) for y_i. The performance of the estimators of the mean function is evaluated in terms of root mean squared error (RMSE). To assess the appropriateness of the uncertainty quantification, we compute the coverage of the PIs and compare it to the nominal prediction level. Finally, we compare the methods in terms of the average value of interval scores, which is a summary measure to assess the quality of prediction intervals (Gneiting and Raftery, 2007). Given a $(1-\alpha)100\%$ PI for y_i with extremes (l_i, u_i), the interval score at y_i is defined as

$$s_\alpha(l_i, u_i; y_i) = (u_i - l_i) + \frac{2}{\alpha}(l_i - y_i)\mathbf{1}(y_i < l_i) + \frac{2}{\alpha}(y_i - u_i)\mathbf{1}(y_i > u_i).$$

We choose this metric to jointly evaluate a family of intervals in terms of precision (i.e. the width of the intervals) and accuracy (i.e., the coverage of the true value). Notably, low values of the score indicate good performance.

4.2 Results

Figure 3 summarizes the resulting RMSEs, PI coverages and averages of the interval scores across the 1000 generated datasets for the two scenarios.

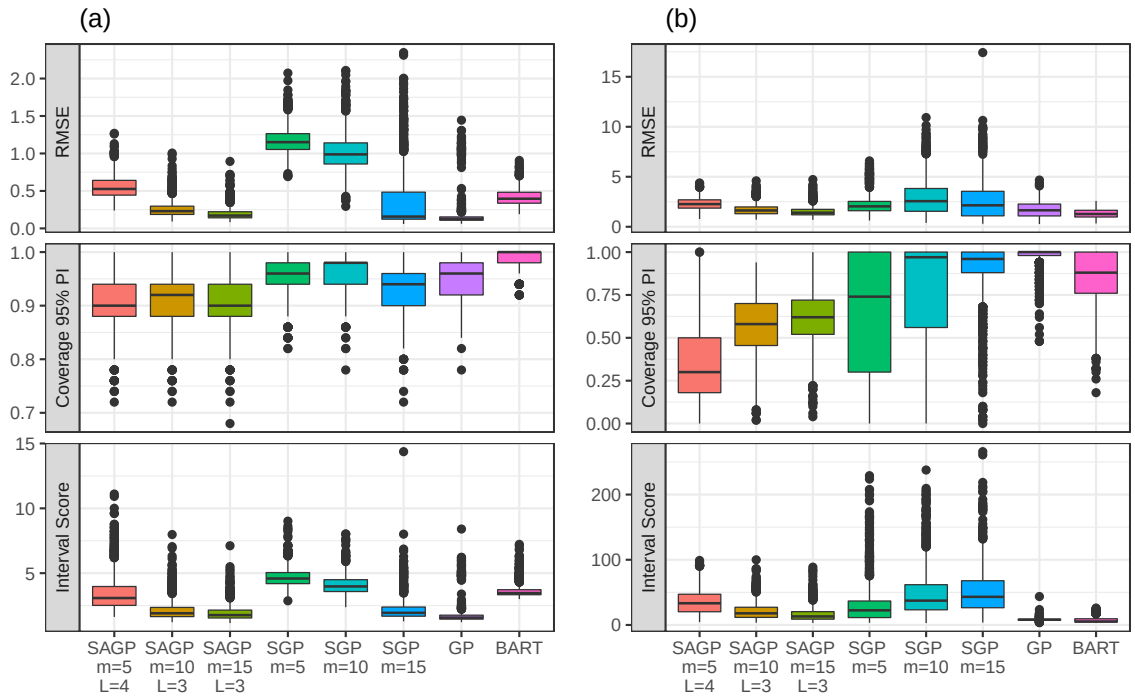


Figure 3: RMSE (top panels), coverage (central panels) and interval score (bottom panels) over the 1000 simulated datasets. Panel (a) shows the results in the case where the testing set is chosen at random over $[0, 1]$. Panel (b) shows the results in the case where the testing set is chosen in a random interval with center uniformly selected from $[0.25, 0.75]$.

Panel (a) provides the results in the scenario where the testing set is selected at random. In terms of RMSE, both the SAGP and SGP models perform better with larger values of m. As expected, the full GP model attains the smallest RMSEs. The SAGP models with $m = 5$ and 10 perform better than the SGP models with the same number of pseudo-inputs. For $m = 15$, the median RMSEs in the SAGP and SGP models are similar, but the performance of the SAGP model is more consistent across simulations (the upper quartile of SAGP with $m = 15$ is considerably smaller than the one of SGP with $m = 15$). With the considered configuration of the parameters, the BART model performs slightly better than the SAGP model with $m = 5$, but worse than the SAGP model with $m = 10$ and $m = 15$. The coverage of the 95% PIs is close to the nominal level for all the methods except for BART. The PIs of the SAGP model appear to be slightly too narrow, as most of the coverages are a little lower than .95. SGP and GP models show coverages perfectly matching the nominal value.

However, the BART model produces overly wide PIs, as the median coverage is 100%. The ranking of the methods in terms of interval score is similar to the one based on the RMSE. Again, better performances are attained by SAGP and SGP models with larger values of m. The SAGP models with $m = 10$ and $m = 15$ perform better than all the other methods, except for the full GP model.

Panel (b) provides the results of the simulations in the scenario where the testing set is an interval with random mid-point. Notably, this prediction problem is much harder than the one evaluated by the previous scenario, as the models are forced to a certain degree of extrapolation due to the lack of data. BART, GP and SAGP with $m = 15$ attain the best performance in terms of RMSE. Overall, the SAGP model seems to perform better than SGP. The coverage is suboptimal for all the methods, being much lower (undercoverage) than expected for SAGP and SGP with $m = 5$ and higher (overcoverage) for GP. A wide range of coverages is observed for all the other methods. With respect to the interval score, GP and BART are the methods that appear to perform best. Among the SAGP and SGP models, SAGP with $m = 15$ is the best performing and competitive with GP and BART.

4.3 Computational Details

As for any Bayesian model that is fit using MCMC algorithms, the convergence to the stationary distribution must be investigated also for the SAGP model. In our specific implementation, we discard the first 10,000 samples as burn-in and keep the following 1,000 samples to compute posterior estimates. We monitor the convergence of σ^2_ϵ, sampled with Gibbs steps, and of the parameters $\eta^{(j)}$, which are sampled with Metropolis-Hastings steps with an adaptive choice of the bandwidth of the proposal distribution to control the acceptance rate. The considered SAGP models turned out to mix well and reasonably fast on the basis of trace-plots of the parameters (not shown) and the diagnostics suggested by Gelman et al. (2013), which are provided in Appendix E. Notably, in our experience, similar satisfying mixing diagnostic for the SAGP model may be achieved with much fewer steps than 10,000.

Table 1: Computation time needed to fit the SAGP model on 1,000 simulated datasets on a 40-core cluster.

m	L	Testing set	CPU time (hh:mm:ss)
10	3	Random	106:53:21
10	3	Interval	107:56:21
5	4	Random	241:17:31
5	4	Interval	175:45:35
15	3	Random	477:22:52
15	3	Interval	479:06:39

With respect to the computation time, setting the burn-in size to 10,000 and the size of posterior samples to 1,000, an SAGP model can be fit on one dataset of size $n = 200$ in

3 to 5 minutes, depending on the configuration of m and L, using a laptop with an Intel Core-i5 2.30GHz processor. The time that was needed to fit the model on one batch of 1,000 simulated datasets are summarized in Table 1.

