Journal of Machine Learning Research 23 (2022) 1-37 Submitted 12/21; Revised 8/22; Published 9/22

Jump Gaussian Process Model for Estimating Piecewise Continuous Regression Functions

Chiwoo Park CPARK5@FSU.EDU
Department of Industrial and Manufacturing Engineering
Florida State University
2525 Pottsdamer St., Tallahassee, FL 32310-6046, USA.

Editor: Matthias Seeger

Abstract

This paper presents a Gaussian process (GP) model for estimating piecewise continuous regression functions. In many scientific and engineering applications of regression analysis, the underlying regression functions are often piecewise continuous in that data follow different continuous regression models for different input regions with discontinuities across regions. However, many conventional GP regression approaches are not designed for piecewise regression analysis. There are piecewise GP models to use explicit domain partitioning and pose independent GP models over partitioned regions. They are not flexible enough to model real datasets where data domains are divided by complex and curvy jump boundaries. We propose a new GP modeling approach to estimate an unknown piecewise continuous regression function. The new GP model seeks a local GP estimate of an unknown regression function at each test location, using local data neighboring the test location. Considering the possibilities of the local data being from different regions, the proposed approach partitions the local data into pieces by a local data partitioning function. It uses only the local data likely from the same region as the test location for the regression estimate. Since we do not know which local data points come from the relevant region, we propose a data-driven approach to split and subset local data by a local partitioning function. We discuss several modeling choices of the local data partitioning function, including a locally linear function and a locally polynomial function. We also investigate an optimization problem to jointly optimize the partitioning function and other covariance parameters using a likelihood maximization criterion. Several advantages of using the proposed approach over the conventional GP and piecewise GP modeling approaches are shown by various simulated experiments and real data studies.

Keywords: Piecewise Regression, Jump Regression, Gaussian Process Regression, Local Data Selection, Local Data Partitioning

1. Introduction

In many engineering and scientific studies, a response variable of interest suddenly jumps or sharply changes for a small perturbation of input variables. For example, in geostatistics, sensing responses from rock strata show sharp changes around sediment layers where rock types change from one to another (Kim et al., 2005). In materials science, the phases of materials suddenly change where phase transitions occur (Park et al., 2021). In econometrics, the responses from participants in social programs significantly differ under a bit of change in treatments (Kang et al., 2019). In environmental ecology, the degree of human-

induced disturbances to nature abruptly changes around environmental thresholds (Toms and Lesperance, 2003).

The input-response relations in these examples show a piecewise continuous nature, i.e., continuous everywhere in a domain but discontinuous across some boundary regions where responses jump. Under these cases, data across boundaries are independent, and only data within the same homogeneous region are correlated. Stationary Gaussian process (GP) regression models are inappropriate for capturing these piecewise continuous input-response relations. The nature of the stationary GP covariance modeling poses high correlations between closely located data points (Rasmussen and Williams, 2006). Thus it promotes more continuity and smoothness. Nonstationary GP modelings may better handle the piecewise nature, locally adjusting covariance parameters (Sampson and Guttorp, 1992). However, the nonstationary GP modeling does not solve the issue completely, because data across boundaries are still correlated under the nonstationary setting. The conventional GP models would only be appropriate for continuous and smooth input-output dynamics. Applying these conventional approaches to data from piecewise continuous response surfaces would give an inadequate representation of the underlying regression function, incurring significant estimation biases. This paper presents a new GP modeling for estimating piecewise continuous regression functions.

A few studies in the GP literature potentially work for piecewise continuous regression functions. By these methods, the input domain is first partitioned into multiple regions, and then the data from each region is modeled as an independent GP model. Therefore, the data are correlated within regions and independent across regions. The best-known examples include piecewise GPs with a tessellation-based domain partitioning (Kim et al., 2005; Heaton et al., 2017; Pope et al., 2021; Luo et al., 2021), binary splitting of the spatial axes (Konomi et al., 2014), and Bayesian treed GP (Gramacy and Lee, 2008; Taddy et al., 2011). Kim et al. (2005) use the Voronoi tessellation to partition an input domain into triangular regions and use an independent and stationary GP to model a response surface within each triangular region. The number and locations of the triangular regions are jointly estimated with the stationary GP models, using a Bayesian sampling procedure. Luo et al. (2021) partitioned a spatial adjacency graph constructed by the Delaunay triangulation. Pope et al. (2021) discuss active learning of the piecewise regression model. One major issue with the tessellation-based methods is that the Voronoi tessellation is practical only in a spatial domain and computationally expensive in three or more dimensions.

A Bayesian treed GP is another piecewise GP approach. Gramacy and Lee (2008) uses a dyadic treed partitioning for splitting an input domain by axis-aligned directions and fits a stationary GP model to data from each leaf of the tree. Konomi et al. (2014) used the Bayesian CART tree to split an input domain. These tree partitioning and piecewise GP models are jointly estimated using the Markov Chain Monte Carlo sampling, which is computationally slow. Dynamic tree model (Taddy et al., 2011) is simpler and more computationally feasible, but it is limited to piecewise constant or linear models. Similar treed regression modelings have been used with non-GP models (Malloy and Nowak, 2014).

There are other partitioned GP models (Park and Huang, 2016; Park and Apley, 2018; Cortes et al., 2019), but they work when the domain partitioning is known as a priori, or the partition can be easily estimated by a simple clustering. These approaches mainly exploits data partitioning to address a big-N issue of the GP. There are many non-GP

regression models that can potentially model piecewise continuous regression functions, including jump regression analysis (Park et al., 2021; Qiu, 2009; Kang et al., 2021). These non-Bayesian approaches only provide mean estimates, but not variance estimates that are useful to understand the prediction uncertainty. In addition, they are developed from two-dimensional (2D) input domains, so their application for higher-dimensional input spaces is limited. Their main applications are 2D image denoising and enhancements (Gijbels et al., 2006).

Some major shortcomings of the existing piecewise GP models are that their partitioning schemes such as axis-aligned partitions or triangulation are too restrictive to model real datasets where the response jumps occur across complex and curvy boundaries. Forcing their use for the real cases would lead to overly partitioned models with few data points per region or under-partitioned models that poorly represent response jumps across boundaries. Accordingly, their predictions are no better than the conventional GP, which is shown by Park et al. (2021) and is also reported in Section 5. By the nature of their domain partitioning, the existing piecewise GP models are more appropriate for spatial domains. They are not easy to extend for higher dimensions.

We believe that a local regression modeling (versus the existing global-scale modelings) would be a more natural choice to adapt to local features like jumps and discontinuities. This paper will develop a new local GP model to estimate piecewise continuous regression surfaces. Our GP model seeks a local GP estimate of an unknown regression function at a test location, using local data neighboring to the test location. Since the local data may come from heterogeneous regions, the proposed method partitions the local data into pieces and takes only a piece of the local data belonging to the same region as the test location for the regression estimate. Since we do not know which local data belongs to the region, we consider several parametric forms of a local data partitioning function and then develop a likelihood maximization formulation to jointly optimize the parameters of the partitioning function with other covariance hyperparameters. The proposed method is referred to as the *Jump GP* or shortly JGP.

The proposed approach has many major advantages over the existing piecewise GP approaches. First, the local data partitioning applied in JGP is a lot simpler and more flexible than the global-level partitioning used in the existing piecewise GP. Complex and curvy boundaries dividing a regression domain can be effectively approximated at a local level by simpler geometries such as locally linear or polynomial boundaries, while the complexities are too substantial to be handled at a global level. We have many numerical illustrations to support this argument in Section 3. Second, compared to many existing approaches that only deal with spatial domains, the proposed approach works for higher dimensions. We have tested at least up to 10 dimensions numerically in Section 4. Last, the proposed approach as a variant of the local GP works for a large N, but many treed GP approaches using expensive MCMC do not.

We also articulate that like the local GP, the proposed approach produces the pointwise estimates of a regression function with pointwise means and variances. However, it does not provide an explicit form of a regression function, which could limit the applicability of the proposed method for a few GP applications requiring an explicit form of a regression function. The proposed method can be used for a vast majority of GP applications which

just need pointwise means and variances, e.g., regression, Bayesian optimization, surrogate modeling, model calibration, uncertainty quantification, and etc.

We organize the remainder of this paper as follows. Section 2 proposes a new GP regression approach, discussing the limitation of conventional local GP approaches. That section explains many modeling choices, such as local data partitioning functions and implementation details for model parameter estimation with some numerical illustrations. Section 3 presents a numerical analysis of the proposed approach with designed simulation experiments with two-dimensional input domains. The main purpose of the section is to visually illustrate how the proposed work for several settings of a noise level, a training data size, a local data size, and different jump boundary configurations. In Section 4, we extend the simulation study with high dimensional problems to see how the proposed approach works for different input dimensions. Section 5 presents numerical analysis with three benchmark datasets, comparing the proposed approach with conventional local GP approaches and two existing piecewise GP models. We conclude this paper in Section 6.

2. Jump Gaussian Process (JGP)

Let $\mathcal{X}$ denote a domain of a function in $\mathbb{R}^d$. We consider a problem of estimating an unknown regression function $f : \mathcal{X} \to \mathbb{R}$ that relates inputs $\boldsymbol{x} \in \mathcal{X}$ to a real response variable. We assume that the regression function is piecewise continuous in the form of

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} f_k(\boldsymbol{x}) 1_{\mathcal{X}_k}(\boldsymbol{x}), \tag{1}$$

where $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_K$ are a partition of $\mathcal{X}$ (i.e. disjoint subsets which union is equal to $\mathcal{X}$), $1_{\mathcal{X}_k}(\boldsymbol{x})$ is an indicator function that determines whether $\boldsymbol{x}$ belongs to region $\mathcal{X}_k$, and $f_k(\boldsymbol{x})$ is a continuous function that represents the regression model on region $\mathcal{X}_k$. Each function $f_k(\boldsymbol{x})$ is assumed a random realization of GP with a constant mean $\mu_k \in \mathbb{R}$ and stationary covariance function $c_k(\cdot, \cdot)$, and we assume mutual independence among the functions,

$$\text{Independence:} \qquad f_k \text{ is independent of } f_j \text{ for } j \neq k. \tag{2}$$

Here, the 'independence' implies a zero correlation between two data points across regions. For a concise development of the proposed idea, we may restrict the covariance function c_k to a parametric family, $\{c(\cdot, \cdot; \theta); \theta \in \Theta\}$, but one can easily extend the proposed idea for other general covariance forms. Let $\theta_k \in \Theta$ denote the region-dependent covariance parameter for region $\mathcal{X}_k$, so we have $c_k(\cdot, \cdot)$ equal to $c(\cdot, \cdot; \theta_k)$. We assume mean heterogeneity across regions,

$$\text{Heterogeneity:} \qquad \mu_j \neq \mu_k \quad \forall \quad j \neq k. \tag{3}$$

The covariance parameters can be different across regions, $\theta_j \neq \theta_k$, or they can be same.

The modeling assumptions (1), (2) and (3) comprises the underlying regression model f. We want to estimate f at a test location $\boldsymbol{x}_* \in \mathcal{X}$, using N noisy observations of the unknown function, where an observation at $\boldsymbol{x}$ is assumed

$$y = f(\boldsymbol{x}) + \epsilon(\boldsymbol{x}), \tag{4}$$

and independent noises $\epsilon(\boldsymbol{x}) \sim \mathcal{N}(0, \sigma^2(\boldsymbol{x}))$. The noise variance $\sigma^2(\boldsymbol{x})$ is assumed to change very smoothly within the same region (but can be significantly different for different regions), so $\sigma^2(\boldsymbol{x})$ is approximately constant around a small neighborhood of $\boldsymbol{x}$ within the region. Here, all $f_k(x)$'s are unknown, and the partition $\{\mathcal{X}_k, k = 1, \ldots, K\}$ are unknown, and even the region number K is unknown. One may seek to explicitly model and estimate the partition, multiple regression models and K with data as in the existing piecewise GP approaches (Gramacy and Lee, 2008; Kim et al., 2005). However, their explicit modeling of the domain partition are not flexible enough to accommodate complex and curvy boundaries between continuous regions. Forcing the explicit modeling of these complexities would lead to overly partitioned models with few data points per sub-region or poorly partitioned models that do not representing intensity jumps across complex boundaries.

We believe that a local and transductive learning approach (Vapnik, 2013) would be a more natural choice to adapt to the local nature of jumps and discontinuities around test points. Here we investigate local GP modeling approaches to take a local GP estimate of f for each test location $\boldsymbol{x}_* \in \mathcal{X}$ without explicit estimation of $f_k(\boldsymbol{x})$ and $\mathcal{X}_k$. We first summarize the conventional local GP approach and discuss its limitation in estimating the regression model (1) to motivate a new approach in Section 2.1. Then, we propose our JGP in Section 2.2, followed by two statistical inference algorithms in Sections 2.3 and 2.4. Sections 2.5 and 2.6 detail the model developments with specific modeling choices.

2.1 Limitation of the Conventional Local GP in Estimating the model (1)

In the conventional local GP, a small subset of training data nearing a test location $\boldsymbol{x}_*$ is first chosen, e.g., choosing the n-nearest neighborhood of $\boldsymbol{x}_*$ or exploiting the existing local data selection criterion (Gramacy and Apley, 2015). Let us denote the local data by

$$\mathcal{D}_n = \{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}, \tag{5}$$

where $\boldsymbol{x}_i$ and y_i represent the input vector and the response value of the ith local data. Please note that D_n is dependent on $\boldsymbol{x}_*$. For notational brevity, we use the notation D_n instead of $D_n(\boldsymbol{x}_*)$.

The conventional local GP model fits a stationary GP model to the local data. Locally, f is assumed a realization of a Gaussian process with a constant mean μ and covariance function $c(\cdot, \cdot; \theta)$, and y_i is a noisy observation of f at $\boldsymbol{x}_i$ with the Gaussian noise and noise variance σ^2. This can be expressed as

$$\begin{aligned} &y_i | f(\boldsymbol{x}_i) \sim \mathcal{N}(f(\boldsymbol{x}_i), \sigma^2) \\ &f(\boldsymbol{x}_i) \sim \mathcal{N}(\mu, c(\boldsymbol{x}_i, \boldsymbol{x}_i; \theta)) \\ &\mathbb{C}\mathrm{ov}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) = c(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta). \end{aligned}$$

It gives the posterior predictive distribution of f at a test location $\boldsymbol{x}_*$ as the Gaussian distribution with

$$\begin{aligned} &\text{mean:} \quad \mu(\boldsymbol{x}_*) = \mu + \boldsymbol{c}_n^T (\sigma^2 \boldsymbol{I}_n + \boldsymbol{C}_n)^{-1} (\boldsymbol{y}_n - \mu \boldsymbol{1}_n), \text{ and} \\ &\text{variance:} \quad s(\boldsymbol{x}_*) = c(\boldsymbol{x}_*, \boldsymbol{x}_*; \theta) - \boldsymbol{c}_n^T (\sigma^2 \boldsymbol{I}_n + \boldsymbol{C}_n)^{-1} \boldsymbol{c}_n, \end{aligned} \tag{6}$$

where $\boldsymbol{c}_n = [c(\boldsymbol{x}_i, \boldsymbol{x}_*; \theta) : i = 1, ..., n]$ is a $n \times 1$ vector of the covariance values between the local data and the test location, $\boldsymbol{C}_n = [c(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta) : i, j = 1, ..., n]$ is a $n \times n$ matrix of the

covariance values evaluated for all pairs of the local data, $\mathbf{1}_n$ is a $n \times 1$ vector of ones, $\boldsymbol{I}_n$ represents a $n \times n$ identity matrix, and $\boldsymbol{y}_n = [y_i : i = 1, ..., n]$. The hyperparameters μ and θ of the Gaussian process prior are optimized by maximizing the likelihood. The estimate of μ is $\bar{\mu}^* = \mathbf{1}_n^T \boldsymbol{C}_n^{-1} \boldsymbol{y}_n / \mathbf{1}_n^T \boldsymbol{C}_n^{-1} \mathbf{1}_n$ (Mu et al., 2017). The estimate $\bar{\mu}^*$ can be written as a weighted average of the local responses in the form of $\sum_{i=1}^n \alpha_i y_i$ for some positive weights $\{\alpha_i, i = 1, ..., n\}$ satisfying $\sum_{i=1}^n \alpha_i = 1$. One can easily show that the bias of the local GP estimate $\mu(\boldsymbol{x}_*)$ is given as

$$\begin{aligned} Bias[\mu(\boldsymbol{x}_*)] &= \mathbb{E}[\mu(\boldsymbol{x}_*) - f(\boldsymbol{x}_*)] \\ &= \mathbb{E}[\bar{\mu}_* + \boldsymbol{c}_n^T (\sigma^2 \boldsymbol{I}_n + \boldsymbol{C}_n)^{-1} (\boldsymbol{y}_n - \bar{\mu}_* \mathbf{1}_n)] - \mathbb{E}[f(\boldsymbol{x}_*)] \\ &= \mathbb{E}[\bar{\mu}_*] - \mathbb{E}[f(\boldsymbol{x}_*)] + \boldsymbol{c}_n^T (\sigma^2 \boldsymbol{I}_n + \boldsymbol{C}_n)^{-1} (\mathbb{E}[\boldsymbol{y}_n] - \mathbb{E}[\bar{\mu}_*] \mathbf{1}_n). \end{aligned} \tag{7}$$

Theorem 1 *The conventional local GP estimate $\mu(\boldsymbol{x}_*)$ is unbiased almost everywhere if and only if $\mathbb{E}[y_i] = \mathbb{E}[f(\boldsymbol{x}_*)]$ for every $i = 1, \ldots, n$.*

The proof is simple as follows. If the condition holds, $\mathbb{E}[\bar{\mu}_*] - \mathbb{E}[f(\boldsymbol{x}_*)] = 0$, and $\mathbb{E}[\boldsymbol{y}_n] - \mathbb{E}[\bar{\mu}_*] \mathbf{1}_n$ becomes a zero vector. Therefore, the last line of (7) would be zero for all possible cases of the training inputs $(\boldsymbol{x}_i, i = 1, \ldots, n)$. So, the condition stated in the theorem should be a sufficient condition for the unbiasedness. We use *proof by contrapositive* to show that the condition is also a necessary condition. Suppose that $I_\pm = \{i : \mathbb{E}[y_i] \neq \mathbb{E}[f(\boldsymbol{x}_*)]\}$ is not empty. The expectation of the estimate $\bar{\mu}_*$ can be written as

$$\mathbb{E}[\bar{\mu}_*] - \mathbb{E}[f(\boldsymbol{x}_*)] = \sum_{i=1}^n \alpha_i (\mathbb{E}[y_i] - \mathbb{E}[f(\boldsymbol{x}_*)]) = \sum_{i \in I_\pm} \alpha_i (\mathbb{E}[y_i] - \mathbb{E}[f(\boldsymbol{x}_*)]).$$

The term goes to zero only if $(\boldsymbol{x}_i, i \in I_\pm)$ are placed such that the associated $(\alpha_i, i \in I_\pm)$ belongs to the null space of $(\mathbb{E}[y_i] - \mathbb{E}[f(\boldsymbol{x}_*)], i \in I_\pm)$. Since the null space has a zero Lebesgue measure in $\mathbb{R}^{|I_\pm|}$, we can say that the term above is non-zero almost everywhere. Similarly, the term $\mathbb{E}[\boldsymbol{y}_n] - \mathbb{E}[\bar{\mu}_*]\mathbf{1}$ would be non-zero almost everywhere, so $Bias[\mu(\boldsymbol{x}_*)]$ could be non-zero almost everywhere. Therefore, the condition stated in the theorem is also a necessary condition.

