Prediction for Distributional Outcomes in High-Performance Computing I/O Variability

Li Xu[1], Yili Hong[2], Max D. Morris[3], and Kirk W. Cameron[4]

[1]Department of Epidemiology, Harvard University, Boston, MA, 02215

[2]Department of Statistics, Virginia Tech, Blacksburg, VA 24061

[3]Department of Statistics, Iowa State University, Ames, IA 50011

[4]Department of Computer Science, Virginia Tech, Blacksburg, VA 24061

Abstract

Although high-performance computing (HPC) systems have been scaled to meet the exponentially-growing demand for scientific computing, HPC performance variability remains a major challenge and has become a critical research topic in computer science. Statistically, performance variability can be characterized by a distribution. Predicting performance variability is a critical step in HPC performance variability management and is nontrivial because one needs to predict a distribution function based on system factors. In this paper, we propose a new framework to predict performance distributions. The proposed model is a modified Gaussian process that can predict the distribution function of the input/output (I/O) throughput under a specific HPC system configuration. We also impose a monotonic constraint so that the predicted function is non-decreasing, which is a property of the cumulative distribution function. Additionally, the proposed model can incorporate both quantitative and qualitative input variables. We evaluate the performance of the proposed method by using the IOzone variability data based on various prediction tasks. Results show that the proposed method can generate accurate predictions, and outperform existing methods. We also show how the predicted functional output can be used to generate predictions for a scalar summary of the performance distribution, such as the mean, standard deviation, and quantiles. Our methods can be further used as a surrogate model for HPC system variability monitoring and optimization.

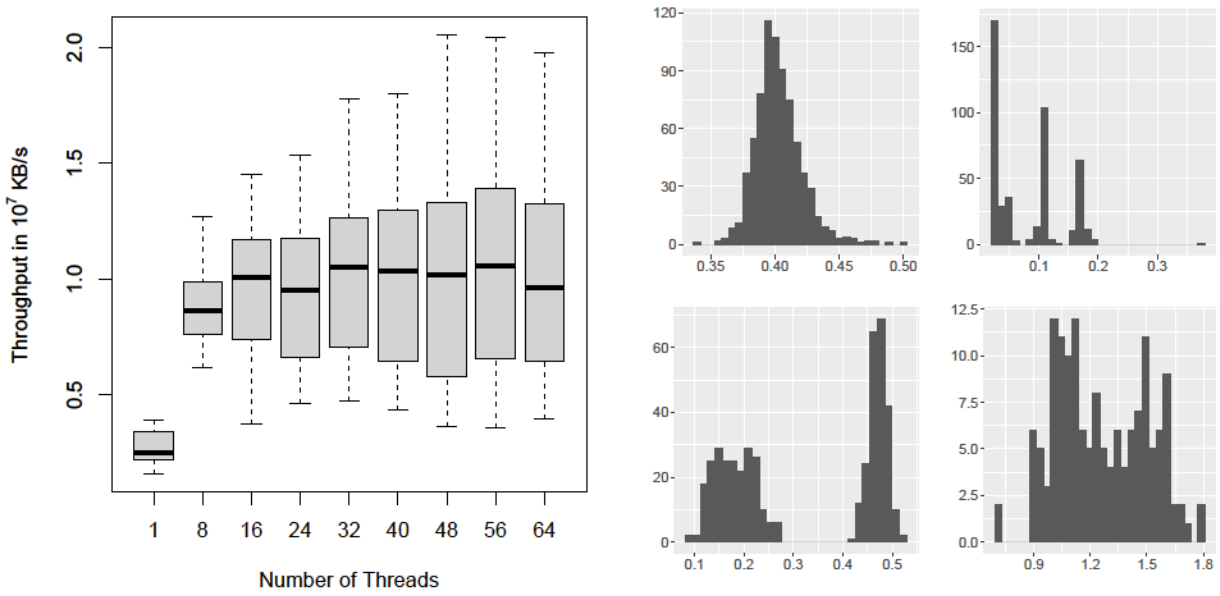
Key Words: Computer Experiments; Gaussian Process; Functional Prediction; HPC Performance Variability; Qualitative and Quantitative Factors; System Variability.

1 Introduction

High-performance computing (HPC) systems aggregate a large number of computers to provide a high level of computing performance. In the past decades, the performance of HPC systems has been increased to meet the exponentially-growing demand for scientific computing. However, existing work (Rahimi et al. 2015; Cameron et al. 2019) has observed that the performance variability increases with HPC system scale and complexity. For example, Figure 1(a) shows that the input/output (I/O) throughput, as one measure of the system performance, increases as the number of threads increases based on a subset of the IOzone data to be introduced in Section 2. However, we observe that the performance variability, as shown by the boxplots, also increases. Existing studies reveal that variability can influence the performance in many aspects from hardware (Kim et al. 2012), middleware (Akkan et al. 2012; Ouyang et al. 2015) to applications (Hammouda et al. 2015). Thus performance variability management has become an important research area in computer science, which is affected by system configurations (e.g., CPU frequency). Unfortunately, the quantitative relationship between the system configuration and variability is not clear, which makes the HPC performance variability management challenging. Studies have discovered that the relationship between HPC variability and system configuration is complicated (Lux et al. 2018; Chang et al. 2018). To study the complicated relationship, statistical tools can be useful for data collection, model building, and performance variability prediction. Large-scale experiments are essential to provide sufficient data for modeling the complex variability map, and experimental design tools have been used for efficient data collection (Wang et al. 2022).

Regarding modeling and prediction, performance variability can be characterized by a distribution. Most existing work in computer science, however, only uses a summary statistic to represent the level of variability. For example, Cameron et al. (2019) study the standard deviation of the IOzone throughput. Xu et al. (2020) show that the throughput distribution is multimodal so a summary statistic like standard deviation cannot represent the system variability. As an illustration, Figure 1(b) shows the histograms of the I/O throughput under four specific HPC system configurations. The top left panel shows a distribution with one mode, and the bottom left panel shows a mixture of two components, while the right two panels show a mixture of three and more than three components. Therefore, the distributions of the throughput are complicated, and it is typically not sufficient to use summary statistics or a simple parametric distribution to describe them.

Because the performance distribution is complicated, it will be ideal to have a general method to predict the entire distribution. Furthermore, various metrics are often of interest in the HPC study. The mean or median of the throughput distribution can be used as an overall



(a) Example of I/O Variability (b) Throughput Distributions

Figure 1: (a) Example of I/O variability in throughput as a function of the number of threads in the IOzone data, and (b) histograms for the I/O throughput under four specific HPC system configurations showing examples of distributions with various shapes. The x-axis is the throughput (10^7 KB/s) and the y-axis is the frequency.

performance measure, while the standard deviation can be used as a measure of variability or stability. Various quantiles of the performance distribution can serve as practical lower or upper bounds of throughput, which leads to a general need for modeling and predicting the performance distribution. This is because once the distribution is predicted, one can derive all the above-mentioned metrics, which brings tremendous benefits in HPC variability management.

To address the challenging problem in HPC variability management, the main objective of this paper is to generate distributional-output predictions for HPC variability study. The prediction framework is outlined as follows. We first use I-splines to smooth the discrete sample quantile function and the obtained spline coefficients matrix is then used to represent the distribution function. Singular value decomposition (SVD) is implemented to reduce the dimension of the coefficient matrix. For prediction, we propose a special Gaussian process (GP) named linear mixed Gaussian process (LMGP), that incorporates both quantitative and qualitative variables. The expectation-maximization (EM) algorithm is used to estimate the parameters. Results show that our prediction framework can achieve accurate predictions under HPC setting. To the best of our knowledge, this work is the first work that develops a statistical framework predicting the distributional outcome with mixed types of inputs and

modeling HPC throughput distributions along with their associated measures of variability.

We give a brief literature review on computer experiments with an emphasis on mixed types of input and output. Computer experiments are often constructed to emulate a physical system. Due to the complexity and expense of evaluating system behavior, a surrogate model is usually used to describe the system behavior based on the data collected by the experiments. Popular surrogate models include response surfaces (Box and Wilson 1951), Gaussian process models (Rasmussen and Williams 2005), localized linear regression (Cleveland 1979), and their extensions. Gramacy and Lee (2008), Chipman et al. (2002), Chipman et al. (2010), and Taddy et al. (2011) use the binary tree to divide the input space and fit separate Gaussian process in each sub-region. Multivariate adaptive regression splines (MARS) uses splines and stepwise regression to model the complex relationships between input and output (Friedman 1991). To determine the best model with respect to node location and number of nodes, a generalized cross-validation procedure (Hastie et al. 2009) is used to do model selection. The linear Shepard (LSP) algorithm uses radial basis functions to design weight and build a localized linear regression model (Thacker et al. 2010).

While most of those models assume the inputs of surrogates are continuous, categorical inputs are common in application. For example, in the HPC setting, the type of storage has two options: solid-state drive (SSD) and hard disk drives (HDD). To utilize categorical variables, Zhou et al. (2011) propose the CGP and Deng et al. (2017) extend the CGP with an additive model structure. In addition, most existing methods focus on scalar prediction, while the output of some engineering models can be complicated (Bayarri et al. 2007). Examples of applications with complicated outputs include the boundary condition of a partial differential equation (Tan 2018), the thermal-hydraulic computations (Auder et al. 2012), and the satellite orbiting carbon observatory (Ma et al. 2022).

For the work on computer experiments modeling with functional outputs, Hung et al. (2015) develop a Monte Carlo expectation-maximization (MCEM) algorithm to convert the irregularly spaced data into a regular grid so that the Kronecker product-based approach can be employed for efficiently fitting a kriging model to the functional data. Higdon et al. (2008) provide a dimension-reduction method to the high-dimensional output computer experiments. Jiang et al. (2021) provide a robust parameter design to computer models with multiple functional outputs. Fruth et al. (2015) conduct sensitivity analysis method for functional input. Drignei (2010) proposes a framework called functional ANOVA to analyze the computer experiments with time series outputs. However, to our best knowledge, there is no work focused on the distributional outcome on computer models with both qualitative and quantitative inputs, which cannot be addressed by straightforward applications of existing methods.

Because of the distributional outcome, the properties of the distribution functions need

to be met. Specifically, the cumulative distribution function is right-continuous and non-decreasing. In addition, effective modeling of output distributions generally requires large datasets because complicated experiments are essential to capture the distributional information. Given the need to predict the distribution and the fact that the distribution is complicated, we use the Gaussian process models as the basis for our work. Compared to parametric models, the Gaussian process can establish a more complicated relationship between the input and response variables. In this paper, we propose a prediction framework with Gaussian process that can predict the distributional output given both quantitative and qualitative inputs.

The rest of this paper is organized as follows. Section 2 describes the HPC IOzone data. Section 3 describes the prediction framework including the curve representation, the formulation of the LMGP model, the EM algorithm for parameter estimation, and the functional prediction. Section 4 presents the prediction results on the IOzone data for different input and output (I/O) operation modes in predicting the quantile functions. Section 5 shows the comparison results with those existing models in predicting summary statistics of the throughputs. Section 6 discusses the results and several areas for future work.

2 HPC Performance Study

While the system variability has many aspects, we concentrate on the I/O tasks as these types of the procedure will reveal the highest variability and exhibit the most interesting system performance characteristics. I/O is identified as a high variation operation and the IOzone benchmark (Capps and Norcott 2008) is used to collect performance data on the various system I/O operations. The reported throughput values are used to represent the system performance and furthermore, the variation of the throughput under identical system configurations is treated as the system variability. The unit of the throughput is KB/s. For convenience, all the throughputs in this paper are on the scale of 10^7 KB/s.

