

Local Gaussian Process Model for Large-scale Dynamic Computer Experiments

Ru Zhang, C. Devon Lin
Department of Mathematics and Statistics,
Queen's University, ON, Canada

and

Pritam Ranjan
Operations Management & Quantitative Techniques,
Indian Institute of Management Indore, MP, India

April 4, 2018

Abstract

The recent accelerated growth in the computing power has generated popularization of experimentation with dynamic computer models in various physical and engineering applications. Despite the extensive statistical research in computer experiments, most of the focus had been on the theoretical and algorithmic innovations for the design and analysis of computer models with scalar responses.

In this paper, we propose a computationally efficient statistical emulator for a large-scale dynamic computer simulator (i.e., simulator which gives time series outputs). The main idea is to first find a good local neighborhood for every input location, and then emulate the simulator output via a singular value decomposition (SVD) based Gaussian process (GP) model. We develop a new design criterion for sequentially finding this local neighborhood set of training points. Several test functions and a real-life application have been used to demonstrate the performance of the proposed approach over a naive method of choosing local neighborhood set using the Euclidean distance among design points.

The supplementary material, which contains proof of the theoretical results, detailed algorithms, additional simulation results and R codes, are available online.

Keywords: Nearest neighbor; Sequential design; Singular value decomposition; Statistical emulator; Time series output.

1. Introduction

Computer experiments are increasingly used in physical, engineering and social sciences as an economical alternative to physical experiments with complex systems/phenomena (Sacks et al. (1989); Santner et al. (2003)). Such experiments are performed on computers with the underlying process represented and implemented by mathematical models. Although cheaper than physical experiments, realistic computer experiments for complex processes can still be time-consuming or sometimes infeasible, and thus, statistical surrogates or emulators are often used for thorough investigation.

Popular objectives of such computer experiments include estimation of pre-specified process features (e.g., overall response surface, global optimum, inverse problem, quantile, and so on), sensitivity analysis, calibration and uncertainty quantification (Jones et al. (1998); Kennedy and O’Hagan (2001); Ranjan et al. (2008); Bingham et al. (2014)). Despite the extensive statistical research in computer experiments, most of the focus had been on the theoretical and algorithmic innovations for the design and analysis of computer models with scalar responses. In this paper we focus on the emulation of dynamic computer models - referred to computer simulators with time series outputs.

Dynamic computer experiments arise in various applications, for example, rainfall-runoff model (Conti et al. (2009)), and vehicle suspension system (Bayarri et al. (2007)). Our motivating application comes from an apple farming industry where the objective is to emulate the population growth curve of European red mites which infest on apple leaves and diminish the crop quality (Teismann et al. (2009)).

With the accelerated growth of computing power, and hence the availability of dynamic computer simulators, there is a desperate need for innovative methodologies and algorithms for the design and analysis of experiments that can particularly handle large data sets. In general, the size of data is a multiple of the length of the time series outputs. Recently, a few attempts on the emulation of dynamic computer experiments have been made by considering time as another input variable in the correlation structure and emulating the response via GP models (Stein, 2005; Conti and O’Hagan, 2010; Hung et al., 2015). Conti et al. (2009) constructed dynamic emulators by using a one-step transition function of state vectors to emulate the computer model movement from one time step to the next. Liu and

West (2009) proposed time varying autoregression (TVAR) models with GP residuals. Farah et al. (2014) extends the TVAR models in Liu and West (2009) by including the input-dependent dynamic regression term. Another clever approach is to represent the time series outputs as linear combinations of a fixed set of basis such as singular vectors (Higdon et al. (2008)) or wavelet basis (Bayarri et al. (2007)) and impose GP models on the linear coefficients. However, fitting GP models over the entire training set can often be computationally infeasible for large-scale dynamic computer experiments involving thousands of training points.

We propose a new approach based on singular value decomposition (SVD) and the local surrogate idea, the latter of which was originally proposed for scalar valued computer simulators with large training data (Emery (2009)). The local surrogate idea was to emulate the process in a local neighborhood of the input location of interest. A naive method of searching for local neighborhood is to select data close to the input location for prediction such that the selected input locations are distributed as uniformly as possible around the location for prediction (as in *k-nearest neighbors*). This method does not take the spatial correlation into account. To search for the most relevant data for local neighborhood in a more intelligent way, Emery (2009) built a local neighborhood by sequentially including data that make the kriging variance decrease more. Gramacy and Apley (2015) further improved the prediction accuracy by using a sequential greedy algorithm and an optimality criterion for finding a non-trivial local neighborhood set. Our objective is to generalize this optimality criterion for the sequential construction of the local neighborhood set for emulating the dynamic computer simulators. We also develop an algorithm for the implementation of the proposed methodology which is efficient from a large-data standpoint.

The subsequent sections are organized as follows. Section 2 reviews the concept of SVD-based GP models and provides a rigorous account for its model assumption and empirical Bayesian inference. Section 3 presents an innovative generalization of the optimality criterion, and a new algorithm for the local approximate SVD-based GP models. We also compare the computational complexity of the algorithms. Section 4 uses two test functions to compare the performance of the *k*-nearest neighbor SVD-based GP models (Euclidean distance based nearest neighbor), the full SVD-based GP models using all training points,

and the proposed methodology in terms of prediction accuracy. The proposed method is also applied to the two-delay blowfly (TDB) model which simulates the population growth curve of European red mites. The concluding remarks are provided in Section 5, and proofs are given in the Supplementary Materials.

2. SVD-based GP Models

Higdon et al. (2008) proposed an SVD-based GP model for the calibration of computer simulators with highly multivariate outputs. They used a full Bayesian approach for model fitting which is exceedingly expensive for large-scale computer experiments, particularly in our proposed sequential procedure for fitting local SVD-based GPs. Thus, we first present a brief review of the SVD-based GP models proposed by Higdon et al. (2008), and then outline an empirical Bayesian procedure to reduce the computational burden.

2.1. Model Formulation

Consider a computer simulator which takes a q -dimensional quantitative input $\mathbf{x} \in \mathbb{R}^q$, and returns a time series output $\mathbf{y}(\mathbf{x}) \in \mathbb{R}^L$ of length L .

For N training points, let $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^T$ be the $N \times q$ input matrix and $\mathbf{Y} = [\mathbf{y}(\mathbf{x}_1), \dots, \mathbf{y}(\mathbf{x}_N)]$ be the $L \times N$ matrix of time series responses. The SVD on \mathbf{Y} gives

$$\mathbf{Y} = \mathbf{U}\mathbf{D}\mathbf{V}^T,$$

where $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_k]$ is an $L \times k$ column-orthogonal matrix of left singular vectors, with k being the minimum of N and L , $\mathbf{D} = \text{diag}(d_1, \dots, d_k)$ is a $k \times k$ diagonal matrix of singular values sorted in decreasing order, and the matrix \mathbf{V} is an $N \times k$ column-orthogonal matrix of right singular vectors. The SVD-based GP model assumes that, for any $\mathbf{x} \in \mathbb{R}^q$,

$$\mathbf{y}(\mathbf{x}) = \sum_{i=1}^p c_i(\mathbf{x})\mathbf{b}_i + \boldsymbol{\epsilon}, \quad (1)$$

where the orthogonal basis $\mathbf{b}_i = d_i\mathbf{u}_i \in \mathbb{R}^L$, for $i = 1, \dots, p$, are the first p vectors of \mathbf{U} scaled by the corresponding singular values. The coefficients c_i 's in (1) are random functions (Rasmussen and Williams (2006)) assumed to be independent Gaussian processes,

i.e., $c_i \sim \text{GP}(0, \sigma_i^2 K_i(\cdot, \cdot; \boldsymbol{\theta}_i))$ for $i = 1, \dots, p$. We use the popular anisotropic Gaussian correlation,

$$K(\mathbf{x}_1, \mathbf{x}_2; \boldsymbol{\theta}) = \exp \left\{ - \sum_{j=1}^q \theta_j (x_{1j} - x_{2j})^2 \right\},$$

for characterizing the spatial correlation structure, however, one can easily use another suitable correlation structure like Matérn or power-exponential (see Santner et al. (2003); Rasmussen and Williams (2006)). The residual error $\boldsymbol{\epsilon}$ in (1) is assumed to be independent Gaussian white noise, that is, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_L)$. For notational simplicity, we denote $\mathbf{U}^* = [\mathbf{u}_1, \dots, \mathbf{u}_p]$, $\mathbf{D}^* = \text{diag}(d_1, \dots, d_p)$, $\mathbf{V}^* = [\mathbf{v}_1, \dots, \mathbf{v}_p]$ and $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_p] = \mathbf{U}^* \mathbf{D}^*$. The j th entry ($1 \leq j \leq N$) of the N -dimensional vector \mathbf{v}_i ($1 \leq i \leq p$) can also be interpreted as a realization of the Gaussian process model for $c_i(\mathbf{x}_j)$.

The number of significant singular values, p in (1), is determined empirically by the cumulative percentage criterion

$$p = \min \left\{ m : \frac{\sum_{i=1}^m d_i}{\sum_{i=1}^k d_i} > \gamma \right\}, \quad (2)$$

where γ is a prespecified threshold of explained variation (we used $\gamma = 0.95$).

Similar to Higdon et al. (2008), we use Bayesian algorithms for model fitting, however, since a full Bayesian implementation is too time consuming, we follow an empirical Bayesian approach. This is particularly crucial here as the GP models have to be fit several times in the proposed sequential procedure.

2.2. Empirical Bayesian Inference

This section briefly reviews the key components of our model fitting procedure. For all the model parameters, we use the maximum a posteriori (MAP) values as the plug-in estimates. The parameters of interest are σ^2 - the error variance, and for $i = 1, 2, \dots, p$, the process variance σ_i^2 and the q -dimensional correlation hyper-parameter $\boldsymbol{\theta}_i = (\theta_{i1}, \dots, \theta_{iq})$. Similar to Gramacy and Apley (2015), we use inverse Gamma priors for σ_i^2 and σ^2 , i.e.,

$$[\sigma_i^2] \sim \text{IG} \left(\frac{\alpha_i}{2}, \frac{\beta_i}{2} \right), i = 1, \dots, p, \quad [\sigma^2] \sim \text{IG} \left(\frac{\alpha}{2}, \frac{\beta}{2} \right),$$

and use the Gamma prior for the hyper-parameter $1/\theta_{ij}$ with the shape parameter $3/2$ and the scale parameter chosen such that the maximum squared distance among any two points

of the design matrix lies at the position of 95% quantile (Gramacy (2016)). As a result, the posterior of $\boldsymbol{\theta}_i$ becomes

$$\pi(\boldsymbol{\theta}_i|\mathbf{v}_i) \propto |\mathbf{K}_i|^{-\frac{1}{2}} \left(\frac{\beta_i + \psi_i}{2} \right)^{-(\alpha_i+N)/2} \pi(\boldsymbol{\theta}_i), \quad (3)$$

where $\pi(\boldsymbol{\theta}_i)$ represents the prior of $\boldsymbol{\theta}_i$, \mathbf{K}_i is the $N \times N$ correlation matrix on the training set \mathbf{X} with the (j, k) th entry being $K(\mathbf{x}_j, \mathbf{x}_k; \boldsymbol{\theta}_i)$, for $j, k = 1, \dots, N$,

$$\psi_i = \mathbf{v}_i^T \mathbf{K}_i^{-1} \mathbf{v}_i,$$

and \mathbf{v}_i is the i th column of \mathbf{V}^* .