2.2 Jump GP: New GP Estimate with Local Data Selection

To mitigate the bias, we propose to partition local data $\mathcal{D}_n$ into two parts: a group of the local data that belong to the same region as a test location $\boldsymbol{x}_*$ and the remainder. Since the second group of the local data is independent of $f(\boldsymbol{x}_*)$ due to independence (2), we define a new local GP estimate of $f(\boldsymbol{x}_*)$ using only the first part of the local data. Since the new estimate satisfies the unbiasedness condition in Theorem 1, it should be unbiased. A major challenge in realizing this idea is that we do not know which local data belong to the same region as the test location. Here we propose a data-driven approach to identify which local data points belong to the region of the test location. The bias of the proposed estimate would be dependent on the accuracy of the local data selection.

We acknowledge similarities between the proposed approach and the local approximate Gaussian process regression (Gramacy and Apley, 2015, shortly referred to as the laGP hereafter). Both produce local GP estimates, and they all select local data points with

particular criteria. The major difference is that the proposed approach exploits the response values of training data to select local data points, while the laGP does not. The laGP selects local data points, starting with a small nearest neighbors and adding additional local data points incrementally. The incremental step does not exploit the response variable values of training data, so it cannot take account of the response jumps. Unsurprisingly, the performance of the laGP remains comparable to the conventional local GP for piecewise continuous regression, as we show later in Section 5.

We start by introducing a binary latent variable $Z_i \in \{0, 1\}$ to represent whether a local data point $\boldsymbol{x}$ belongs to the same region as the test location $\boldsymbol{x}_*$ as

$$Z_i = \begin{cases} 1 & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_* \text{ belong to the same region} \\ 0 & \text{otherwise.} \end{cases}$$

When the latent variable values are known, we can use the latent variables to divide the local data $\mathcal{D}_n$ into two groups:

$$\begin{aligned} \mathcal{D}_o &= \{(\boldsymbol{x}_i, y_i) : Z_i = 0\} \text{ and} \\ \mathcal{D}_* &= \{(\boldsymbol{x}_i, y_i) : Z_i = 1\}. \end{aligned} \tag{8}$$

Please note that the local data $\mathcal{D}_*$ and the test location $\boldsymbol{x}_*$ belong to the same region. According to our modeling proposition (1), they should be governed by one common stationary GP model. If $\boldsymbol{x}_* \in \mathcal{X}_k$, the governing GP model would be f_k that has constant mean function μ_k and covariance function $c(\cdot, \cdot; \theta_k)$. Since we do not know which region $\boldsymbol{x}_*$ belongs to, we denote the governing GP model by f_* with a constant mean m_* and a covariance function $c(\cdot, \cdot; \theta_*)$; if $\boldsymbol{x}_* \in \mathcal{X}_k$, f_*, m_*, and θ_* actually refer to f_k, m_k, and θ_k, respectively. Accordingly, conditioned on $Z_i = 1$, we model y_i as a noisy output of an unknown $f_*(\boldsymbol{x}_i)$,

$$\begin{aligned} p(y_i | Z_i = 1, f_{*,i}) &= \mathcal{N}_1(y_i | f_{*,i}, \sigma^2) \\ f_* &\sim GP(m_*, c(\cdot, \cdot; \theta_*)), \end{aligned} \tag{9}$$

where we use a short notation $f_{*,i}$ to represent $f_*(\boldsymbol{x}_i)$, and $\mathcal{N}_1(\cdot | f_{*,i}, \sigma^2)$ represents an univariate normal density function with mean $f_{*,i}$ and variance σ^2. The other local data $\mathcal{D}_o$ really do not affect f_*, because they are independent of f_*. We use a simple dummy likelihood for $\mathcal{D}_o$, which is defined as a uniform density,

$$p(y_i | Z_i = 0) = U \tag{10}$$

for a positive constant U. The constant value is related to how much deviations of y_i from the model (9) are tolerable to declare $Z_i = 1$ (i.e. whether y_i and $f_{*,i}$ are from the same GP realization). Intuitively, if $Z_i = 1$, they are correlated through covariance function $c(\cdot, \cdot; \theta_*)$, so their values become close. Otherwise, they are statistically independent so become more distinct with high possibilities. The constant U determines how much deviation of y_i from $f_{i,*}$ is sufficient to declare the independence. If y_i deviates from $f_{*,i}$ to the degree for inducing $p(y_i | Z_i = 1, f_{*,i}) < U$, so the data is likely classified as $Z_i = 0$. Due to the Gaussianity of the model $p(y_i | Z_i = 1, f_{*,i})$, we propose to set

$$U = \phi\left(c \cdot \sigma\right),$$

where $\phi(\cdot)$ is a standard normal density function. With the choice, data deviating from $f_{*,i}$ by more than $c\sigma$ gives $p(y_i|Z_i = 1, f_{*,i}) < U$, so the data is likely classified as $Z_i = 0$. Typically, data outside ± 3 sigma range from the normal are considered outliers in literature, e.g., 3σ limit in quality control (Goh and Xie, 2003), so a practical choice of c would be 2 or 3. We use $c = 2.5$ for all numerical examples. We admit that this choice may not be optimal. We consider it a practical choice, given that considering it as another model parameter may cause a model non-identifiability issue.

The uncertainty over the latent variable Z_i is modeled by a sigmoid function π of an unknown partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega})$,

$$p(Z_i = 1|\boldsymbol{x}_i, \boldsymbol{\omega}) = \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega})), \tag{11}$$

where the partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega})$ is parameterized by a parameter $\boldsymbol{\omega}$. The partitioning function determines whether $\boldsymbol{x}_i$ belongs to $\mathcal{D}_o$ or $\mathcal{D}_*$. When $g(\boldsymbol{x}_i, \boldsymbol{\omega}) \geq 0$, $p(Z_i = 1|\boldsymbol{x}_i, \boldsymbol{x}_*, \boldsymbol{\omega})$ is greater than $p(Z_i = 0|\boldsymbol{x}_i, \boldsymbol{\omega})$, so $\boldsymbol{x}_i$ is likely classified as a data point in $\mathcal{D}_*$. Otherwise, it is likely to be classified as a data point in $\mathcal{D}_o$. The form of the function g influences the boundary dividing $\mathcal{D}_o$ and $\mathcal{D}_*$. For example, if g is chosen as a linear function $\boldsymbol{\omega}^T[1, \boldsymbol{x}]$, we have a linear dividing boundary. One can use a non-linear form $\boldsymbol{\omega}^T[1, \boldsymbol{\psi}(\boldsymbol{x})]$ with a choice of non-linear basis functions, $\boldsymbol{\psi}(\boldsymbol{x}) = (\psi_\ell(\boldsymbol{x}); \ell = 1, \ldots, L)$. Another option would be to use another layer of GP modeling for $g(\boldsymbol{x}, \boldsymbol{\omega})$, which will constitute a deep GP model with two GP layers. The model estimations would be more complicated. For a concise representation of the overall idea, we will use a non-GP parametric model,

$$g(\boldsymbol{x}, \boldsymbol{\omega}) = \boldsymbol{\omega}^T[1, \boldsymbol{\psi}(\boldsymbol{x})].$$

The function should satisfy one constraint

$$g(\boldsymbol{x}_*, \boldsymbol{\omega}) \geq 0 \tag{12}$$

to constrain that $\mathcal{D}_*$ and $\boldsymbol{x}_*$ are on the same side of the boundary.

We need to estimate a set of latent variables, $\boldsymbol{Z} = (Z_i, i = 1, \ldots, n)$ and $\boldsymbol{f}_* = (f_{*,i}, i = 1, \ldots, n)$, as well as unknown model parameters, $\boldsymbol{\Theta} = \{\boldsymbol{\omega}, m_*, \theta_*, \sigma^2\}$. One natural choice for the mixture model may be to use the expectation maximization (EM), which iterates an E-step (taking the expectation of the joint log density with respect to the latent variables) and an M-step (maximizing the expectation with respect to the model parameters). The major difficulty here is that obtaining the joint posterior distribution of $\boldsymbol{Z}$ and $\boldsymbol{f}_*$ is not tractable, which creates complications in the E-step. To avoid that complication, we can use EM variants such as the *classification EM* (Bryant and Williamson, 1978; Gupta and Chen, 2010), *Monte Carlo EM* (Levine and Casella, 2001), and *variational EM* (Bishop, 2006). We realized that the Monte Carlo EM is too computationally slow. Here we describe inference steps for the other two EM variants.

2.3 JGP-CEM: Classification EM Inference for Jump GP

Here we describe the statistical inference of JGP with the *classification EM* (Bryant and Williamson, 1978; Gupta and Chen, 2010), which replaces the E-step with a pointwise maximum a posteriori (MAP) estimation of latent variables.

To describe the steps, we first write priors and likelihood. From (11), the prior for $\boldsymbol{Z}$ is

$$p(\boldsymbol{Z}|\boldsymbol{\omega}) = \prod_{i=1}^{n} \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega}))^{Z_i}(1 - \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega})))^{1-Z_i}.$$

From the Gaussian process prior (9),

$$p(\boldsymbol{f}_*|m_*, \theta_*) = \mathcal{N}_n(\boldsymbol{f}_*|m_*\mathbf{1}_n, \boldsymbol{C}_n),$$

where $\mathcal{N}_n(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ represents a n-variate normal density with mean $\boldsymbol{\mu} \in \mathbb{R}^n$ and a $n \times n$ covariance $\boldsymbol{\Sigma}$, and $\boldsymbol{C}_n = [c(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta_*) : i, j = 1, ..., n]$ is a square matrix of the covariance values evaluated for all pairs of the local data. The conditional distribution of all local data $\boldsymbol{y}_n$ given the latent variables is

$$p(\boldsymbol{y}_n|\boldsymbol{f}_*, \boldsymbol{Z}, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}_1(y_i|f_{*,i}, \sigma^2)^{Z_i} U^{1-Z_i}. \tag{13}$$

The classification EM algorithm iterates the following E-step and M-step to estimate the model parameter Θ:

• **E-step:** Given the parameter estimates $\Theta^{(t)} = \{m_*^{(t)}, \theta_*^{(t)}, \boldsymbol{\omega}^{(t)}, \sigma^{2(t)}\}$ from the past iteration, take the MAP estimate of $\boldsymbol{Z}$ and $\boldsymbol{f}_*$ by minimizing the negative log posterior distribution,

$$(\boldsymbol{Z}^{(t+1)}, \boldsymbol{f}_*^{(t+1)}) = \underset{\boldsymbol{Z}, \boldsymbol{f}_*}{\operatorname{argmin}} \quad -\log\{p(\boldsymbol{Z}|\boldsymbol{\omega}^{(t)})p(\boldsymbol{f}_*|m_*^{(t)}, \theta_*^{(t)})p(\boldsymbol{y}_n|\boldsymbol{f}_*, \boldsymbol{Z}, \sigma^{2(t)})\} \tag{14}$$

We use the block coordinate descent algorithm to solve the problem (14), which minimizes the objective function along with one coordinate direction at a time. The coordinate direction to be optimized can be chosen cyclically or randomly. Below we describe coordinate-wise update steps. Conditioned on $\boldsymbol{f}_*$, the posterior density of $Z_i = 1$ is given by

$$\begin{aligned} &P(Z_i = 1|\boldsymbol{y}_n, \boldsymbol{f}_*, \Theta^{(t)}) \\ &\quad = \frac{\pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)}))\mathcal{N}_1(y_i|f_{*,i}, \sigma^{2(t)})}{\pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)}))\mathcal{N}_1(y_i|f_{*,i}, \sigma^{2(t)})) + \{1 - \pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)}))\}U}. \end{aligned} \tag{15}$$

We set Z_i to its posterior mode,

$$Z_i = \begin{cases} 1 & \text{if } P(Z_i = 1|\boldsymbol{y}_n, \boldsymbol{f}_*, \Theta^{(t)}) \geq 0.5 \\ 0 & \text{if } P(Z_i = 1|\boldsymbol{y}_n, \boldsymbol{f}_*, \Theta^{(t)}) < 0.5. \end{cases} \tag{16}$$

Based on Z_i, we can classify the local data indices into $I_{n,*} = \{i = 1, \ldots, n : Z_i = 1\}$ and $I_{n,o} = \{i = 1, \ldots, n : Z_i = 0\}$. Let $\boldsymbol{y}_* = (y_i, i \in I_{n,*})$, and let n_* denote the number of the elements in $I_{n,*}$. Given $I_{n,*}$, we can find the posterior mode of $\boldsymbol{f}_*$ in the closed form:

$$\boldsymbol{f}_* = m_*^{(t)}\mathbf{1}_n + \boldsymbol{C}_{n*}^{(t)}(\sigma^{2(t)}\boldsymbol{I} + \boldsymbol{C}_{**}^{(t)})^{-1}(\boldsymbol{y}_* - m_*^{(t)}\mathbf{1}_{n_*}), \tag{17}$$

where $\boldsymbol{C}_{n*}^{(t)} = [c(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta_*^{(t)}) : i = 1, \ldots, n, j \in I_{n,*}]$ is a $n \times n_*$ matrix of the covariance values between $\boldsymbol{y}_n$ and $\boldsymbol{y}_*$, and $\boldsymbol{C}_{**}^{(t)} = [c(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta_*^{(t)}) : i, j \in I_{n,*}]$ is a square matrix of the covariance values evaluated for all pairs of the elements in $\boldsymbol{y}_*$. The block coordinate descent iteration converges because it satisfies a sufficient condition for the convergence (Beck and Tetruashvili, 2013), i.e., the objective function is block-wise convex, and its gradient is block-coordinatewise Lipschitz continuous. Let $\boldsymbol{f}_*^{(t+1)}$ and $\boldsymbol{Z}^{(t+1)}$ represent the latent variable values at the convergence.