5. Real Data Applications

In this section, the proposed model is applied to real data. We considered four datasets that differ in terms of sample size and number of predictors:

- Heart rate data: $n = 1,664$, $d = 1$;
- Temperature data: $n = 247$, $d = 2$;
- Ice Sheet data: $n = 2,226$, $d = 2$;
- UK Housing data: $n = 1,519$ and $d = 8$.

The performance is evaluated quantitatively with the out-of-sample RMSE, coverage of 95% PIs and average interval score on a 25% test set. Our model is compared to other popular methods. We considered two Bayesian models: BART and Bayesian CART (BCART) (Chipman et al., 1998), implemented in the `BayesTree` package on CRAN (version 0.3-1.4). We also considered two frequentist models: full GP and Local Approximate GP (laGP) (Gramacy et al., 2016), implemented in the `laGP` package on CRAN (version 1.5-5). For the $d = 1$ and $d = 2$ datasets, we also provide a qualitative assessment of the fits via graphical plots.

5.1 Heart Rate Data

The heart rate (HR) dataset we study here can be used to evaluate the level of physical preparation and design training/rehabilitation activities (Zakynthinaki, 2015). In this study, a single runner was asked to run on a treadmill at constant speed. The HR (in beats/minute) was recorded for about 7 minutes from the beginning of the exercise. After the exercise, the HR of the subject was measured for about 10 minutes during the recovery. The experiment was repeated four times, varying the speed of the exercise ($v = 13.4, 14.4, 15.7$ and 17 km/h). For our illustrative purposes, we use the data of the exercise performed at speed $v = 13.4$ km/h, which are graphically represented in Figure 5.

We consider SAGP models with $m = 5$ and $m = 10$ pseudo-inputs, and use 10-fold cross-validation to select the number of layers L as shown in Figure 4(a). This plot demonstrates the trade-off between the values of pseudo imputs m and the number of layers L, with $L = 3, m = 10$ being a good choice. The resulting fitted SAGP model, which consists of 7 additive components, is summarized in Figure 5, both in terms of how the fit is decomposed by layer in panel (a) and the overall fit shown in panel (b). An alternative fit with $L = 4, m = 5$ is provided in Appendix F.

5.2 Temperature Data

In this section we study a moderate sized 2-dimensional dataset of average daily maximum temperature in degrees centigrade at 247 locations in Colorado during 1997 (`https://`

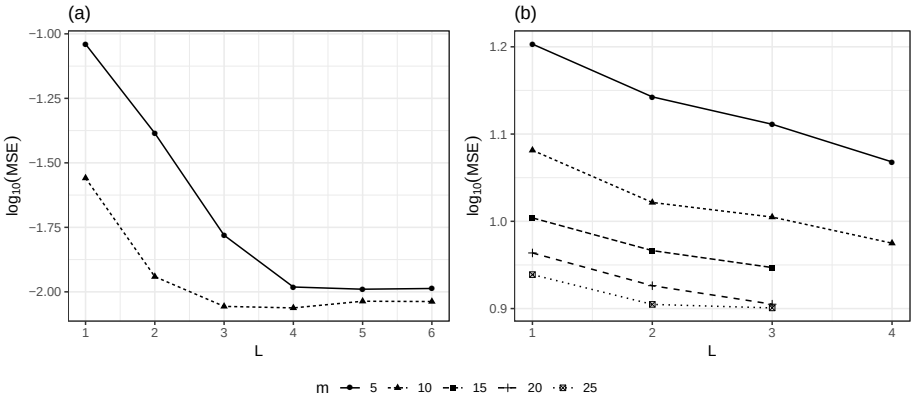


Figure 4: (a) Out-of-sample MSE (on $\log_{10}$ scale) attained by different models fitted on 1 dimensional heart-rate dataset with $m = 5, 10$ and $L = 1, \ldots, 6$.
(b) Out-of-sample MSE (on $\log_{10}$ scale) attained by different models fitted on 2 dimensional temperature dataset with $m = 5, 10, 15, 20, 25$ and $L = 1, \ldots, 4$. For any $L > 4$, our pruning algorithm 1 will reduce it to $L = 4$; for $m = 25$ our pruning algorithm will reduce SAGP model to $L = 3$

`www.image.ucar.edu/Data/US.monthly.met/USmonthlyMet.shtml`, US precipitation and temperature (1895-1997) dataset).

Qualitative comparisons of GP, BART and SAGP are shown in Figure 6. For GP regression we used MLE estimates with the Matern$(5/2)$ kernel. For BART we use the default settings (Chipman et al., 2010). For SAGP, we choose $L = 3, m = 25$ and calibrate the α, β of the noise prior in SAGP and the noise estimate in BART according to MLE of noise estimate from GP. The GP model shows reasonable predictions, however, the prediction comes with high predictive variance in locations away from the observations and especially near the boundary (not shown). The predictive mean of BART shows it has a slight grid-like artifact due to its decision tree construction. In addition, the shape of the response around the mode is noticeably more rectangular than suggested by the other models.

This dataset provides us a 2-dimensional example where the data is limited, which is actually a disadvantage for SAGP since the sparsification does not cut down the computational cost significantly yet some information is lost in the procedure. Nonetheless, the SAGP method captures the major trends and even some of the extremal temperatures close to 40 degrees centigrade. Compared to BART and GP, the SAGP model behaves "in-between" these two methods and provides us with very competitive performance.

5.3 Ice Sheet Data

The Ice Sheet data is a larger 2-dimensional dataset but this time with noticeably uneven sampling as discussed in Park and Apley (2018).The response is ice sheet thickness in meters collected over a region of west Antarctica (Blankenship et al., 2004).We used the data from 1991, first converting the longitude and latitude into 2-dimensional Euclidean coordinates and standardizing the dataset to $[0, 1]^2$.

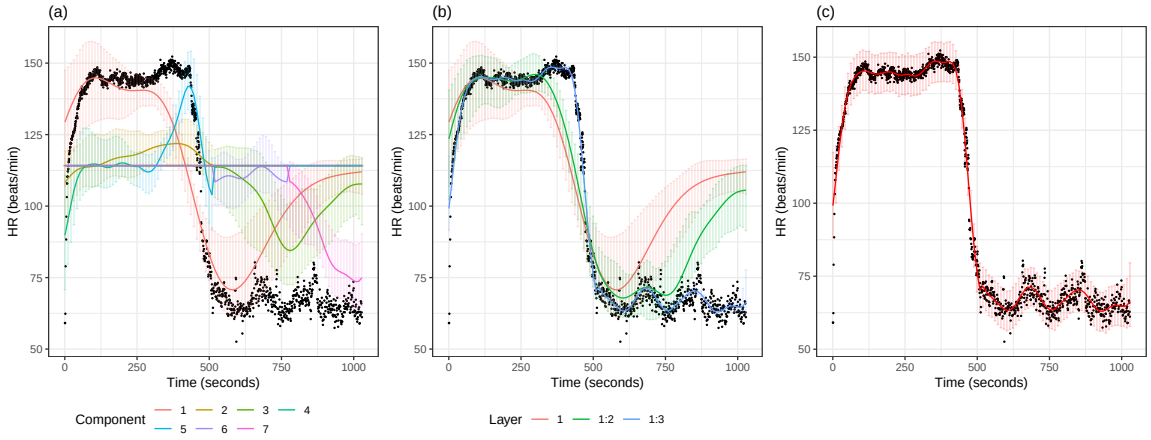


Figure 5: The panels show the observed HR values over time as black dots and results about the fit of the SAGP model with $m = 10$ and $L = 3$. Panel (a) shows the posterior means and the 95% CIs of the 7 additive components of the SAGP model on 100 equispaced locations on the support of the data. Panel (b) shows the posterior means and the 95% CIs of the sole component in layer 1 (red), of the components belonging to layer 1 and 2 (green) and of the complete model, including components from layer 1, 2 and 3. Panel (c) provides the predictive mean and the corresponding 95% prediction intervals.