It can also be shown that, for any input \mathbf{x}_0 , the conditional distribution of $c_i(\mathbf{x}_0)$ given $(\mathbf{v}_i, \boldsymbol{\theta}_i)$ is independent non-central t distribution with $N + \alpha_i$ degrees of freedom, for $i = 1, \dots, p$, i.e.,

$$[c_i(\mathbf{x}_0)|\mathbf{v}_i, \boldsymbol{\theta}_i] \sim t_{N+\alpha_i}(\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \boldsymbol{\theta}_i), \hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \boldsymbol{\theta}_i)),$$

where the location parameter is

$$\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \boldsymbol{\theta}_i) = \mathbf{k}_i^T(\mathbf{x}_0) \mathbf{K}_i^{-1} \mathbf{v}_i,$$

with $\mathbf{k}_i(\mathbf{x}_0) = [K(\mathbf{x}_0, \mathbf{x}_1; \boldsymbol{\theta}_i), \dots, K(\mathbf{x}_0, \mathbf{x}_N; \boldsymbol{\theta}_i)]^T$, and the scale parameter is

$$\hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \boldsymbol{\theta}_i) = \frac{(\beta_i + \psi_i) \left(1 - \mathbf{k}_i^T(\mathbf{x}_0) \mathbf{K}_i^{-1} \mathbf{k}_i(\mathbf{x}_0) \right)}{\alpha_i + N}.$$

Finally, the posterior distribution of σ^2 given \mathbf{Y} is

$$\begin{aligned} \pi(\sigma^2|\mathbf{Y}) &\propto \pi(\mathbf{Y}|\sigma^2)\pi(\sigma^2) \\ &= (\sigma^2)^{-\frac{NL}{2}} \exp\left\{-\frac{\mathbf{r}^T \mathbf{r}}{2\sigma^2}\right\} (\sigma^2)^{-\frac{\alpha}{2}-1} \exp\left\{-\frac{\beta}{2\sigma^2}\right\} \\ &= (\sigma^2)^{-\frac{NL}{2}-\frac{\alpha}{2}-1} \exp\left\{-\frac{\mathbf{r}^T \mathbf{r} + \beta}{2\sigma^2}\right\}, \end{aligned} \quad (4)$$

where $\pi(\sigma^2)$ is the prior distribution of σ^2 , and $\mathbf{r} = \text{vec}(\mathbf{Y}) - (\mathbf{I}_N \otimes \mathbf{B})\text{vec}(\mathbf{V}^{*T})$, is the vectorization of residual matrix $\mathbf{Y} - \mathbf{B}\mathbf{V}^{*T}$. The notation \otimes represents the Kronecker product and the operator $\text{vec}(\cdot)$ performs vectorization for a matrix. Thus, $[\sigma^2|\mathbf{Y}]$ follows the inverse Gamma distribution $\text{IG}((NL + \alpha)/2, (\mathbf{r}^T \mathbf{r} + \beta)/2)$, and

$$\hat{\sigma}^2 = \underset{\sigma^2}{\text{argmax}} \pi(\sigma^2|\mathbf{Y}) = \frac{1}{NL + \alpha + 2} (\mathbf{r}^T \mathbf{r} + \beta). \quad (5)$$

The posterior predictive distribution of $\mathbf{y}(\mathbf{x}_0)$ is given by

$$\pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \Theta, \sigma^2) \propto \int_{\mathbb{R}^p} \pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{c}(\mathbf{x}_0), \sigma^2) \prod_{i=1}^p \pi(c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i) \prod_{i=1}^p dc_i(\mathbf{x}_0),$$

where $\Theta = \{\theta_1, \dots, \theta_p\}$, and $\mathbf{c}(\mathbf{x}_0) = (c_1(\mathbf{x}_0), \dots, c_p(\mathbf{x}_0))$. For a reasonably large value of N , a normal approximation can be imposed on the non-central $t_{N+\alpha_i}$ distribution of $[c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i]$, i.e.,

$$\pi(c_i(\mathbf{x}_0)|\mathbf{v}_i, \theta_i) \approx \mathcal{N}(\hat{c}_i(\mathbf{x}_0|\mathbf{v}_i, \theta_i), \hat{\sigma}_i^2(\mathbf{x}_0|\mathbf{v}_i, \theta_i)). \quad (6)$$

Furthermore, the results from Section 14.2 of Gelman et al. (2014) can be summarized into Lemma 1 for further simplification of the predictive distribution of $\mathbf{y}(\mathbf{x}_0)$.

Lemma 1 *Suppose $[\mathbf{y}|\boldsymbol{\beta}, \sigma^2] \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}_n)$ and $[\boldsymbol{\beta}] \sim \mathcal{N}(\mathbf{b}, \mathbf{V})$, where $\mathbf{y} \in \mathbb{R}^n$, $\boldsymbol{\beta}, \mathbf{b} \in \mathbb{R}^m$, \mathbf{X} is an $n \times m$ matrix, and \mathbf{V} is an $m \times m$ positive definite covariance matrix. Then, $[\mathbf{y}|\sigma^2] \sim \mathcal{N}(\mathbf{X}\mathbf{b}, \mathbf{X}\mathbf{V}\mathbf{X}^T + \sigma^2\mathbf{I}_n)$.*

Combining (1) and (6) with Lemma 1, we get

$$\pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \Theta, \sigma^2) \approx \mathcal{N}(\mathbf{B}\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \Theta), \mathbf{B}\boldsymbol{\Lambda}(\mathbf{V}^*, \Theta)\mathbf{B}^T + \sigma^2\mathbf{I}_L), \quad (7)$$

where $\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \Theta) = [\hat{c}_1(\mathbf{x}_0|\mathbf{v}_1, \theta_1), \dots, \hat{c}_p(\mathbf{x}_0|\mathbf{v}_p, \theta_p)]^T$, and $\boldsymbol{\Lambda}(\mathbf{V}^*, \Theta) = \text{diag}(\hat{\sigma}_1^2(\mathbf{x}_0|\mathbf{v}_1, \theta_1), \dots, \hat{\sigma}_p^2(\mathbf{x}_0|\mathbf{v}_p, \theta_p))$. The parameters Θ and σ^2 cannot be integrated out analytically, and Kennedy and O'Hagan (2001) suggested using the MAP estimator into the predictive distribution. Following their paradigm, we plug $\hat{\sigma}^2$ and

$$\hat{\boldsymbol{\theta}}_i = \underset{\boldsymbol{\theta}_i}{\text{argmax}} \pi(\boldsymbol{\theta}_i|\mathbf{v}_i), \quad i = 1, \dots, p, \quad (8)$$

into (7) to obtain the approximate predictive distribution

$$\pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{Y}) \approx \pi(\mathbf{y}(\mathbf{x}_0)|\mathbf{V}^*, \hat{\sigma}^2, \hat{\Theta}) \approx \mathcal{N}(\mathbf{B}\hat{\mathbf{c}}(\mathbf{x}_0|\mathbf{V}^*, \hat{\Theta}), \mathbf{B}\boldsymbol{\Lambda}(\mathbf{V}^*, \hat{\Theta})\mathbf{B}^T + \hat{\sigma}^2\mathbf{I}_L). \quad (9)$$

where $\pi(\boldsymbol{\theta}_i|\mathbf{v}_i)$ and $\pi(\sigma^2|\mathbf{Y})$ are given by (3) and (4), respectively. As a result, with the data \mathbf{X} and \mathbf{Y} , the pre-specified hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, as well as the threshold γ in (2), the SVD-based GP model fitting via Bayesian procedure provides the approximate predictive distribution in (9) with the plug-in estimates of $\hat{\sigma}^2$ in (5) and $\hat{\boldsymbol{\theta}}_i$'s in (8). It shall

be noted that, the MAP estimator of θ_i 's can be shown to be robust (Gu et al., 2017). Algorithm 1 of the supplementary material summarizes the important steps in estimating the necessary parameters of the posterior predictive distribution (9) for a full SVD-based GP model.

Fitting the i th GP model ($1 \leq i \leq p$) to N training data points involves numerous evaluations of the posterior (3), and the computation of \mathbf{K}_i^{-1} and $|\mathbf{K}_i|$ requires $O(N^3)$ floating point operations (flops), which can quickly become infeasible even for moderately large N . Thus, we propose to use a localized SVD-based GP model that aims to achieve the same prediction accuracy at a substantially less computational cost.

3. Local SVD-based GP Model

The main idea is to use a small subset of n ($\ll N$) points instead of the entire training set of N points for approximating the predicted response at an arbitrary \mathbf{x}_0 in the input space. Let \mathbf{X} be the training set of N points, and $\mathbf{X}^{(n)}(\mathbf{x}_0)$ or $\mathbf{X}^{(n)}$ (in short) denote the desired subset of \mathbf{X} which defines the n -point neighborhood of \mathbf{x}_0 contained in \mathbf{X} . In this section, we discuss two methods of constructing this neighborhood set $\mathbf{X}^{(n)}$.

The first one, called as the *naive* approach, assumes the elements of the neighborhood set $\mathbf{X}^{(n)}$ by finding n nearest neighbors of \mathbf{x}_0 in \mathbf{X} as per the Euclidean distance in the *k-nearest neighbor* method. The emulator obtained via fitting an SVD-based GP model (as described in Section 2) to this local set of points is referred to as *k-nearest neighbor SVD-based GP model* (in short, knnsvdGP). Though, knnsvdGP is computationally much cheaper than the *full SVD-based GP model* (referred to as svdGP) trained on N points, its prediction accuracy may not be satisfactory.

The second method (main focus of this paper) finds the neighborhood set $\mathbf{X}^{(n)}(\mathbf{x}_0)$ (for every \mathbf{x}_0) using a greedy approach. Gramacy and Apley (2015) developed a greedy sequential algorithm for constructing a neighborhood set for a scalar-valued simulator. In this paper, we propose a generalization of this algorithm for dynamic computer simulators. For every test point, the generalized greedy algorithm finds a local set of points in the training set to build an SVD-based GP model. Such model is referred to as the *local approximate SVD-based GP model* (in short, lasvdGP).

3.1. Local Approximate SVD-based GP Model

For every given \mathbf{x}_0 in the input space, the proposed approach starts with finding a smaller neighborhood set $\mathbf{X}^{(n_0)}(\mathbf{x}_0)$, which consists of n_0 ($< n$) nearest neighbors of \mathbf{x}_0 in \mathbf{X} (with respect to the Euclidean distance). This step is the same as in knnsvdGP with n replaced by n_0 . The remaining $n - n_0$ neighborhood points are chosen sequentially one at-a-time by optimizing a merit-based criterion over the input space. The prime objective is to reduce the overall prediction error. The key steps of the proposed lasvdGP approach is summarized in Algorithm 2 of the supplementary material.