• **M-step:** Fixing $\boldsymbol{f}_*^{(t+1)}$ and $\boldsymbol{Z}^{(t+1)}$, the M-step finds Θ that minimizes the negative log complete likelihood,

$$\underset{\Theta}{\text{Minimize}} \quad -\log(p(\boldsymbol{Z}^{(t+1)}|\boldsymbol{\omega})p(\boldsymbol{f}_*^{(t+1)}|m_*, \theta_*)p(\boldsymbol{y}_n|\boldsymbol{f}_*^{(t+1)}, \boldsymbol{Z}^{(t+1)}, \sigma^2)).$$

The solutions for m_* and σ^2 are given in the following closed form expressions:

$$\begin{aligned} m_*^{(t+1)} &= \frac{\mathbf{1}_n^T (\boldsymbol{C}_n^{(t)})^{-1} \boldsymbol{f}_*^{(t+1)}}{\mathbf{1}_n^T (\boldsymbol{C}_n^{(t)})^{-1} \mathbf{1}_n}, \text{ and} \\ \sigma^{2(t+1)} &= \frac{\sum_{i=1}^n Z_i^{(t+1)} (y_i - f_{*,i}^{(t+1)})^2}{\sum_{i=1}^n Z_i^{(t+1)}}, \end{aligned} \tag{18}$$

where $f_{*,i}^{(t+1)}$ and $Z_i^{(t+1)}$ are the ith elements of $\boldsymbol{f}_*^{(t+1)}$ and $\boldsymbol{Z}^{(t+1)}$, respectively, and $\boldsymbol{C}_n^{(t)}$ represents $\boldsymbol{C}_n$ evaluated with covariance parameter $\theta_*^{(t)}$. The optimal values for the other parameters should be achieved numerically using gradient algorithms. By the additive nature of the objective function, optimization of $\boldsymbol{\omega}$ and covariance hyperparameters θ_* can be solved independently. To optimize $\boldsymbol{\omega}$, we solve

$$\begin{aligned} \underset{\boldsymbol{\omega}}{\text{Minimize}} \quad & -\log p(\boldsymbol{Z}^{(t+1)}|\boldsymbol{\omega}) \\ \text{subject to} \quad & g(\boldsymbol{x}_*, \boldsymbol{\omega}) \geq 0. \end{aligned} \tag{19}$$

Putting the parametric form of $g(\boldsymbol{x}_i, \boldsymbol{\omega})$ into the expression, we will write problem (19) as

$$\begin{aligned} \underset{\boldsymbol{\omega}}{\text{Minimize}} \quad & -\sum_{i=1}^n [\log(1 + \exp\{-\boldsymbol{\omega}^T [1, \boldsymbol{\psi}(\boldsymbol{x}_i)]\}) - (1 - Z_i^{(t+1)}) \boldsymbol{\omega}^T [1, \boldsymbol{\psi}(\boldsymbol{x}_i)]] \\ \text{subject to} \quad & \boldsymbol{\omega}^T [1, \boldsymbol{\psi}(\boldsymbol{x}_*)] \geq 0. \end{aligned} \tag{20}$$

The constraint optimization with a linear constraint can be easily solved by the Lagrangian dual method (Boyd et al., 2004, Chapter 5). Optimizing the covariance hyper parameters θ_* is equivalent to the marginal likelihood maximization,

$$\underset{\theta_*}{\text{Minimize}} \quad -\log \mathcal{N}_n(\boldsymbol{f}_*^{(t+1)}|m_*^{(t+1)}\mathbf{1}, \boldsymbol{C}_n). \tag{21}$$

• **Convergence:** Check if $\rho^{(t)} - \rho^{(t+1)}$ is greater than a threshold ϵ, where

$$\rho^{(t+1)} = -\log(p(\boldsymbol{Z}^{(t+1)}|\boldsymbol{\omega}^{(t+1)})p(\boldsymbol{f}_*^{(t+1)}|m_*^{(t+1)}, \theta_*^{(t+1)})p(\boldsymbol{y}_n|\boldsymbol{f}_*^{(t+1)}, \boldsymbol{Z}^{(t+1)}, \sigma^{2(t+1)})). \tag{22}$$

After the convergence, we would have the parameter estimates of Θ. Given the parameter estimates, the posterior predictive distribution of $f_*(\boldsymbol{x})$ at a test location $\boldsymbol{x}_*$ is described as the Gaussian distribution with

$$\begin{array}{ll} \text{mean:} & \mu_1(\boldsymbol{x}_*) = m_* + \boldsymbol{c}_*^T(\sigma^2\boldsymbol{I}_{n_*} + \boldsymbol{C}_{**})^{-1}(\boldsymbol{y}_* - m_*\boldsymbol{1}_{n_*}), \text{ and} \\ \text{variance:} & s_1(\boldsymbol{x}_*) = c(\boldsymbol{x}_*, \boldsymbol{x}_*; \theta_*) - \boldsymbol{c}_*^T(\sigma^2\boldsymbol{I}_{n_*} + \boldsymbol{C}_{**})^{-1}\boldsymbol{c}_*, \end{array} \tag{23}$$

where $\boldsymbol{c}_* = [c(\boldsymbol{x}_i, \boldsymbol{x}_*; \theta_*) : i \in I_{n,*}]$ is a column vector of the covariance values between the local data $\boldsymbol{y}_*$ and the test location. If the estimate $\boldsymbol{Z}^{(t)}$ is correct, the mean estimate $\mu_1(\boldsymbol{x}_*)$ would satisfy the condition in Theorem 1, so it should be unbiased. The overall bias of the estimate is largely influenced by the accuracy of the estimate $\boldsymbol{Z}^{(t)}$. We refer to this GP estimate as *Jump Gaussian Process Regression by the classification EM* or shortly JGP-CEM.

• **Initialization:** For JGP-CEM, we should run the coordinate-wise optimization algorithm to estimate the model parameters. It is crucial to feed good initial estimates of the parameters as an initial solution to the optimization algorithm. This section describes our approach to get an initial solution.

For an initial estimate of $\boldsymbol{\omega}$, we propose to use a local kernel estimate. We approximate the local response surface of $f(\boldsymbol{x})$ by a partitioning function $\boldsymbol{\omega}^T[1, \boldsymbol{\psi}(\boldsymbol{x})]$ around a test location $\boldsymbol{x}_*$ that minimizes the following local fit error,

$$\boldsymbol{\omega}^{(1)} = \underset{\boldsymbol{\omega}\in\mathbb{R}^{L+1}}{\operatorname{argmin}} \sum_{i=1}^{n} K_h(\boldsymbol{x}_i, \boldsymbol{x}_*) \left\{ y_i - \boldsymbol{\omega}^T[1, \boldsymbol{\psi}(\boldsymbol{x}_i)] \right\}^2, \tag{24}$$

where $K_h(\boldsymbol{x}_i, \boldsymbol{x}_*) = \phi\left(\frac{||\boldsymbol{x}_i - \boldsymbol{x}_*||}{h}\right)$ is the Gaussian kernel weight with the choice of bandwidth h; here we choose $h = \max_{i=1,\ldots,n} ||\boldsymbol{x}_i - \boldsymbol{x}_*||$. The fitted partitioning function $(\boldsymbol{\omega}^{(1)})^T\boldsymbol{\psi}(\boldsymbol{x})$ now serves as a local surrogate to $f(\boldsymbol{x})$ around a test location $\boldsymbol{x}_*$, and thresholding the local surrogate value $(\boldsymbol{\omega}^{(1)})^T[1, \boldsymbol{\psi}(\boldsymbol{x}_i)]$ by a threshold ω_0 would give a division of the local data into two groups distinguished by y_i values. Hence, the fitted $(\boldsymbol{\omega}^{(1)})^T[1, \boldsymbol{\psi}(\boldsymbol{x}_i)]$ would serve a good partitioning function of the local data. We propose to use the local kernel estimate $\boldsymbol{\omega}^{(1)}$ as an initial solution of $\boldsymbol{\omega}$ after adjusting it with ω_0. The adjusted estimates of a partitioning function are given to

$$\hat{g}(\boldsymbol{x}) = (\boldsymbol{\omega}^{(1)})^T[1, \boldsymbol{\psi}(\boldsymbol{x})] - \omega_0 = 0, \tag{25}$$

where ω_0 is chosen to minimize the within-region variance, calculated as the sample variance of $\{y_i : \hat{g}(\boldsymbol{x}_i) \geq 0\}$ plus the sample variance of $\{y_i : \hat{g}(\boldsymbol{x}_i) < 0\}$. The corresponding estimate of $\boldsymbol{\omega}$ is $(\boldsymbol{\omega}^{(1)})^T - [\omega_0, \boldsymbol{0}_{L-1}^T]^T$, where $\boldsymbol{0}_{L-1}$ is the $L-1$ dimensional vector of zeros.

The initial estimate of Z_i can be obtained by applying the partitioning function estimate,

$$Z_i^{(1)} = \begin{cases} 1 & \text{if } (\boldsymbol{\omega}^{(1)})^T[1, \boldsymbol{\psi}(\boldsymbol{x}_i)] \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

With the initial estimates $\boldsymbol{\omega}^{(1)}$ and $\boldsymbol{Z}^{(1)}$, we can solve problem (21) to estimate θ_* and then use (17) to estimate the remainder. The step-by-step of JGP-CEM with the initial estimation is summarized in Algorithm 1.

Algorithm 1: Jump Gaussian Process Regression by the Classification EM (JGP-CEM)

Input. local training data $\mathcal{D}_n$, a test location $\boldsymbol{x}_* \in \mathcal{X}$, basis functions $\boldsymbol{\psi}(\boldsymbol{x})$, a small tolerance $\epsilon > 0$

Output. predictive mean $\mu_1(\boldsymbol{x}_*)$, predictive variance $\sigma_1(\boldsymbol{x}_*)$

Initialization:

Set $\boldsymbol{Z}^{(1)}$, $\boldsymbol{f}_*^{(1)}$ and $\Theta^{(1)}$, using (24) and (26).

Set $\rho^{(0)} = \infty$ and calculate $\rho^{(1)}$ using (22).

Set $t = 1$.

[Perform the classification EM algorithm.]

While $\rho^{(t-1)} - \rho^{(t)} > \epsilon$

E-step:

[Iterate the two steps below until convergence or can take only one iteration.]

Calculate $\boldsymbol{Z}^{(t+1)}$, using (16).

Calculate $\boldsymbol{f}_*^{(t+1)}$, using (17).

M-step:

Calculate $m_*^{(t+1)}$ and $\sigma^{2(t+1)}$, using (18).

Calculate $\boldsymbol{\omega}^{(t+1)}$ using (20).

Calculate $\theta_*^{(t+1)}$, using (21).

Calculate $\rho^{(t+1)}$ and increase t by one.

End While

2.4 JGP-VEM: Variational EM Inference for Jump GP

This section describes the variational EM (Bishop, 2006) to estimate $\boldsymbol{Z}$, $\boldsymbol{f}_*$ and Θ.

• **E-step:** Given the parameter estimates $\Theta^{(t)} = \{m_*^{(t)}, \theta_*^{(t)}, \boldsymbol{\omega}^{(t)}, \sigma^{2(t)}\}$ from the past iteration, take a factorization approximation $q(\boldsymbol{Z})q(\boldsymbol{f}_*)$ to the joint posterior distribution of $\boldsymbol{Z}$ and $\boldsymbol{f}_*$. The factorization approximation $q(\boldsymbol{Z})q(\boldsymbol{f}_*)$ are obtained so as to minimize the Kullback-Leibler divergence between the factorized approximation and the true joint posterior distribution. We can easily show that $q(\boldsymbol{Z}) = \prod_{i=1}^{N} q(Z_i)$ with $q(Z_i = 1)$ given as

$$\gamma_i = \frac{\pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)})) \exp\left\{-\frac{\mathbb{V}\mathrm{ar}[f_{*,i}]}{2\sigma^{2(t)}}\right\} \mathcal{N}_1(y_i | \mathbb{E}[f_{*,i}], \sigma^{2(t)})}{\pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)})) \exp\left\{-\frac{\mathbb{V}\mathrm{ar}[f_{*,i}]}{2\sigma^{2(t)}}\right\} \mathcal{N}_1(y_i | \mathbb{E}[f_{*,i}], \sigma^{2(t)}) + \{1 - \pi(g(\boldsymbol{x}, \boldsymbol{\omega}^{(t)}))\} U}, \tag{27}$$

and $q(\boldsymbol{f}_*)$ is the multivariate Gaussian density function with mean $\tilde{\boldsymbol{\mu}}$ and covariance $\tilde{\boldsymbol{\Sigma}}$,

$$\begin{aligned} \tilde{\boldsymbol{\mu}} &= \tilde{\boldsymbol{\Sigma}} \left\{ (\sigma^{2(t)})^{-1} \boldsymbol{D}_Z^{-1} \boldsymbol{y} + m_*^{(t)} (\boldsymbol{C}_n^{(t)})^{-1} \mathbf{1} \right\} \\ \tilde{\boldsymbol{\Sigma}} &= \left\{ (\sigma^{2(t)})^{-1} \boldsymbol{D}_Z^{-1} + (\boldsymbol{C}_n^{(t)})^{-1} \right\}^{-1}, \end{aligned} \tag{28}$$

where $\boldsymbol{D}_Z$ is a $n \times n$ diagonal matrix with $1/\mathbb{E}[Z_i]$ as its ith diagonal element. Here, achieving $q(\boldsymbol{Z})$ requires $\mathbb{E}[f_{*,i}]$ and $\mathbb{V}\mathrm{ar}[f_{*,i}]$, while achieving $q(\boldsymbol{f}_*)$ requires $\mathbb{E}[Z_i]$. Numerically, $\mathbb{E}[f_{*,i}]$ and $\mathbb{V}\mathrm{ar}[f_{*,i}]$ can be estimated from $q(\boldsymbol{f}_*)$, and $\mathbb{E}[Z_i]$ can be estimated from $q(\boldsymbol{Z})$. Each of

$q(\boldsymbol{Z})$ and $q(\boldsymbol{f}_*)$ can be updated iteratively while fixing another until the iteration reaches to convergence. The convergence is guaranteed because the Kullback-Leibler divergence is convex with respect to each of the factor distributions (Bishop, 2006). Let $\gamma_i^{(t+1)}$, $\tilde{\boldsymbol{\mu}}^{(t+1)}$ and $\tilde{\boldsymbol{\Sigma}}^{(t+1)}$ represent the values of γ_i, $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\Sigma}}$ at the convergence. Let $\boldsymbol{D}_\gamma^{(t+1)}$ represent a $n \times n$ diagonal matrix with $1/\gamma_i^{(t+1)}$ as its ith diagonal element. In the E-step, we take an expectation of the complete log likelihood $\log\{p(\boldsymbol{Z}|\boldsymbol{\omega})p(\boldsymbol{f}_*|m_*, \theta_*)p(\boldsymbol{y}_n|\boldsymbol{f}_*, \boldsymbol{Z}, \sigma^2)\}$ with respect to $q(\boldsymbol{Z})q(\boldsymbol{f}_*)$, which forms

$$
\begin{aligned}
L(\Theta) = \sum_{i=1}^{n} \Big[& \gamma_i^{(t+1)} \log \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega})) + (1 - \gamma_i^{(t+1)}) \left\{ \log(1 - \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega}))) + \log U \right\} \Big] \\
& - \frac{1}{2} \log |\sigma^2 \boldsymbol{D}_\gamma^{(t+1)}| - \frac{1}{2} \log |\boldsymbol{C}_n| \\
& - \frac{1}{2} tr \left[(\sigma^{-2} (\boldsymbol{D}_\gamma^{(t+1)})^{-1} + \boldsymbol{C}_n^{-1}) (\tilde{\boldsymbol{\mu}}^{(t+1)} (\tilde{\boldsymbol{\mu}}^{(t+1)})^T + \tilde{\boldsymbol{\Sigma}}^{(t+1)}) \right] \\
& + (\tilde{\boldsymbol{\mu}}^{(t+1)})^T (\sigma^{-2} (\boldsymbol{D}_\gamma^{(t+1)})^{-1} \boldsymbol{y} + m_* \boldsymbol{C}_n^{-1} \mathbf{1}) - \frac{1}{2\sigma^2} \boldsymbol{y}^T (\boldsymbol{D}_\gamma^{(t+1)})^{-1} \boldsymbol{y} - \frac{m_*^2}{2} \mathbf{1}^T \boldsymbol{C}_n^{-1} \mathbf{1}.
\end{aligned}
$$

• **M-step:** We maximizes $L(\Theta)$ with respect to Θ, and the optimal solutions of σ^2 and $\boldsymbol{\omega}$ can be achieved independently of the other parameters. The optimal solution of σ^2 is achieved in the closed forms,

$$
\sigma^{2(t+1)} = \frac{tr \left[(\boldsymbol{D}_\gamma^{(t+1)})^{-1} \left\{ \tilde{\boldsymbol{\Sigma}}^{(t+1)} + (\boldsymbol{y} - \tilde{\boldsymbol{\mu}}^{(t+1)})(\boldsymbol{y} - \tilde{\boldsymbol{\mu}}^{(t+1)})^T \right\} \right]}{n}. \tag{29}
$$

The parameter $\boldsymbol{\omega}$ is only dependent on the first two terms of $L(\Theta)$, so the optimal solution for $\boldsymbol{\omega}$ can be achieved, using the following optimization

$$
\begin{aligned}
\boldsymbol{\omega}^{(t+1)} = \underset{\boldsymbol{\omega}}{\text{argmin}} \quad & - \sum_{i=1}^{n} [\gamma_i^{(t+1)} \log \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega})) + (1 - \gamma_i^{(t+1)}) \log(1 - \pi(g(\boldsymbol{x}_i, \boldsymbol{\omega})))] \\
\text{subject to} \quad & \boldsymbol{\omega}^T [1, \boldsymbol{\psi}(\boldsymbol{x}_*)] \geq 0.
\end{aligned} \tag{30}
$$

Optimizing the covariance hyper parameters θ_* and m_* is equivalent to the marginal likelihood maximization,

$$
\begin{aligned}
& (m_*^{(t+1)}, \theta_*^{(t+1)}) \\
& = \underset{m_*, \theta_*}{\text{argmin}} \frac{1}{2} \log |\boldsymbol{C}_n| + \frac{1}{2} (\tilde{\boldsymbol{\mu}}^{(t+1)} - m_* \mathbf{1})^T \boldsymbol{C}_n^{-1} (\tilde{\boldsymbol{\mu}}^{(t+1)} - m_* \mathbf{1}) + \frac{1}{2} tr \left[\boldsymbol{C}_n^{-1} \tilde{\boldsymbol{\Sigma}}^{(t+1)} \right].
\end{aligned} \tag{31}
$$

After the convergence of the variational EM algorithm, we would have the parameter estimates of Θ and corresponding posterior distribution $q(\boldsymbol{f}_*)$. Given them, the posterior predictive distribution of $f_*(\boldsymbol{x})$ at a test location $\boldsymbol{x}_*$ can be achieved as

$$
q(f_*(\boldsymbol{x})) = \int p(f_*(\boldsymbol{x})|\boldsymbol{f}_*) q(\boldsymbol{f}_*) d\boldsymbol{f}_*. \tag{32}
$$

We can easily see that the predictive distribution is a normal distribution with

$$
\begin{aligned}
& \text{mean:} \quad \mu_2(\boldsymbol{x}_*) = m_* + \boldsymbol{c}_*^T \boldsymbol{C}_n^{-1} (\tilde{\boldsymbol{\mu}} - m_* \mathbf{1}_{n_*}), \text{ and} \\
& \text{variance:} \quad s_2(\boldsymbol{x}_*) = c(\boldsymbol{x}_*, \boldsymbol{x}_*; \theta_*) - \boldsymbol{c}_*^T (\boldsymbol{C}_n^{-1} - \boldsymbol{C}_n^{-1} \tilde{\boldsymbol{\Sigma}} \boldsymbol{C}_n^{-1}) \boldsymbol{c}_*.
\end{aligned} \tag{33}
$$

Algorithm 2: Jump Gaussian Process Regression with Variation EM (JGP-VEM)

Input. local training data $\mathcal{D}_n$, a test location $\boldsymbol{x}_* \in \mathcal{X}$, basis functions $\boldsymbol{\psi}(\boldsymbol{x})$, a small tolerance $\epsilon > 0$

Output. predictive mean $\mu_2(\boldsymbol{x}_*)$, predictive variance $\sigma_2(\boldsymbol{x}_*)$

Initialization:

Initialize $\Theta^{(1)}$ as in Algorithm 1.