A plot of the data and predictive fits for the GP (exponential correlation), laGP, BART, treed GP (TGP; (Gramacy et al., 2007)) and SAGP models are shown in Figure 7. We included TGP in this plot as we thought it may be helpful with the unevenly sampled data but did not end up including it in our overall quantitative results below. For the SAGP model, we show the fit obtained with $L = 3, m = 10$.

The fits obtained among these models show quite different behaviors. The full GP fit possess extreme boundary behavior due to the lack of data near the boundary. The BART model shows more noticeable grid-like artifacts in this dataset, but does not suffer from the boundary effects seen with the GP. The TGP regression also does not exhibit boundary effects but has much higher variability of the mean response in the data-rich region which does not agree with the other models. The dynamic partitioning of TGP also introduces considerable computational cost. The laGP model with its default settings and MSPE criterion exhibits some degree of variability in the fitted mean response, particularly near the boundaries, however, it is the most computationally efficient method.

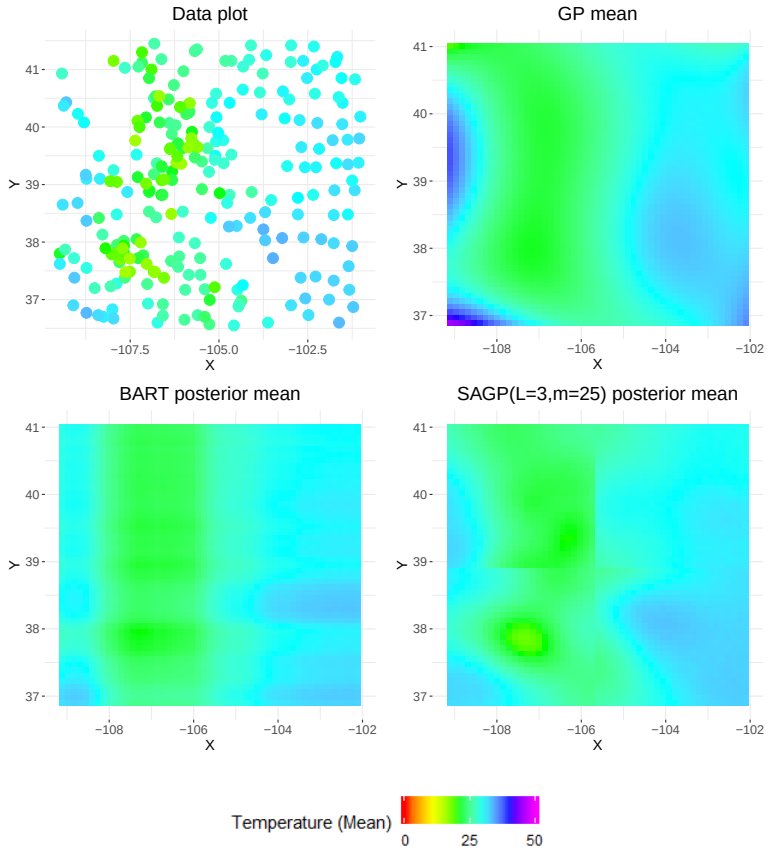


Figure 6: The original max temperature dataset for Colorado in 1997 summer. The horizontal axis is longitude; the vertical axis is latitude; the response is the observed values of maximal temperature in degrees Celsius. The typical raster plot for predictive means of Gaussian Process regression (Universal kriging with Gaussian kernel and MLE nuggets)/BART(number of trees $m = 200$)/SAGP($m = 25, L = 3$) evaluated on a fine meshed grid (generated by steplengths of 0.1) on the original input domain.

Table 2: Performance of SAGP model and of other competing methods on four datasets.

Dataset	Model	Details	RMSE	Coverage (%)	Avg. Int. Score ($\log_{10}$ scale)
Heart Rate (n=1,664, d=1)	SAGP	L=3, m=10	1.340×10^2	88.8	2.800
	SAGP	L=4, m=5	1.342×10^2	89.5	2.796
	GP	-	2.727×10^2	12.0	3.855
	SGP	m=5	1.366×10^2	100.0	4.605
	SGP	m=15	1.347×10^2	100.0	4.633
	SGP	m=150	1.325×10^2	100.0	4.720
	laGP	ALC	1.325×10^2	91.7	2.777
	laGP	MSPE	1.879×10^2	91.1	2.821
	BART	-	1.331×10^2	18.1	3.494
	BCART	-	1.355×10^2	94.7	2.751
Temperature (n=247, d=2)	SAGP	L=2, m=5	3.412×10^0	79.7	1.345
	SAGP	L=4, m=10	2.910×10^0	77.4	1.356
	GP	-	3.041×10^0	92.5	1.228
	SGP	m=5	3.401×10^0	100.0	2.010
	SGP	m=15	3.345×10^0	100.0	2.033
	SGP	m=150	3.043×10^0	100.0	1.709
	laGP	ALC	3.206×10^0	86.6	1.247
	laGP	MSPE	3.431×10^0	85.3	1.273
	BART	-	3.123×10^0	52.3	1.658
	BCART	-	3.432×10^0	90.8	1.195
Ice Sheet (n=2,226, d=2)	SAGP	L=3, m=10	1.944×10^2	89.6	3.048
	SAGP	L=3, m=15	1.858×10^2	89.1	3.038
	SAGP	L=4, m=5	2.126×10^2	89.7	3.073
	GP	-	0.766×10^2	93.8	2.570
	SGP	m=5	2.575×10^2	100.0	5.638
	SGP	m=15	2.278×10^2	100.0	5.841
	SGP	m=150	1.637×10^2	100.0	6.365
	laGP	ALC	1.672×10^2	88.7	2.892
	laGP	MSPE	1.715×10^2	88.7	2.894
	BART	-	1.532×10^2	49.9	3.322
	BCART	-	2.231×10^2	91.3	3.026
UK Budget (n=1,519, d=8)	SAGP	L=2, m=10	3.486×10^1	92.3	2.327
	SAGP	L=2, m=15	3.370×10^1	92.8	2.312
	SAGP	L=3, m=10	3.478×10^1	92.2	2.327
	SAGP	L=3, m=15	3.366×10^1	92.7	2.313
	GP	-	3.105×10^1	94.2	2.245
	SGP	m=5	3.087×10^1	5.0	2.904
	SGP	m=15	3.053×10^1	13.9	2.855
	SGP	m=150	3.112×10^1	49.0	2.647
	laGP	ALC	4.459×10^1	55.3	2.679
	laGP	MSPE	4.529×10^1	54.7	2.685
	BART	-	3.065×10^1	48.1	2.605
	BCART	-	3.613×10^1	92.4	2.286