Let k denote the current number of points in the neighborhood set, $\mathbf{X}^{(k)}$ and $\mathbf{X} \setminus \mathbf{X}^{(k)}$ be the sets of selected and unselected (remaining) training points, respectively, and $\hat{\Theta}^{(k)} = \{\hat{\theta}_1^{(k)}, \dots, \hat{\theta}_p^{(k)}\}$ be the estimated correlation parameters using $\mathbf{X}^{(k)}$ and $\mathbf{Y}(\mathbf{X}^{(k)})$. Then the next follow-up point in the neighborhood set is chosen as

$$\mathbf{x}_{k+1}^* = \operatorname{argmin}_{\mathbf{x} \in \mathbf{X} \setminus \mathbf{X}^{(k)}} J(\mathbf{x}_0, \mathbf{x}),$$

where

$$J(\mathbf{x}_0, \mathbf{x}) = \mathbb{E} \left\{ \mathbb{E} \left[\left\| \mathbf{y}(\mathbf{x}_0) - \hat{\mathbf{y}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}) \right\|^2 \middle| \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \middle| \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right\}, \quad (10)$$

with

$$\begin{aligned} \hat{\mathbf{y}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}) &= \mathbb{E} \left[\mathbf{y}(\mathbf{x}_0) \middle| \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \\ &= \mathbf{B}^{(k)} \hat{\mathbf{c}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}), \end{aligned} \quad (11)$$

where $\mathbf{B}^{(k)}$ and $\mathbf{V}^{*(k)}$ are the matrices of basis vectors and the right singular vectors, p_k is the number of bases selected in this iteration, and $\mathbf{c}(\mathbf{x}) = [c_1(\mathbf{x}), \dots, c_{p_k}(\mathbf{x})]^T$ with $c_i \sim \text{GP}(0, \sigma_i^2 K(\cdot, \cdot; \boldsymbol{\theta}_i^{(k)}))$. The predictive mean vector of coefficients $\hat{\mathbf{c}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)})$ is calculated in the exact same way as (7) except $[(\mathbf{V}^{*(k)})^T, \mathbf{c}(\mathbf{x})]^T$ and $\Theta^{(k)}$ are used in place of \mathbf{V}^* and Θ , respectively.

This J -criterion is a generalization of the active learning Cohn (ALC) criterion (Cohn et al. (1996); Cohn (1996); Gramacy and Apley (2015))

$$\int_{\mathbf{x}} \left[\int_{\mathbf{y}} (\hat{\mathbf{y}}(\mathbf{x}) - \mathbf{y}(\mathbf{x}))^2 dP(\mathbf{y} | \mathbf{x}) \right] dP(\mathbf{x}),$$

where $y(\mathbf{x})$ and $\hat{y}(\mathbf{x})$ are the observed and predicted scalar-valued outputs, respectively, at input \mathbf{x} , $P(y|\mathbf{x})$ is the approximate predictive distribution, and the marginal distribution $P(\mathbf{x})$ is uniform. For dynamic computer simulators, we use L_2 norm discrepancy instead of the squared error.

The closed form expression for $J(\mathbf{x}_0, \mathbf{x})$ can be derived by taking the outer expectation in (10) with respect to the approximate posterior distribution in (7) and substituting $(\mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2)$ for $(\mathbf{V}^*, \Theta, \sigma^2)$. Similarly, the expectation in (11) is computed with respect to (7), and by substituting $([(\mathbf{V}^{*(k)})^T, \mathbf{c}(\mathbf{x})]^T, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2)$ for $(\mathbf{V}^*, \Theta, \sigma^2)$. Proposition 1 states the closed form expression of $J(\mathbf{x}_0, \mathbf{x})$, and the proof is shown in the Appendix A of the supplementary materials.

Proposition 1 *Suppose the expectations in (10) and (11) are taken with respect to the approximate predictive distribution (7). Then, for any $\mathbf{x} \in \mathbf{X} \setminus \mathbf{X}^{(k)}$*

$$J(\mathbf{x}_0, \mathbf{x}) = (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)}),$$

where $d_i^{(k)}$ is the i th largest singular value of $\mathbf{Y}^{(k)}$,

$$\begin{aligned} \hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)}) &= \frac{\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})}{\alpha_i + k} \left(\beta_i + \frac{\alpha_i + k}{\alpha_i + k - 1} \psi_i^{(k)} \right), \\ \rho_i^{(k)}(\mathbf{x}_0, \mathbf{x}) &= 1 - \tilde{\mathbf{k}}_i(\mathbf{x}_0, \mathbf{x})^T \tilde{\mathbf{K}}_i^{-1}(\mathbf{x}) \tilde{\mathbf{k}}_i(\mathbf{x}_0, \mathbf{x}), \\ \psi_i^{(k)} &= (\mathbf{v}_i^{(k)})^T (\mathbf{K}_i^{(k)})^{-1} \mathbf{v}_i^{(k)}, \\ \tilde{\mathbf{k}}_i(\mathbf{x}_0, \mathbf{x}) &= [K(\mathbf{x}_0, \mathbf{x}_1^{(k)}; \hat{\theta}_i^{(k)}), \dots, K(\mathbf{x}_0, \mathbf{x}_k^{(k)}; \hat{\theta}_i^{(k)}), K(\mathbf{x}_0, \mathbf{x}; \hat{\theta}_i^{(k)})]^T, \\ \tilde{\mathbf{K}}_i(\mathbf{x}) &= \begin{bmatrix} \mathbf{K}_i^{(k)} & \mathbf{k}_i^{(k)}(\mathbf{x}) \\ \mathbf{k}_i^{(k)}(\mathbf{x})^T & 1 \end{bmatrix}, \end{aligned}$$

with $\mathbf{v}_i^{(k)}$ being the i th column of $\mathbf{V}^{*(k)}$, for $i = 1, \dots, p_k$, $\mathbf{x}_j^{(k)}$ being the j th point of $\mathbf{X}^{(k)}$ for $j = 1, \dots, k$, $\mathbf{K}_i^{(k)}$ being a $k \times k$ matrix with $K(\mathbf{x}_j^{(k)}, \mathbf{x}_l^{(k)}; \hat{\theta}_i^{(k)})$, as the (j, l) th entry, and $\mathbf{k}_i^{(k)}(\mathbf{x}) = [K(\mathbf{x}, \mathbf{x}_1^{(k)}; \hat{\theta}_i^{(k)}), \dots, K(\mathbf{x}, \mathbf{x}_k^{(k)}; \hat{\theta}_i^{(k)})]^T$.

As $(\hat{\sigma}^{(k)})^2 L$ is a constant with respect to \mathbf{x} , finding \mathbf{x}_{k+1}^* , by minimizing the J -criterion

in Proposition 1, is equivalent to obtaining

$$\mathbf{x}_{k+1}^* = \operatorname{argmin}_{\mathbf{x} \in \mathbf{X} \setminus \mathbf{X}^{(k)}} \left[\sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{x}, \mathbf{v}_i^{(k)}, \hat{\boldsymbol{\theta}}_i^{(k)}) \right]. \quad (12)$$

Note that the simplified design criterion in (12) turns out to be the weighted sum of the predictive variance of the singular vector coefficients, where the weights are $(d_i^{(k)})^2$ which represents the total variation explained by the i th singular vector basis. Therefore, the chosen follow-up point \mathbf{x}_{k+1}^* minimizes the expected L_2 prediction error at \mathbf{x}_0 evaluated at stage k .

As compared to knnsvdGP, the proposed algorithm, lasvdGP, requires many more GP model fitting steps, which increase the computational cost, however, it is still substantially faster than the svdGP implementation. The matrix inverse updating procedure employed in Hager (1989) (also used in Gramacy and Apley (2015)) can be used to achieve further time saving from $O(k^3)$ to $O(k^2)$ in evaluating $J(\mathbf{x}_0, \mathbf{x})$ of Proposition 1 for each $\mathbf{x} \in \mathbf{X} \setminus \mathbf{X}^{(k)}$, where $k = n_0, \dots, n-1$, is the number of neighborhood points in the current neighborhood set $\mathbf{X}^{(k)}$. This is because, the evaluation of the J -criterion requires inverting $(k+1) \times (k+1)$ correlation matrices $\{\tilde{\mathbf{K}}_i(\mathbf{x})\}_{i=1}^p$, and applying the matrix inverse update, we have

$$\tilde{\mathbf{K}}_i(\mathbf{x})^{-1} = \begin{bmatrix} (\mathbf{K}_i^{(k)})^{-1} + \mathbf{g}_i \mathbf{g}_i^T \phi_i & \mathbf{g}_i \\ \mathbf{g}_i^T & \phi_i^{-1} \end{bmatrix},$$

where $\mathbf{g}_i = -(\mathbf{K}_i^{(k)})^{-1} \mathbf{k}_i^{(k)}(\mathbf{x}) / \phi_i$ and $\phi_i = 1 - \mathbf{k}_i^{(k)}(\mathbf{x})^T (\mathbf{K}_i^{(k)})^{-1} \mathbf{k}_i^{(k)}(\mathbf{x})$. Thus, the computation of $\tilde{\mathbf{K}}_i(\mathbf{x})^{-1}$ attributes to computing both $(\mathbf{K}_i^{(k)})^{-1}$ and \mathbf{g}_i , the former of which has been calculated and stored in the process of estimating range parameters and thus no additional computing time is required for evaluating $(\mathbf{K}_i^{(k)})^{-1}$ in the J -criterion. On the other hand, the evaluation of $\mathbf{k}_i^{(k)}(\mathbf{x})$ in \mathbf{g}_i requires $O(k)$ time, and the complexity of the matrix multiplication $(\mathbf{K}_i^{(k)})^{-1} \mathbf{k}_i^{(k)}(\mathbf{x})$ is $O(k^2)$. Note that the anticipated boost in the prediction accuracy at the cost of a small increase in the computational cost is perhaps worth it. The computational complexities of the two methods knnsvdGP and lasvdGP are more extensively discussed in Section 3.2.

3.2. Computational Complexity

In this section, we discuss the computational complexity of (1) full SVD-based GP model (svdGP), (2) k -nearest neighbor SVD-based GP model (knnsvdGP), and (3) local approximate SVD-based GP model (lasvdGP). For this comparison, let \mathbf{X} contain N training points, \mathbf{X}^* consist of M test points, L be the length of time series response, each neighborhood set in knnsvdGP and lasvdGP consists of n training points, and $N > L > n$ (assuming N is large). Furthermore, we only compute the diagonal entries of the predictive covariance matrix of $\mathbf{y}(\mathbf{x}_0)$, i.e., the marginal predictive variances, for each $\mathbf{x}_0 \in \mathbf{X}^*$.

(1) **svdGP**: A single call of the empirical Bayesian inference for SVD-based GP model on the full training data requires $O(N^3)$ floating point operations (flops). Since we assume $N > L$, the estimation of Θ is the dominant part of the empirical Bayesian inference computation, which requires $O(N^3)$ flops. The complexity of the prediction step is $O(M(N^2 + L)) = O(MN^2)$, and thus, the total cost of svdGP is $O(N^2 \max\{M, N\})$.

(2) **knnsvdGP**: For each $\mathbf{x}_0 \in \mathbf{X}^*$, the neighborhood set construction needs $O(nN)$ flops, and one call of singular value decomposition takes $O(nL \min\{n, L\})$ flops (Gentle (2007)), which is $O(n^2L)$ since $n < L$ is assumed. The estimation of $\hat{\sigma}^2$ and $\hat{\Theta}$ based on n neighborhood points requires $O(nL)$ and $O(n^3)$ flops, respectively. That is, the cost of the empirical Bayesian inference for SVD-based GP models based on n neighborhood points is $O(n^2L + nL + n^3) = O(n^2L)$. Furthermore, the computational complexity of the prediction step is $O(n^2 + L)$. Consequently, the total cost of knnsvdGP algorithm is $O(Mn \max\{nL, N\})$.

(3) **lasvdGP**: The cost of empirical Bayesian inference for SVD-based GP models based on k neighborhood points is $O(k^2L)$, as in knnsvdGP, $n_0 \leq k \leq n$. Optimization of the J -criterion costs $O(k^2N)$ (as per the quick update formula by Gramacy and Apley (2015)). Thus the cost of building a local approximate SVD-based GP model at the k -th iteration is $O(k^2N)$, $k = n_0, \dots, n - 1$, and thus the entire process of fitting a local approximate SVD-based GP model requires $\sum_{k=n_0}^{n-1} O(k^2N) = O(n^3N)$ flops. Note that the prediction cost in lasvdGP is not significant compared to the neighborhood selection and inference of the GP models. As a result, the total cost of this algorithm is $O(n^3NM)$.