The configurations are characterized by a list of variables, which are referred to as inputs. There are two kinds of inputs, namely, numerical inputs and categorical inputs. We have four numerical inputs, the file size, the record size, the CPU frequency, and the number of threads. The record size is fixed at 16 KB throughout the whole experiment. Thus, the numerical variables we model in this paper are file size, CPU frequency, and the number of threads. The categorical input is the I/O operation mode, which has six levels. There are various combinations of those three continuous inputs under each level of the categorical input (i.e., the I/O operation mode). Table 1 shows the system configurations and all possible levels we have considered in our data collecting experiments. In total, we have 22,734 combinations (system

Table 1: System factors and their levels used in the study of I/O variability.

System Parameters	No. of Levels	Levels
CPU Clock Frequency (GHz)	7	1.2, 1.6, 2.0, 2.3, 2.8, 3.2, 3.5
Number of Threads	9	1, 8, 16, 24, 32, 40, 48, 56, 64
File Size (KB)	10	4, 16, 64, 256, 1024, 4096, 8192, 32768, 65536
I/O Operation Mode	6	random_reader, initial_writer, random_writer, rereader, reader, rewriter

configurations) in the IOzone database. Figure 2 shows the combinations of continuous inputs under I/O operation mode initial_writer. Because the levels of the file size and the number of threads are spaced on an exponential scale, we take the binary logarithm of the two variables in our subsequent analyses.

The configurations are denoted by $\{\boldsymbol{x}_i, \boldsymbol{z}_i\}, i = 1, \ldots, n$. Here, $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^{\mathrm{T}}$ is a $p \times 1$ vector that denotes the numerical inputs, $\boldsymbol{z}_i$ is a $q \times 1$ vector that denotes the categorical inputs, and n is the number of configurations. In the IOzone data, $p = 3$ and $q = 1$. We denote the numerical input matrix by $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^{\mathrm{T}}$, which is of size $n \times p$, and denote the categorical input matrix by $\mathbf{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)^{\mathrm{T}}$, which is of size $n \times q$. Thus the input is represented by $\{\mathbf{X}, \mathbf{Z}\}$.

To collect the data which reflects the distributional information, we fix the system configuration at a given combination in Table 1 and run the IOzone benchmark for a specified number of replicates. The output of each experiment run is the throughput of IOzone, which measures the I/O speed under the current system configuration. The throughput data at configuration i are denoted by y_{ib}, $i = 1, \ldots, n$, and $b = 1, \ldots, m_i$. Let $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{im_i})^{\mathrm{T}}$, where m_i is the number of replicates for ith configuration. The values of m_i vary from 150 to 900, depending on the specific system configuration. It took several months to collect all the data over a Linux server. In particular, the experiments were conducted on a 12-node server and all the nodes are identical Dell PowerEdge R630s. Each node is equipped with Intel(R) Xeon(R) CPU E5-2637 v4@3.50 GHz, 16 GB DRAM (2 DIMMs), and a new 200 GB SSD with Intel model SSDSC2BA200G4R. There are 2 sockets with 4 cores per socket. In total, there are 8 physical cores and 16 CPUs with hyper-threading enabled. The operating system is Debian GNU/Linux with kernel version 4.14 and the IOzone version is 3.465. Note that we are working with real performance data from HPC systems (not with emulator data as in some computer experiment literature).

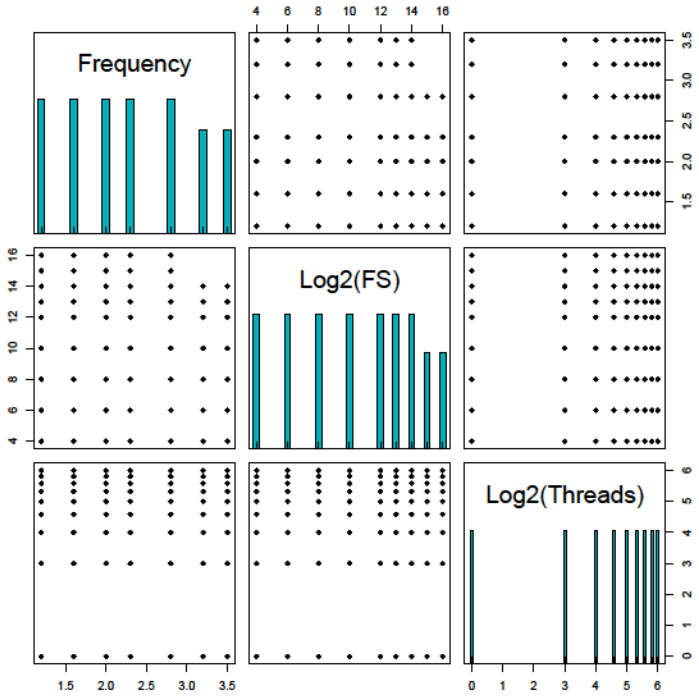


Figure 2: Scatter plots and histograms for three continuous factors under I/O operation mode initial_writer. "Log2(FS)" means the binary logarithm of the file size. The unit for CPU Frequency is GHz and the unit for file size is KB.

The data $\boldsymbol{y}_i$ are then used to estimate a distribution function which we will treat as a functional response in modeling. For the development of the model and notational convenience, we need to first sort the data by $\mathbf{Z}$. Let c be the number of unique combinations of the categorical variables (i.e., the unique rows in $\mathbf{Z}$). We sort the data $\{\boldsymbol{y}_i, \boldsymbol{x}_i, \boldsymbol{z}_i\}, i = 1, \ldots, n$, by the unique categorical combinations. Let n_k be the number of rows in the kth categorical combination in $\mathbf{Z}$ for $k = 1, \ldots, c$.

3 The Prediction Framework

Our proposed framework for predicting HPC throughput distribution has three components: curve representation, Gaussian process for prediction, and reconstruction of functional curves.

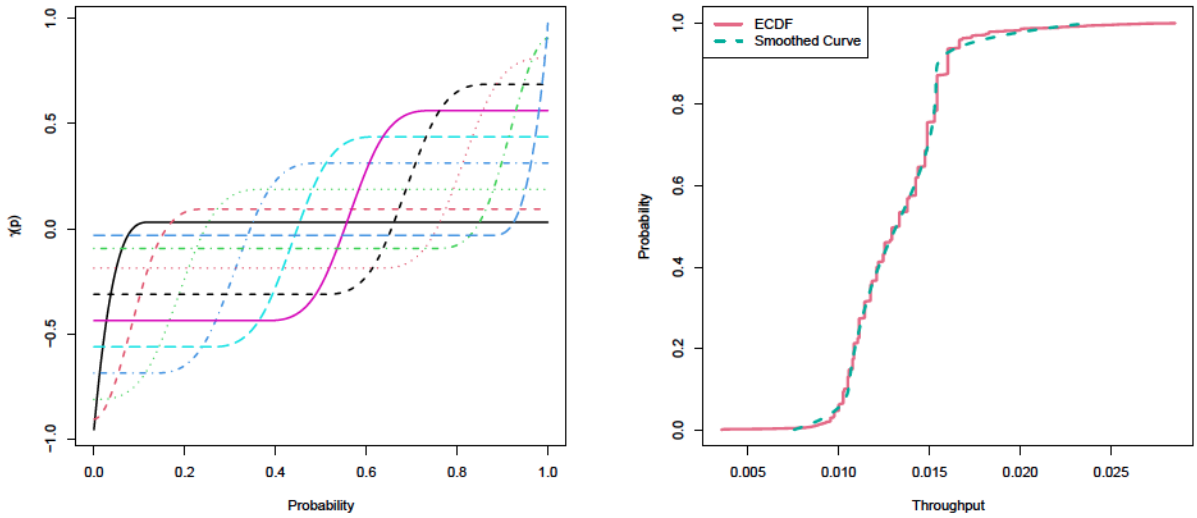
3.1 Curve Representation

For the throughput data, $\boldsymbol{y}_i$, from configuration i, we are interested in its cumulative distribution function (CDF), $F_i(y)$. Because the distribution of the throughput is usually complicated and cannot be adequately described by commonly used parametric distributions, we use the empirical cumulative distribution function (ECDF) to estimate the distribution function. In particular, the ECDF is computed as

$$\widehat{F}_i(y) = m_i^{-1} \sum_{b=1}^{m_i} \mathbb{1}(y_{ib} \leq y).$$

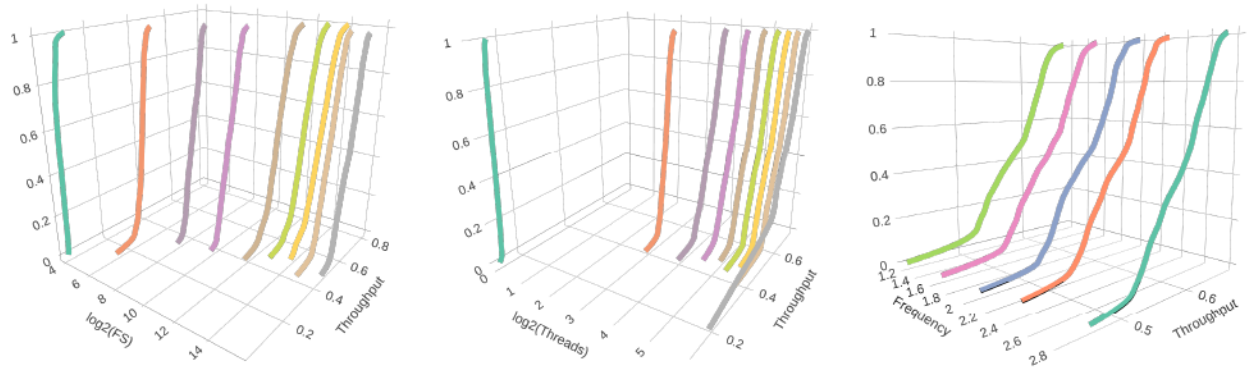
The critical points in the ECDF are $\{[y_{i(b)}, b/m_i], b = 1, \ldots, m_i\}$, where $y_{i(b)}$ is the sorted version of y_{ib} in the ascending order.

Because the ECDF is only right continuous and always has jump discontinuities, for the convenience of modeling, we use a smooth function to approximate it. In addition, because the CDF is a non-decreasing function, we use monotonic splines for smoothing. In particular, we use I-splines (Ramsay 1988). I-splines are a set of functions that are positive and monotone increasing in a closed interval and constants outside this closed interval. Figure 3(a) shows the curve of a set of I-spline basis functions. Figure 3(b) shows one example of the ECDF and the curve after being smoothed. Figure 4 shows how the smoothed CDF curve changes when we vary on one of the three continuous configuration factors. We find a complicated relationship between the CDF curves and input configurations. For example, in Figure 4(c), when we have more threads, the range of the throughputs will have a larger range and the shape of the curve also changes. Figure 4 indicates that predicting the distributional outcome is challenging.



(a) I-spline Bases (b) ECDF

Figure 3: Examples of I-spline bases (a), and plots of the ECDF as a step function and its corresponding smoothed curve for a specific system configuration (b).