Fixing $\Theta^{(1)}$, initialize $\boldsymbol{\gamma}_i^{(1)}$, $\tilde{\boldsymbol{\mu}}^{(1)}$ and $\tilde{\boldsymbol{\Sigma}}^{(1)}$, using (27) and (28).

Set $\rho^{(0)} = \infty$ and calculate $\rho^{(1)} = -L(\Theta^{(1)})$

Set $t = 1$.

[Perform the variational EM algorithm.]

While $\rho^{(t-1)} - \rho^{(t)} > \epsilon$

E-step:

[You can iterate the two steps below until convergence.]

Calculate $\boldsymbol{\gamma}_i^{(t+1)}$ and $\boldsymbol{D}_\gamma^{(t+1)}$, using (27).

Calculate $\tilde{\boldsymbol{\mu}}^{(t+1)}$ and $\tilde{\boldsymbol{\Sigma}}^{(t+1)}$, using (28).

M-step:

Calculate $\sigma^{2(t+1)}$, using (29).

Calculate $\boldsymbol{\omega}^{(t+1)}$ using (30).

Calculate $\theta_*^{(t+1)}$, using (31).

Calculate $\rho^{(t+1)} = -L(\Theta^{(t+1)})$ and increase t by one.

End While

We refer to this GP estimate as *Jump Gaussian Process Regression with Variation EM* or shortly JGP-VEM. The initial solution of the iterative algorithm can be achieved as in the JGP-CEM. The step-by-step of JGP-VEM is summarized in Algorithm 2.

2.5 Local Design Selection with a Linear Partitioning Function

In this section, we use toy examples to illustrate how JGP works with a linear partitioning function, i.e., $g(\boldsymbol{x}, \boldsymbol{\omega}) = \boldsymbol{\omega}^T[1, \boldsymbol{x}]$ and discuss potential issues with it for motivating a higher-order model. With the linear form, the local data is split into two parts by a linear boundary $g(\boldsymbol{x}, \boldsymbol{\omega}) = 0$. We will use several synthetic examples to show that the linear split is simple and good enough. In particular, when a regional boundary nearing a test location is smooth with continuous boundary derivatives, the first-order Taylor approximation to the boundary works well, as illustrated in Figure 1.

We use five toy examples. For an effective visualization, we use a two-dimensional rectangular domain $[-0.5, 0.5]^2$, which is partitioned into two or more regions, and the response function for each region is randomly drawn from an independent GP with different constant mean functions and a square exponential covariance function parameterized by $\theta_k = (\theta_{k1}, \theta_{k2})$,

$$m_k(\boldsymbol{x}) = 0 \text{ or } 27$$
$$c(\boldsymbol{x}, \boldsymbol{x}'; \theta_k) = \theta_{k1} \exp\left\{-\frac{1}{\theta_{k2}}(\boldsymbol{x} - \boldsymbol{x}')^T(\boldsymbol{x} - \boldsymbol{x}')\right\}.$$

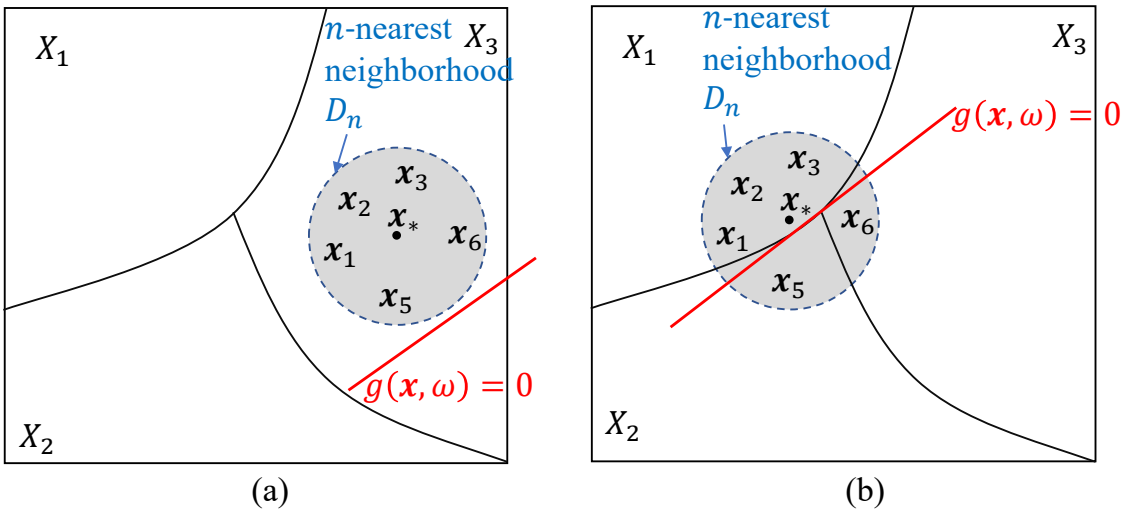

Figure 1: A linear partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega}) = 0$ would work when (a) a test location $\boldsymbol{x}_*$ is deeply interior of one homogeneous region. In this case, $g(\boldsymbol{x}, \boldsymbol{\omega})$ is formed such that all local data are selected and used for a local estimate, and (b) the boundary dividing the region of a test location $\boldsymbol{x}_*$ and other neighboring regions are locally smooth. In that case, the smooth boundary can be well approximated linearly at a local level.

We used $\theta_{k1} = 9$ and $\theta_{k2} = 200$, and $\mu_k \sim Uniform(\{0, 27\})$. Both of μ_k and θ_k are assumed unknown during the training stage, and they are estimated with training data. For the first three examples, the domain is divided into two regions by a linear boundary in the first synthetic example (Figure 2-(a)), by a quadratic boundary in the second example (Figure 2-(b)), and by a triangular boundary with a sharp corner in the third example (Figure 2-(c)). For the fourth and fifth examples, the domain is divided into more than two regions, as illustrated in Figure 2-(d) and 2-(e). The fifth example is most challenging with many sharp and narrow regions.

We randomly generate training datasets of different sizes ($N = 600, 1200, 2500$). Each training input $\boldsymbol{x}_i$ is randomly drawn from the uniform distribution over the input domain. The noisy response variable is from the ground-truth response plus Gaussian noise with zero mean and variance σ^2. We varied σ^2 over $\{1, 4, 9\}$, which correspond to the signal-to-noise ratio, 14 decibel (dB), 8 dB and 4.4 dB respectively. The 4.4 dB is considered a very noisy scenario. The test set consists of 2,601 ground truth responses over $[51 \times 51]$ grid locations of the input domain. We used the training set to fit the proposed local GP model and the conventional local GP model. We choose local data using the n-nearest neighborhood for both of the models. We tried different n values, $n \in \{25, 35, 50\}$.

We evaluate the fitted models at various test locations. Figures 3 and 4 show how the partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega})$ and latent variables $\boldsymbol{Z}$ are estimated for the first two toy datasets. When test locations are interior of a region and all of the n-nearest neighbors belong to one homogeneous region, the fitted linear partitioning functions locate far outside the n-nearest neighborhood, so all local data are used to fit JGP, which is desirable; see the last columns of Figure 3 and Figure 4. When test locations are near regional boundaries,

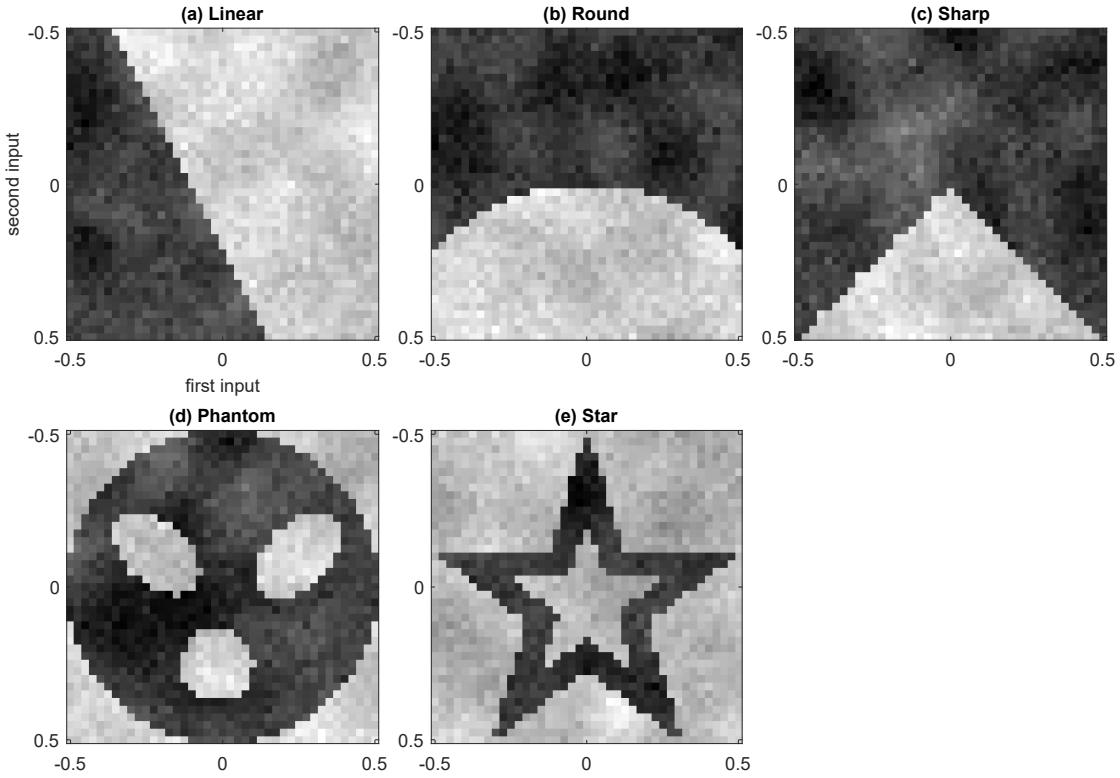

Figure 2: Noisy regression functions of five toy examples on two-dimensional input domains. The x and y axes represent the first and second inputs of the domains, and the grayscale intensities represent the regression function values over the domains.

the linear partitioning functions are approximately tangential to the regional boundaries. Therefore, $g(\boldsymbol{x}, \boldsymbol{\omega})$ accurately splits local data into homogeneous groups.

Figures 5 and 6 show the results with the third and fourth toy examples. The third toy example contains a non-smooth regional boundary at a sharp tip. As we expected, the JGP with a linear partitioning function does not work perfectly when a test location is near the sharp tip (that corresponds to the third column of Figure 5). Around the tip, the boundary is non-differentiable, so the first-order Taylor approximation of the boundary may not work. In such cases, a linear hyperplane is less effective in splitting local data into homogeneous regions. The fourth toy example is a more extreme case, where a domain consists of eight regions. Some test locations are on narrow canals of a dark region, sandwiched by two other white regions, as illustrated in the first column of Figure 6. The local data around such test locations may contain data points from more than two regions. A linear partitioning function cannot effectively separate the local data. As illustrated in the first column of Figure 6, the local data selected by JGP still contains many data points from the two white region regions. Therefore, the JGP with a linear partitioning function would not work well for predicting f at these test locations.

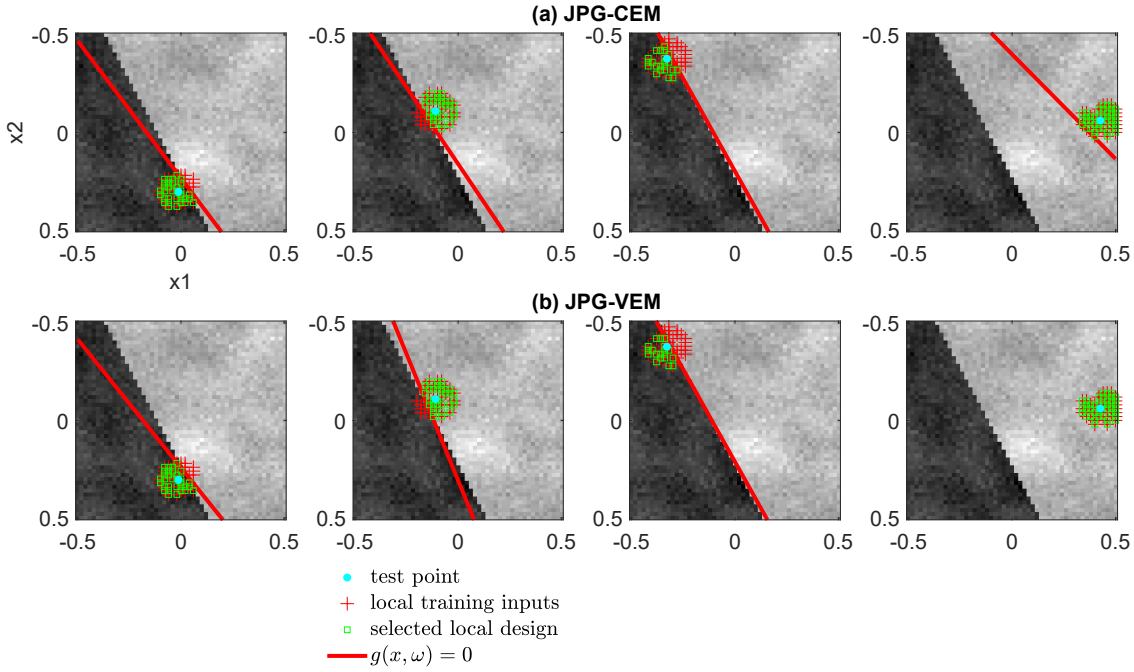

Figure 3: JGP with a linear partitioning function for the 1st toy example ($\sigma^2 = 2^2$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes

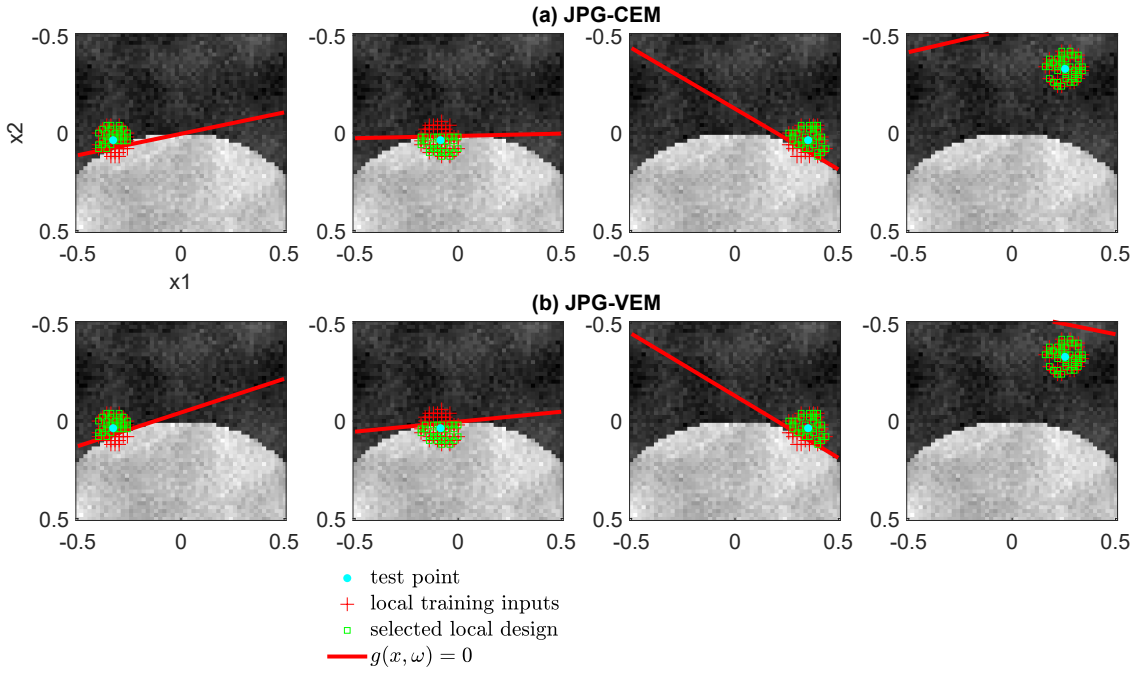

Figure 4: JGP with a linear partitioning function for the 2nd toy example ($\sigma^2 = 2^2$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes for those with $\hat{Z}_i = 1$ and red crosses for those with $\hat{Z}_i = 0$.