Table 1 summarizes the computational complexity of the three methods.

Table 1: The computational cost of fitting GP models under the three methods.

Method	svdGP	knnsvdGP	lasvdGP
Cost	$O(N^2 \max\{M, N\})$	$O(Mn \max\{nL, N\})$	$O(n^3 NM)$

It is easy to see that lasvdGP is computationally more expensive than knnsvdGP, however, the gain in the prediction accuracy is perhaps worth more. Assuming $M = O(N)$, it is also straightforward to notice that knnsvdGP and lasvdGP are substantially faster than svdGP (full model) as long as $n = O(N^{1/3})$.

3.3. Implementation

Both local SVD-based GP models (knnsvdGP and lasvdGP) are run in a parallel computing environment using the *R* package **parallel** (R Core Team (2017)). One quick option is to divide the job into M parts and fit independent local GP models. In contrast, svdGP models cannot be parallelized in such an easy manner, except the prediction component. Of course, one could use parallelization for SVD of \mathbf{Y} , and/or computing the determinant and inverse of the correlation matrices within the optimization step.

We implemented the three methods in *R* (R Core Team (2017)). The parallelization of the empirical Bayesian estimation and the prediction at M untried inputs are implemented via the package *parallel*. The optimization in empirical Bayesian inference for all the three methods is performed with the assistance of the **laGP** package with default priors (Gramacy (2016)). We shall also mention that in searching for the best follow-up point \mathbf{x}_{k+1}^* from the candidate set $\mathbf{X} \setminus \mathbf{X}^{(k)}$, we adopt the limit search scheme suggested by (Gramacy (2016)) instead of the exhaustive search. This allows us to save tremendous computational time without sacrificing prediction accuracy, as indicated by our empirical studies.

Fitting GP models to a large number of observations in low input dimension can often run into numerical instability due to near-singularity, and typically a small nugget is used in the correlation structure to address this numerical issue (e.g., Ranjan et al. (2011); Gramacy and Lee (2012); Peng and Wu (2014)). Similar to Gramacy and Apley (2015), we fix the nugget η at a pre-determined small value to avoid near-singularity issue of the correlation matrix frequently emerged in the Gaussian correlation family (Gu et al., 2017).

4. Applications

In this section, we consider two examples with different test functions that represent dynamic computer models. We also consider a real-life application where the computer simulator (TDB model) generates population growth curve. The complexity of the examples considered here range from $N = 10,000$ to $30,000$ (size of the training set), and $q = 3$ to 11 (input dimension).

The performance of the three methods svdGP, knmsvdGP and lasvdGP is evaluated by comparing the normalized mean squared prediction error (NMSPE),

$$\text{NMSPE}(\mathbf{x}) = \frac{\sum_{t=1}^L (y_t(\mathbf{x}) - \hat{y}_t(\mathbf{x}))^2}{\sum_{t=1}^L (y_t(\mathbf{x}) - \bar{y}(\mathbf{x}))^2}, \quad (13)$$

and the proper scoring rule (Gneiting and Raftery (2007)) defined as

$$S(P_{\hat{\mathbf{y}}(\mathbf{x})}, \mathbf{y}(\mathbf{x})) = -\frac{1}{L} \sum_{t=1}^L \frac{(y_t(\mathbf{x}) - \hat{y}_t(\mathbf{x}))^2}{\hat{\sigma}_t^2(\mathbf{x})} - \frac{1}{L} \sum_{t=1}^L \log \hat{\sigma}_t^2(\mathbf{x}), \quad (14)$$

where $P_{\hat{\mathbf{y}}(\mathbf{x})}$ is the predictive distribution of the response at \mathbf{x} , $\mathbf{y}(\mathbf{x}) = [y_1(\mathbf{x}), \dots, y_L(\mathbf{x})]^T$ is the (typically unknown) true response time-series at \mathbf{x} , $\hat{y}_t(\mathbf{x})$ is the corresponding predicted mean response, and $\hat{\sigma}_t^2(\mathbf{x})$ is the associated variance given by the t th diagonal entry of $\mathbf{B}\mathbf{\Lambda}(\mathbf{V}^*, \hat{\boldsymbol{\Theta}})\mathbf{B}^T + \hat{\sigma}^2\mathbf{I}_L$. Furthermore, the temporal mean is given by $\bar{y}(\mathbf{x}) = \sum_{t=1}^L y_t(\mathbf{x})/L$. As model ranking criteria, the objective is to minimize average NMSPE and maximize the mean proper scoring rule.

For all these methods, we use the default priors of the *R* package **laGP**, i.e., the vague scale-invariant priors (Gramacy (2005)) with α_i 's, β_i 's, α and β set to be 0, and for the correlation parameters $\boldsymbol{\theta}_i$'s, the priors are explained at the beginning of Section 2.2. We adopt zero-mean function in all GP models, apply the models to the normalized outputs that have zero mean and add the mean back for prediction. For the simulated test functions, Examples 1 and 2, we repeat the emulation procedure 50 times with different (randomly chosen) training and test data sets and compare the average performance. For the real application in Example 3, we used Monte Carlo cross-validation approach for quantifying uncertainty in the prediction process (Shao (1993)).

4.1. Example 1 (Forrester et al. (2008))

Consider the following test function with 3-dimensional inputs to generate simulator responses with time-series outputs,

$$f(\mathbf{x}, t) = (x_1 t - 2)^2 \sin(x_2 t - x_3), \quad (15)$$

where $\mathbf{x} = (x_1, x_2, x_3)^T \in [4, 10] \times [4, 20] \times [1, 7]$, and $t \in [1, 2]$ is on a 200-point equidistant time-grid.

For each of 50 replications, we randomly generate the training data of size 10,000 and the test data of size 2,000 using random Latin hypercube designs (LHDs) (McKay et al. (1979)) from the input space $[4, 10] \times [4, 20] \times [1, 7]$. The local approximate methods are implemented on the neighborhood sets of size $n = 20$ and 40 points. For lasvdGP, we assume the initial neighborhood size to be $n_0 = \lceil n/4 \rceil$, and $\lceil n/2 \rceil$, where $\lceil x \rceil$ represents the smallest integer greater than or equal to x . Figures 1 and 2 summarize the log of mean NMSPE and mean proper scoring rule values, respectively, for different models. Notation: lasvdGP_ n_0 denotes that the proposed method uses n_0 points in the initial neighborhood set chosen as nearest points based on Euclidean distance, and the remaining $n - n_0$ points are chosen sequentially by optimising the J -criterion.

Figures 1 and 2 reveal that the proposed algorithm outperforms its naive counterpart irrespective of the total neighborhood size (n). As the neighborhood size gets large, the prediction accuracy of the local approximation algorithms improves.

For a fixed data set of size 10,000, we also computed Monte Carlo cross-validation based values for the two measures, log of mean NMSPE and mean proper scoring rule, and compared the three models. We considered one-fifth of the data as the test set and the remaining as the training set. The boxplots of the two measures over 50 random splits of the data show the similar trend as in Figures 1 and 2.

4.2. Example 2 (Bliznyuk et al. (2008))

Consider the environmental model in Bliznyuk et al. (2008) which models a pollutant spill caused by a chemical accident. The simulator output is given by

$$f(\mathbf{x}, t) = \frac{M}{\sqrt{Dt}} \exp\left(\frac{-s^2}{4Dt}\right) + \frac{M}{\sqrt{D(t-\tau)}} \exp\left(-\frac{(s-L)^2}{4D(t-\tau)}\right) I(\tau < t), \quad (16)$$

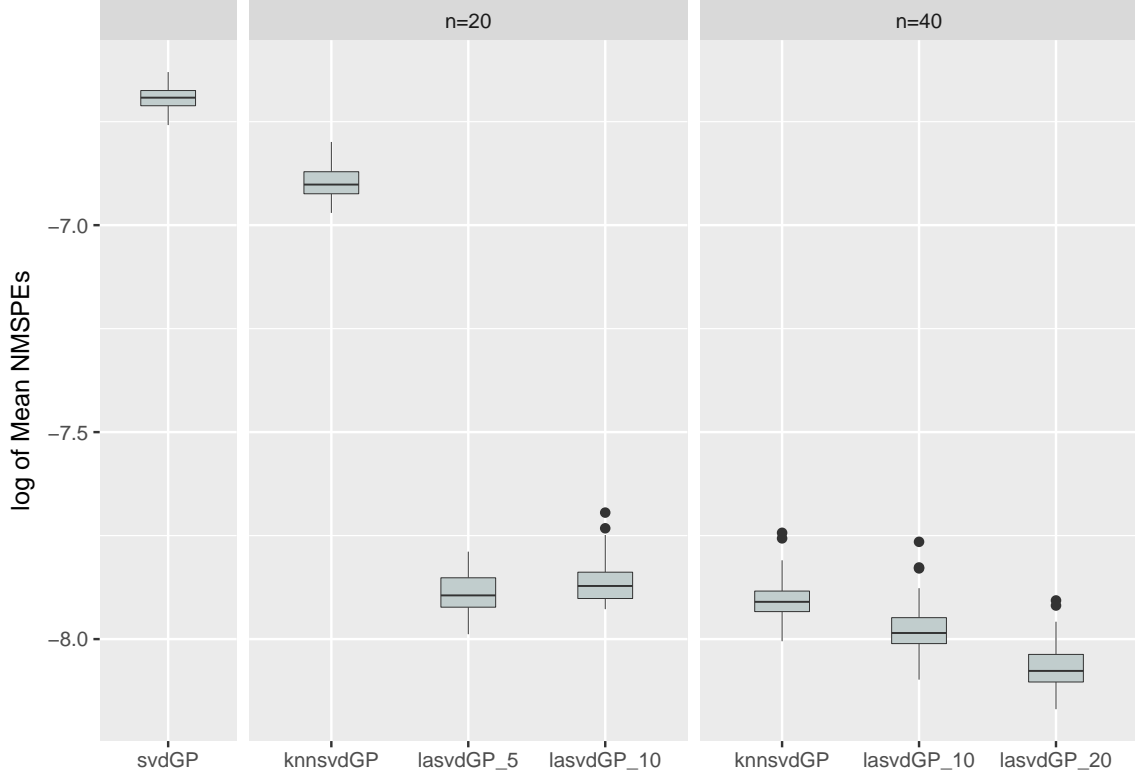


Figure 1: The boxplots of the log of mean NMSPE computed from 2,000 test points over 50 simulations for the computer simulator (15). The proposed lasvdGP approach achieves much smaller log of mean NMSPE values than the competitors.

where $\mathbf{x} = (M, D, L, \tau, s)^T$, M denotes the mass of pollutant spilled at each location, D is diffusion rate in the channel, L is location of the second spill, τ is time of the second spill, $\mathbf{x} \in [7, 13] \times [0.02, 0.12] \times [0.01, 3] \times [30.01, 30.295] \times [0, 3]$, and $t \in [0.3, 60]$ is on a regular 200-point equidistant time grid.

In this example as well, we use the training data of size $N = 10,000$ and the test data of size $M = 2,000$ obtained using a random LHD. Similar to the previous example, Figures 3 and 4 display the boxplots of 50 log of mean NMSPEs and mean proper scoring rule values, respectively, computed over the test set.