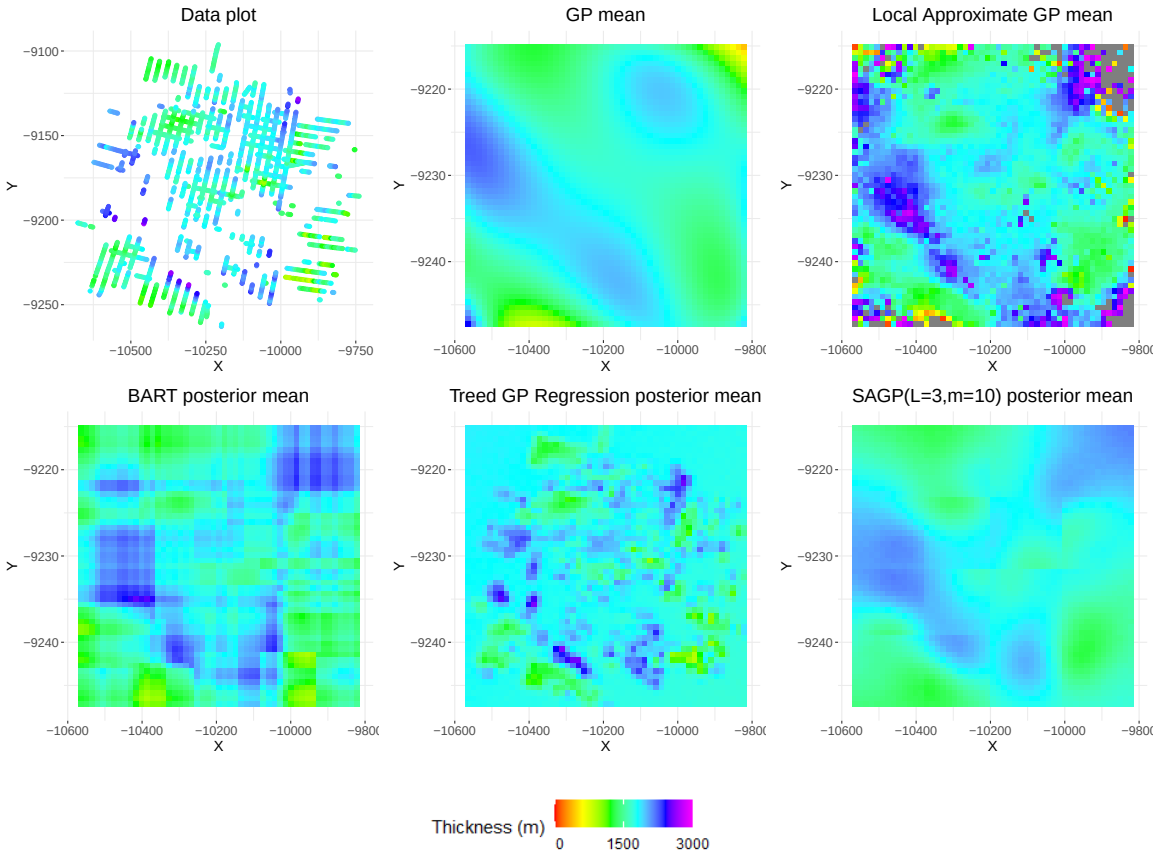


Figure 7: The scatter point plot shows the ice thickness in a region of Antarctica. The horizontal and the vertical axis are geographical coordinates in kilometers (km). The raster plot for predictive means of GP/laGP/BART(number of trees $m = 200$)/TGP/SAGP($m = 10, L = 3$) for ice sheet dataset and evaluation on a fine meshed grid (generated by steplengths of 0.02) on the original input domain. The color scales and the axis are the same in these plots.

The SAGP model fit is reasonable. It does not have extreme values, high variability in the mean response or boundary effects like some of the other models, yet retains most of the smoothness suggested by the full GP fit.

5.4 United Kingdom Budget Data

This dataset is a well-known econometric dataset first studied in Blundell et al. (1998). The dataset consists of a cross-section of 1,519 households drawn from the 1980-1982 British Family Expenditure Surveys. We attempt to predict the total household expenditure (rounded to the nearest 10 UK pounds sterling) with 8 variables as inputs. We do not use the variable of the number of children per household in the regression, since the dataset is cleaned in such a way that it contains only households with one or two children, as presented in Blundell et al. (1998).

We choose this dataset to explore the performance of our SAGP model in the higher-dimensional scenario. As mentioned by Gramacy et al. (2016), such a dataset of high-dimensionality ($d = 8$) will usually present computational challenges to classical GP models. Our main goal is to show that with reasonable increase of computational time, SAGP model has competitive performance. Since this dataset cannot be easily visualized, we only present quantitative results as shown in the next section.

5.5 Quantitative Performance Summary

The performance of SAGP and the alternative models considered is summarized quantitatively in Table 2. As in Section 4, we summarize the quantitative performance using 25% test set of original dataset to calculate out-of-sample RMSE, coverage of 95% credible intervals and interval scores. SAGP, SGP, GP, laGP, BART and BCART models were applied to all datasets. For SAGP, we generally selected $L = 2 \sim 4$ and $m = 5 \sim 15$ while for SGP we selected $m = 5, 15$ or 150. BART and BCART models were fit using their defaults, and laGP was fit using defaults but with both ALC and MSPE local design criteria.

Generally, we see that models could excel in one aspect (say RMSE) typically at the expense of another aspect of model fit quality, where the quality of fit depends on the dataset and application scenario. We notice that BART generally had lower coverage probability for the 95% PI and higher interval score. BCART had better coverage probability but generally was not the best in terms of RMSE. For the frequentist GP, two datasets exhibited good RMSE and two exhibited weaker RMSE performance. The GP is also less informative in terms of uncertainty quantification than the Bayesian models we considered. The laGP models often provided good RMSE performance, particularly with the ALC criterion, however the coverage was lower on the UK dataset.

In comparison, the SAGP model generally provided RMSE performance on par or near the best model for each dataset. The coverage also shows that SAGP models were consistent performers, especially compared to BART and laGP. We also see that SGP is uniformly worse than SAGP, often having either higher RMSE or worse coverage behavior. Overall, it is clear that SAGP is competitive with the best models for each dataset as summarized in Table 2, and we often prefer the qualitative aspect of the SAGP fits compared to some of the alternative models, as demonstrated earlier.

6. Discussion

The SAGP model effectively borrows ideas from both sparsification and localization. In particular, we divide the input domain X in such a way that we can choose enough pseudo-inputs and fit a sparse GP regression within the sub-region block of the partition, which also produces a trade-off for model parameters. We also showed that SAGP can achieve an effective reduction in computational cost (see Proposition 1) since all components within a layer can effectively be fit in parallel.

As a Bayesian additive model, SAGP provides uncertainty quantification and leads to accurate posterior inference. Along the model building process, we exhibit how the pseudo-inputs can be sampled to capture the aspect of model uncertainty, which is ignored with the fixed pseudo-inputs of SGP. The RP partition scheme outlined not only serves as a localization construction but also guarantees adequate pseudo-inputs for this resampling are available in each SAGP model component.As shown in the data analysis examples, the SAGP model is a competitive candidate compared to other generalization of GP regression methods. SAGP model can easily be generalized to higher dimensions, and our RP partition scheme is very flexible since it carries a hierarchical structure that allows us to analyze dataset in a multiscale way. With the homogeneous partition in one dimension, our RP scheme is similar to the one proposed by Bui and Turner (2014) and Lee et al. (2017); with heterogeneous partition in higher dimensions, our RP partition scheme is more flexible. For example, we can use binary partitioning in the first dimension but ternary partitioning in the second dimension. This will also preserve the hierarchical structure and allow us to decompose the high-dimensional data through different layers.

As for future works, there are various possible extensions of the proposed SAGP model. In terms of generalization of our current base model, we are interested in making the SAGP model admit different covariance kernels and different number of pseudo-inputs in each component. It is also of interest to extend the SAGP model to binary, count and categorical responses. To push the computational implementation of SAGP further and since independent sparse Gaussian process (SGP) regression models are fitted for each local component, it is readily seen that our model is parallelizable for efficient computation. Theoretically, we would also like to see a (frequentist) consistency result (Rocková and van der Pas, 2017) for the SAGP model and a careful analysis of the effect of the choice of priors in this model.