(a) File Size (b) Threads (c) Frequency

Figure 4: Smoothed CDFs when one continuous configuration changes. The z-axis is the probability. In (a), file size changes while (frequency, threads) is fixed at $(2.3, 32)$. In (b), the number of threads changes while (file size, frequency) is fixed at $(65536, 2.3)$. In (c), frequency changes while (file size, threads) is fixed at $(65536, 32)$.

A set of knots are needed to construct the spline bases. Based on an initial exploration of the data, we find that the supports of the CDF are quite different for different configurations. Figure 5(a) shows a typical example in the IOzone data that the distributions of the throughput from different configurations have different supports. The supports of each CDF vary largely, which is challenging to choose both the number of and the locations of the spline knots. In order to cover the entire range of the CDF support and ensure smoothing accuracy, we would need a large number of knots. To overcome this difficulty, we need to set our predicted probability function to have common support with a fixed boundary. We show ten examples of smoothed quantile functions in Figure 5(a). Each CDF is smoothed individually by a unique set of I-splines. The number of knots is 20 and the range of knots is equal to the range of throughputs under this configuration. From the figure, we can see that the supports of different smoothed CDFs are different. To set common bases for all configurations, we smooth the estimated quantile function, instead of the ECDF. Because a quantile function, $Q(p)$, is defined in $(0, 1)$ which is bounded, one can easily set the knots in the bounded domain.

To summarize the idea, we want to model and predict CDF's, and use spline fits to represent them. However, splines require knot locations, which is impractical here because the support and complexity of CDF varies substantially for different experimental conditions. Thus, we instead directly model the inverse of the CDF, the quantile function, which is always defined on the same interval.

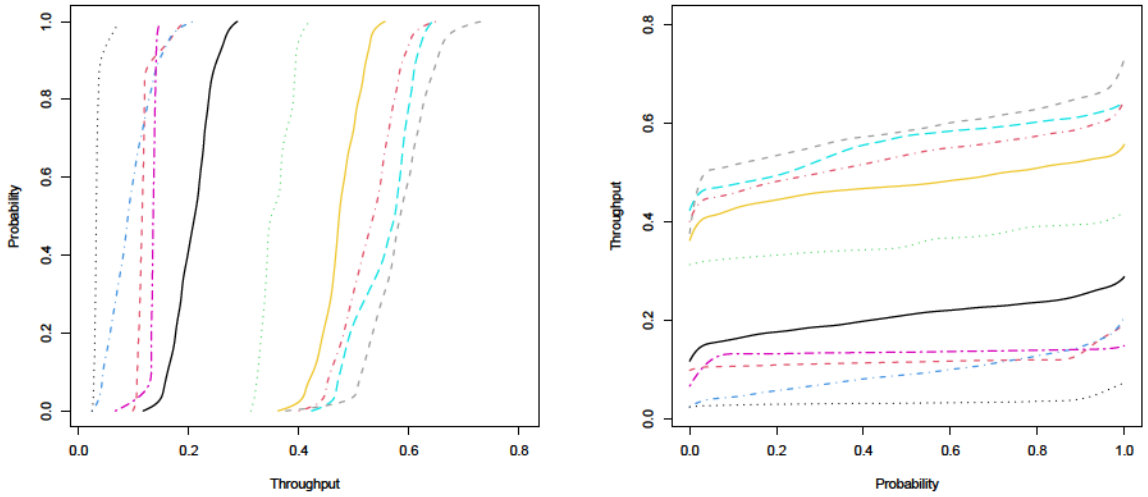
Specifically, we first construct $(d-1)$ common spline bases, $\gamma_j(p), j = 1, \ldots, (d-1)$ and use these base to smooth the points $\{[b/m_i, y_{i(b)}], b = 1, \ldots, m_i\}$ separately for configuration i. Let $\boldsymbol{\beta}_i = (\beta_{i0}, \beta_{i1}, \ldots, \beta_{i,d-1})^{\mathrm{T}}$ be the coefficients of the spline fitting for configuration i. The first element β_{i0} is an intercept term, and the rest elements $(\beta_{i1}, \ldots, \beta_{i,d-1})^{\mathrm{T}}$ are the coefficients for those $(d-1)$ spline basis functions. Thus $\boldsymbol{\beta}_i$ is of size $d \times 1$. The smoothed quantile function is

$$\widehat{Q}(p) = \widehat{\beta}_{i0} + \sum_{j=1}^{d-1} \widehat{\beta}_{ij}\gamma_j(p).$$

For I-splines, the intercept β_{i0} is unconstrained, and the spline coefficients $(\beta_{i1}, \ldots, \beta_{i,d-1})^{\mathrm{T}}$ are constrained to be nonnegative, to ensure monotonicity. The constrained least-squares method (e.g., Meyer 2008) is used to find the spline coefficients. Let $\mathbf{B} = (\boldsymbol{\beta}_1^{\mathrm{T}}, \ldots, \boldsymbol{\beta}_n^{\mathrm{T}})^{\mathrm{T}}$ be the spline coefficient matrix, which is of size $n \times d$. Using the spline representation, the distributional data are now represented by the coefficient matrix $\mathbf{B}$.

We apply the singular value decomposition (SVD) to de-correlate the columns of $\mathbf{B}$, which is similar to the treatment in Higdon et al. (2008). That is, we express $\mathbf{B}$ as

$$\mathbf{B} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^{\mathrm{T}},$$



(a) CDF (b) Quantile Function

Figure 5: Examples of I-spline smoothed CDF and quantile function for the throughputs under 10 randomly picked configurations.

where $\mathbf{U}$ is an $n \times n$ unitary matrix, $\mathbf{\Lambda}$ is an $n \times d$ diagonal matrix with diagonal elements $\{\lambda_1, \ldots, \lambda_d\}$ which are the singular values, and $\mathbf{V}$ is a $d \times d$ unitary matrix in the SVD. Let

$$\mathbf{W} = \mathbf{U}\mathbf{\Lambda} = \mathbf{B}\mathbf{V}. \tag{1}$$

Note here $\mathbf{W}$ is an $n \times d$ matrix. Let $\boldsymbol{w}_j$ be jth column of $\mathbf{W}$ and w_{ij} be the (i,j)th element of $\mathbf{W}$. After re-expression of the $\mathbf{B}$ matrix using the SVD, we focus on the resulting $\mathbf{W}$ matrix. We perform separate modeling for $\boldsymbol{w}_j$ because the $\boldsymbol{w}_j$'s are linear independent components.

3.2 Gaussian Process Modeling

We first summarize the formulas for modeling and prediction with the Gaussian process for continuous scalar output $\boldsymbol{y}$ and continuous covariates $\mathbf{X}$. Then, with the overall mean μ, variance σ^2, the length-scale parameter vector $\boldsymbol{\nu}$ and the nugget g, the Gaussian process for the data $\{\boldsymbol{y}, \mathbf{X}\}$ is that $\boldsymbol{y}$ follows a multivariate normal distribution $\boldsymbol{y} \sim \mathrm{N}\left[\mu \mathbf{1}_n, \sigma^2 \mathbf{\Omega}(\boldsymbol{\nu}, g)\right]$, where $\mathbf{1}_n$ is an n-element vector with all ones. The construction for the matrix $\mathbf{\Omega}(\boldsymbol{\nu})$ is the distance-inverse kernel as follows,

$$\mathbf{\Omega}(\boldsymbol{\nu}, g)_{ii'} = \exp\left[-d(\boldsymbol{x}_i, \boldsymbol{x}_{i'}, \boldsymbol{\nu}, g)\right], \text{ and } d(\boldsymbol{x}_i, \boldsymbol{x}_{i'}, \boldsymbol{\nu}, g) = \sum_{l=1}^{p} \frac{(x_{il} - x_{i'l})^2}{\nu_l} + g\delta_{ii'}.$$

Here $\delta_{ii'}$ is the Kronecker delta. The parameters μ, σ^2, and $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_p)^{\mathrm{T}}$ can be estimated through the maximum likelihood estimation (MLE) procedure.

For prediction, the joint distribution for $\boldsymbol{y}$ and $\boldsymbol{y}_0$ is

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_0 \end{pmatrix} \sim \mathbf{N}\left\{ \begin{pmatrix} \mu \mathbf{1}_n \\ \mu \mathbf{1}_{n_0} \end{pmatrix}, \sigma^2 \begin{bmatrix} \boldsymbol{\Omega} & \boldsymbol{\Omega}_{12} \\ \boldsymbol{\Omega}_{12}^{\mathrm{T}} & \boldsymbol{\Omega}_0 \end{bmatrix} \right\},$$

where $(\boldsymbol{\Omega}_{12})_{ii'} = \exp\left[-d(\boldsymbol{x}_i, \boldsymbol{x}_{0i'}, \boldsymbol{\nu}, g)\right], (\boldsymbol{\Omega}_0)_{ii'} = \exp\left[-d(\boldsymbol{x}_{0i}, \boldsymbol{x}_{0i'}, \boldsymbol{\nu}, g)\right]$, n_0 is the number of predicted points, and $\boldsymbol{x}_{0i'}$ is the i'th input variable in the set of predicted points. The prediction for $\boldsymbol{y}_0$ is the conditional mean

$$\widehat{\boldsymbol{y}}_0 = \mathbf{E}\left[\boldsymbol{y}_0 | \boldsymbol{y}\right] = \mu \mathbf{1}_{n_0} + \boldsymbol{\Omega}_{12}^{\mathrm{T}}(\boldsymbol{\nu}) \boldsymbol{\Omega}^{-1}(\boldsymbol{\nu})(\boldsymbol{y} - \mu \mathbf{1}_n).$$

Estimation of parameters $\boldsymbol{\nu}$ and μ will be discussed in Section 3.4.

3.3 The Linear Mixed Gaussian Process

We construct d separate models for the columns of matrix $\mathbf{W}$. For each model, we fix j and use the data $\{\boldsymbol{w}_j, \mathbf{X}, \mathbf{Z}\}$ to build the model. We consider the following model for the $\boldsymbol{w}_j$,

$$w_{ij} = \mu + \alpha_{ij} + \varepsilon_{ij}, \quad i = 1, \ldots, n, \tag{2}$$

where μ is the grand mean, α_{ij} is the categorical random effect, and ε_{ij} is the random error. For notation convenience, we drop the index j but keep in mind that the model in (2) will be applied separately for $j = 1, \ldots, d$. With the dropping of index j, the model in (2) is represented as

$$w_i = \mu + \alpha_i + \varepsilon_i, \quad i = 1, \ldots, n, \tag{3}$$

and the vector formulation of (3) is

$$\boldsymbol{w} = \boldsymbol{\mu} + \boldsymbol{\alpha} + \boldsymbol{\varepsilon},$$

where $\boldsymbol{w} = \boldsymbol{w}_j$, $\boldsymbol{\mu} = (\mu, \ldots, \mu)^{\mathrm{T}} = \mu \mathbf{1}_n$, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^{\mathrm{T}}$, and $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^{\mathrm{T}}$.