Focussing on the local GP models, Figures 3 and 4 demonstrate that the proposed approach (lasvdGP) is more accurate than the naive one (knnsvdGP), and $n = 50$ exhibits more accurate prediction than $n = 30$. As in the previous example, the Monte Carlo cross-

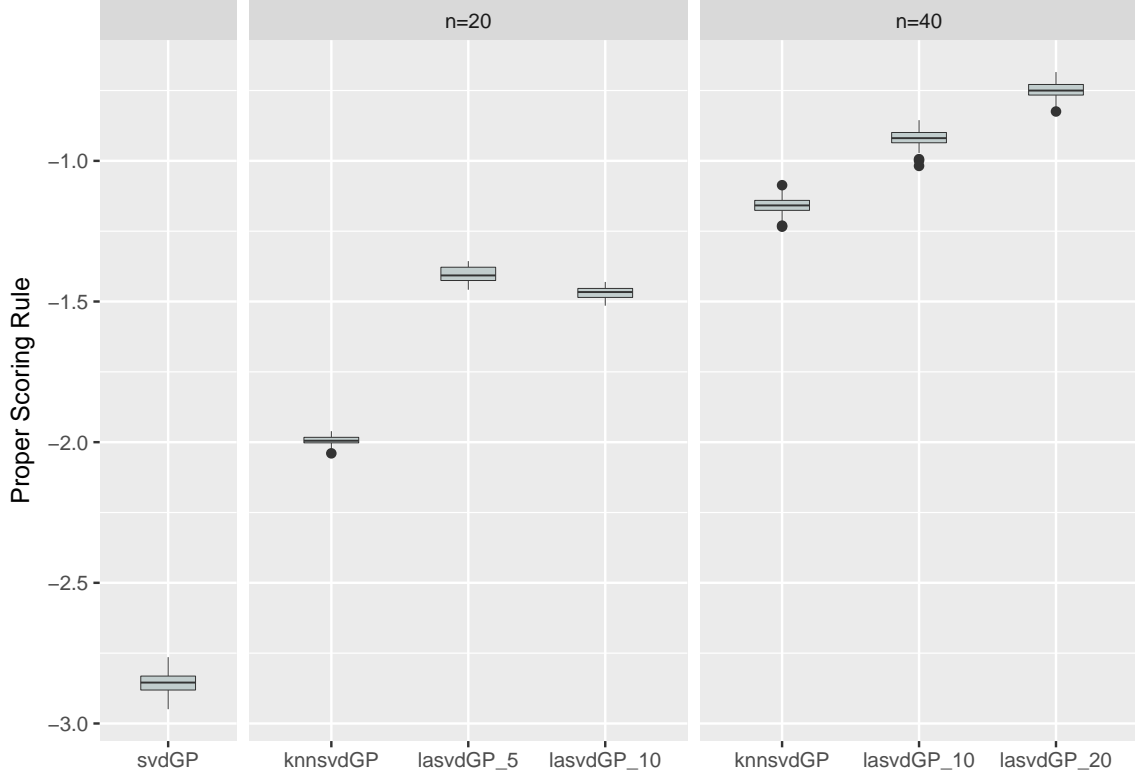


Figure 2: The boxplots of the mean proper scoring rule computed from 2,000 test points over 50 simulations for the computer simulator (15). The proposed lasvdGP approach achieves higher values of mean proper scoring rule than the competitors.

validation approach shows consistent findings. To investigate this further, we compared the prediction accuracy of lasvdGP for different n and $n_0 = \lceil n/2 \rceil$, Figure 5 summarizes the findings.

Figure 5 shows the expected increasing trend of the average prediction accuracy. Though the prediction accuracy increases with n , the rate of increment in the accuracy slows down as n increases, and more importantly, note that fitting a lasvdGP model, requires $O(n^3NM)$ flops, which becomes prohibitively large very quickly.

4.3. Example 3 (TDB simulator - Teismann et al. (2009))

The two-delay blowfly (TDB) model (Teismann et al. (2009)) simulates European red mites (ERM) population dynamics under predator-prey interactions in apple orchards via

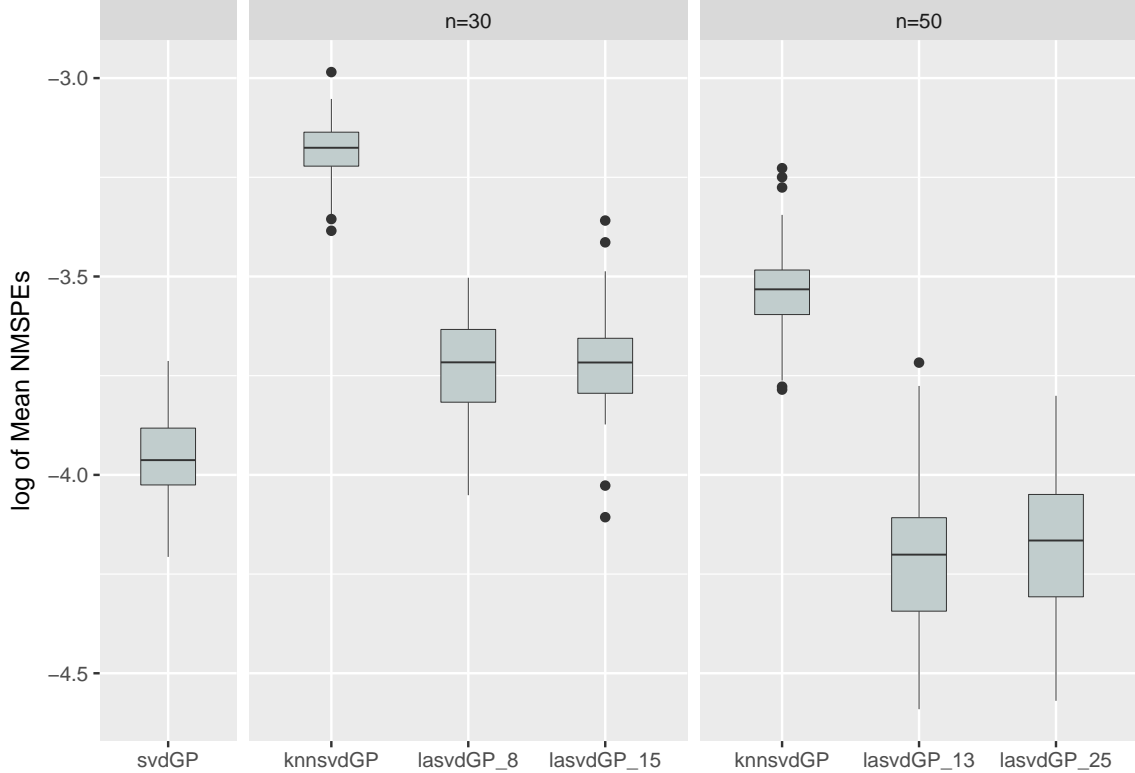


Figure 3: The boxplots of the log of mean NMSPEs computed from 2,000 test points over 50 simulations for the simulator given by (16). The proposed lasvdGP approach achieves much smaller log of mean NMSPE values than the competitors.

numerically solving the Nicholson’s blowfly differential equation (Gurney et al. (1980)). Unmanaged ERM population growth could incur massive infestation which inflicts heavy loss in apple industry. Therefore, the monitoring and subsequent intervention of ERM population dynamics is of vital importance for apple orchards management. The objective here is to emulate this simulator for deeper insight in the process.

The TDB model takes eleven input variables (e.g., death rates for different stages, fecundity, hatching time, survival rates, and so on) and returns the time series (at 28 time points) of ERM population evolutions at three stages, i.e., eggs, juveniles and adults (see Ranjan et al. (2016) for details). In this paper, we focus on the population dynamics of juveniles. Figure 6 shows the model output at five randomly chosen input points.

The input variable domains are decided by expert knowledge. For convenience, we

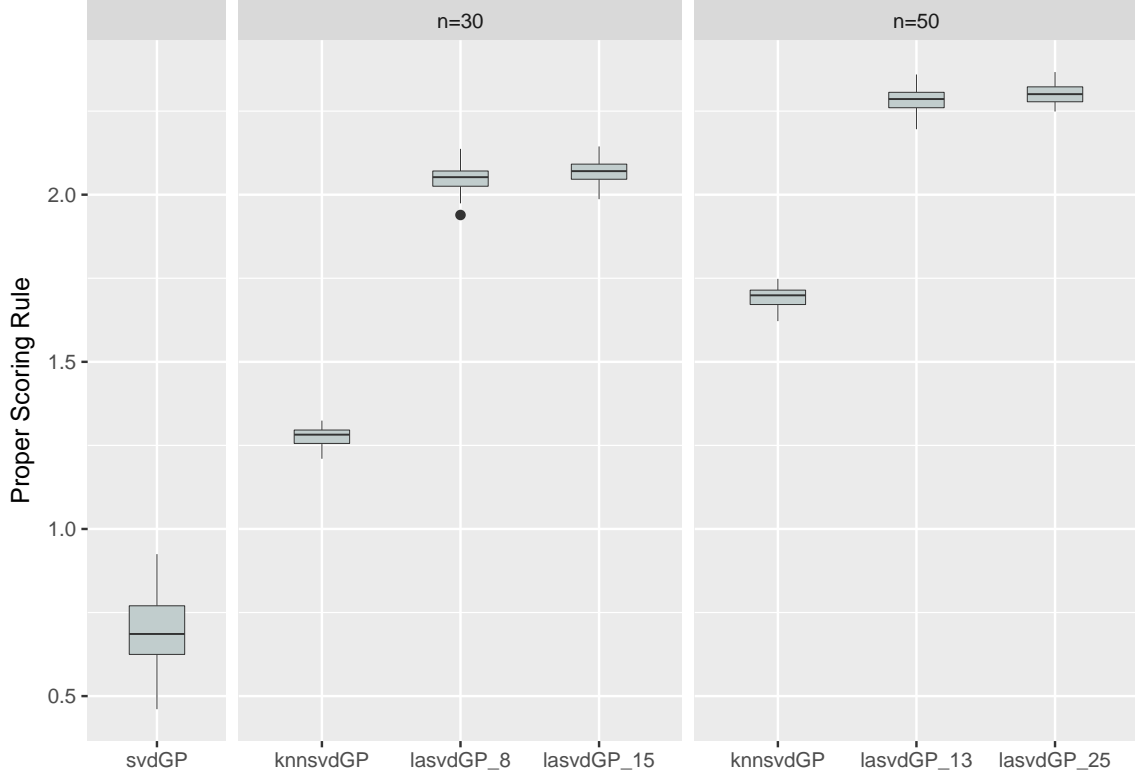


Figure 4: The boxplots of mean proper scoring rule computed from 2,000 test points over 50 simulations for the simulator given by (16). The proposed lasvdGP approach achieves higher values of mean proper scoring rule than the competitors.

transform the inputs into 11-dimensional unit hypercube. Given that we have a limited (data) budget from the simulator, we rely on the Monte Carlo cross-validation error alone. We had access to a data set of size 30,000 for the emulation and prediction accuracy measurements. For such a large scale dynamic computer model, svdGP is computationally infeasible. We used $n = 80$ and $n_0 = \lceil n/2 \rceil$ for the proposed local SVD-based GP models. For each method, the total data was partitioned into training and test set in 4:1 ratio, and then the prediction accuracy measures were computed on the test set. Figure 7 shows the boxplots of the log of mean NMSPEs and mean proper scoring rule values over 50 randomly chosen Monte Carlo partitions.