Acknowledgments

The authors wish to thank the helpful feedback of the editor, associate editor and two anonymous reviewers, which helped to substantially improve the paper. The work of M.T.P. was supported in part by the National Science Foundation under Agreement DMS-1916231 and in part by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under Award No. OSR-2018-CRG7-3800.3.

Appendix A. Correlation between Targets as Function of the Distance between Inputs

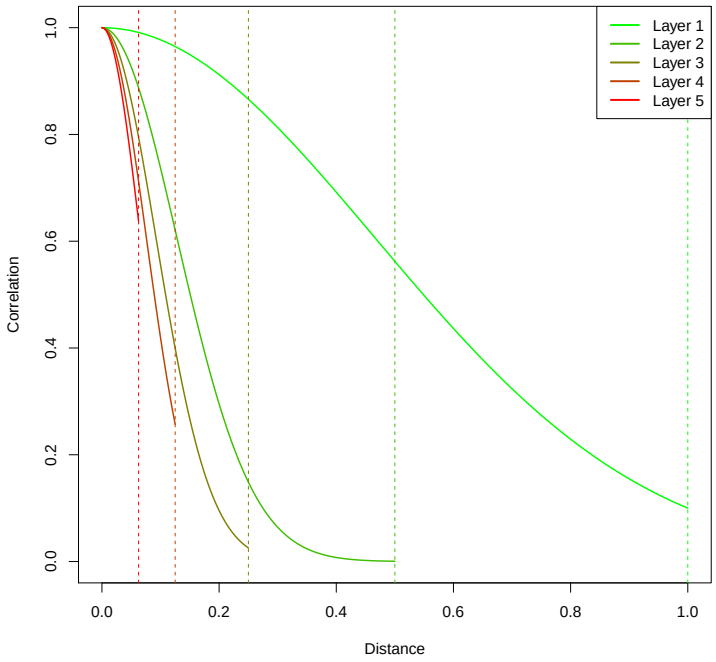


Figure 8: Given two one-dimensional inputs $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$ ($d = 1$), the figure represents the correlation between $f_j(\boldsymbol{x}_i)$ and $f_j(\boldsymbol{x}_{i'})$ (i.e., the targets of component j) as a function of the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$. We represent with different colors the correlation that is assumed for components at different layers ($L = 5$ in the Figure). Notably, the size of a component's domain B_j depends on the layer where the component is defined (the higher the layer index, the smaller the domain). Therefore, in our binary recursive partition scheme, the maximum possible distance between two inputs (highlighted with a vertical dashed line in the Figure) halves at each layer.

Appendix B. Proof of Proposition 1

We first calculate the total number of components in the SAGP models when the input-domain is $[0, 1]^d$. For each dimension $i = 1, 2, \ldots, d$, if we b_i-ary subdivide the $[0, 1]$ interval, then there are at most $|\mathcal{L}_\ell| = \prod_{i=1}^{d} b_i^{\ell-1}$ individual components in form of $B(\boldsymbol{c}_j, \boldsymbol{w}_j)$ in the ℓ-th layer of the RP scheme for $\ell = 1, 2, \ldots, L$.

For each component in the ℓ-th layer, the number of observations fitted to the j-th component is at most $|\mathcal{X}^{(j)}| \leq n$. Then we fit a SGP model with m pseudo-inputs, whose complexity is $\mathcal{O}(|\mathcal{X}^{(j)}| \cdot m^2)$. Then for the ℓ-th layer the total complexity is $\mathcal{O}(\sum_{B_j \in \mathcal{L}_\ell} |\mathcal{X}^{(j)}| \cdot m^2)$.

Therefore, for the ℓ-th layer the complexity is at most $\mathcal{O}(|\mathcal{L}_\ell|\cdot n\cdot m^2)$. Therefore we can compute the total complexity of the model as $\mathcal{O}(\sum_{\ell=1}^{L}|\mathcal{L}_\ell|\cdot n\cdot m^2) \asymp \mathcal{O}\left(\sum_{\ell=1}^{L}\prod_{i=1}^{d} b_i^{\ell-1}\cdot n\cdot m^2\right)$.

Appendix C. MCMC Algorithm for SAGP Model Fitting (Algorithm 3)

Algorithm 3 MCMC algorithm for SAGP model.

Input : RP partition scheme consisting of N components, Number of pseudo-inputs for each component m_j, Hyper-parameters for the prior of parameters, Observed dataset $\{\mathcal{X}, \boldsymbol{y}\}$.

Output: Posterior samples for parameters, $\bar{\mathcal{X}}^{(j)}, \bar{\boldsymbol{f}}_j$, Predictive posterior samples for $\boldsymbol{y}_*, \boldsymbol{f}_j, \boldsymbol{y}$.

Initialization of the parameter values
while *not converged* **do**
  Sample $\bar{\mathcal{X}}^{(j)}, j = 1, \ldots, N$ as in Algorithm 2.
  for j *in* $1:N$ **do**
    $\boldsymbol{r}_j \leftarrow \boldsymbol{y} - \sum_{l\neq j} \boldsymbol{K}_{nm_l}^{(l)} \left(\boldsymbol{K}_{m_l}^{(l)}\right)^{-1} \bar{\boldsymbol{f}}_l$
    $\bar{\boldsymbol{f}}_j \leftarrow N_{m_j}(\boldsymbol{Mean}_j, \boldsymbol{Var}_j)$ as in (10)
    $\eta_{j,\text{new}} \leftarrow \text{Uniform}(\eta_j \pm \text{bandwidth})$
    $\alpha \leftarrow \min(1, C\cdot$ `Model Likelihood`$(\eta_{j,new})/C\cdot$ `Model Likelihood`$(\eta_j))$
    if *Uniform(0,1)*$\leq \alpha$ **then**
      $\eta_j \leftarrow \eta_{j,\text{new}}$
    end
    `// For every burn-in steps/20 steps, we adjust bandwidth.`
    if *Acceptance rate of* $\eta_j \notin (0.39, 0.49]$ **then**
      Band width for proposing $\eta_j \leftarrow$ Acceptance rate of $\eta_j/0.44$
    end
  end
  $\sigma_\epsilon^2 \leftarrow \text{InverseGamma}\left(\alpha_\epsilon + \frac{n}{2}, \beta_\epsilon + \frac{1}{2}(\boldsymbol{y}-\hat{\boldsymbol{y}})^T(\boldsymbol{y}-\hat{\boldsymbol{y}})\right)$
end

Appendix D. Detailed Derivation of Posterior Distribution in Section 3.2.5

To clarify our derivations, we first stated following simple lemma that will be used, which can be derived from Woodbury identity (Horn and Johnson, 1990) or a direct verification (Rasmussen and Williams, 2006).