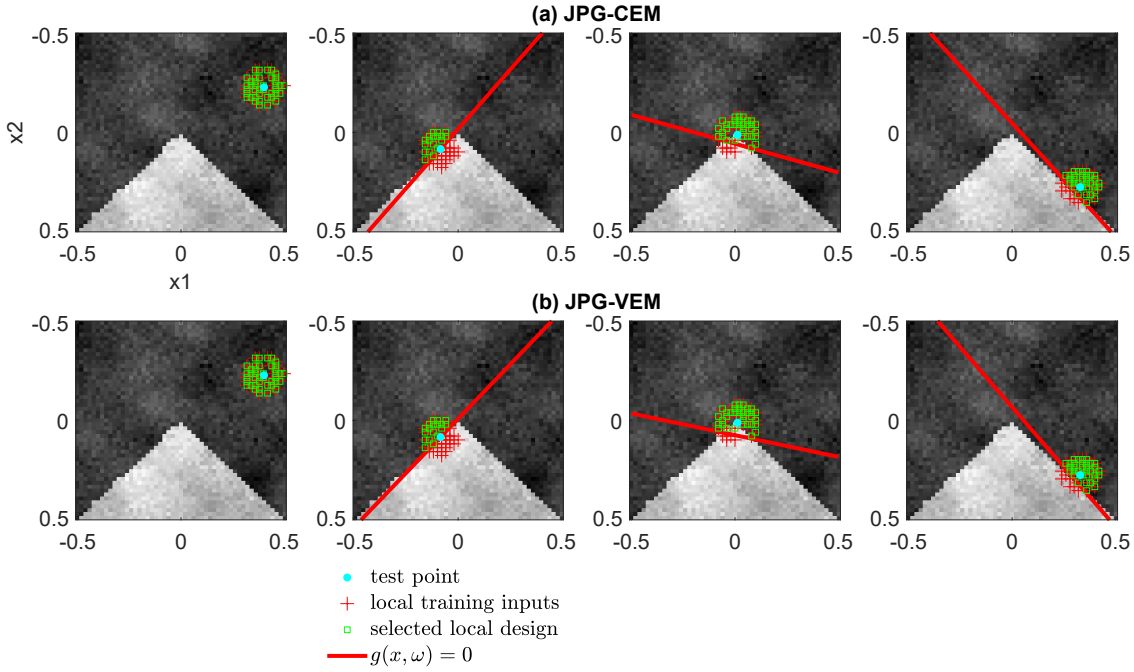

Figure 5: JGP with a linear partitioning function for the 3rd toy example ($\sigma^2 = 2^2$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes

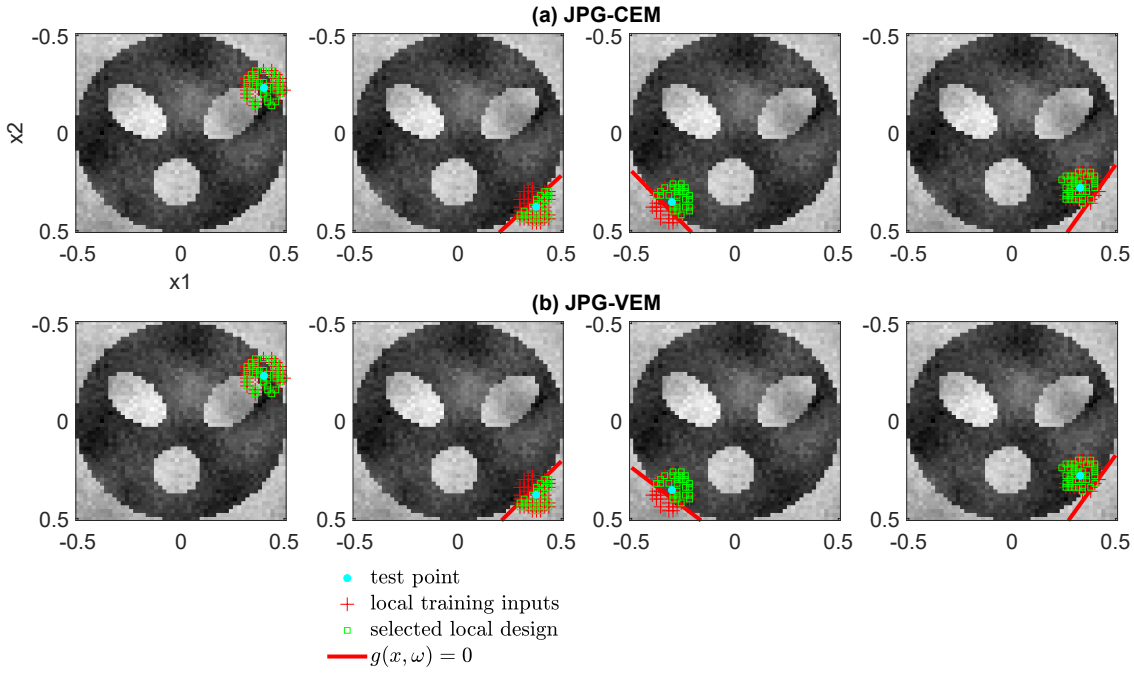

Figure 6: JGP with a linear partitioning function for the 4th toy example ($\sigma^2 = 2^2$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes for those with $\hat{Z}_i = 1$ and red crosses for those with $\hat{Z}_i = 0$.

Figure 7 presents the JGP predictions at test locations over a dense grid of $[-0.5, 0.5] \times [-0.5, 0.5]$ for the first four toy examples. The predictions for the fourth toy example have some discrepancies from the ground truth. In particular, the discrepancy was more severe in test locations in a dark region sandwiched by inner and outer white regions. That is because a simple linear partitioning function cannot split local data around these locations into homogeneous regions effectively, as we illustrated before in Figure 6. To resolve this issue, one needs to use a partitioning function composed of higher-order polynomial terms. We will explore this possibility in the next section. We also have comparisons of two JGP variants: JGP-CEM (top panels) versus JGP-VEM (bottom panels). The predictions from JGP-CEM are more accurate. Later, in Section 3.1, we will show that it is attributed to difference in the $Z_i$estimates from the two variants

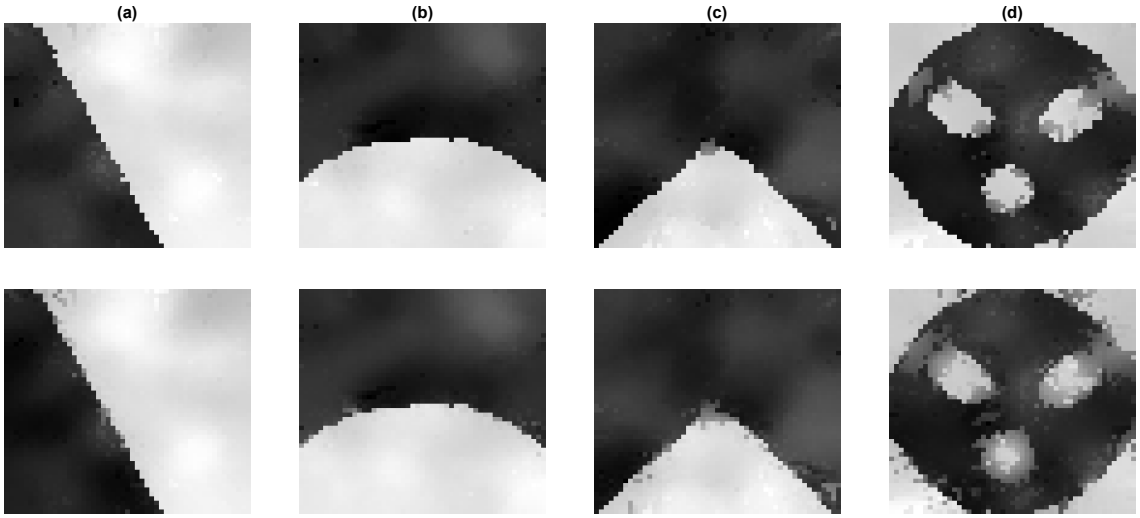

Figure 7: JGP estimates of f with a linear partitioning function. The panels show the JGP estimates at 2,601 grid locations over $[-0.5, 0.5] \times [-0.5, 0.5]$. The first row shows the results from JGP-CEM, in contrast with the second row from JGP-VEM. Here we used the setting $n = 50$ and $\sigma^2 = 4$.

2.6 Local Data Selection with a Nonlinear Partitioning Function

As we have shown in Section 2.5, the JGP with a linear partitioning function is not good enough for some complicated scenarios. Here we investigate the JGP with a nonlinear form of a partitioning function,

$$g(\boldsymbol{x}, \boldsymbol{\omega}) = \boldsymbol{\omega}^T [1, \boldsymbol{\psi}(\boldsymbol{x})].$$

Given a small number of local data, the number of the basis functions should not be very large. A practical choice for the nonlinear basis $\boldsymbol{\psi}(\boldsymbol{x})$ could be quadratic polynomial functions or cubic polynomial functions. Here we present a numerical study with quadratic polynomial basis functions, comprising

$$1, \{x_i, i = 1, \ldots, p\}, \{x_i x_j, i = 1, \ldots, p, j = 1, \ldots, p\},$$

where x_i is the ith dimension of a p-dimensional input vector $\boldsymbol{x}$. We found that the quadratic basis is flexible enough for most tested cases.

Figure 8 shows how quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ are $\boldsymbol{Z}$ are estimated for various test locations in the fourth toy example. JGP with a linear $g(\boldsymbol{x}, \boldsymbol{\omega})$ was not very successful for the toy example. The quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ works well for all of the test cases of the fourth toy example. As shown in Figure 8, the fitted quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ form curvy boundaries that effectively split local data into homogeneous regions. When linear splits are good enough, the fitted quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ becomes closer to linear boundaries; see the second panel of Figure 8. This means that the partition functions from JGP do not overfit.

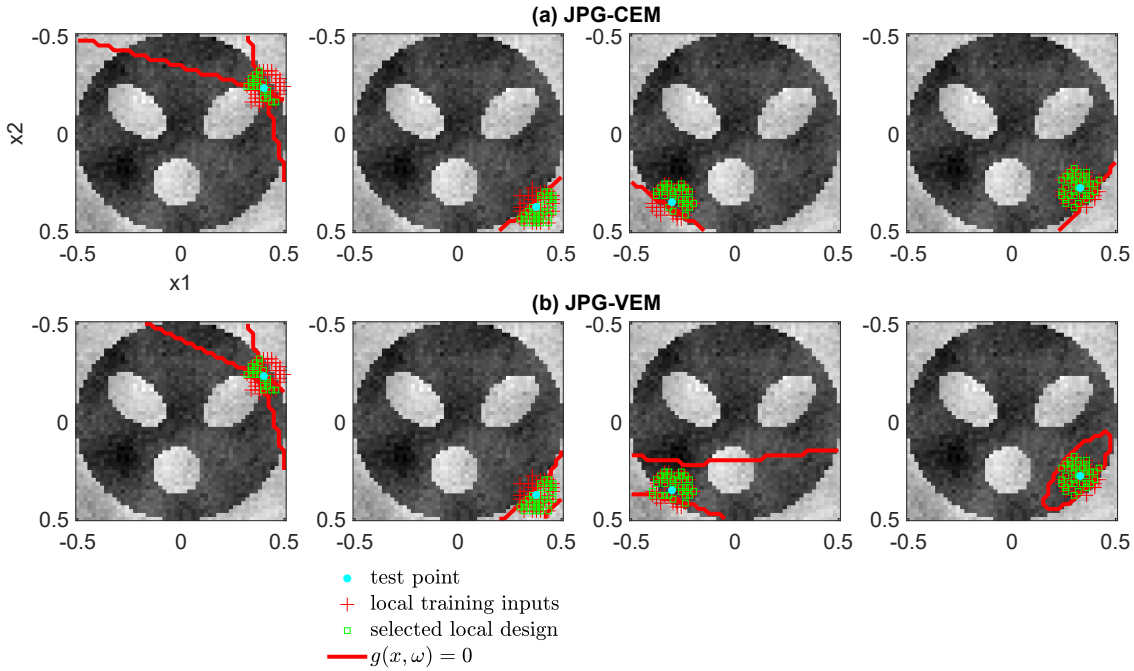

Figure 8: JGP with a quadratic partitioning function for the 4th toy example ($\sigma^2 = 2^2$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes for those with $\hat{Z}_i = 1$ and red crosses for those with $\hat{Z}_i = 0$.

We also tested the quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ for a more complicated case with the fifth toy example, where a domain is still $[-0.5, 0.5]^2$ and is partitioned to several regions as shown in Figure 9. There are many thin regions with sharp corners. We tested JGP with a quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ for various test locations of the toy example. As shown in the first column of Figure 9, a quadratic partitioning function works well for thin white regions. The second and fourth columns of the figure show that the quadratic model also works well on sharp corners. We may have more complicated scenarios, but we believe what we emulated may be more complicated than what we see from real datasets. For most real cases, a quadratic partitioning function would be good enough.

Figure 10 shows the JGP estimates (with quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$) of f at 2,601 grid locations over $[-0.5, 0.5] \times [-0.5, 0.5]$ for the fourth and fifth toy examples. The estimates are comparable to the ground truth for the two examples except for a few test locations around sharp

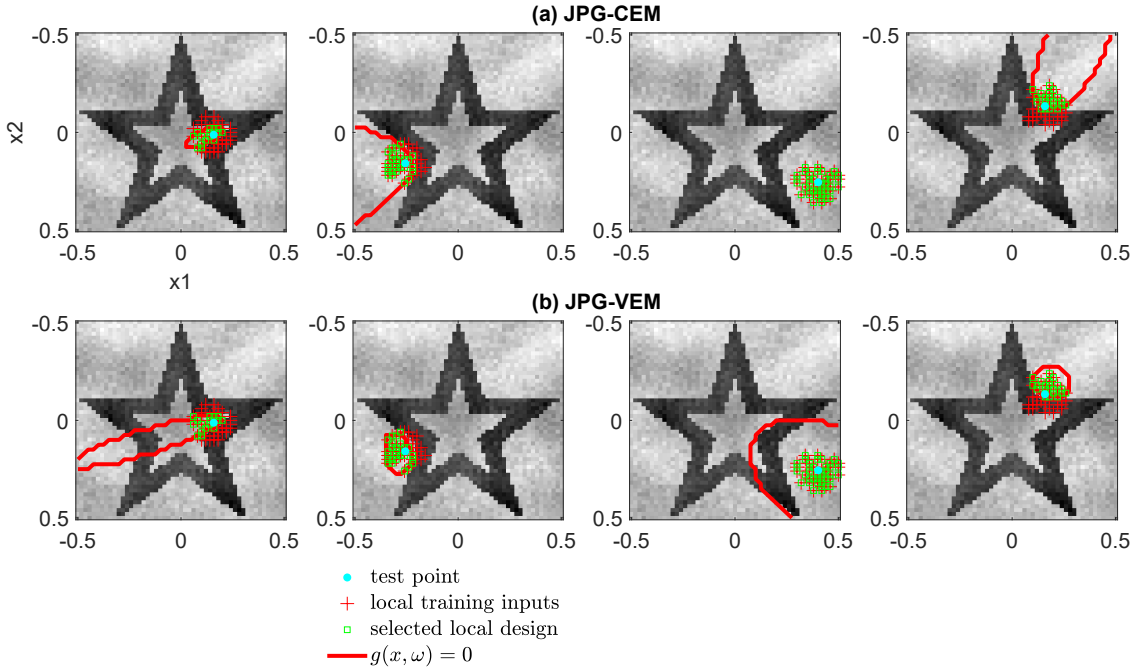

Figure 9: JGP with a quadratic partitioning function for the 5th toy example ($\sigma^2 = 4$, $n = 50$). The input locations of 50 local data around each test location are described as green boxes for those with $\hat{Z} = 1$ and red crosses for those with $\hat{Z} = 0$.

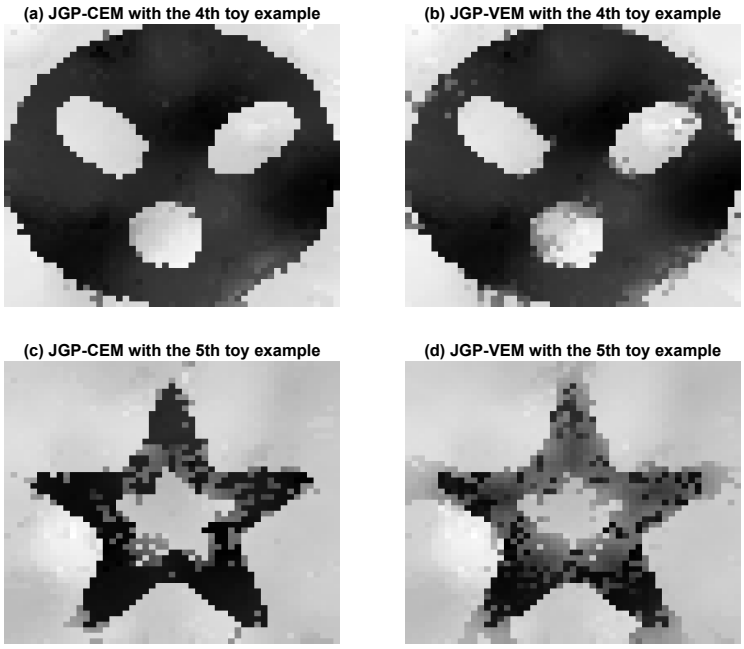

Figure 10: JGP estimates with a quadratic partitioning function. The four panels show the JGP estimates of f at 2,601 grid locations over $[-0.5, 0.5] \times [-0.5, 0.5]$. Here we used the setting $n = 50$ and $\sigma^2 = 4$.

corners in the star-shape frame of the fifth toy example. We also have comparisons of two JGP variants: JGP-CEM (top panels) versus JGP-VEM (bottom panels). The predictions from JGP-CEM are more accurate. Later, in Section 3.1, we will show that it is attributed to difference in the $Z_i$estimates from the two variants. Compared to the conventional local GP predictors, the prediction results are much better. We will provide some quantitative comparison later in Section 3 to support this argument.