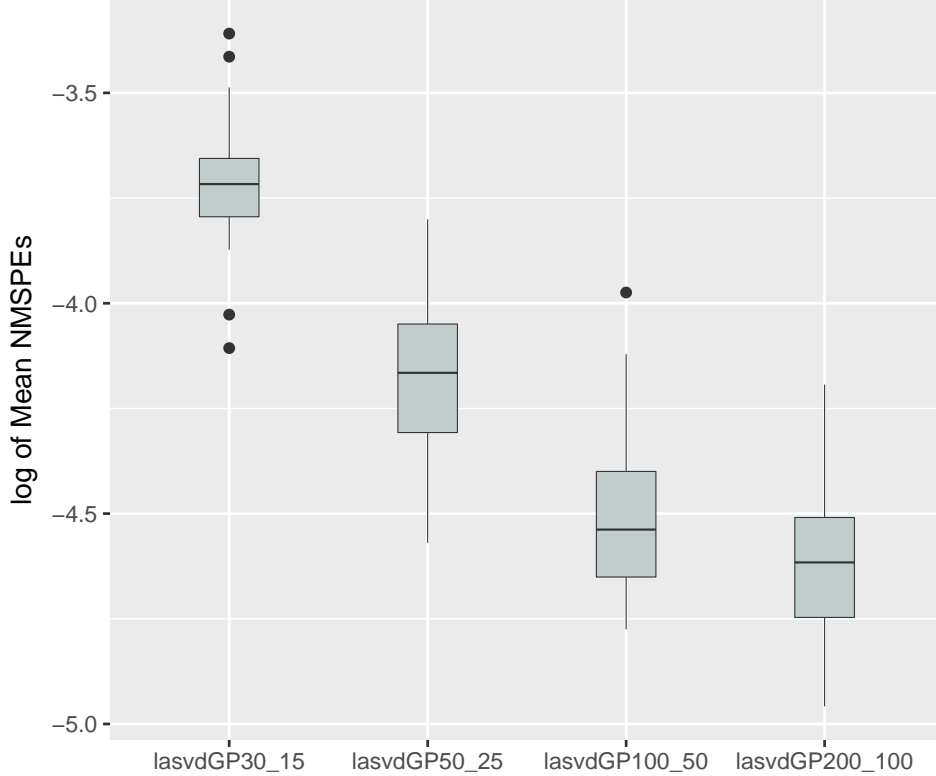


Figure 5: The boxplots of the log of mean NMSPEs computed using 2,000 test points over 50 simulations with $n = 30, 50, 100, 200$ and $n_0 = n/2$ for model (16). As the neighborhood size increases, the log of mean NMSPE values by the proposed lasvdGP approach decrease, resulting in more accurate prediction.

5. Concluding Remarks

We have proposed local approximate SVD-based GP models for large-scale dynamic computer experiments. The proposed local SVD-based GP models with the proposed neighborhood selection algorithm reduce the time complexity of the full SVD-based GP models. Though slightly more time consuming than its naive counterpart, lasvdGP has been shown to be much more accurate in prediction for both simulation examples and the real data analysis. With the assistance of parallel computation, the proposed algorithm can easily handle dynamic computer experiments with training set as large as (approx) 25,000 points, which is beyond the capacity of the full model.

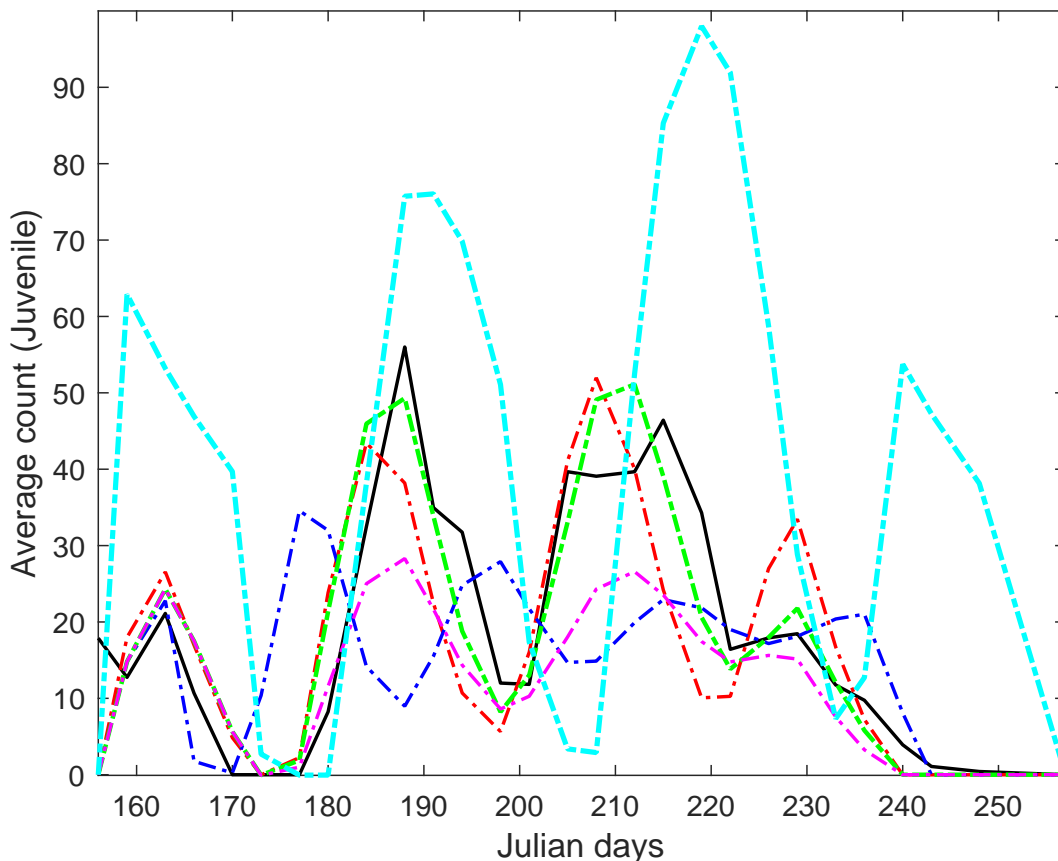


Figure 6: Juvenile ERM population dynamics as outputs of the TDB model at five different inputs. The solid curve shows the field data, and the dashed curves show the TDB outputs.

There are a few remarks worth mentioning. First, in this article, we refer to large-scale dynamic computer experiments as those with a large number of inputs. This is different from the large data aspect in Gu et al. (2016) where the spatial-temporal applications with small run sizes (in the order of hundreds) but large numbers of time points (in the order of tens of thousands) were considered. In their application, fitting full SVD-based GP models are still computationally feasible, as the number of significant singular values might be large but the number of inputs (for $n \times n$ correlation matrix factorization) would be small.

Second, the formula (10) does not consider the possible update of the estimated correlation parameters. If new data arrives, the empirical Bayesian estimators of correlation parameters θ 's in (8) are expected to change with the training set. To address this issue,

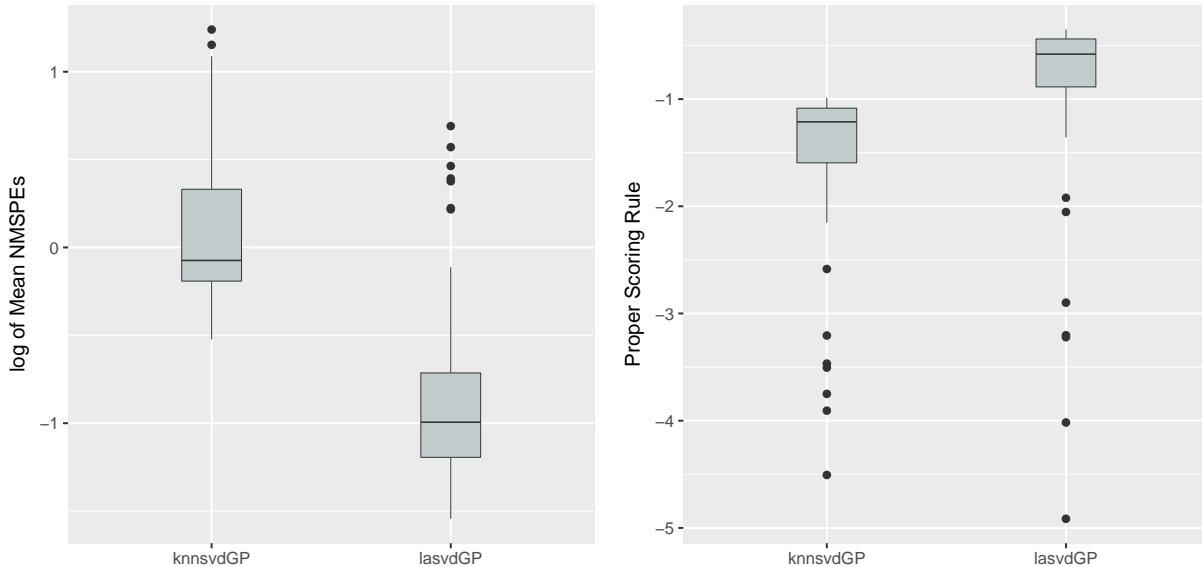


Figure 7: The boxplots of the log of mean NMSPEs (left) and mean proper scoring rule (right) for the TDB application obtained via Monte Carlo cross-validation. The proposed lasvdGP approach outperforms the knnsvdGP approach in terms of both log of mean NMSPEs and the mean proper scoring rule.

Gramacy and Apley (2015) suggested the second order Taylor polynomial approximation.

Third, there are possible improvements in terms of computational efficiency. In searching for neighborhood set, it has been suggested to consider more sophisticated searches such as using graphical processing units (GPUs) and approximating discrete neighborhood searches via continuous ones along the rays emanating from each predictive point (Franeý et al. (2012); Gramacy and Haaland (2016)). Another way to boost the computational efficiency is that instead of searching the neighborhood set for each individual point in the prediction set \mathbf{X}^* , some clustering algorithms could be performed on \mathbf{X}^* to divide it into groups on which the proposed neighborhood selection is executed. These are interesting topics for our future research.

Acknowledgement

We would like to the Editor, the AE and the two referees for their valuable comments and suggestions that led to significant improvements in the article. Ranjan’s research was

supported by the Extra Mural Research Funding (EMR/2016/003332/MS) from the Science and Engineering Research Board, Department of Science and Technology, Government of India. Lin’s research was supported by the Discovery grant from Natural Sciences and Engineering Research Council of Canada. We also thank Dr. Holger Teismann for providing the field data and outputs for the TDB model.

Supplementary Materials

The supplementary material includes the following:

Appendix: Section A contains the proof of Proposition 1. Section B presents two algorithms (in the formal algorithm format) for fitting local approximate SVD-based GP models (lasvdGP) described in Sections 2 and 3 of this article. Section C summarizes simulation results for establishing the reliability of the estimated range parameters (or equivalently, the correlation parameters) for the proposed lasvdGP model fits in Examples 1 and 2. (Appendix.pdf, PDF file)

Code: R codes to reproduce results in the article are available in the zip file. Details can be found in the readme.txt file included. (code.zip, zipped folder)

REFERENCES

- Bayarri, M., J. Berger, J. Cafeo, G. Garcia-Donato, F. Liu, J. Palomo, R. Parthasarathy, R. Paulo, J. Sacks, and D. Walsh (2007). Computer model validation with functional output. *The Annals of Statistics* 35(5), 1874–1906.
- Bingham, D., P. Ranjan, and W. J. Welch (2014). Statistics in action: A canadian outlook. *Sequential design of computer experiments for optimization, estimating contours, and related objectives*, 109–124.
- Bliznyuk, N., D. Ruppert, C. Shoemaker, R. Regis, S. Wild, and P. Mugunthan (2008). Bayesian calibration and uncertainty analysis for computationally expensive models using optimization and radial basis function approximation. *Journal of Computational and Graphical Statistics* 17(2), 270–294.