The random error term ε_i models the within-class correlation. Here, each class is one level of the categorical input combinations. Across classes, the ε_i's are independent. Specifically, we model $\boldsymbol{\varepsilon}$ as a realization from a multivariate normal distribution $\mathbf{N}(\mathbf{0}, \boldsymbol{\Sigma}_\varepsilon)$, and the variance-covariance matrix for $\boldsymbol{\varepsilon}$, denoted by $\boldsymbol{\Sigma}_\varepsilon$, is a block diagonal matrix. In particular,

$$\boldsymbol{\Sigma}_\varepsilon = \begin{pmatrix} \boldsymbol{\Sigma}_{\varepsilon 1} & & \\ & \ddots & \\ & & \boldsymbol{\Sigma}_{\varepsilon c} \end{pmatrix} = \sigma_\varepsilon^2 \begin{pmatrix} \boldsymbol{\Omega}_{\varepsilon 1} & & \\ & \ddots & \\ & & \boldsymbol{\Omega}_{\varepsilon c} \end{pmatrix} = \sigma_\varepsilon^2 \boldsymbol{\Omega}_\varepsilon.$$

Here, σ_ε^2 is the variance of ε_i, and the $\boldsymbol{\Omega}_{\varepsilon k}$ are $n_k \times n_k$ correlation matrix. In particular, the correlation of ε_i and $\varepsilon_{i'}$ is also the distance-inverse kernel which defined as

$$(\boldsymbol{\Omega}_\varepsilon)_{ii'} = \mathrm{Corr}(\varepsilon_i, \varepsilon_{i'}) = \begin{cases} \exp\left[-\sum_{l=1}^{p}(x_{il} - x_{i'l})^2/\nu_l\right] + g\delta_{ii'}, & \text{if } \boldsymbol{z}_i = \boldsymbol{z}_{i'} \\ 0 \qquad\qquad\qquad\qquad\quad , & \text{if } \boldsymbol{z}_i \neq \boldsymbol{z}_{i'} \end{cases}.$$

The term α_i is the categorical random effect which models the between-class correlation. We also model $\boldsymbol{\alpha}$ with a multivariate normal distribution $\mathbf{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha)$. Let $\boldsymbol{\Sigma}_\alpha = \sigma_\alpha^2 \boldsymbol{\Omega}_\alpha$, where σ_α^2 is the variance of α_i, and $\boldsymbol{\Omega}_\alpha$ is the corresponding correlation matrix. The structure of $\boldsymbol{\Omega}_\alpha$ is specified as follows,

$$\mathrm{Corr}(\alpha_i, \alpha_{i'}) = \begin{cases} \rho(\boldsymbol{z}_i, \boldsymbol{z}_{i'})\kappa(r_{ii'}, r_{\max}), & \text{if } r_{ii'} \leq r_{\max} \\ 0 \qquad\qquad\qquad , & \text{if } r_{ii'} > r_{\max} \end{cases}, \tag{4}$$

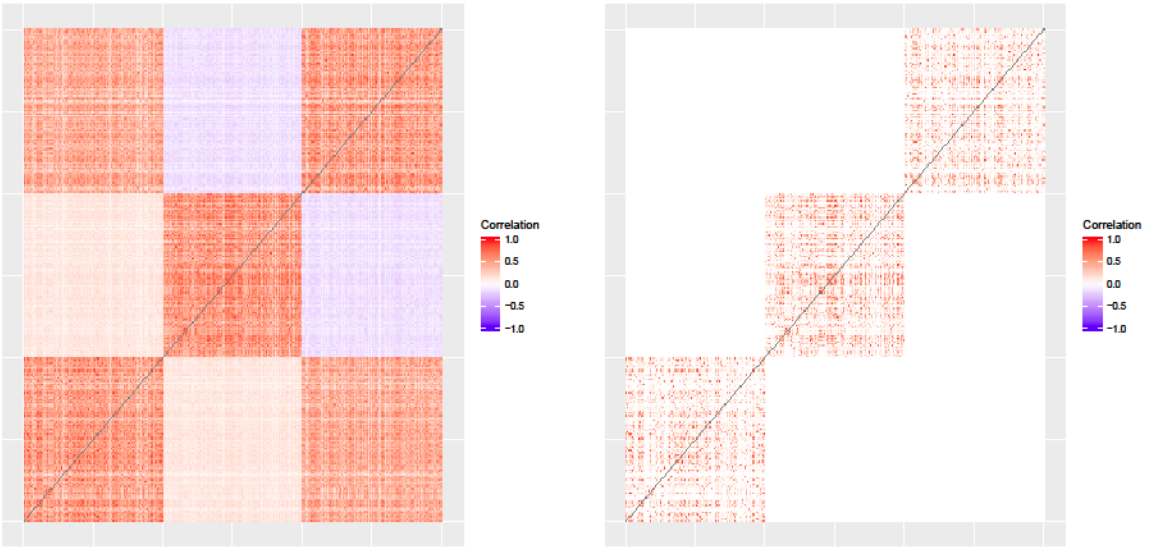
where $r_{ii'} = ||\boldsymbol{x}_i - \boldsymbol{x}_{i'}||$ is the Euclid distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$, $\rho(\boldsymbol{z}_i, \boldsymbol{z}_{i'})$ defines the correlation between category i and i', and $\kappa(\cdot, r_{\max})$ is a compact support kernel with prespecified range parameter $r_{\max}$ to allow for sparsity. The formula of the compact support kernel we use is

$$\kappa(r_{ii'}, r_{\max}) = \left(1 - \frac{r_{ii'}}{r_{\max}}\right)_+^v, \text{ where } v \leq \frac{p+1}{2},$$

which is defined by Wendland (1995). This functional form ensures that the $\boldsymbol{\Omega}_\alpha$ is positive definite, as required for variance-covariance matrices. Let $\rho_{kk'} = \rho(\boldsymbol{z}_i, \boldsymbol{z}_{i'})$, where k and k' are the corresponding coded class labels for $\boldsymbol{z}_i$ and $\boldsymbol{z}_{i'}$, respectively. We use the formulation in Simonian (2010) and Zhou et al. (2011) for $\rho_{kk'}$. Note that the total number of categorical level combinations is c. To ensure that the matrix defined by using (4) is a valid variance-covariance matrix, the $c \times c$ matrix $\mathbf{P} = (\rho_{kk'})$ must be a positive definite matrix with unit diagonal values. Let $\mathbf{P} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$, where $\mathbf{L} = (l_{rs})$ is a lower triangle matrix with positive diagonal values. Let $l_{11} = 1$ and for $k = 2, \ldots, c$ the formula for kth row of $\mathbf{L}$ is given as

$$l_{k1} = \cos(\theta_{k1}), l_{ks} = \left[\prod_{j=1}^{s-1}\sin(\theta_{kj})\right]\cos(\theta_{ks}), \text{ for } s = 2, \ldots, k-1, \text{ and } l_{kk} = \prod_{j=1}^{k-1}\sin(\theta_{kj}).$$

The parameters for $\boldsymbol{\Omega}_\alpha$ is $\theta_{ks} \in (0, \pi)$, $k = 2, \ldots, c$, $s = 1, \ldots, k-1$. As a result, for c categorical levels, we have $c \times (c-1)/2$ parameters for $\rho_{kk'}$. To visualize the model variance-covariance matrix structure, we provide the heatmap of a typical $\boldsymbol{\Omega}_\alpha$ and $\boldsymbol{\Omega}_\varepsilon$ in Figure 6. From Figure 6(b), the distance inverse kernel can only model positive correlation, which is suitable for the data within the same categorical variable combination. The $\boldsymbol{\Omega}_\alpha$ in Figure 6(a) can model the negative correlation (the block in the top center area) between different categorical variables.



(a) $\boldsymbol{\Omega}_{\boldsymbol{\alpha}}$ (b) $\boldsymbol{\Omega}_{\boldsymbol{\varepsilon}}$

Figure 6: Example correlation matrix heatmaps for $\boldsymbol{\Omega}_{\boldsymbol{\alpha}}$ and $\boldsymbol{\Omega}_{\boldsymbol{\varepsilon}}$, when $p = 4$, $c = 3$, $\boldsymbol{\rho} = (1.33, 0.56, 2.66)^{\mathrm{T}}$, and $\boldsymbol{\nu} = (2.0, 2.0, 2.0, 2.0)^{\mathrm{T}}$ based on the randomly generated data.

3.4 The Estimation Procedure

Let $\boldsymbol{\theta}_{\boldsymbol{\alpha}} = (\sigma^2_{\boldsymbol{\alpha}}, \boldsymbol{\rho}^{\mathrm{T}})^{\mathrm{T}}$ and $\boldsymbol{\theta}_{\boldsymbol{\varepsilon}} = (\sigma^2_{\boldsymbol{\varepsilon}}, \mu, \boldsymbol{\nu}^{\mathrm{T}}, g)^{\mathrm{T}}$. All the parameters are denoted by $\boldsymbol{\theta} = (\boldsymbol{\theta}^{\mathrm{T}}_{\boldsymbol{\varepsilon}}, \boldsymbol{\theta}^{\mathrm{T}}_{\boldsymbol{\alpha}})^{\mathrm{T}}$, $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}} = \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}(\boldsymbol{\theta}_{\boldsymbol{\alpha}})$, $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}} = \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}(\boldsymbol{\theta}_{\boldsymbol{\varepsilon}})$. The complete likelihood is

$$L(\boldsymbol{\theta}; \boldsymbol{w}, \boldsymbol{\alpha}) = f(\boldsymbol{w}, \boldsymbol{\alpha}; \boldsymbol{\theta}) = f(\boldsymbol{w}|\boldsymbol{\alpha}; \boldsymbol{\theta}) f(\boldsymbol{\alpha}; \boldsymbol{\theta}). \tag{5}$$

Note that,

$$\begin{aligned} f(\boldsymbol{w}, \boldsymbol{\alpha}; \boldsymbol{\theta}) &= \frac{1}{(2\pi)^n |\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}|^{1/2} |\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}|^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{w} - \boldsymbol{\alpha} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}_{\boldsymbol{\varepsilon}} (\boldsymbol{w} - \boldsymbol{\alpha} - \boldsymbol{\mu}) - \frac{1}{2} \boldsymbol{\alpha}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}_{\boldsymbol{\alpha}} \boldsymbol{\alpha} \right\} \\ &\propto \exp[g(\boldsymbol{\alpha})], \end{aligned}$$

where $g(\boldsymbol{\alpha}) = (\boldsymbol{w} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1}_{\boldsymbol{\varepsilon}} \boldsymbol{\alpha} - \boldsymbol{\alpha}^{\mathrm{T}} (\boldsymbol{\Sigma}^{-1}_{\boldsymbol{\varepsilon}} + \boldsymbol{\Sigma}^{-1}_{\boldsymbol{\alpha}}) \boldsymbol{\alpha}/2$. Note that $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}}$ is a block diagonal matrix, and its inverse can be obtained relatively easily, and $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}$ can be a sparse matrix when data size is large. We use an EM procedure to do the estimation. The advantage is that the procedure is scalable to sample size n and parameters can be estimated separately, which can reduce the difficulty of optimization.