Lemma 1 *For a joint Gaussian distribution* $\boldsymbol{a}\in\mathbb{R}^n, \boldsymbol{b}\in\mathbb{R}^n$ *if*

$$\begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{pmatrix} \propto N_{n+m}\left(\begin{pmatrix} \boldsymbol{\mu_a} \\ \boldsymbol{\mu_b} \end{pmatrix}, \begin{pmatrix} C_{\boldsymbol{aa}} & C_{\boldsymbol{ab}} \\ C_{\boldsymbol{ba}} & C_{\boldsymbol{bb}} \end{pmatrix}\right) \tag{12}$$

then its conditional distribution is:

$$\boldsymbol{a} \mid \boldsymbol{b} \sim N_n\left(\boldsymbol{\mu_a} + C_{\boldsymbol{ab}}\left(C_{\boldsymbol{bb}}\right)^{-1}(\boldsymbol{b}-\boldsymbol{\mu_b}), C_{\boldsymbol{aa}} - C_{\boldsymbol{ab}}\left(C_{\boldsymbol{bb}}\right)^{-1} C_{\boldsymbol{ba}}\right) \tag{13}$$

In particular, for $f_l = f(\boldsymbol{x}_l), \bar{\boldsymbol{f}}_j = (\bar{f}_j(\bar{\boldsymbol{x}}_1), \ldots, \bar{f}_j(\bar{\boldsymbol{x}}_{m_j}))^T$ *and covariance kernel function* $K = K^{(j)}, K^{(j)}_{ll} = K^{(j)}(\boldsymbol{x}_l, \boldsymbol{x}_l)$ *if*

$$\left(\begin{array}{c} f_l \\ \bar{\boldsymbol{f}}_j \end{array} \right) \Bigg| \bar{\mathcal{X}}^{(j)}, \boldsymbol{x}_l \propto N_{1+m_j} \left(\left(\begin{array}{c} 0 \\ \boldsymbol{0}_{m_j} \end{array} \right), \left(\begin{array}{cc} K^{(j)}_{ll} & \boldsymbol{k}^{(j)T}_l \\ \boldsymbol{k}^{(j)}_l & \boldsymbol{K}^{(j)}_{m_j} \end{array} \right) \right) \tag{14}$$

then its conditional distribution is:

$$f_l \mid \bar{\boldsymbol{f}}_j, \bar{\mathcal{X}}^{(j)}, \boldsymbol{x}_l \sim N_1 \left(\boldsymbol{k}^{(j)T}_l \left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \bar{\boldsymbol{f}}_j, K^{(j)}_{ll} - \boldsymbol{k}^{(j)T}_l \left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \boldsymbol{k}^{(j)}_l \right) \tag{15}$$

We assume a Gaussian prior on the pseudo-targets as in (3.2.2).

$$P(\bar{\boldsymbol{f}}_j \mid \bar{\mathcal{X}}^{(j)}) \sim N_{m_j} \left(\bar{\boldsymbol{f}}_j \mid \boldsymbol{0}_{m_j}, \boldsymbol{K}^{(j)}_{m_j} \right) \tag{16}$$

and then use Bayesian rule on the parameter $\bar{\boldsymbol{f}}_j$, recalling that (2) determines the form of mean and variance of the Gaussian distribution $P(\boldsymbol{y} \mid \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \mathcal{X}, \boldsymbol{\kappa})$.

$$\begin{aligned}
&P(\bar{\boldsymbol{f}}_j \mid \boldsymbol{y}, \mathcal{X}, \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}) \\
\propto &P(\boldsymbol{y} \mid \bar{\boldsymbol{f}}_1, \ldots \bar{\boldsymbol{f}}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}) \times \\
&P(\bar{\boldsymbol{f}}_j \mid \{\boldsymbol{x}\}_n, \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_{j-1}, \bar{\boldsymbol{f}}_{j+1}, \ldots \bar{\boldsymbol{f}}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}) \qquad (17) \\
= &N_n \left(\boldsymbol{y} - \sum_{l \neq j} \boldsymbol{K}^{(l)}_{nm_l} \left(\boldsymbol{K}^{(l)}_{m_l} \right)^{-1} \bar{\boldsymbol{f}}_l \Bigg| \boldsymbol{K}^{(j)}_{nm_j} \left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \bar{\boldsymbol{f}}_j, \boldsymbol{\Lambda}^{(j)}_n + \sigma^2_\epsilon \boldsymbol{I}_n \right) \times \\
&N_{m_j} \left(\bar{\boldsymbol{f}}_j | \boldsymbol{0}_{m_j}, \boldsymbol{K}^{(j)}_{m_j} \right) \qquad (18)
\end{aligned}$$

We can derive the posterior using the normal normal conjugacy:

$$\begin{aligned}
&P(\bar{\boldsymbol{f}}_j \mid \boldsymbol{y}, \mathcal{X}, \bar{\boldsymbol{f}}_1, \ldots, \bar{\boldsymbol{f}}_{j-1}, \bar{\boldsymbol{f}}_{j+1}, \ldots \bar{\boldsymbol{f}}_N, \bar{\mathcal{X}}^{(1)}, \ldots, \bar{\mathcal{X}}^{(N)}, \boldsymbol{\kappa}) \\
\propto &\frac{1}{\sqrt{\left| 2\pi \left(\boldsymbol{\Lambda}^{(j)}_n + \sigma^2_\epsilon \boldsymbol{I}_n \right) \right|}} \exp \left\{ -\frac{1}{2} \left[\boldsymbol{y} - \sum_{l=1}^{N} \boldsymbol{K}^{(l)}_{nm_l} \left(\boldsymbol{K}^{(l)}_{m_l} \right)^{-1} \bar{\boldsymbol{f}}_l \right]^T \left(\boldsymbol{\Lambda}^{(j)}_n + \sigma^2_\epsilon \boldsymbol{I}_n \right)^{-1} \times \right. \\
&\left. \left[\boldsymbol{y} - \sum_{l=1}^{N} \boldsymbol{K}^{(l)}_{nm_l} \left(\boldsymbol{K}^{(l)}_{m_l} \right)^{-1} \bar{\boldsymbol{f}}_l \right] \right\} \times \\
&\frac{1}{\sqrt{\left| 2\pi \boldsymbol{K}_{m_j} \right|}} \exp \left\{ -\frac{1}{2} \bar{\boldsymbol{f}}^T_j \boldsymbol{K}^{-1}_{m_j} \bar{\boldsymbol{f}}_j \right\} \qquad (19)
\end{aligned}$$

We complete the squares inside the exponent,

$$\begin{aligned}
\propto &\exp \left\{ -\frac{1}{2} \bar{\boldsymbol{f}}^T_j \left(\boldsymbol{K}^{-1}_{m_j} + \left[\left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \boldsymbol{K}^{(j)}_{m_j n} \left(\boldsymbol{\Lambda}^{(j)}_n + \sigma^2_\epsilon \boldsymbol{I}_n \right)^{-1} \boldsymbol{K}^{(j)}_{nm_j} \left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \right] \right) \bar{\boldsymbol{f}}_j \right. \\
&\left. - \boldsymbol{f}^T_j \left(\boldsymbol{\Lambda}^{(j)}_n + \sigma^2_\epsilon \boldsymbol{I}_n \right)^{-1} \boldsymbol{K}^{(j)}_{nm_j} \left(\boldsymbol{K}^{(j)}_{m_j} \right)^{-1} \bar{\boldsymbol{f}}_j + \text{proportionally constant terms} \right\} \qquad (20)
\end{aligned}$$

After completing square we can obtain the mean and variance of the j-th component:

$$\begin{aligned}\boldsymbol{Mean}_j =& \left(\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} + \left[\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\right]\right)\times \\ & \left(\left(\boldsymbol{y}-\sum_{l\neq j}\boldsymbol{K}_{nm_l}^{(l)}\left(\boldsymbol{K}_{m_l}^{(l)}\right)^{-1}\bar{\boldsymbol{f}}_l\right)^T\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\right)^T \\ =& \boldsymbol{K}_{m_j}^{(j)}\boldsymbol{Q}_{m_j}^{(j)-1}\boldsymbol{K}_{m_jn}^{(j)}\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)^{-1}\left(\boldsymbol{y}-\sum_{l\neq j}\boldsymbol{K}_{nm_l}^{(l)}\left(\boldsymbol{K}_{m_l}^{(l)}\right)^{-1}\bar{\boldsymbol{f}}_l\right) \end{aligned} \tag{21}$$