- Cohn, D. A. (1996). Neural network exploration using optimal experiment design. *Neural networks* 9(6), 1071–1083.
- Cohn, D. A., Z. Ghahramani, and M. I. Jordan (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research* 4, 129–145.
- Conti, S., J. P. Gosling, J. E. Oakley, and A. O’Hagan (2009). Gaussian process emulation of dynamic computer codes. *Biometrika* 96(3), 663–676.
- Conti, S. and A. O’Hagan (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of statistical planning and inference* 140(3), 640–651.
- Emery, X. (2009). The kriging update equations and their application to the selection of neighboring data. *Computational Geosciences* 13(3), 269–280.
- Farah, M., P. Birrell, S. Conti, and D. D. Angelis (2014). Bayesian emulation and calibration of a dynamic epidemic model for A/H1N1 influenza. *Journal of the American Statistical Association* 109(508), 1398–1411.
- Forrester, A., A. Sobester, and A. Keane (2008). *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons.
- Franey, M., P. Ranjan, and H. Chipman (2012). A short note on Gaussian process modeling for large datasets using graphics processing units. *arXiv:1203.1269*.
- Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (2014). *Bayesian data analysis*, Volume 2. Chapman & Hall/CRC Boca Raton, FL, USA.
- Gentle, J. E. (2007). *Matrix algebra: theory, computations, and applications in statistics*. Springer Science & Business Media: New York.
- Gneiting, T. and A. E. Raftery (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association* 102(477), 359–378.
- Gramacy, R. B. (2005). *Bayesian treed Gaussian process models*. Ph. D. thesis, University of California Santa Cruz.

- Gramacy, R. B. (2016). laGP: Large-scale spatial modeling via local approximate Gaussian processes in R. *Journal of Statistical Software* 72(1), 1–46.
- Gramacy, R. B. and D. W. Apley (2015). Local Gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics* 24(2), 561–578.
- Gramacy, R. B. and B. Haaland (2016). Speeding up neighborhood search in local Gaussian process prediction. *Technometrics* 58(3), 294–303.
- Gramacy, R. B. and H. K. Lee (2012). Cases for the nugget in modeling computer experiments. *Statistics and Computing* 22(3), 713–722.
- Gu, M., J. O. Berger, et al. (2016). Parallel partial Gaussian process emulation for computer models with massive output. *The Annals of Applied Statistics* 10(3), 1317–1347.
- Gu, M., X. Wang, and J. O. Berger (2017). Robust Gaussian stochastic process emulation. *arXiv:1708.04738*.
- Gurney, W., S. Blythe, and R. Nisbet (1980). Nicholson’s blowflies revisited. *Nature* 287, 17–21.
- Hager, W. W. (1989). Updating the inverse of a matrix. *SIAM review* 31(2), 221–239.
- Higdon, D., J. Gattiker, B. Williams, and M. Rightley (2008). Computer model calibration using high-dimensional output. *Journal of the American Statistical Association* 103(482), 570–583.
- Hung, Y., V. R. Joseph, and S. N. Melkote (2015). Analysis of computer experiments with functional response. *Technometrics* 57(1), 35–44.
- Jones, D. R., M. Schonlau, and W. J. Welch (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization* 13(4), 455–492.
- Kennedy, M. C. and A. O’Hagan (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B* 63(3), 425–464.

- Liu, F. and M. West (2009). A dynamic modelling strategy for Bayesian computer model emulation. *Bayesian Analysis* 4(2), 393–411.
- McKay, M. D., R. J. Beckman, and W. J. Conover (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 42(1), 55–61.
- Peng, C.-Y. and C. J. Wu (2014). On the choice of nugget in kriging modeling for deterministic computer experiments. *Journal of Computational and Graphical Statistics* 23(1), 151–168.
- R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.
- Ranjan, P., D. Bingham, and G. Michailidis (2008). Sequential experiment design for contour estimation from complex computer codes. *Technometrics* 50(4), 527–541.
- Ranjan, P., R. Haynes, and R. Karsten (2011). A computationally stable approach to Gaussian process interpolation of deterministic computer simulation data. *Technometrics* 53(4), 366–378.
- Ranjan, P., M. Thomas, H. Teismann, and S. Mukhoti (2016). Inverse problem for a time-series valued computer simulator via scalarization. *Open Journal of Statistics* 6(03), 528–544.
- Rasmussen, C. E. and C. K. I. Williams (2006). *Gaussian processes for machine learning*. The MIT Press.
- Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn (1989). Design and analysis of computer experiments. *Statistical Science* 4(4), 409–423.
- Santner, T. J., B. J. Williams, and W. I. Notz (2003). *The design and analysis of computer experiments*. Springer Science & Business Media: New York.
- Shao, J. (1993). Linear model selection by cross-validation. *Journal of the American statistical Association* 88(422), 486–494.

Stein, M. L. (2005). Space–time covariance functions. *Journal of the American Statistical Association* 100(469), 310–321.

Teismann, H., R. Karsten, R. Hammond, J. Hardman, and J. Franklin (2009). On the possibility of counter-productive intervention: the population mean for blowflies models can be an increasing function of the death rate. *Journal of Biological Systems* 17(04), 739–757.

Supplementary Materials

A. Proof Of Proposition 1

Following (7), the inner expectation in (10) can be written as

$$\begin{aligned}
& \mathbb{E} \left[\left\| \mathbf{y}(\mathbf{x}_0) - \hat{\mathbf{y}}(\mathbf{x}_0 | \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}) \right\|^2 \middle| \mathbf{c}(\mathbf{x}), \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \\
&= \text{tr} \left(\mathbf{B}^{(k)} \mathbf{\Lambda}(\mathbf{V}^{*(k)}(\mathbf{x}), \hat{\Theta}^{(k)}) (\mathbf{B}^{(k)})^T + (\hat{\sigma}^{(k)})^2 \mathbf{I}_L \right) \\
&= (\hat{\sigma}^{(k)})^2 L + \text{tr} \left(\mathbf{\Lambda}(\mathbf{V}^{*(k)}(\mathbf{x}), \hat{\Theta}^{(k)}) (\mathbf{B}^{(k)})^T \mathbf{B}^{(k)} \right) \\
&= (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{v}_i^{(k)}(\mathbf{x}), \hat{\theta}_i^{(k)}),
\end{aligned} \tag{A.17}$$

where $\mathbf{V}^{*(k)}(\mathbf{x}) = [(\mathbf{V}^{*(k)})^T, \mathbf{c}(\mathbf{x})]^T$ and $d_i^{(k)}$ is the i th largest singular value of $\mathbf{Y}^{(k)}$,

$$\mathbf{\Lambda}(\mathbf{V}^{*(k)}(\mathbf{x}), \hat{\Theta}^{(k)}) = \text{diag} \left(\hat{\sigma}_1^2(\mathbf{x}_0 | \mathbf{v}_1^{(k)}(\mathbf{x}), \hat{\theta}_1^{(k)}), \dots, \hat{\sigma}_{p_k}^2(\mathbf{x}_0 | \mathbf{v}_{p_k}^{(k)}(\mathbf{x}), \hat{\theta}_{p_k}^{(k)}) \right),$$

and

$$\hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{v}_i^{(k)}(\mathbf{x}), \hat{\theta}_i^{(k)}) = \frac{\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})}{\alpha_i + k} (\beta_i + \psi_i^{(k)}(\mathbf{x})),$$

for $i = 1, \dots, p_k$, where $\psi_i^{(k)}(\mathbf{x}) = \mathbf{v}_i^{(k)}(\mathbf{x})^T \tilde{\mathbf{K}}_i^{-1}(\mathbf{x}) \mathbf{v}_i^{(k)}(\mathbf{x})$, and $\mathbf{v}_i^{(k)}(\mathbf{x}) = [(\mathbf{v}_i^{(k)})^T, c_i(\mathbf{x})]^T$ is the i th column of $\mathbf{V}^{*(k)}(\mathbf{x})$.

The first equality of (A.17) follows from Theorem 3.2b.1 of Mathai and Provost (1992). The third equality is derived from the column-orthogonality of $\mathbf{B}^{(k)}$, i.e. $(\mathbf{B}^{(k)})^T \mathbf{B}^{(k)} = (\mathbf{D}^{*(k)})^2$. Plugging (A.17) into (10), we get

$$\begin{aligned}
J(\mathbf{x}_0, \mathbf{x}) &= \mathbb{E} \left[(\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{v}_i^{(k)}(\mathbf{x}), \hat{\theta}_i^{(k)}) \middle| \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2 \right] \\
&= (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \left(\frac{\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})}{\alpha_i + k} (\beta_i + \mathbb{E}[\psi_i^{(k)}(\mathbf{x}) | \mathbf{V}^{*(k)}, \hat{\Theta}^{(k)}, (\hat{\sigma}^{(k)})^2]) \right) \\
&= (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \left(\frac{\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})}{\alpha_i + k} (\beta_i + \mathbb{E}[\psi_i^{(k)}(\mathbf{x}) | \mathbf{v}_i^{(k)}, \hat{\theta}_i^{(k)}]) \right) \\
&= (\hat{\sigma}^{(k)})^2 L + \sum_{i=1}^{p_k} (d_i^{(k)})^2 \left(\frac{\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})}{\alpha_i + k} (\beta_i + \frac{\alpha_i + k}{\alpha_i + k - 1} \psi_i^{(k)}) \right).
\end{aligned}$$

The second equality holds because $\rho_i^{(k)}(\mathbf{x}_0, \mathbf{x})$ is a deterministic function of \mathbf{x}_0, \mathbf{x} and $\hat{\theta}_i^{(k)}$. The third equality follows from the independence among c_i 's. The validity of the fourth equality is due to Gramacy and Apley (2015).

B. Algorithms

Algorithm 1 summarizes the key steps required for estimating the necessary parameters in the posterior predictive distribution (Equation (9) of the main article) of a full SVD-based GP model fitted to a training data of size N .

Algorithm 1: SVD-based GP model

Input : (1) Training set: $\mathbf{X}_{N \times q}$, (2) response matrix: $\mathbf{Y}_{L \times N}$, (3) threshold γ ,

(4) prior parameters: $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_p, \alpha]^T$, $\boldsymbol{\beta} = [\beta_1, \dots, \beta_p, \beta]^T$.

Output: (1) Basis $\mathbf{B}_{N \times p}$, (2) singular values $\mathbf{D}_{p \times p}^*$, (3) coefficients \mathbf{V}^* ,

(4) correlation parameters $\hat{\boldsymbol{\Theta}}$, (5) variance $\hat{\sigma}^2$.