3.4.1 Expectation Step

In the expectation step (E-step), at the tth iteration, we have $\boldsymbol{\theta}^{(t-1)} = \left\{ \left[\boldsymbol{\theta}_{\varepsilon}^{(t-1)}\right]^{\mathrm{T}}, \left[\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right]^{\mathrm{T}} \right\}^{\mathrm{T}}$. Let $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}\left[\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right] = \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}$ and $\boldsymbol{\Sigma}_{\varepsilon}\left[\boldsymbol{\theta}_{\varepsilon}^{(t-1)}\right] = \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)}$. The expectation is

$$\mathcal{Q}\left[\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}\right] = \mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}} \log\left[L(\boldsymbol{\theta};\boldsymbol{w},\boldsymbol{\alpha})\right].$$

We need to derive the distribution of $\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}$. The joint distribution for $\boldsymbol{w}$ and $\boldsymbol{\alpha}$ given $\boldsymbol{\theta}_{\varepsilon}^{(t-1)}$ and $\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}$ is

$$\begin{bmatrix} \boldsymbol{\alpha}|\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)} \\ \boldsymbol{w}|\boldsymbol{\theta}_{\varepsilon}^{(t-1)} \end{bmatrix} \sim \mathrm{N}\left\{ \begin{bmatrix} \mathbf{0}_n \\ \boldsymbol{\mu}^{(t-1)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} & \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} \\ \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} & \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} + \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)} \end{bmatrix} \right\},$$

where $\mathbf{0}_n$ is an n-element vector with all zero entries. By the properties of normal distribution, the distribution of $\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}$ is normal with the mean and covariance matrix as

$$\begin{aligned} \mathbf{E}\left[\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right] &= \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}\left[\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} + \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)}\right]^{-1}\left[\boldsymbol{w} - \boldsymbol{\mu}^{(t-1)}\right], \\ \mathrm{Cov}\left[\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right] &= \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} - \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}\left[\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} + \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)}\right]^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}, \end{aligned}$$

respectively. Here, to ensure model estimability, we introduce the zero-sum constraint for $\boldsymbol{\alpha}$, that is $\sum_{i=1}^{n}\alpha_i = 0$. To achieve this, we multiply $\boldsymbol{\alpha}$ by a centering matrix $\mathbf{C} = \mathbf{I} - n^{-1}\mathbf{J}_n$, where $\mathbf{J}_n$ is an $n \times n$ all-ones matrix. The centralized $\boldsymbol{\alpha}$ has a singular multivariate normal distribution with mean and covariance

$$\begin{aligned} \mathbf{E}\left[\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right] &= \mathbf{C}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}\left[\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} + \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)}\right]^{-1}\left[\boldsymbol{w} - \boldsymbol{\mu}^{(t-1)}\right], \\ \mathrm{Cov}\left[\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_{\varepsilon}^{(t-1)},\boldsymbol{\theta}_{\boldsymbol{\alpha}}^{(t-1)}\right] &= \mathbf{C}\left\{\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} - \boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}\left[\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)} + \boldsymbol{\Sigma}_{\varepsilon}^{(t-1)}\right]^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{(t-1)}\right\}\mathbf{C}, \end{aligned}$$

respectively. Expanding the complete likelihood function in (5), we obtain

$$\mathcal{L}(\boldsymbol{\theta}) = \log[L(\boldsymbol{\theta};\boldsymbol{w},\boldsymbol{\alpha})] = \mathcal{L}_1(\boldsymbol{\theta}_{\varepsilon};\boldsymbol{w}|\boldsymbol{\alpha}) + \mathcal{L}_2(\boldsymbol{\theta}_{\boldsymbol{\alpha}};\boldsymbol{\alpha}).$$

Here,

$$\mathcal{L}_1(\boldsymbol{\theta}_{\varepsilon};\boldsymbol{w}|\boldsymbol{\alpha}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Sigma}_{\varepsilon}|) - \frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu}-\boldsymbol{\alpha})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon}^{-1}(\boldsymbol{w}-\boldsymbol{\mu}-\boldsymbol{\alpha}),$$

and

$$\mathcal{L}_2(\boldsymbol{\theta}_{\boldsymbol{\alpha}};\boldsymbol{\alpha}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}|) - \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{-1}\boldsymbol{\alpha}.$$

Taking the expectation with respect to $\boldsymbol{\alpha}$, we have

$$\begin{aligned} \mathcal{Q}_1\left[\boldsymbol{\theta}_{\varepsilon}|\boldsymbol{\theta}_{\varepsilon}^{(t-1)}\right] = &-\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Sigma}_{\varepsilon}|) - \frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon}^{-1}(\boldsymbol{w}-\boldsymbol{\mu}) \\ &+ (\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)} - \frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Sigma}_{\varepsilon}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right] - \frac{1}{2}\left[\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}, \end{aligned} \tag{6}$$

and

$$\mathcal{Q}_2\left[\boldsymbol{\theta}_\alpha|\boldsymbol{\theta}_\alpha^{(t-1)}\right] = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Sigma}_\alpha|) - \frac{1}{2}\text{tr}\left[\boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{\Sigma}_{\alpha|w}^{(t-1)}\right] - \frac{1}{2}\left[\boldsymbol{\mu}_{\alpha|w}^{(t-1)}\right]^{\text{T}}\boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{\mu}_{\alpha|w}^{(t-1)}.$$

The derivation for $\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right]$ and $\mathcal{Q}_2\left[\boldsymbol{\theta}_\alpha|\boldsymbol{\theta}_\alpha^{(t-1)}\right]$ are provided in Appendix A.

3.4.2 M-Step for Parameter Estimation

The updating formulas for $\boldsymbol{\theta}_\varepsilon$ and $\boldsymbol{\theta}_\alpha$ are

$$\boldsymbol{\theta}_\varepsilon^{(t)} = \arg\max_{\boldsymbol{\theta}_\varepsilon}\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right] \text{ and } \boldsymbol{\theta}_\alpha^{(t)} = \arg\max_{\boldsymbol{\theta}_\alpha}\mathcal{Q}_2\left[\boldsymbol{\theta}_\alpha|\boldsymbol{\theta}_\alpha^{(t-1)}\right],$$

respectively. For μ, σ_ε^2, and σ_α^2, we have closed forms for updating as follows,

$$\begin{aligned}
\widehat{\mu} =& \frac{\mathbf{1}^{\text{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}(\boldsymbol{w}-\boldsymbol{\mu}_{\alpha|w})}{\mathbf{1}^{\text{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}\mathbf{1}},\\
n\widehat{\sigma}_\varepsilon^2 =& (\boldsymbol{w}-\widehat{\boldsymbol{\mu}})^{\text{T}}\boldsymbol{\Omega}_\varepsilon^{-1}(\boldsymbol{w}-\widehat{\boldsymbol{\mu}}) - 2(\boldsymbol{w}-\widehat{\boldsymbol{\mu}})^{\text{T}}\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\mu}_{\alpha|w}\\
&+ \text{tr}\left[\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\Sigma}_{\alpha|w}^{(t-1)}\right] + \left[\boldsymbol{\mu}_{\alpha|w}^{(t-1)}\right]^{\text{T}}\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\mu}_{\alpha|w}^{(t-1)},\\
n\widehat{\sigma}_\alpha^2 =& \text{tr}\left[\boldsymbol{\Omega}_\alpha^{-1}\boldsymbol{\Sigma}_{\alpha|w}^{(t-1)}\right] + \left[\boldsymbol{\mu}_{\alpha|w}^{(t-1)}\right]^{\text{T}}\boldsymbol{\Omega}_\alpha^{-1}\boldsymbol{\mu}_{\alpha|w}^{(t-1)}.
\end{aligned}$$

Substituting $\widehat{\mu}$, $\widehat{\sigma}_\varepsilon^2$, and $\widehat{\sigma}_\alpha^2$ into $\mathcal{Q}_1$ and $\mathcal{Q}_2$, we have the profile likelihood for $\boldsymbol{\nu}$, g and $\boldsymbol{\rho}$ as follows,

$$\begin{aligned}
\mathcal{Q}_1\left[\boldsymbol{\nu}, g|\boldsymbol{\theta}_\varepsilon^{(t-1)}, \widehat{\mu}, \widehat{\sigma}_\varepsilon^2\right] &= -\frac{1}{2}\log(|\boldsymbol{\Omega}_\varepsilon|) - \frac{n}{2}\log(\widehat{\sigma}_\varepsilon^2),\\
\mathcal{Q}_2\left[\boldsymbol{\rho}|\boldsymbol{\theta}_\alpha^{(t-1)}, \widehat{\mu}, \widehat{\sigma}_\alpha^2\right] &= -\frac{1}{2}\log(|\boldsymbol{\Omega}_\alpha|) - \frac{n}{2}\log(\widehat{\sigma}_\alpha^2).
\end{aligned}$$

We use the "L-BFGS-B" in the R routine "optim", which is a gradient-based method (Zhu et al. 1995), to solve the optimization problem for $\boldsymbol{\nu}$, g and $\boldsymbol{\rho}$. We use the estimated value of $\boldsymbol{\nu}$, g and $\boldsymbol{\rho}$ by GP as the optimization starting values.

3.4.3 Different μ, σ_ε^2, g, and $\boldsymbol{\nu}$ for Each Category

The model we construct so far shares the same μ, σ_ε^2, g, and $\boldsymbol{\nu}$ in all categories. But in some applications, it is possible that data in different categories behave differently. For example, the throughput for the I/O modes random_reader and reader are different because random_reader tests the speed of reading large amounts of small files while reader tests the speed of reading large files. As a result, we also provide the formula for the model with different μ, σ_ε^2, g, and $\boldsymbol{\nu}$ separately for each category in this section. We refer to this model as LMGP-S. Note that the LMGP-S model still has correlations among different categories.

For category k, $k = 1, \ldots, c$, let $\mathbb{I}_k$ be the set of indexes that all the data points belong to category k. In other words, $\mathbb{I}_k$ is an index set with n_k elements. Let μ_k, $\sigma^2_{\varepsilon,k}$, g_k, and $\boldsymbol{\nu}_k$ be the parameters for the $\boldsymbol{\Sigma}_{\varepsilon,k}$ in category k. Then updating formulas for μ_k and $\sigma^2_{\varepsilon,k}$ in the M step are:

$$\begin{aligned}
\widehat{\mu}_k =& \frac{\mathbf{1}^{\mathrm{T}}_{\mathbb{I}_k} \boldsymbol{\Sigma}^{-1}_{\varepsilon,\mathbb{I}_k} (\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k})}{\mathbf{1}^{\mathrm{T}}_{\mathbb{I}_k} \boldsymbol{\Sigma}^{-1}_{\varepsilon,\mathbb{I}_k} \mathbf{1}_{\mathbb{I}_k}}, \\
n\widehat{\sigma}^2_{\varepsilon,k} =& (\boldsymbol{w}_{\mathbb{I}_k} - \widehat{\boldsymbol{\mu}}_k)^{\mathrm{T}} \boldsymbol{\Omega}^{-1}_{\varepsilon,\mathbb{I}_k} (\boldsymbol{w}_{\mathbb{I}_k} - \widehat{\boldsymbol{\mu}}_k) - 2(\boldsymbol{w}_{\mathbb{I}_k} - \widehat{\boldsymbol{\mu}}_k)^{\mathrm{T}} \boldsymbol{\Omega}^{-1}_{\varepsilon,\mathbb{I}_k} \boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k} \\
& + \operatorname{tr}\left[\boldsymbol{\Omega}^{-1}_{\varepsilon,\mathbb{I}_k} \boldsymbol{\Sigma}^{(t-1)}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}\right] + \left[\boldsymbol{\mu}^{(t-1)}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}\right]^{\mathrm{T}} \boldsymbol{\Omega}^{-1}_{\varepsilon,\mathbb{I}_k} \boldsymbol{\mu}^{(t-1)}_{\boldsymbol{\alpha}|w,\mathbb{I}_k},
\end{aligned}$$

where $\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}$, $\boldsymbol{\Omega}_{\varepsilon,\mathbb{I}_k}$, and $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}$ are the corresponding $n_k \times n_k$ block matrix for all the data points in category k in $\boldsymbol{\Sigma}_{\varepsilon}$, $\boldsymbol{\Omega}_{\varepsilon}$, and $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|w}$. Other formulas for the EM algorithm are the same as derived before. The derivations for $\widehat{\mu}_k$ and $\widehat{\sigma}^2_{\varepsilon,k}$ are provided in Appendix B. Some further technical details for derivatives are given in Appendix C.