Remark 2 ***Consideration of Higher-order Partitioning Function.*** *One may perhaps consider higher-order models for a partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega})$ to handle more complex scenarios, e.g., cubic polynomial functions and nonlinear functions. Extending the proposed approach for higher-order partitioning functions is straightforward as changing basis functions $\boldsymbol{\psi}(\boldsymbol{x})$. We still recommend the quadratic model because it performed satisfactorily for all our benchmark examples in Sections 3 and 5. In the simulated benchmark datasets of Section 3, individual regional boundaries are various, from linear (the first toy example), quadratic (second and fourth toy examples), and piecewise linear (third and fifth toy examples). In addition, a composition of regional boundaries makes more complicated patterns locally, e.g., around the eyes of phantoms in the fourth toy example or around sharp corners of a star shell in the fifth toy example. The real datasets used in Section 5 have simpler regional boundaries than the simulated scenarios. The Taylor series approximation theory supports the sufficiency of a quadratic model. A high-order polynomial boundary (even non-polynomial smooth boundaries) can be locally approximated by a quadratic polynomial boundary reasonably well up to some degrees.*

3. Performance Analysis of JGP with 2D Toy Examples

This section uses various simulated examples to illustrate how the proposed JGP works compared to the conventional local GP model. We use the five toy examples in Figure 2. To simulate various scenarios, we varied the training data size N over $\{600, 1200, 2500\}$. Each training input $\boldsymbol{x}_i$ is randomly drawn from the uniform distribution over the input domain. We also varied the noise σ^2 over $\{1, 4, 9\}$, which correspond to the signal-to-noise ratio, 14 decibel (dB), 8 dB and 4.4 dB respectively. For a model fit, we use the n-nearest neighborhood as local data, where the value of n varies over $\{25, 35, 50\}$. We have 27 simulation configurations with different values of the three simulation inputs. For each configuration, we performed ten replicated experiments. The fitted models are evaluated at test sites of 2,601 grid locations over $[-0.5, 0.5] \times [-0.5, 0.5]$.

3.1 Accuracy of Z_i Estimates

Per Theorem 1, the bias of the JGP estimate is highly dependent on the accuracy of Z_i estimates. In this section, we evaluate the estimation accuracy for the simulated configurations. We match the estimates with the Z_i values used for generating training data. We measure the percent of correct matches.

Figure 11 shows the estimation accuracy for five toy examples. For the first three toy examples with simple regional boundaries, the JGP-CEM and JGP-VEM work comparably. The JGP-VEM is only marginally better. For the other two with more complex regional boundaries, the JGP-CEM has shown better accuracy. The JGP-CEM with a quadratic

partitioning function provides more than 95% accuracy for most test scenarios. There is a relatively small variation in the percent due to noise levels and data sizes. From now on, we investigate only the JGP-CEM model numerically as a suggested JGP model. We use shortnamings JGP-L and JGP-Q to represent JGP-CEM with a linear partitioning function and JGP-CEM with a quadratic partitioning function, respectively.

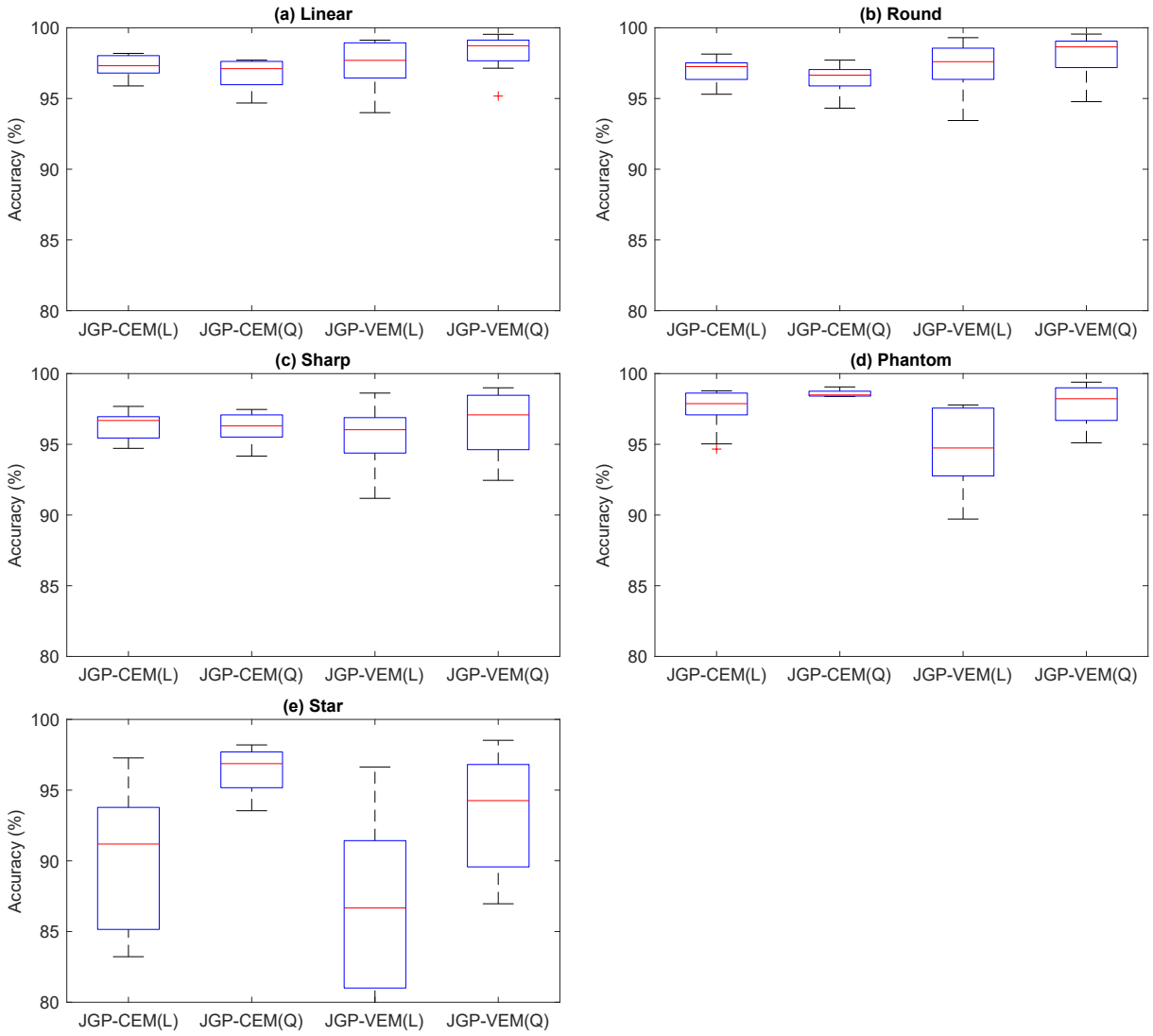

Figure 11: Accuracy of Z_i estimates for five toy examples with various simulation configurations. The '(L)' and '(Q)' in the x labels represent linear and quadratic partitioning functions, respectively.

Figure 12 shows the trend of the estimation accuracy over training data sizes. We can see the upward trend of the accuracy as N increases for all test scenarios. This desirable numerical result shows numerically statistical consistency of the proposed estimator.

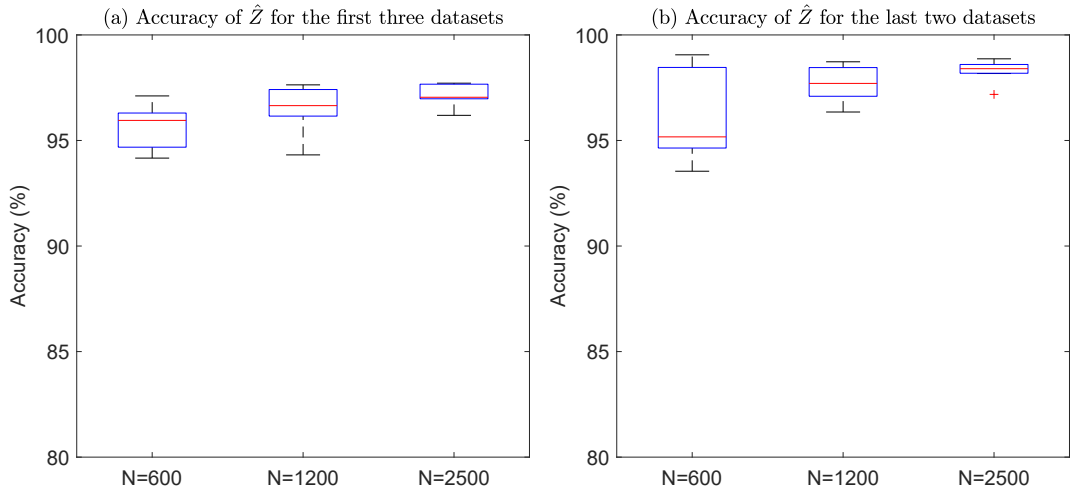

Figure 12: Accuracy of Z_i estimates for different training data sizes N. Here we illustrate the accuracy for JGP-Q, i.e., JGP-CEM with a quadratic partitioning function.

3.2 Prediction Accuracy of JGP Estimates

We use the training data to fit the proposed JGP and the conventional local GP. For the model fit, we use the n-nearest neighborhood as local data, where the value of n varies over $\{25, 35, 50\}$. The fitted models are evaluated at test sites of 2,601 grid locations over $[-0.5, 0.5] \times [-0.5, 0.5]$. We compare the evaluations with the ground truth response values in terms of three performance metrics: the mean square prediction error (MSE), the negative log posterior density (NLPD), and the average $0-1$ loss

$$\begin{aligned} \text{MSE} &= \frac{1}{N_t} \sum_{t=1}^{N_t} (\hat{y}_t - y_t)^2 \\ \text{NLPD} &= \frac{1}{N_t} \sum_{t=1}^{N_t} -\log \phi \left(\frac{\hat{y}_t - y_t}{s_t} \right), \end{aligned} \tag{34}$$

where N_t is the number of the test sites, $\hat{y}_t$ and y_t are the posterior predictive mean estimate and the ground truth of f at the tth test site respectively, s_t is the posterior predictive standard deviation estimate at the tth test site, and $\phi(\cdot)$ is a standard normal density function. The MSE measures the overall error of the posterior mean estimates. The NLPD is the negative logarithm of the posterior predictive density evaluated at test data y_t, which quantifies how the posterior predictive distributions fit the test data.

We expect the performance gap between JGP and the local GP to be substantial at test locations near boundaries dividing homogeneous regions. Therefore, we report the performance metrics for two separate groups of test locations: a group of test locations within a short distance (0.05) from the nearest region boundary and a group of test locations outside the distance. Figure 13 shows the summary statistics of the performance metrics for the first group. For the first three toy examples, the MSE and NLPD values of JGP-L and JGP-Q

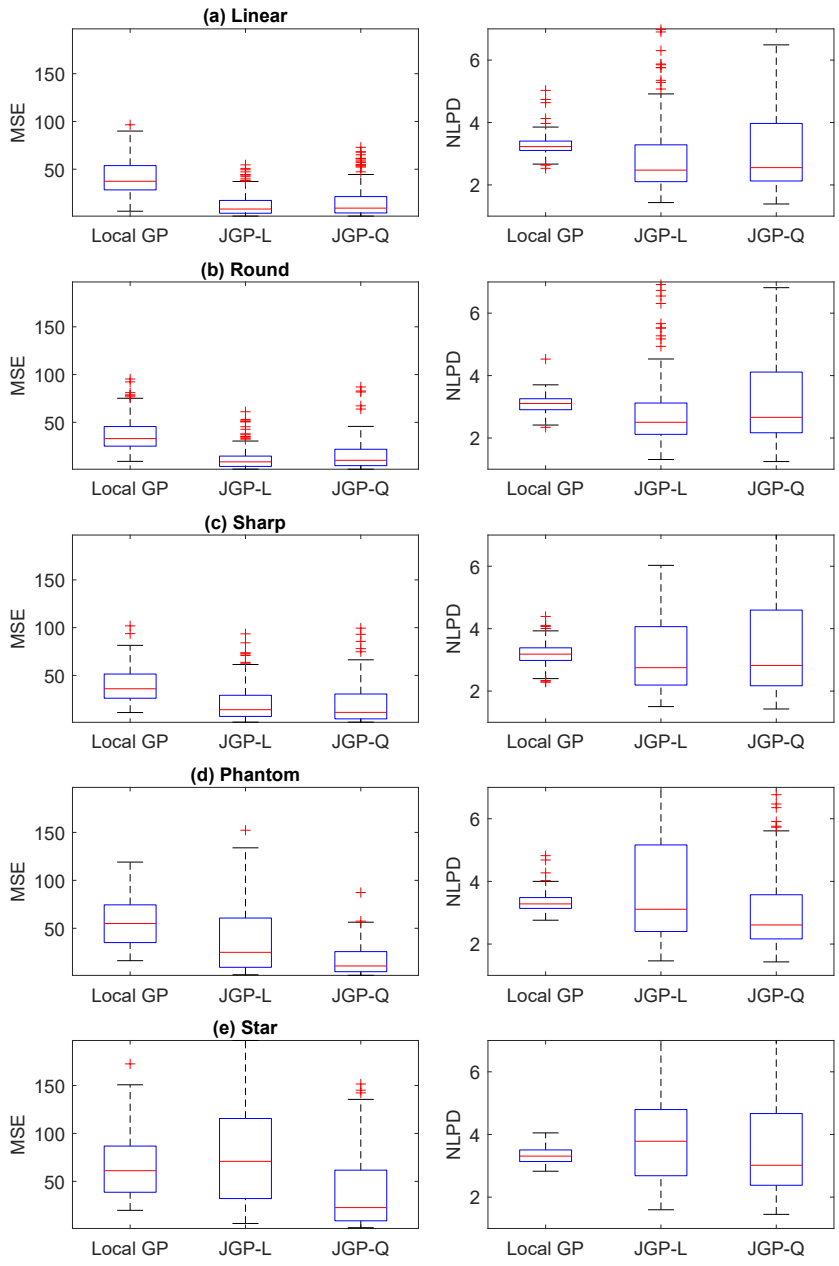

Figure 13: MSE and NLPD metrics of local GP, JGP-CEM with a linear partitioning function (JGP-L), and JGP-CEM with a quadratic partitioning function (JGP-Q) at test locations near regional boundaries for five toy examples. In the box plots above, the red center mark, top-side, and bottom-side of a box represent the median, 25th, and 75th percentiles of the metrics over 27 simulated scenarios with ten replicated experiments. The upper and lower black bars represent the median $\pm$ 1.5 (75th percentile - 25th percentile). Red crosses represent the outliers outside the upper and lower bars.

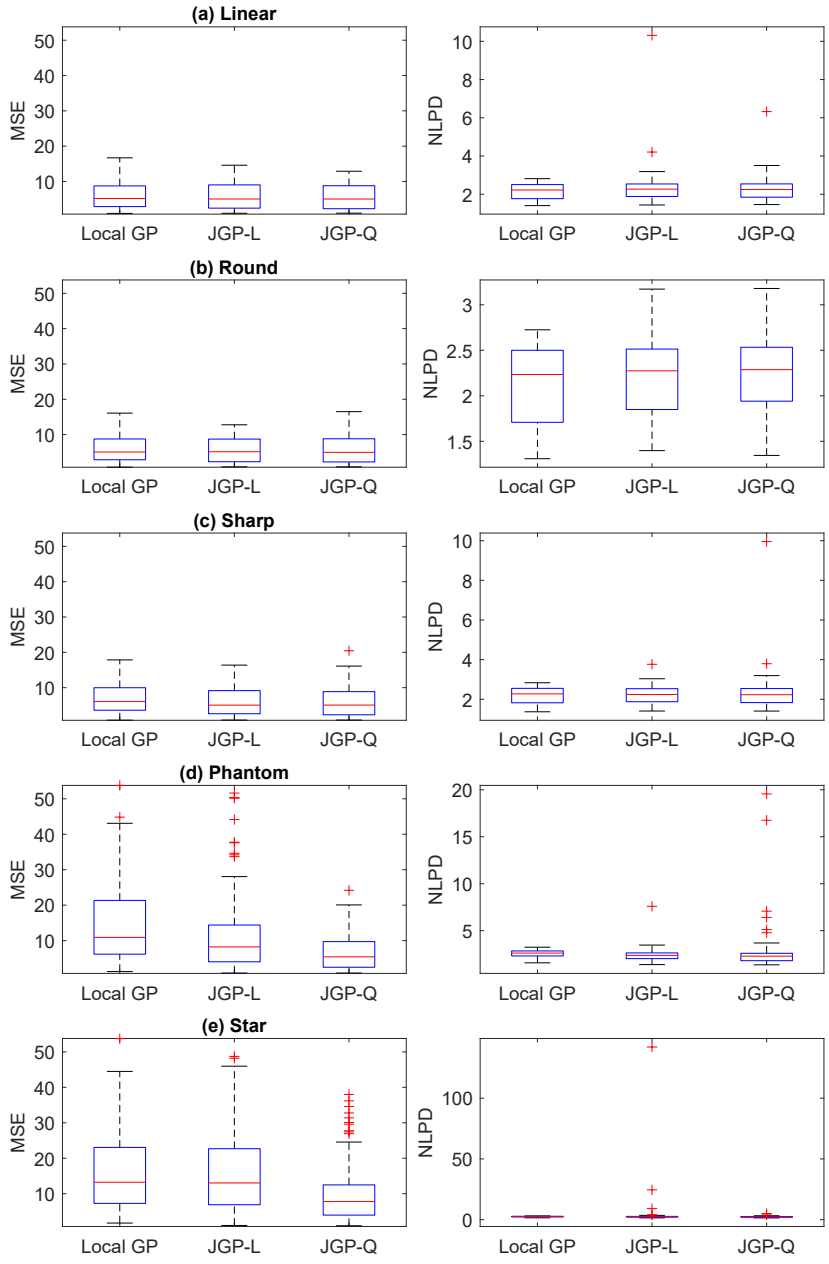

Figure 14: MSE and NLPD metrics of local GP, JGP-CEM with a linear partitioning function (JGP-L), and JGP-CEM with a quadratic partitioning function (JGP-Q) at test locations interior homogeneous regions for five toy examples. In the box plots above, the red center mark, top-side, and bottom-side of a box represent the median, 25th, and 75th percentiles of the metrics over 27 simulated scenarios with ten replicated experiments. The upper and lower black bars represent the median $\pm$ 1.5 (75th percentile - 25th percentile). Red crosses represent the outliers outside the upper and lower bars.

are significantly better than those of the conventional local GP. The JGP-Q and JGP-L are comparable for the first three examples, while the MSE of JGP-Q is twice better than the MSE of JGP-L for the last two examples. The results clearly show that our proposed local data partitioning is effective for piecewise regression, and a quadratic partitioning function works better than a linear partitioning function for complicated scenarios by a significant performance gain.