1 **Function** svdGP($\mathbf{X}, \mathbf{Y}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma$)

2 $[\mathbf{B}, \mathbf{D}^*, \mathbf{V}^*, p] \leftarrow \text{buildBasis}(\mathbf{Y}, \gamma)$

3 $\mathbf{r} \leftarrow \text{vec}(\mathbf{Y}) - (\mathbf{I}_N \otimes \mathbf{B})\text{vec}(\mathbf{V}^{*T})$

4 $\hat{\sigma}_i^2(\mathbf{x}_0 | \mathbf{v}_i, \boldsymbol{\theta}_i) = (\beta_i + \psi_i) \left(1 - \mathbf{k}_i^T(\mathbf{x}_0) \mathbf{K}_i^{-1} \mathbf{k}_i(\mathbf{x}_0) \right) / (\alpha_i + N),$

$\hat{\sigma}^2 \leftarrow (\mathbf{r}^T \mathbf{r} + \beta) / (NL + \alpha + 2)$

5 $\hat{\boldsymbol{\Theta}} \leftarrow \text{inference}(\mathbf{V}^*, p, \boldsymbol{\alpha}, \boldsymbol{\beta})$

6 **return** $\mathbf{B}, \mathbf{D}^*, \mathbf{V}^*, \hat{\boldsymbol{\Theta}}, \hat{\sigma}^2$

7 **Subroutine** buildBasis(\mathbf{Y}, γ)

8 $[\mathbf{U}, \mathbf{D}, \mathbf{V}] \leftarrow \text{SVD}(\mathbf{Y})$ /* perform SVD on matrix \mathbf{Y} . */

9 $p \leftarrow \min \left\{ m : \frac{\sum_{i=1}^m d_i}{\sum_{i=1}^k d_i} > \gamma \right\}$ /* where $\mathbf{D} = \text{diag}(d_1, \dots, d_N), k = \min\{N, L\}$ */

10 $\mathbf{B} \leftarrow \mathbf{U}^* \mathbf{D}^*$ /* as in Section 2.1 */

11 **return** $\mathbf{B}, \mathbf{D}^*, \mathbf{V}^*, p$

12 **Subroutine** inference($\mathbf{V}^*, p, \boldsymbol{\alpha}, \boldsymbol{\beta}$)

13 **for** $i \leftarrow 1$ **to** p **do**

14 $\hat{\boldsymbol{\theta}}_i \leftarrow \underset{\boldsymbol{\theta}_i}{\text{argmax}} \pi(\boldsymbol{\theta}_i | \mathbf{v}_i)$ /* fit p independent GPs by finding the MAPs */

15 **return** $\hat{\boldsymbol{\Theta}} = [\hat{\boldsymbol{\theta}}_1, \dots, \hat{\boldsymbol{\theta}}_p]^T$

Algorithm 2 presents the steps required for fitting the proposed local approximate SVD-based GP model (lasvdGP) with the neighbourhood points selected using the J -criterion in Section 3.1 of the main article.

Algorithm 2: Proposed local SVD-based GP model

Input : (1) Training set: $\mathbf{X}_{N \times q}$, (2) response matrix: $\mathbf{Y}_{L \times N}$, (3) test set $\mathbf{X}_{M \times q}^*$,
(4) neighborhood size n , (5) initial neighborhood size n_0 , (6) threshold γ ,
(7) prior parameters $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_p, \alpha]^T$ and $\boldsymbol{\beta} = [\beta_1, \dots, \beta_p, \beta]^T$.

Output: (1) The predicted mean response, and (2) the associated posterior variance in estimating $\mathbf{y}(\mathbf{x}_0)$ for each $\mathbf{x}_0 \in \mathbf{X}^*$.

```

1 for each  $\mathbf{x}_0 \in \mathbf{X}^*$  do
2    $\mathbf{X}^{(n_0)} \leftarrow \{\mathbf{x}_i, i = 1, \dots, n_0\}$  /*  $n_0$  nearest neighbours of  $\mathbf{x}_0$  in  $\mathbf{X}$  as in knn */
3    $\mathbf{Y}^{(n_0)} \leftarrow \{y(\mathbf{x}) : \mathbf{x} \in \mathbf{X}^{(n_0)}\}$ 
4   for  $k \leftarrow n_0$  to  $n - 1$  do
5      $[\mathbf{B}^{(k)}, \mathbf{D}^{*(k)}, \mathbf{V}^{*(k)}, p_k, \hat{\boldsymbol{\Theta}}^k, (\hat{\sigma}^{(k)})^2, (\hat{\boldsymbol{\sigma}}^{(k)})^2] \leftarrow \text{svdGP}(\mathbf{X}^{(k)}, \mathbf{Y}^{(k)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma)$ 
6      $\mathbf{x}_{k+1}^* \leftarrow \underset{\mathbf{x} \in \mathbf{X} \setminus \mathbf{X}^{(k)}}{\text{argmin}} J(\mathbf{x}_0, \mathbf{x})$ 
7      $\mathbf{X}^{(k+1)} \leftarrow \mathbf{X}^{(k)} \cup \mathbf{x}_{k+1}^*$ 
8      $\mathbf{Y}^{(k+1)} \leftarrow \mathbf{Y}^{(k)} \cup \mathbf{y}(\mathbf{x}_{k+1}^*)$ 
9    $[\mathbf{B}^{(n)}, \mathbf{D}^{*(n)}, \mathbf{V}^{*(n)}, p_n, \hat{\boldsymbol{\Theta}}^{(n)}, (\hat{\sigma}^{(n)})^2, (\hat{\boldsymbol{\sigma}}^{(n)})^2] \leftarrow \text{svdGP}(\mathbf{X}^{(n)}, \mathbf{Y}^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma)$ 
10  Predict  $\mathbf{y}(\mathbf{x}_0)$  through  $\pi(\mathbf{y}(\mathbf{x}_0) | \mathbf{V}^{*(n)}, \hat{\boldsymbol{\Theta}}^{(n)}, (\hat{\sigma}^{(n)})^2, (\hat{\boldsymbol{\sigma}}^{(n)})^2)$  in Eqn. (9)

```

C. Additional Simulation Results

We now investigate the reliability of the estimated range parameters (or equivalently, the correlation parameters) for the proposed local approximate SVD-based GP model (lasvdGP) fits in Examples 1 (Forrester et al., 2008 – $q = 3, N = 10000, M = 2000, n = 40$ and $n_0 = 20$) and 2 (Bliznyuk et al., 2008 – $q = 5, N = 10000, M = 2000, n = 50$ and $n_0 = 25$) of the main article.

To explain the results, recall that for each point in the test set, the SVD-based GP model

fitted on the neighbourhood set is represented using a p -dimensional basis as in Equation (1), where p is selected by the cumulative percentage criterion (Equation (2)). That is, for each test point, p independent GP models for each $c_i(\mathbf{x})$ are fitted in the respective neighbourhood searched. The value of p may be different for different test points. The frequency table of the number of leading basis functions for 2,000 test points for each of the two examples are displayed in Table 2.

	p						total
	3	4	5	6	7	8	
Example 1	266	1734	0	0	0	0	2000
Example 2	15	161	873	825	124	2	2000

Table 2: Frequency of p among the 2,000 test points in Examples 1 and 2.

For simplicity, we only report the estimated range parameters in the GP models corresponding to $c_1(\mathbf{x})$, $c_2(\mathbf{x})$ and $c_3(\mathbf{x})$ from the final fits, i.e., after $n - n_0$ follow-up points were added. Figures 8 and 9 display the boxplots of those 2,000 estimates for each range parameter in Examples 1 and 2, respectively.

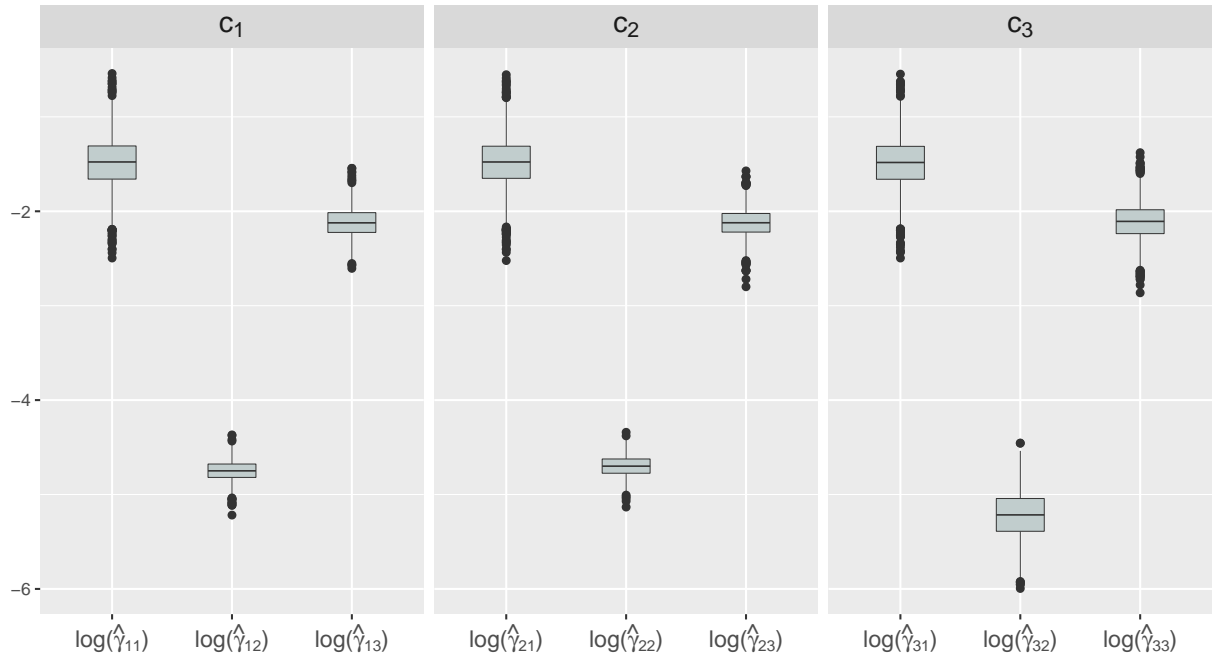


Figure 8: The boxplots of the 2,000 estimates of the log-range parameters in the GP models for $c_1(\mathbf{x})$, $c_2(\mathbf{x})$ and $c_3(\mathbf{x})$ in Example 1 (Forrester et al., 2008).

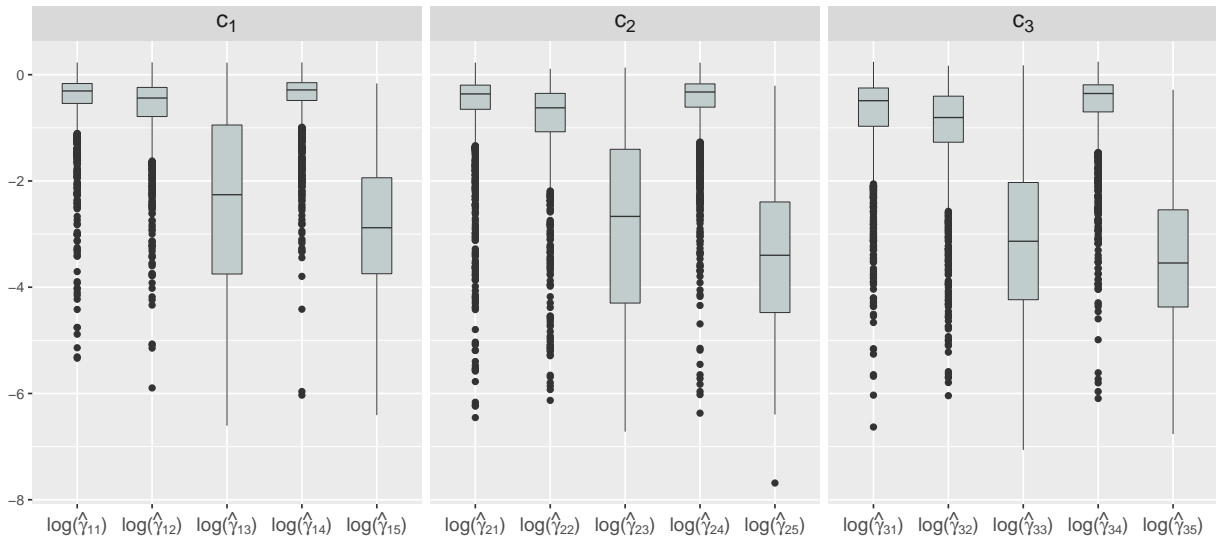


Figure 9: The boxplots of the 2,000 estimates of the log-range parameters in the GP models for $c_1(\mathbf{x})$, $c_2(\mathbf{x})$ and $c_3(\mathbf{x})$ in Example 2 (Bliznyuk et al., 2008).