3.5 Prediction for Distributional Outcomes

For a new configuration $(\boldsymbol{x}^{\mathrm{T}}_0, \boldsymbol{z}^{\mathrm{T}}_0)^{\mathrm{T}}$, the goal is to predict its distribution function and we can do this by predicting $\boldsymbol{w}_0 = (w_{01}, \ldots, w_{0d})^{\mathrm{T}}$. The prediction is based on those d separate models in (2). Here, we describe how to make the prediction for the jth element of $\boldsymbol{w}_0$ based on the following model,

$$w_{0j} = \mu + \alpha_{0j} + \varepsilon_{0j}, j = 1, \ldots, d.$$

For notation convenience, we drop the index j and work on the following model.

$$w_0 = \mu + \alpha_0 + \varepsilon_0.$$

We construct

$$\begin{pmatrix} w_0 \\ \boldsymbol{w} \end{pmatrix} \sim \mathbf{N}\left[\begin{pmatrix} \mu \\ \mu \mathbf{1}_n \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Sigma}_{00} & \boldsymbol{\Sigma}_{01} \\ \boldsymbol{\Sigma}_{10} & \boldsymbol{\Sigma}_{11} \end{pmatrix}\right].$$

With estimated $\widehat{\boldsymbol{\theta}}$, the predicted $\widehat{w}_0$ is the conditional mean

$$\widehat{w}_0 = \mathbf{E}(w_0|\boldsymbol{w}) = \mu + \boldsymbol{\Sigma}_{01} \left(\boldsymbol{\Sigma}_{11}\right)^{-1} (\boldsymbol{w} - \mu \mathbf{1}_n). \tag{7}$$

Repeat the prediction in (7) for $j = 1, \ldots, d'$ to obtain the prediction for w_0 to obtain $\widehat{w}_0$. Here, $d' \leq d$, d' is the number of SVD components and is chosen by computing budget and prediction accuracy. Another way to determine d' is to let d' be the smallest integer such that $\sum_{j=1}^{d'} \lambda_j / \sum_{j=1}^{d} \lambda_j \geq$ threshold, which is selected to ensure sufficient modeling fidelity for

predictive purposes. Let $\boldsymbol{\beta}_0 = (\beta_{00}, \beta_{01}, \ldots, \beta_{0d})^{\mathrm{T}}$. The predicted coefficients for the splines are recovered by

$$\widehat{\boldsymbol{\beta}}_0 = \widehat{\boldsymbol{w}}_0 \mathbf{V}_{d'}^{\mathrm{T}}, \tag{8}$$

according to the SVD, where $\mathbf{V}_{d'}$ is the first d' columns of $\mathbf{V}$. Because it is possible that some of the elements in $\widehat{\boldsymbol{\beta}}_0$ are negative. These negative entries are truncated to 0. The prediction for the quantile function $Q_0(p)$ is then obtained as

$$\widehat{Q}_0(p) = \widehat{\beta}_{00} + \sum_{j=1}^{d-1} \widehat{\beta}_{0j} \gamma_j(p). \tag{9}$$

Note that truncation at zero is justified because it results in the nearest point to $\widehat{\boldsymbol{\beta}}_0$ in the convex hull that makes $\widehat{Q}_0(p)$ a monotone function. The prediction of the CDF, $\widehat{F}_0(y)$, can be obtained by inverting $\widehat{Q}_0(p)$.

4 Prediction Performance Study Using HPC Data

We first introduce the prediction model variants and the error metric for comparisons. We then demonstrate that the SVD can reduce the dimension of the $\mathbf{B}$ without much loss of accuracy. We compare the prediction performance of the four model variants under the proposed prediction framework. We also visualize the results of predicting quantile functions.

4.1 Prediction Models and Performance Metrics

Our prediction framework can have four variants, depending on the GP model used for predicting $\boldsymbol{w}$. In particular, they are

- LMGP: Linear mixed Gaussian process with common μ, σ_ε^2, g, and $\boldsymbol{\nu}$ for all categories.
- LMGP-S: Linear mixed Gaussian process with different μ, σ_ε^2, g, and $\boldsymbol{\nu}$ for each category.
- GP: Separate simple Gaussian process fitting for each category.
- CGP: Categorical Gaussian process in Zhou et al. (2011). The CGP is a modified GP using the $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}$ with no threshold for $r_{ii'}$ as the variance-covariance matrix.

The GP and CGP can be treated as two special cases of the LMGP. Based on the four model variants, the quantile function $Q_0(p)$ can be predicted using (8) and (9). Because there is no existing methods for comparisons, we compare the prediction accuracy under the four model variants.

The prediction accuracy is measured by comparing the discrepancy between the smoothed CDF and predicted CDF. Let $F(y)$ be the sample CDF and $\widehat{F}(y)$ be the predicted CDF; we

use the errors based on the L^1-norm (EL_1) for error measurement:

$$EL_1 = ||F(y) - \widehat{F}(y)||_1.$$

All the EL_1's in this paper are on the scale of 10^7 KB/s.

We show the prediction accuracy for different prediction tasks on IOzone data. The algorithms are implemented in R (R Core Team 2021). Because our focus is on prediction, we test the prediction framework on real datasets, instead of using simulated datasets. To create multiple datasets for training/testing purposes, we obtain subsets with three I/O modes from the IOzone database.

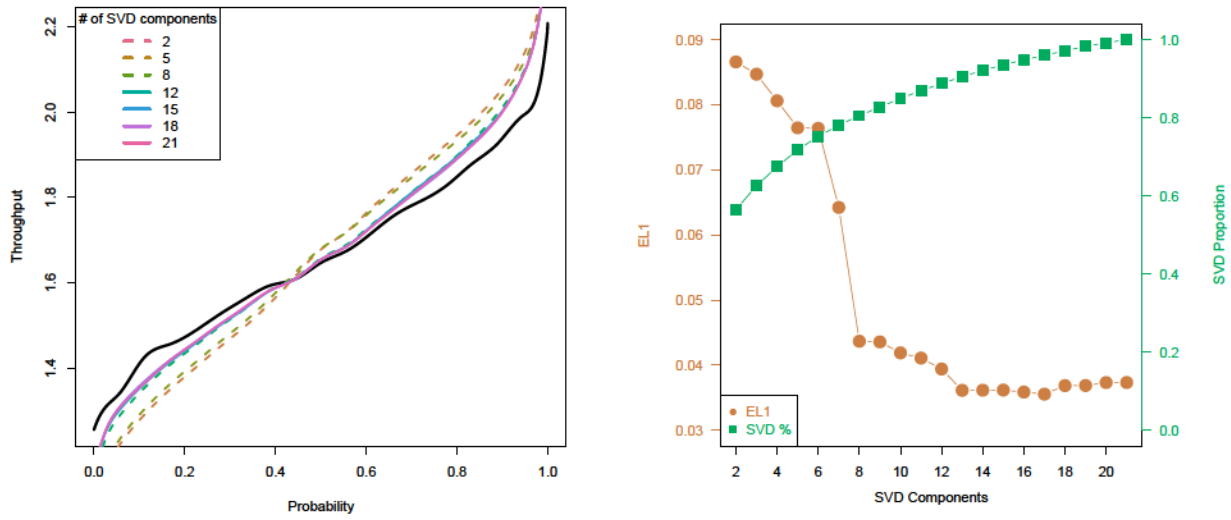
4.2 Dimension Reduction by Selection of d'

In this section, we show the model's potential in reducing the dimension by selecting the number of SVD components d'. The dataset used here contains three modes: random_writer, rereader, and reader. We randomly choose 20% of the data as the test set. Figure 7 shows the predicted curves by LMGP using different numbers of SVD components. In Figure 7(a), when we use more than 12 components, the predicted curves (solid lines) are quite similar and are very close to the true black quantile function. This observation is also confirmed by Figure 7(b). The decreasing trend of the EL_1 vanishes when the number of components increases beyond eight, where about 80% of the singular values are covered. Thus the dimension of $\boldsymbol{\beta}$ can be reduced by SVD without much loss of accuracy. In the rest of this paper, all of our predictions are based on the first 12 SVD components (i.e., $d' = 12$).

4.3 Average Error for Different Training Set Proportions

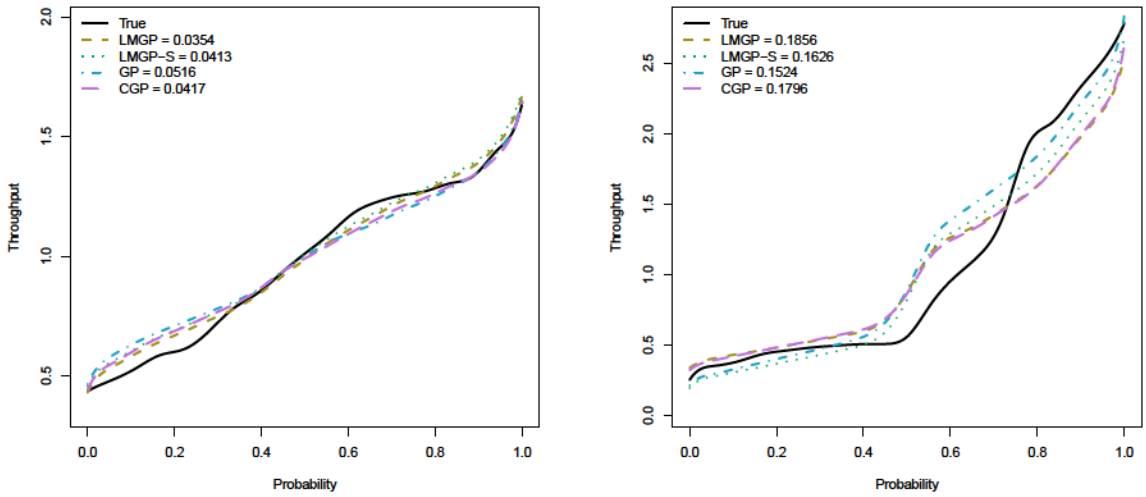
In this section, we discuss the prediction accuracy on different training set proportions. We create five datasets and each dataset is a different combination of the three IO operation modes (the categorical input) from the large IOzone database. For each dataset, the training proportions are from 30% to 70%. To obtain the average error, the random train-test splitting is repeated 100 times. The results for five datasets are shown in Table 2.

From Table 2, we can see that the prediction accuracy generally increases when the proportion of the dataset used for training increases. For most cases, the LMGP-S model variant has the best performance, the LMGP variant is the next one, and the performance of GP and CGP is worse than the LMGP-S, which reveals that there are correlations among data in different I/O modes. In some cases, the GP model variant has the best performance and the performance of LMGP-S is close to that of GP. Overall, the LMGP-S model variant provides the most consistently accurate results.



(a) Predicted Quantile (b) EL_1 and SVD Proportion

Figure 7: Prediction under different numbers of the SVD components. The black solid line in Figure (a) is the smoothed sample quantile function.



(b) Prediction for an Interior Point (a) Prediction for a Boundary Point

Figure 8: Two examples of the predicted quantile functions by using the four models. The $\boldsymbol{x}$ for the left panel is an interior point (Freq=2.3, FS=8192, Thread=24) in the training set while the $\boldsymbol{x}$ for the right panel is at the boundary (Freq=3.5, FS=1024, Thread=8). The legends show the EL_1 for each model.