By Woodbury identity, we know that for $\boldsymbol{Q}_{m_j}^{(j)} = \boldsymbol{K}_{m_j}^{(j)} + \boldsymbol{K}_{m_jn}^{(j)}\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}_{nm_j}^{(j)}$ we can write its inverse as

$$\boldsymbol{Q}_{m_j}^{(j)-1} = \left\{\boldsymbol{K}_{m_j}^{(j)-1} - \boldsymbol{K}_{m_j}^{(j)-1}\boldsymbol{K}_{m_jn}^{(j)}\left[\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)+\boldsymbol{K}_{nm_j}^{(j)}\boldsymbol{K}_{m_j}^{(j)-1}\boldsymbol{K}_{m_jn}^{(j)}\right]^{-1}\boldsymbol{K}_{nm_j}^{(j)}\boldsymbol{K}_{m_j}^{(j)-1}\right\}$$

Using this $m_j \times m_j$ matrix $\boldsymbol{Q}_{m_j}$, we can write down the covariance matrix $\boldsymbol{Var}_j$:

$$\boldsymbol{Var}_j = \left(\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} + \left[\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\right]\right)^{-1} \tag{22}$$

$$\begin{aligned} =& \boldsymbol{K}_{m_j}^{(j)} - \boldsymbol{K}_{m_j}^{(j)}\left[\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\right]\times \\ & \left[\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)+\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\right]^{-1}\times \\ & \boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_j}^{(j)} \end{aligned} \tag{23}$$

$$\begin{aligned} =& \boldsymbol{K}_{m_j}^{(j)} - \boldsymbol{K}_{m_jn}^{(j)}\left[\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)+\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\right]^{-1}\boldsymbol{K}_{nm_j}^{(j)} \\ =& \boldsymbol{K}_{m_j}^{(j)}\left\{\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1} - \right. \\ & \left.\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\left[\left(\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n\right)+\boldsymbol{K}_{nm_j}^{(j)}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\boldsymbol{K}_{m_jn}^{(j)}\right]^{-1}\left(\boldsymbol{K}_{m_j}^{(j)}\right)^{-1}\right\}\boldsymbol{K}_{nm_j}^{(j)} \end{aligned} \tag{24}$$

$$= \boldsymbol{K}_{m_j}^{(j)}\boldsymbol{Q}_{m_j}^{(j)-1}\boldsymbol{K}_{m_j}^{(j)} \tag{25}$$

Note that although we do need to invert an $n\times n$ matrix $\boldsymbol{\Lambda}_n^{(j)}+\sigma_\epsilon^2\boldsymbol{I}_n$, it is a diagonal matrix and hence easy to invert as claimed before.

Appendix E. Diagnostic Statistics for the SAGP Model on 1000 Batches of Simulated Dataset

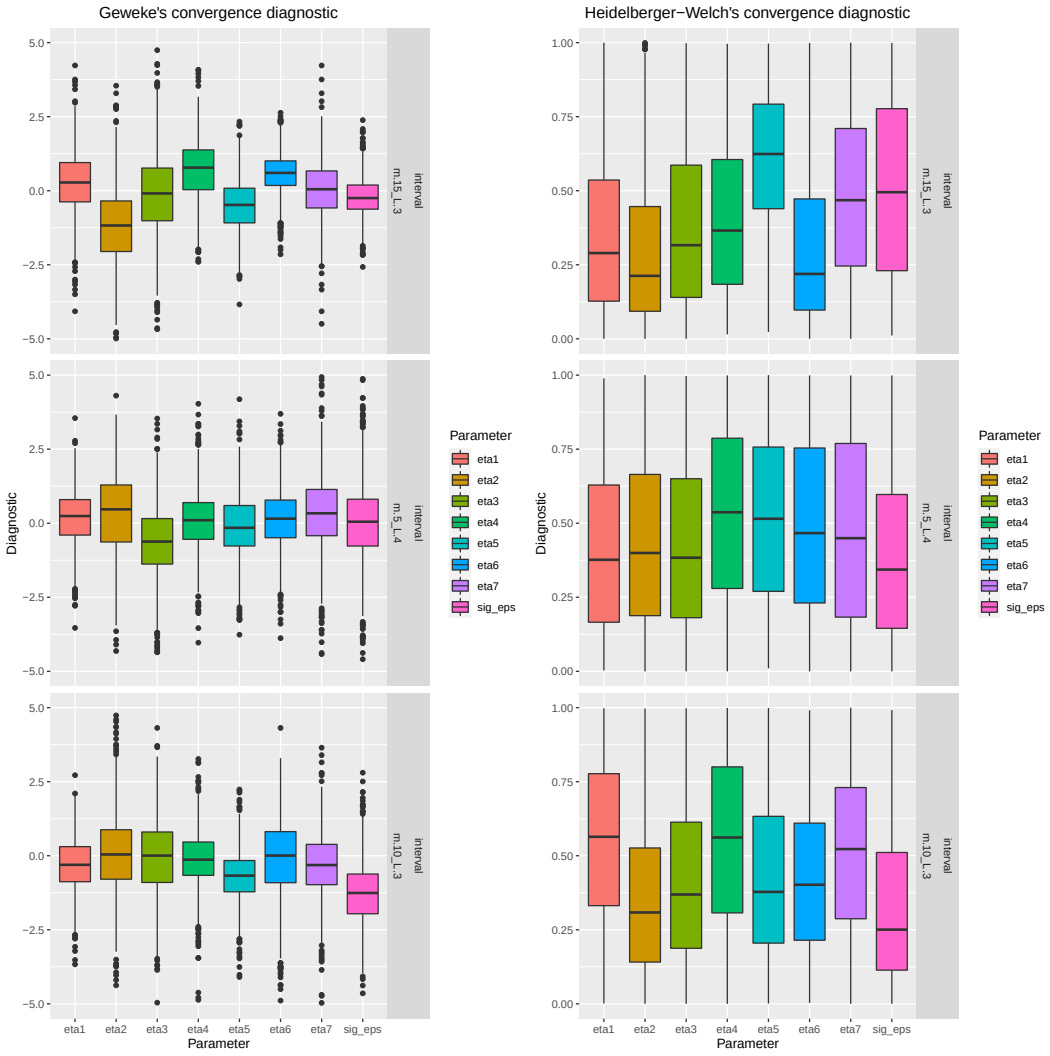


Figure 9: The panels of box plots show the Geweke's convergence diagnostic (Geweke et al., 1991) and Heidelberger-Welch's convergence diagnostic (Heidelberger and Welch, 1983) based on the MCMC sample of SAGP model, for parameter $\eta^{(j)}$ and σ_ϵ^2, calculated from the 1000 batches of simulated dataset from formula (11) with the testing set is random or interval.