We also report the performance metrics for test locations interior homogeneous regions (distance from the nearest regional boundary more than 0.05) to look at how the JGP estimates compare to the conventional local GP estimates for interior test locations. Figure 14 shows the summary statistics of the performance metrics. All the three compared GP methods perform comparably for all the five toy examples. That implies that the JGP performs like the conventional local GP when test locations are interior one homogeneous region and local data only contain data from the same region. As we illustrated in Figure 5 (first panel) and Figure 9 (fourth panel), a partitioning function $g(\boldsymbol{x}, \boldsymbol{\omega})$ is formed to select all local data for these interior test points. Therefore, the JGP works as the conventional local GP for these test locations. The result is desirable because the JGP does not take out some useful local data erroneously.

3.3 Performance versus noise level σ^2

We report how the JGP and local GP work for different noise levels. We only present the trend of MSE versus σ^2, because the trend of NLPD is similar. The trends of MSE are similar for the first to third toy examples, and they are similar for the fourth and fifth examples. Figure 15 shows the trends of MSE for two representative toy examples: the second and fourth ones. For the second toy example, the average MSE values of the JGP estimates are comparable to the noise level σ^2, which are the lowest MSE achievable. The local GP's MSE values are significantly higher than σ^2. For the fourth toy example, the MSE levels of the JGP with linear $g(\boldsymbol{x}, \boldsymbol{\omega})$ are significantly elevated. That is because a linear partitioning function is inflexible to split local data into homogeneous regions for this example, as we illustrated in Figure 6 (first three panels). When we use a quadratic partitioning function, the MSE levels are reduced comparably to the noise level σ^2.

3.4 Effect of local data size n and training data size N

We also investigate the effect of training data size N and local data size n on JGP estimates to make recommendations on the choice of n. We use JGP-CEM with quadratic $g(\boldsymbol{x}, \boldsymbol{\omega})$ for the investigation. Figure 16 shows trend of MSE as n and N change. According to the medians and variances of MSE, the choice of n between 15 and 35 achieves the best performance for $N = 600$ and $N = 1200$. Smaller or larger choices would increase MSE. When data are very dense, as in $N = 2500$ (for which the distance from a test location to the closest training data point is only 0.02), the MSE tends to keep decreasing as n decreases further. However, if n goes below 15, the variance of MSE increases for most tested scenarios. The choice $n = 25$ balances the MSE and variances, so our suggestion based on the numerical study is $n = 25$.

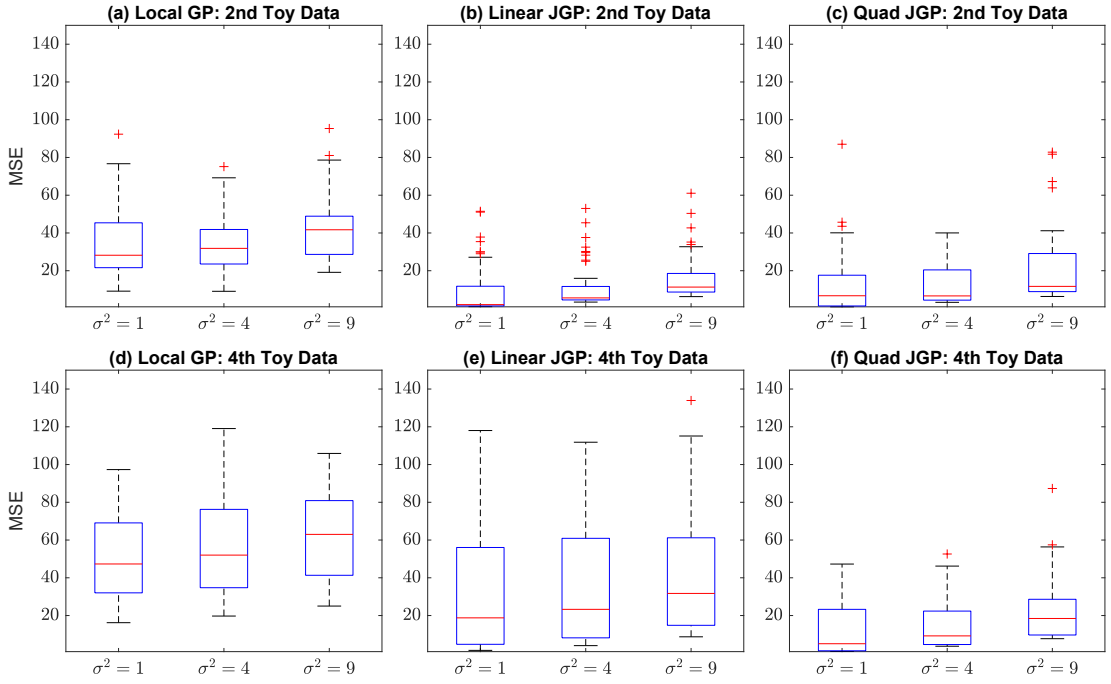

Figure 15: Effect of noise level σ^2 on Local GP, JGP-L, and JGP-Q at test locations near regional boundaries for the second and fourth toy datasets.

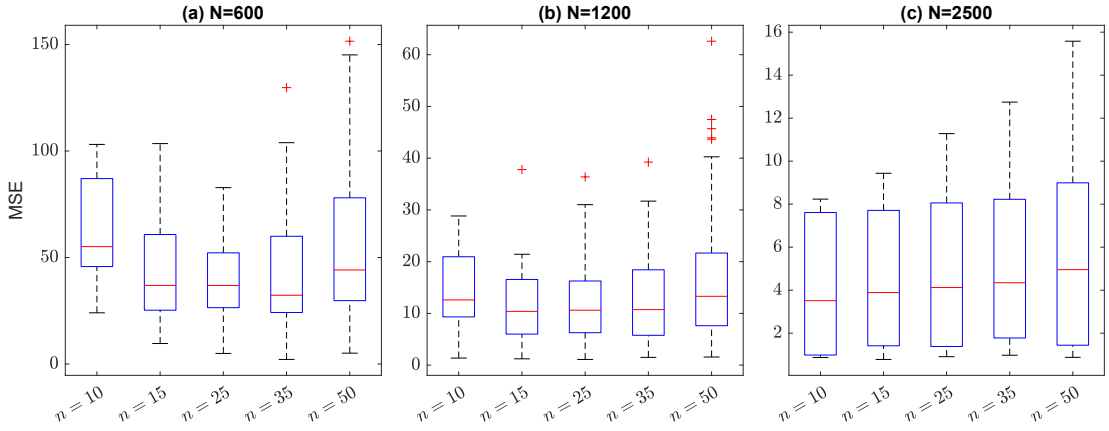

Figure 16: Effect of training data size N and local data size n on the MSE of JGP-Q. The trends are similar for JGP-L.

4. Performance Analysis of JGP with Higher Input Dimensions

We extend our simulated scenarios to study how JGP performs with higher input dimensions. We consider a d-dimensional input domain $[-0.5, 0.5]^d$, and the domain is partitioned

into multiple domains by $(d+1)$ partitioning functions,

$$g_0(\boldsymbol{x}) = \sum_{i=1}^{d} x_i^2 - 0.4^2$$
$$g_j(\boldsymbol{x}) = \sum_{i=1}^{d} x_i^2 - x_j^2 + (x_j + r_j 0.5)^2 - 0.3^2 \text{ for } j = 1, \ldots, d.$$

where x_i represents the ith coordinate of an input vector $\boldsymbol{x}$, and r_j is randomly sampled with 50% for +1 and 50% for -1 for each $j = 1, \ldots, d$. The input domain can be bisected by each g_j into $\mathcal{X}_{j,+} = \{\boldsymbol{x} \in [-0.5, 0.5]^d : g_j(\boldsymbol{x}) \geq 0\}$ and $\mathcal{X}_{j,-} = \{\boldsymbol{x} \in [-0.5, 0.5]^d : g_j(\boldsymbol{x}) < 0\}$. A combination of the bisections determines the domain partitioning. The region index for an input location $\boldsymbol{x}$ is determined by

$$\text{region}(\boldsymbol{x}) = \sum_{j=0}^{d} 2^j I_{\mathcal{X}_{j,+}}(\boldsymbol{x}).$$

In total, N training inputs, $\{\boldsymbol{x}_i, i = 1, \ldots, N\}$, are randomly drawn uniformly over the input domain, and the corresponding responses are randomly drawn from GP with region dependent means and co variances. Responses from region k are randomly sampled from a GP with a constant mean function μ_k and a square exponential covariance function parameterized by $\theta_k = (\theta_{k1}, \theta_{k2})$,

$$c(\boldsymbol{x}, \boldsymbol{x}'; \theta_k) = \theta_{k1} \exp\left\{-\frac{1}{\theta_{k2}}(\boldsymbol{x} - \boldsymbol{x}')^T(\boldsymbol{x} - \boldsymbol{x}')\right\}. \tag{35}$$

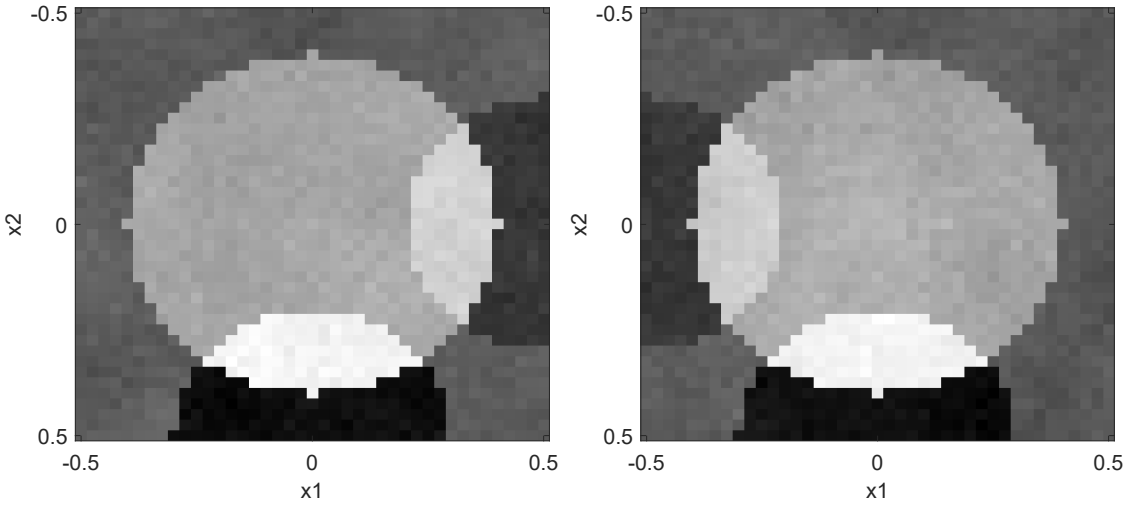

Figure 17: Two random realizations of simulated response surfaces on a 2D domain $[-0.5, 0.5]^2$ are presented here for illustration purposes. We also simulated similar datasets for higher dimensional domains.

Dimension (d)	JGP-L	JGP-Q	JGP-Q / JGP-L
2	4.0179	3.4779	1.15
5	1234.1183	935.1615	1.31

Table 1: MSE for a test case with two regions separated by a linear regional boundary, $x_1 + \ldots + x_d = 0$. We applied $N = 10,000$, and $n = 25$. The results reported here are the averages over ten replicated experiments.

The mean μ_k is randomly selected from a uniform distribution over a set $\{-13.5k, +13.5k\}$, and we used $\theta_{k1} = 9$ and $\theta_{k2} = 200$. Both of μ_k and θ_k are assumed unknown during training, and they are estimated with training data. The prior mean function values for different regions are quite distinct so create discontinuities across regions. We added zero-mean Gaussian noises with variance 4 to the sampled responses for emulating noisy data. We also generated N_t test data similarly but without noise additions. The test input locations are chosen within 0.05 distance from regional boundaries, because we realized from Section 3 that the performance of JGP and local GP are comparable for test locations deeply inside homogeneous regions.

We varied the input dimension d from 2 to 10 and the training data size N over $\{1000, 3000, 5000, 10000, 20000, 30000\}$, while we fixed $N_t = 150$ and chose the local neighborhood size $n = 25$, as suggested from our simulation study in Section 3. We have 54 different configurations with different choices of d and N. For each configuration, we run the randomized experiments 100 times for replicated experiments. Figure 18 shows the summary statistics of the results with the medians, quantiles of the MSE values for different N and d. Figure 19 presents how the median MSE values trend for different N and d. The JGP's MSE trends down as N increases and d decreases. As shown in Figure 19(b), the median MSE values are proportional to N^d. This JGP's MSE trend is very similar to the MSE trend of a stationary GP for stationary data (Park et al., 2022). That is, JGP provides a good solution to high dimensional piecewise regression problems.

We also acknowledge the issue of overfitting with JGP-Q for high-dimensional cases. The JGP-Q model has $d^2 + d + 1$ degrees of freedom. As d increases, the degrees increase significantly relative to the local data size n. It can cause a model overfit. To illustrate this issue, we carefully designed a simulated experiment with the d-dimensional input domain $[-0.5, 0.5]^d$, and the domain is partitioned into two regions by a linear boundary $\sum_{i=1}^{d} x_i = 0$. The response surface for the first region is sampled from the Gaussian process with mean -13.5 and covariance function (35), while the one for the second region is from the Gaussian process with mean 13.5 and covariance function (35). We set the training data size $N = 10,000$ and the local neighborhood size $n = 5(d^2 + d + 1)$. We learned JGP-L and JGP-Q with the training dataset, and the trained models are tested and compared against a test dataset of size 200. Table 1 compares the test MSE for JGP-L and JGP-Q. As seen in the table, the MSE of JGP-Q degrades quicker than JGP-L, which illustrates the overfit issue with JGP-Q. The experiment result suggests the use of JGP-L for high dimensional cases ($d > 3$), while JGP-Q would serve better low dimensional cases with complex regional boundaries.

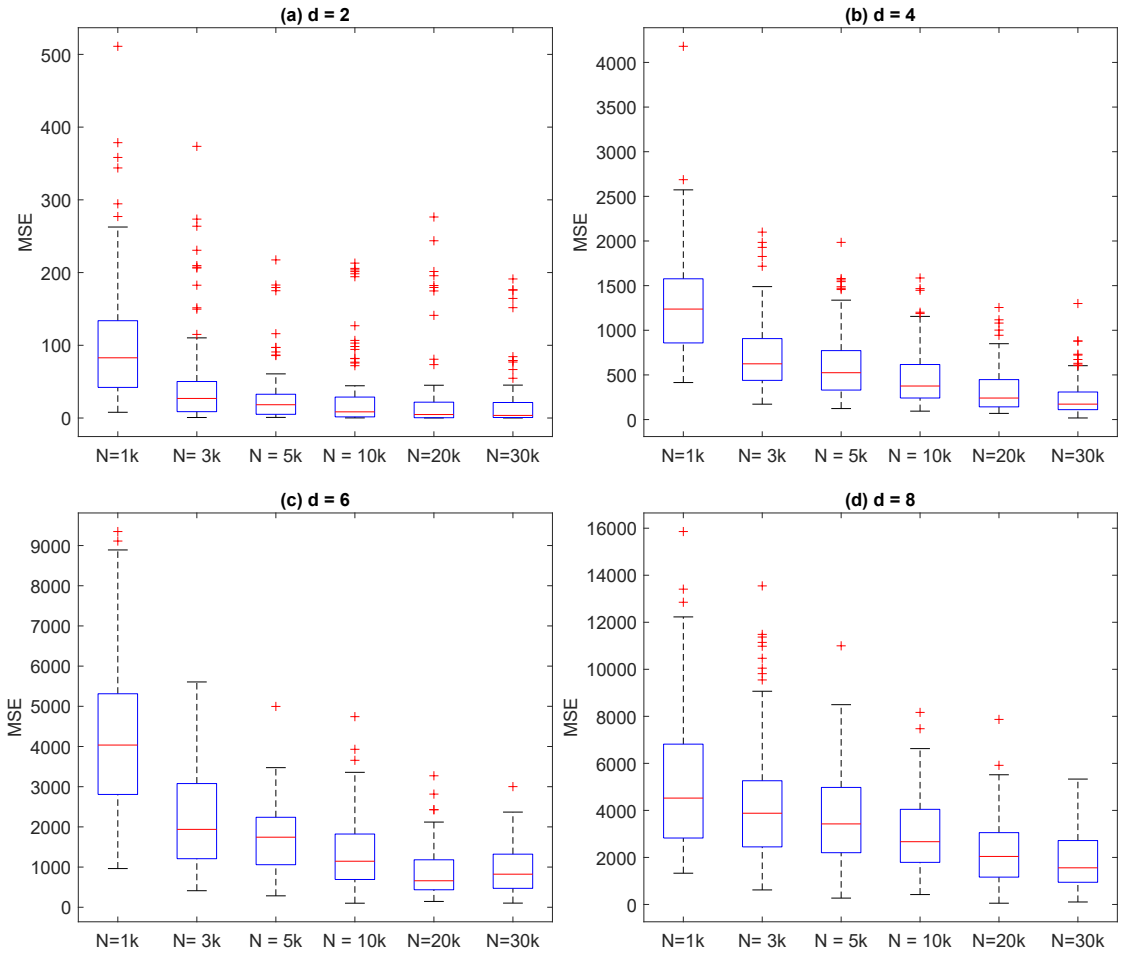

Figure 18: Effect of training data size N and input dimensions d on JGP-Q.