Table 2: Average EL_1 after multiple train-test splits on multiple datasets with different 3-mode combinations.

Modes in Dataset	Training Proportion	EL_1			
		LMGP	LMGP-S	GP	CGP
random_reader random_writer rereader	0.3	0.0428	0.0427	0.0472	0.0449
	0.4	0.0387	0.0385	0.0430	0.0405
	0.5	0.0367	0.0363	0.0411	0.0381
	0.6	0.0347	0.0342	0.0389	0.0360
	0.7	0.0340	0.0337	0.0382	0.0353
random_writer rereader reader	0.3	0.0423	0.0421	0.0465	0.0443
	0.4	0.0381	0.0380	0.0428	0.0400
	0.5	0.0355	0.0349	0.0404	0.0371
	0.6	0.0339	0.0335	0.0388	0.0359
	0.7	0.0347	0.0340	0.0375	0.0349
rereader reader rewriter	0.3	0.0422	0.0417	0.0453	0.0442
	0.4	0.0381	0.0377	0.0421	0.0399
	0.5	0.0360	0.0355	0.0401	0.0375
	0.6	0.0345	0.0337	0.0384	0.0356
	0.7	0.0343	0.0330	0.0370	0.0342
initial_writer random_reader random_writer	0.3	0.0319	0.0305	0.0307	0.0340
	0.4	0.0298	0.0292	0.0278	0.0305
	0.5	0.0294	0.0268	0.0258	0.0282
	0.6	0.0288	0.0252	0.0248	0.0268
	0.7	0.0297	0.0257	0.0239	0.0258
initial_writer random_writer rereader	0.3	0.0285	0.0295	0.0285	0.0310
	0.4	0.0266	0.0307	0.0267	0.0288
	0.5	0.0256	0.0283	0.0252	0.0269
	0.6	0.0245	0.0257	0.0239	0.0252
	0.7	0.0255	0.0261	0.0235	0.0248

To visualize the prediction results, we provide two examples of the predicted quantile functions in the test set as shown in Figure 8. When the test point is an interior point of the training set (e.g., as the point shown in Figure 8(a)), the predicted quantile functions are quite close to the sample quantile. The predicted curves of LMGP and LMGP-S are closer to the true curve compared with CGP and GP. When the test point is close to the boundary (as shown in Figure 8(b)), the prediction is poor. GP-based models are intended for interpolation (i.e., the $\boldsymbol{x}_0$ is inside the convex hull of the data). When $\boldsymbol{x}_0$ is near the boundary or outside the convex hull of the data, the performance tends to be poor.

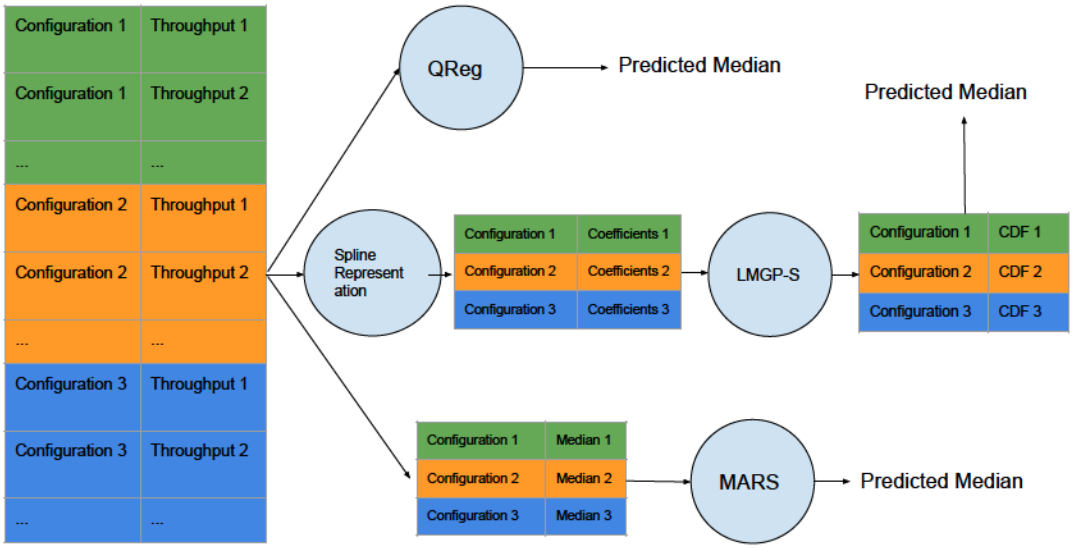


Figure 9: Flow chart that illustrates how data are processed before being fed to different models for median prediction.

5 Predicting Summary Statistics and Comparisons

One application of distributional predictions is to predict the summary statistics of a distribution from the predicted quantile function/CDF. Typical summary statistics can be the mean, standard deviation (SD), and quantile values of the underlying distribution. For predicting summary statistics, there are also existing methods available. Thus, we make comparisons with existing methods in predicting summary statistics in this section.

Xu et al. (2021) study the accuracy of predicting throughput standard deviations using multiple surrogates. For comparison, we have two baseline models which can incorporate both quantitative and qualitative factors. The first baseline model is quantile regression (Koenker and Bassett 1978; Li and Zhang 2021). Quantile regression (QReg) can predict the quantile of the throughput given all the replicated throughputs. The other comparison method is MARS. We use the R package "earth" (Milborrow 2020) for implementing MARS and "quantreg" (Koenker 2021) for implementing QReg. The summary statistics are calculated from the throughputs under a given configuration. Figure 9 provides a flow chart on how data are processed before being fitted to different models. The QReg can take the raw data with replicated throughputs and predict the median directly. For MARS, the sample median is calculated and then used in model training. We use the LMGP-S model variant here. For LMGP-S, the median is calculated from the predicted quantile function. The summary statistic of interest in Figure 9 is the median (i.e., the 0.5 quantile).

The dataset we used for summary statistics prediction has three I/O modes: random_writer,

Table 3: MSEs for the predictions of multiple summary statistics. The QReg cannot predict the sample mean and standard deviation. MSEs are in the unit 10^{14} $(\text{KB/s})^2$.

	LMGP-S	QReg	MARS
Mean	0.0012	n/a	0.0091
SD	0.0013	n/a	0.0040
$Q(0.05)$	0.0033	0.3443	0.0065
$Q(0.10)$	0.0028	0.3830	0.0068
$Q(0.25)$	0.0019	0.4565	0.0067
$Q(0.50)$	0.0109	0.5727	0.0168
$Q(0.75)$	0.0030	0.4220	0.0238
$Q(0.90)$	0.0046	0.4325	0.0365
$Q(0.95)$	0.0057	0.4554	0.0420

rereader, and reader. 20% of the data are randomly chosen to be the test set. The error measure for the summary statistics (a scalar output) is the mean squared error (MSE). Figure 10 shows the scatter plots between the LMGP-S predicted and true summary statistics on the test set. Figure 10 shows the LMGP-S has accurate predictions even for the 0.05 and 0.95 quantiles. Almost all the points are close to the $y = x$ line. Table 3 shows the MSEs for different summary statistics and models. The LMGP-S's MSE is about 1% of the QReg and 20% of the MARS.

The functional prediction framework (implemented with LMGP-S here) can utilize more information in the data. As a result, it can achieve much better results for all summary statistics predictions. The traditional quantile regression does not have good predictions when dealing with complicated data with non-normal underlying distributions. Surrogates like MARS that use the scalar-form summary statistics directly also lose information, which shows the advantage of the proposed model framework.

6 Conclusions and Areas for Future Research

In this paper, we focus on using the spline representation and Gaussian process to predict the I/O throughput distributions given the HPC system configuration. I-splines are used to represent the quantile function and the SVD is used to reduce the dimension. GP-based models are used to predict the SVD scores. The two LMGP models can be viewed as a mixture of the GP and CGP, and they can determine the proportion of GP and CGP automatically. We conduct comparisons between our model framework with some baseline methods. Numerical results show that our prediction framework has good performance in predictions for different

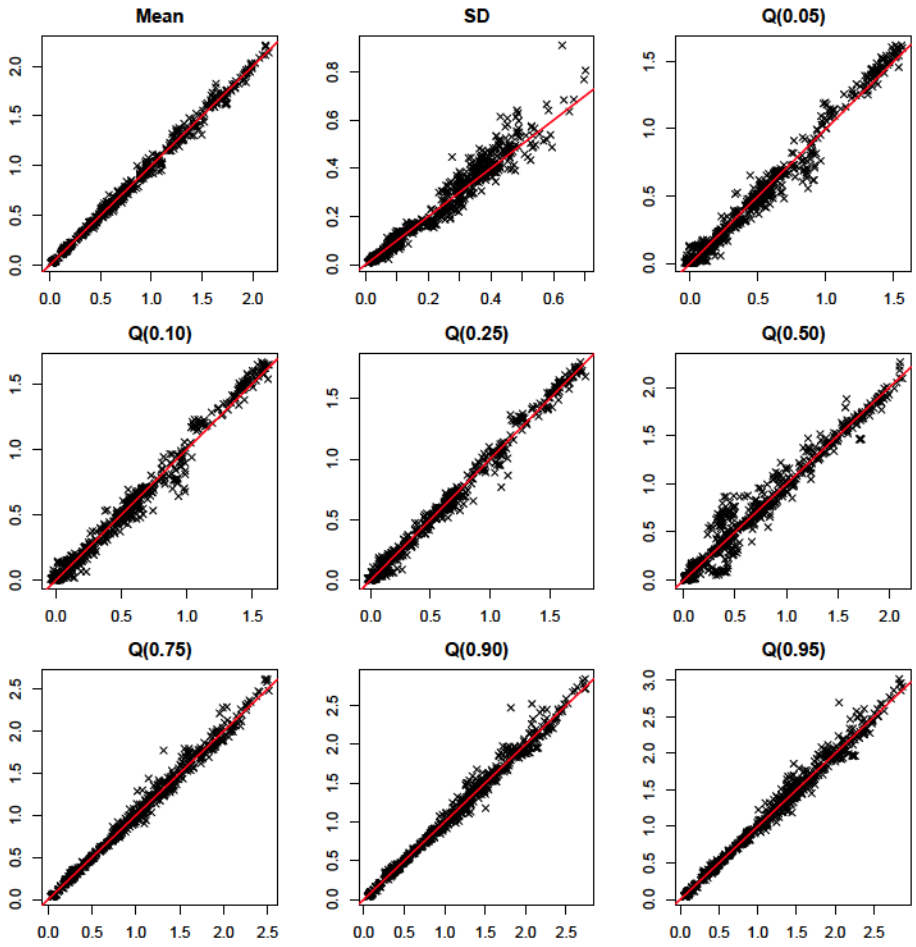


Figure 10: Scatter plots of several predicted summary statistics using LMGP-S. The x-axis is the predicted value and the y-axis is the true value.

subsets of the IOzone data, both in distributional and summary-statistic levels.

One important future step in the management of performance variability is to develop a general tool to predict the throughput distribution for a new system configuration. Our LMGP models capture the relation between the system configuration and throughput distribution. One direct engineering application of the LMGP models is that one can utilize the predicted summary statistics to optimize the HPC system for different perspectives. For example, if we want to ensure a lower bound of the throughput, the 0.2 quantile can be part of the optimization objective. The advantage of our models is that LMGP models can use the distributional information and conduct much more accurate predictions on summary statistics.