Appendix F. Heart Rate Dataset Analyzed by SAGP Model Fitted with $m = 5$ and $L = 4$ (Figure 10)

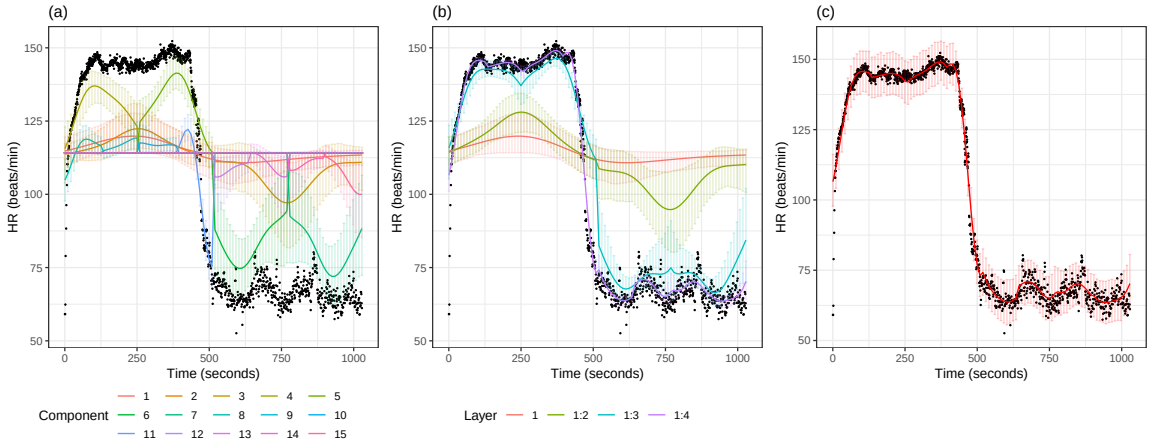


Figure 10: The panels show the observed HR values over time as black dots and results about the fit of the SAGP model with $m = 5$ and $L = 4$. Panel (a) shows the posterior means and the 95% CIs of the 15 additive components of the SAGP model on 100 equispaced locations on the support of the data. Panel (b) shows the posterior means and the 95% CIs of the sole component in layer 1 (red), of the components belonging to layer 1 and 2 (green) , of the components belonging to layer 1, 2, 3 and of the complete model, including components from layer 1, 2, 3 and 4. Panel (c) provides the predictive mean and the corresponding 95% prediction intervals.

References

Anjishnu Banerjee, David B. Dunson, and Surya T. Tokdar. Efficient gaussian process regression for large datasets. *Biometrika*, 100.1:75–89, 2012.

D. D. Blankenship et al. Ice thickness and surface elevation, southeastern ross embayment, west antarctica. *U.S. Antarctic Program (USAP) Data Center*, 2004. doi: doi:10.7265/N5WW7FKC. URL `https://www.usap-dc.org/view/dataset/609099`.

Richard Blundell, Alan Duncan, and Krishna Pendakur. Semiparametric estimation and consumer demand. *Journal of applied econometrics*, 13(5):435–461, 1998.

Leo Breiman. *Classification and Regression Trees*. Belmont, California: Wadsworth, 1984.

Thang D. Bui and Richard E. Turner. Tree-structured gaussian process approximations. In *Advances in Neural Information Processing Systems*, 2014.

Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. Bayesian cart model search. *Journal of the American Statistical Association*, 93.443:935–948, 1998.

Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4.1:266–298, 2010.

Hugh A. Chipman et al. High-dimensional nonparametric monotone function estimation using bart. *arXiv preprint arXiv:1612.01619*, 2016.

Andreas Damianou and Neil D. Lawrence. Deep gaussian processes. *Artificial Intelligence and Statistics*, 2013.

David G.T. Denison, Bani K. Mallick, and Adrian F.M. Smith. A bayesian cart algorithm. *Biometrika*, 85.2:363–377, 1998.

Emily Fox and David B. Dunson. Multiresolution gaussian processes. *Advances in Neural Information Processing Systems*, 2012.

Alan E. Gelfand, Susan E. Hills, Amy Racine-Poon, and Adrian F. M. Smith. Illustration of bayesian inference in normal data models using gibbs sampling. *Journal of the American Statistical Association*, 85.412:972–985, 1990.

Andrew Gelman et al. *Bayesian Data Analysis*. New York : CRC Press : Taylor & Francis Group, 2013.

John Geweke et al. *Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments*, volume 196. Federal Reserve Bank of Minneapolis, Research Department Minneapolis, MN, 1991.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102.477:359–378, 2007.

Robert B. Gramacy and Daniel W. Apley. Local gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, 24.2:561–578, 2015.

Robert B. Gramacy et al. tgp: an r package for bayesian nonstationary, semiparametric nonlinear regression and design by treed gaussian process models. *Journal of Statistical Software*, 19.9:1–46, 2007.

Robert B. Gramacy et al. lagp: large-scale spatial modeling via local approximate gaussian processes in r. *Journal of Statistical Software*, 72.1:1–46, 2016.

Trevor J. Hastie and Robert J. Tibshirani. *Generalized Additive Models*. New York : Chapman and Hall, 1990.

Trevor J. Hastie and Robert J. Tibshirani. Bayesian backfitting (with comments and a rejoinder by the authors. *Statistical Science*, 15.3:196–223, 2000.

Philip Heidelberger and Peter D. Welch. Simulation run length control in the presence of an initial transient. *Operations Research*, 31.6:1109–1144, 1983.

Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge: Cambridge university press, 1990.

Cari G. Kaufman, Mark J. Schervish, and Douglas W. Nychka. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103.484:1545–1555, 2008.

Neil D. Lawrence, Ralf Herbrich, and Matthias Seeger. Fast sparse gaussian process methods: The informative vector machine. In *Advances in neural information processing systems*, 2003.

Byung-Jun Lee, Jongmin Lee, and Kee-Eung Kim. Hierarchically-partitioned gaussian process approximation. In *Artificial Intelligence and Statistics*, 2017.

Haitao Liu et al. When gaussian process meets big data: A review of scalable gps. *IEEE Transactions on Neural Networks and Learning Systems*, 31(11):4405–4423, 2020.

Duy Nguyen-Tuong, Matthias Seeger, and Jan Peters. Model learning with local gaussian process regression. *Advanced Robotics*, 23.15:2015–2034, 2009.

Chiwoo Park and Daniel Apley. Patchwork kriging for large-scale gaussian process regression. *The Journal of Machine Learning Research*, 19.1:269–311, 2018.

Chiwoo Park and Jianhua Z. Huang. Efficient computation of gaussian process regression for large spatial data sets by patching local gaussian processes. *Journal of Machine Learning Research*, 17.174:1–29, 2016.

Matthew T. Pratola, C Devon Lin, and Peter F. Craigmile. Optimal design emulators: A point process approach. *arXiv preprint arXiv:1804.02089*, 2019.

Matthew T. Pratola, Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. Heteroscedastic bart via multiplicative regression trees. *Journal of Computational and Graphical Statistics*, 29:405–417, 2020.

Joaquin Quinonero-Candela and Carl E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

Carl E. Rasmussen and Christopher K.I. Williams. *Gaussian Process for Machine Learning*. Cambridge: MIT press, 2006.

Veronika Rocková and Stéphanie van der Pas. Posterior concentration for bayesian regression trees and forests. *arXiv preprint arXiv:1708.08734*, 2017.

Olivier Roustant, David Ginsbourger, and Yves Deville. Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software, Articles*, 51.1, 2012.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, 2006.

Edward Snelson and Zoubin Ghahramani. Local and global sparse gaussian process approximations. In *Artificial Intelligence and Statistics*, 2007.

James Andrew Storer. *An Introduction to Data structures and Algorithms*. New York : Springer, 2012.

Michali Titsias. Variational learning of inducing variables in sparse gaussian processes. *Artificial Intelligence and Statistics*, 2009.

Maria S. Zakynthinaki. Modelling heart rate kinetics. *PloS one*, 10.4, 2015.