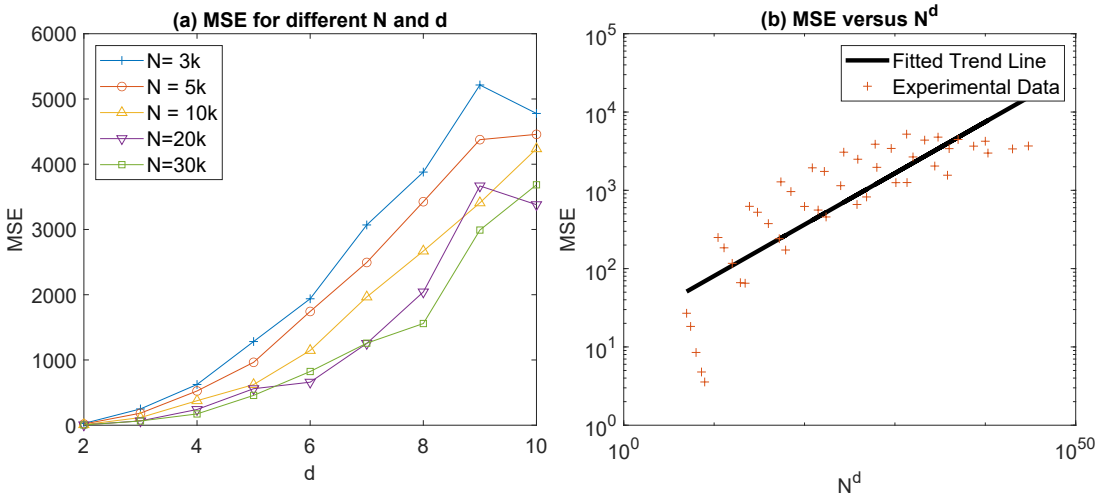

Figure 19: MSE of JGP-Q versus training data size N and input dimensions d.

5. Real Data Study

We perform a numerical comparison of the proposed JGP to three benchmarks, local GP, laGP with MSPE criterion (Gramacy and Apley, 2015), Bayesian Treed GP (Gramacy and Lee, 2008, TGP) and the Dynamic Tree Model (Taddy et al., 2011, DynaTree). We use the R implementations of laGP, TGP and DynaTree from the authors. We use three real datasets. The first dataset contains 10,153 corrosion sensor readings collected under various environmental conditions comprising air temperature, surface temperature, relative humidity, and effective humidity. The sensor readings measure a corrosion current flowing on a metal specimen, which has shown abrupt changes around unknown threshold conditions (Park et al., 2022). We randomly split the dataset into 90% training dataset and 10% test dataset. We use the training dataset to fit the proposed model and the three benchmarks, and we evaluate the trained models using the test dataset in terms of MSE. The second dataset contains a compressive sensing (CS) result of nanoparticle agglomerates. The inputs are (x, y) coordinates of sensing locations, and the corresponding responses are the electron microscope intensities at the input locations. The microscope intensities suddenly jump around the boundaries between nanoparticle agglomerates and substrate materials. There are 17,519 sensing locations selected by an adaptive sensing algorithm (Park et al., 2021). We randomly split the data points into 90% training data and 10% test data. We use them for training and testing the compared methods in the same manner as the first dataset for evaluating the MSE. The third dataset contains 52 carbon nanotube yields collected for 52 different carbon nanotube growth conditions described by a reaction temperature and a log ratio of two reactants. Nanotube yields are almost zero for some growth conditions, and they suddenly jump around the conditions where chemical catalysts are activated. The conditions vary significantly depending on the reaction temperature. Since the dataset size is quite small, we use a leave-one-out cross-validation scheme instead of the conventional train and test data split. We report the average leave-one-out cross-validation error.

Dataset	N	d	JGP-L	JGP-Q	Local GP	laGP	TGP	DynaTree
Corrosion	10153	4	0.623	0.636	0.972	0.983	1.157	1.188
Comp. Sensing	17519	2	0.150	0.156	0.195	1.289	0.224	2.711
Nanotube	52	2	0.894	1.052	1.540	1.109	1.013	1.062

Table 2: MSE Comparison of the proposed JGP with a linear partitioning function (JGP-L), JGP with a quadratic partitioning function (JGP-Q), Local GP, laGP with MSPE criterion, Bayesian Treed GP (TGP) and Dynamic Treed Regression (DynaTree) for three benchmark datasets.

Table 2 summarizes the numerical comparison. Among the six compared methods, the two treed methods (TGP and DynaTree) use global-scale domain partitioning schemes, either a treed partitioning or a tessellation. The JGP-L and JGP-Q use local-scale domain partitioning schemes. The Local GP and the laGP do not perform any data partitioning. As shown in the summary table, the MSEs of the localized methods are 12%-50% smaller than those of the global-scaled approaches. The differences signify as the training data size N increases. We also note that the two global-scale approaches are not better than the local GP for the first two benchmark cases. As we discussed in the introduction and

literature review, the global-scale domain partitioning schemes adopted by the two global approaches are too restrictive. From the two benchmark datasets, we observe that the response jumps and discontinuity occur over curvy and complex boundaries, which are not effectively modeled by the restrictive global-scale partitioning. The wrong partitioning is the main cause of their prediction inaccuracies around regional boundaries, as also illustrated in Figure 20.

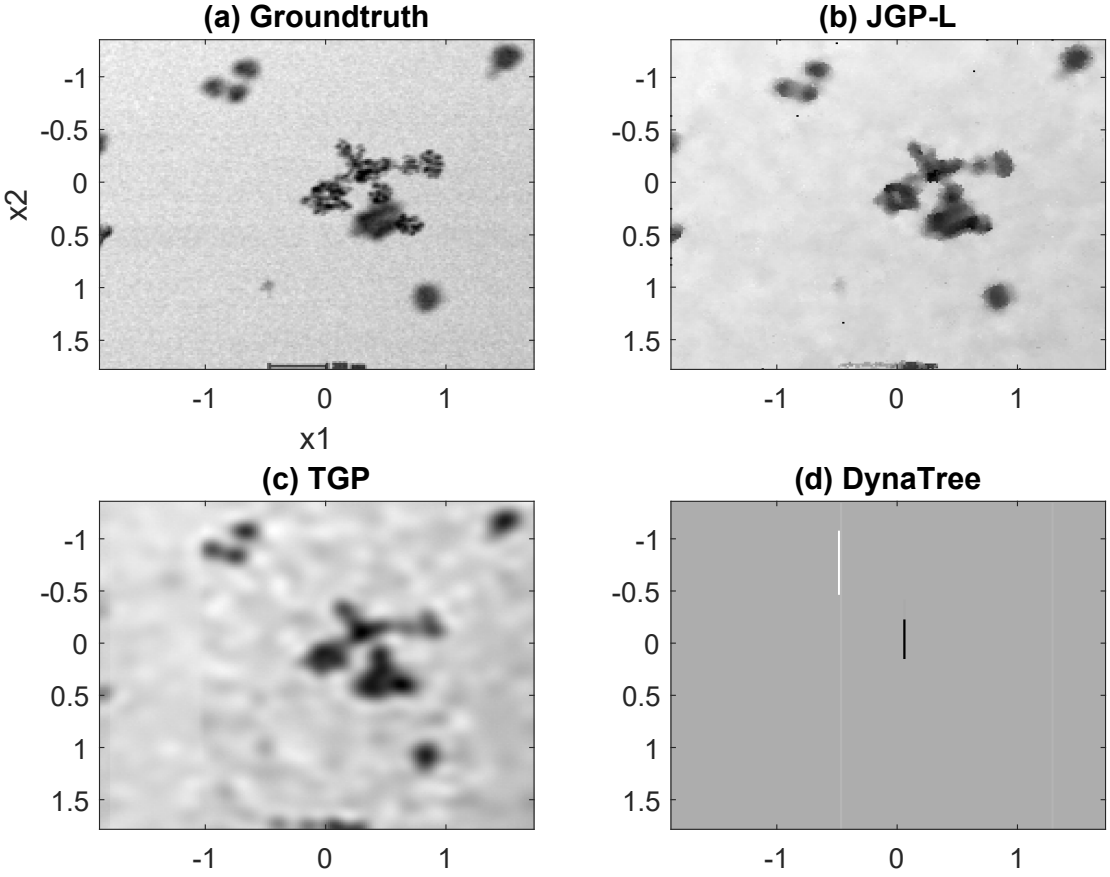

Figure 20: Predictive means of three compared methods for the Compressive Sensing dataset. The test locations are selected densely over a 2D image grid.

Figure 20 illustrates the predictive means of the JGP-L, TGP, and DynaTree for the Compressive Sensing dataset. This dataset has a two-dimensional input domain, so it is easy to visualize the outcomes. The dataset comes with a full-scan microscope image data. The training data for this example corresponds to a small subset (about 5%) of the full-scan data. We select the test locations over the 2D grid locations of the full-scan data, so the prediction over the test locations would serve as an estimate of the full-scan data. We can cross-check the estimation with the full-scan data for visual inspection. As seen in Figure 20(b), the prediction from JGP-L is visually comparable to the ground truth. The

prediction from TGP is also reasonable, but the predictions around regional boundaries are inaccurate, as evidenced by the visually smoothed response jumps around the boundaries. That is mainly because the TGP could not properly partition the input domain; from the figure, the TGP partitions the 2D domain into four regions along the first input dimension. The DynaTree did not work very well for this case. It generates many tiny partitions around the first input dimension.

The ineffectiveness of the global partitioning motivated us to develop our proposed JGP method. Among the three localized approaches, the JGP-L and JGP-Q have a significant performance gain over the Local GP by 29%-42% percent. It shows that the local data partitioning idea of the JGP is effective. The laGP approach is comparable to the local GP approach. On the other hand, for the third benchmark case, the two global approaches outperform the Local GP significantly. The result is mainly due to a small training data size. We should admit that we cannot collect more than 50 data points due to experimental costs. I believe that the same applies to a majority of experimental datasets. The proposed JGP is still better than the global approaches significantly. This example shows that the proposed JGP also works for small N cases.

One last point noted is that the JGP-L is slightly better than the JGP-Q for all three benchmarks. The boundaries dividing homogeneous regions in the real datasets may be less complex than the fourth and fifth toy examples we used for the simulation study, so a linear partitioning function of the JGP-L would suffice to analyze many real datasets. We expect a quadratic partitioning function of JGP-Q be just sufficient for most real scenarios.

6. Conclusion

There are many regression problems where the underlying regression functions are piecewise continuous, i.e., continuous within regions of a domain and discontinuous at regional boundaries. We proposed a Jump Gaussian process (JGP) model that accurately estimates an unknown piecewise continuous regression function. The proposed approach uses the local data neighboring to a test location to predict the unknown regression function value at the test location. Unlike the conventional local GP approach, the proposed approach considers the possibilities of the local data being from different continuous regions. In the proposed approach, the local data is partitioned into pieces by a local data partitioning function. Only the piece of the local data belonging to the same region as the test location is used for the regression estimate. We proposed a parametric form of the local partitioning function and developed a likelihood-maximization algorithm to optimize the parameters of the partitioning function jointly with other GP hyperparameters. The local data split idea has two significant advantages over the existing piecewise GP models that partition the entire input domain globally in pieces and pose independent regression functions for different areas of the input domain. First, the local split is simpler than the global split. While continuous regions divided by complex and curvy regional boundaries are difficult to model at a global level, the complex boundaries can be effectively approximated at a local level by simple linear or polynomial boundaries. Therefore, simple partitioning models can perform the local data split effectively. Second, due to the simplicity, the model estimation of the proposed GP approach is simpler and faster than the approaches based on the global domain split. We showed the two advantages using simulated scenarios and

three real benchmark datasets. Piecewise continuous regression modeling finds many other applications, e.g., experimental analysis of material behaviors with sudden regime changes, spatial data analysis with spatial heterogeneity, and social study that naturally handles survey data with different heterogeneity. The proposed JGP approach significantly advances a study of regression modeling and analysis with the critical applications to these engineering problems.

Acknowledgment

We acknowledge support for this work from the Air Force Office of Scientific Research (FA9550-18-1-0144) and the National Science Foundation (NSF-2152655/NSF-2152679).

References

Amir Beck and Luba Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer Science and Business Media, New York, NY, 2006.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge UK, 2004.

Peter Bryant and John A Williamson. Asymptotic behaviour of classification maximum likelihood estimates. *Biometrika*, 65(2):273–281, 1978.

Corinna Cortes, Giulia DeSalvo, Claudio Gentile, Mehryar Mohri, and Ningshan Zhang. Region-based active learning. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89, pages 2801–2809, Naha, Okinawa, Japan, 2019. PMLR.

Irene Gijbels, Alexandre Lambert, and Peihua Qiu. Edge-preserving image denoising and estimation of discontinuous surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1075–1087, 2006.

Thong Ngee Goh and Min Xie. Statistical control of a six sigma process. *Quality Engineering*, 15(4):587–592, 2003.

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 and Herbert K H Lee. Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103 (483):1119–1130, 2008.

Maya R Gupta and Yihua Chen. Theory and use of the EM algorithm. *Foundations and Trends in Signal Processing*, 4(3):223–296, 2010.

Matthew J Heaton, William F Christensen, and Maria A Terres. Nonstationary Gaussian process models using spatial hierarchical clustering from finite differences. *Technometrics*, 59(1):93–101, 2017.

Yicheng Kang, Xiaodong Gong, Jiti Gao, and Peihua Qiu. Errors-in-variables jump regression using local clustering. *Statistics in Medicine*, 38(19):3642–3655, 2019.

Yicheng Kang, Yueyong Shi, Yuling Jiao, Wendong Li, and Dongdong Xiang. Fitting jump additive models. *Computational Statistics & Data Analysis*, 162:107266, 2021.

Hyoung-Moon Kim, Bani K Mallick, and CC Holmes. Analyzing nonstationary spatial data using piecewise Gaussian processes. *Journal of the American Statistical Association*, 100 (470):653–668, 2005.

Bledar A. Konomi, Huiyan Sang, and Bani K. Mallick. Adaptive Bayesian nonstationary modeling for large spatial datasets using covariance approximations. *Journal of Computational and Graphical Statistics*, 23(3):802–829, 2014.

Richard A Levine and George Casella. Implementations of the Monte Carlo EM algorithm. *Journal of Computational and Graphical Statistics*, 10(3):422–439, 2001.

Z Luo, Huiyan Sang, and Bani Mallick. A Bayesian contiguous partitioning method for learning clustered latent variables. *Journal of Machine Learning Research*, 22(37):1–52, 2021.

Matthew L Malloy and Robert D Nowak. Near-optimal adaptive compressed sensing. *IEEE Transactions on Information Theory*, 60(7):4001–4012, 2014.

Rongji Mu, Lixiang Dai, and Jin Xu. Sequential design for response surface model fit in computer experiments using derivative information. *Communications in Statistics-Simulation and Computation*, 46(2):1148–1155, 2017.

Chiwoo Park and Daniel W Apley. Patchwork kriging for large-scale Gaussian process regression. *Journal of Machine Learning Research*, 19(7):1–43, 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.

Chiwoo Park, Peihua Qiu, Jennifer Carpena-Núñez, Rahul Rao, Michael Susner, and Benji Maruyama. Sequential adaptive design for jump regression estimation. *IISE Transactions*, 0(0):1–18, 2021. doi: 10.1080/24725854.2021.1988770.

Chiwoo Park, David J. Borth, Nicholas S. Wilson, and Chad N. Hunter. Variable selection for Gaussian process regression through a sparse projection. *IISE Transactions*, 54(7): 699–712, 2022.

Christopher A Pope, John Paul Gosling, Stuart Barber, Jill S Johnson, Takanobu Yamaguchi, Graham Feingold, and Paul G Blackwell. Gaussian process modeling of heterogeneity and discontinuities using Voronoi tessellations. *Technometrics*, 63(1):53–63, 2021.

Peihua Qiu. Jump-preserving surface reconstruction from noisy data. *Annals of the Institute of Statistical Mathematics*, 61(3):715–751, 2009.

Carl E Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

Paul D Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417):108–119, 1992.

Matthew A Taddy, Robert B Gramacy, and Nicholas G Polson. Dynamic trees for learning and design. *Journal of the American Statistical Association*, 106(493):109–123, 2011.

Judith D Toms and Mary L Lesperance. Piecewise regression: a tool for identifying ecological thresholds. *Ecology*, 84(8):2034–2041, 2003.

Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer Science & Business Media, New York, NY, 2013.