When comparing the discrepancy between two distributions, the Kolmogorov–Smirnov (KS) distance is used in some applications. We did not use the KS distance because, in certain situations, the KS distance can be misleading and is sensitive to a distributional shift. In particular, the KS distance can be misleading when a CDF $F(x)$ has a steep behavior (i.e., throughputs have multiple modes), and Xu et al. (2020) show that multimodal behaviors commonly exist through the IOzone data. The KS distance measures the maximal error while EL_1 provides an average discrepancy. We use EL_1 as the error measurements in this paper, which is more appropriate.

One future research area is on data collection. As the number of factors becomes large, it will become impractical to collect a dataset as "dense" as in Figure 2 with functional responses. In this case, it will be interesting to explore a more sparse design in higher dimension as a "screening" step (e.g., Dean and Lewis 2006) to determine which system parameters are most critical for prediction, followed by more extensive data collection in the corresponding subspace.

Restricting Gaussian process models is not a simple matter (e.g., Mitchell and Morris 1992). The approach we used by projecting those negative weights back to the constrained space provides a convenient solution and the results are reasonably well. In the future, it will be interesting to explore other methods in constraining Gaussian process models, such as those described in Swiler et al. (2020).

Note that the number of parameters for $\mathbf{\Sigma}_{\boldsymbol{\alpha}}$ increases at the order of $O(c^2)$. So when we have many categorical levels, the optimization for $\mathcal{L}_2$ will be difficult. As a result, a better parametrization for the categorical inputs can be investigated in future research. In addition, the EM algorithm is computationally intensive when we have a large dataset. Inversion of $\mathbf{\Sigma}_{\boldsymbol{\alpha}}$ is expensive because it becomes a dense matrix when $r_{\max}$ is large. In the future, the estimation efficiency for using a sparse $\mathbf{\Sigma}_{\boldsymbol{\alpha}}$ can be studied.

Another future research is using positive matrix factorizations instead of SVD when decorrelating $\mathbf{B}$. Currently, we use SVD and will introduce negative entries in $\mathbf{W}$. Several methods

for positive matrix factorizations in Hopke (2000) can be studied for decorrelating $\mathbf{B}$. We will also perform simulation studies to further study the model estimability to predict $\boldsymbol{w}$. Last but not least, it will be interesting to study how the errors are propagated from I-spline smoothing to LMGP modeling in our prediction framework.

Acknowledgments

The authors acknowledge Advanced Research Computing at Virginia Tech for providing computational resources. The research was supported by National Science Foundation Grants CNS-1565314 and CNS-1838271 to Virginia Tech.

A Formulas for the $\mathcal{Q}$ Functions

We show the formulas for the $\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right]$ and $\mathcal{Q}_2\left[\boldsymbol{\theta}_\alpha|\boldsymbol{\theta}_\alpha^{(t-1)}\right]$. After the conditional expectation is taken, we obtain,

$$
\begin{aligned}
&\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right] = \mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_\varepsilon^{(t-1)},\boldsymbol{\theta}_\alpha^{(t-1)}}\mathcal{L}_1(\boldsymbol{\theta}_\varepsilon;\boldsymbol{w}|\boldsymbol{\alpha})\\
=&\mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_\varepsilon^{(t-1)},\boldsymbol{\theta}_\alpha^{(t-1)}}\left[-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log(|\boldsymbol{\Sigma}_\varepsilon|)-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu}-\boldsymbol{\alpha})^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}(\boldsymbol{w}-\boldsymbol{\mu}-\boldsymbol{\alpha})\right]\\
=&-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log(|\boldsymbol{\Sigma}_\varepsilon|)-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}(\boldsymbol{w}-\boldsymbol{\mu})\\
&+\frac{1}{2}\mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_\varepsilon^{(t-1)},\boldsymbol{\theta}_\alpha^{(t-1)}}\left[2\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}(\boldsymbol{w}-\boldsymbol{\mu})-\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon\boldsymbol{\alpha}\right]\\
=&-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log(|\boldsymbol{\Sigma}_\varepsilon|)-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}(\boldsymbol{w}-\boldsymbol{\mu})\\
&+(\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}-\frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Sigma}_\varepsilon^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right]-\frac{1}{2}\left[\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Sigma}_\varepsilon^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)},
\end{aligned}
$$

and

$$
\begin{aligned}
&\mathcal{Q}_2\left[\boldsymbol{\theta}_\alpha|\boldsymbol{\theta}_\alpha^{(t-1)}\right] = \mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_\varepsilon^{(t-1)},\boldsymbol{\theta}_\alpha^{(t-1)}}\mathcal{L}_2(\boldsymbol{\theta}_\alpha;\boldsymbol{\alpha})\\
=&\mathbf{E}_{\boldsymbol{\alpha}|\boldsymbol{w},\boldsymbol{\theta}_\varepsilon^{(t-1)},\boldsymbol{\theta}_\alpha^{(t-1)}}\left[-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log(|\boldsymbol{\Sigma}_\alpha|)-\frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{\alpha}\right]\\
=&-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log(|\boldsymbol{\Sigma}_\alpha|)-\frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right]-\frac{1}{2}\left[\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|\boldsymbol{w}}^{(t-1)}.
\end{aligned}
$$

B Derivations for Different Parameters in Each Category

When we have different $\mu, \boldsymbol{\nu}, g$, and σ_ε^2 for each category, the $\boldsymbol{\theta}_\varepsilon$ becomes

$$\boldsymbol{\theta}_\varepsilon = (\sigma_{\varepsilon,1}^2, \mu_1, \boldsymbol{\nu}_1^{\mathrm{T}}, g_1, \ldots, \sigma_{\varepsilon,k}^2, \mu_k, \boldsymbol{\nu}_k^{\mathrm{T}}, g_k, \ldots, \sigma_{\varepsilon,c}^2, \mu_c, \boldsymbol{\nu}_c^{\mathrm{T}}, g_c)^{\mathrm{T}}.$$

Let $\boldsymbol{\theta}_{\varepsilon,k} = (\sigma_{\varepsilon,k}^2, \mu_k, \boldsymbol{\nu}_k^{\mathrm{T}}, g_k)^{\mathrm{T}}$, the $\boldsymbol{\Sigma}_\varepsilon$ becomes

$$\boldsymbol{\Sigma}_\varepsilon = \mathrm{Diag}\left(\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_1}, \ldots, \boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}, \ldots, \boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_c}\right) = \mathrm{Diag}\left(\sigma_{\varepsilon,1}^2\boldsymbol{\Omega}_{\varepsilon,\mathbb{I}_1}, \ldots, \sigma_{\varepsilon,k}^2\boldsymbol{\Omega}_{\varepsilon,\mathbb{I}_k}, \ldots, \sigma_{\varepsilon,c}^2\boldsymbol{\Omega}_{\varepsilon,\mathbb{I}_c}\right).$$

Using the block diagonal structure of $\boldsymbol{\Sigma}_\varepsilon$, the $\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right]$ in (6) becomes:

$$\begin{aligned}
&\mathcal{Q}_1\left[\boldsymbol{\theta}_\varepsilon|\boldsymbol{\theta}_\varepsilon^{(t-1)}\right] = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\sum_{k=1}^{c}\log(|\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}|) - \frac{1}{2}\sum_{k=1}^{c}(\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}(\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k}) \\
&+ \sum_{k=1}^{c}(\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)} - \frac{1}{2}\sum_{k=1}^{c}\mathrm{tr}\left[\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)}\right] - \frac{1}{2}\sum_{k=1}^{c}\left[\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)} \\
=&\sum_{k=1}^{c}\mathcal{Q}_1\left[\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}|\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}^{(t-1)}\right],
\end{aligned}$$

where,

$$\begin{aligned}
&\mathcal{Q}_1\left[\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}|\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}^{(t-1)}\right] = -\frac{n_k}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}|) - \frac{1}{2}(\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}(\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k}) \\
&+ (\boldsymbol{w}_{\mathbb{I}_k} - \boldsymbol{\mu}_{\mathbb{I}_k})^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)} - \frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)}\right] - \frac{1}{2}\left[\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Sigma}_{\varepsilon,\mathbb{I}_k}^{-1}\boldsymbol{\mu}_{\boldsymbol{\alpha}|w,\mathbb{I}_k}^{(t-1)}.
\end{aligned}$$

Thus, $\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}$ can be estimated through maximizing $\mathcal{Q}_1\left[\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}|\boldsymbol{\theta}_{\varepsilon,\mathbb{I}_k}^{(t-1)}\right]$ separately.

C Formulas for the Score Functions

We use the gradient-based method for optimization. Here we provide the approximated score function for the likelihoods $\mathcal{Q}_1$ and $\mathcal{Q}_2$ for the LMGP model. For $\mathcal{Q}_1$, we assume $\widehat{\mu}$ does not depend on $\boldsymbol{\nu}$ and g. Then, for ν_l ,$l = 1, \ldots, p$, we have

$$\begin{aligned}
\frac{\partial\mathcal{Q}_1}{\partial\boldsymbol{\nu}} &= -\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Omega}_\varepsilon^{-1}\frac{\partial\boldsymbol{\Omega}_\varepsilon}{\partial\boldsymbol{\nu}}\right) - \frac{n}{2\widehat{\sigma}_\varepsilon^2}\frac{\partial\widehat{\sigma}_\varepsilon^2}{\partial\boldsymbol{\nu}}, \\
\frac{\partial\left(\boldsymbol{\Omega}_\varepsilon\right)_{ij}}{\partial\nu_l} &= -\frac{(x_{id} - x_{jd})^2}{\nu_l^2}\exp\left[-\sum_{l=1}^{p}\frac{(x_{il} - x_{jl})^2}{\nu_l}\right], \\
n\frac{\partial\widehat{\sigma}_\varepsilon^2}{\partial\nu_l} &= (\boldsymbol{w} - \widehat{\boldsymbol{\mu}})^{\mathrm{T}}\boldsymbol{\Omega}_\varepsilon^{-1}\frac{\partial\boldsymbol{\Omega}_\varepsilon}{\partial\nu_l}\boldsymbol{\Omega}_\varepsilon^{-1}(\boldsymbol{w} - \widehat{\boldsymbol{\mu}}) - 2(\boldsymbol{w} - \widehat{\boldsymbol{\mu}})^{\mathrm{T}}\boldsymbol{\Omega}_\varepsilon^{-1}\frac{\partial\boldsymbol{\Omega}_\varepsilon}{\partial\nu_l}\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\mu}_{\alpha|w} \\
&\quad + \mathrm{tr}\left[\boldsymbol{\Omega}_\varepsilon^{-1}\frac{\partial\boldsymbol{\Omega}_\varepsilon}{\partial\nu_l}\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\Sigma}_{\alpha|w}^{(t-1)}\right] + \left[\boldsymbol{\mu}_{\alpha|w}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Omega}_\varepsilon^{-1}\frac{\partial\boldsymbol{\Omega}_\varepsilon}{\partial\nu_l}\boldsymbol{\Omega}_\varepsilon^{-1}\boldsymbol{\mu}_{\alpha|w}^{(t-1)}.
\end{aligned}$$

For g, we can get $\frac{\partial \mathcal{Q}_1}{\partial g}$ similarly using $\frac{\partial \boldsymbol{\Omega}_\varepsilon}{\partial g} = \mathbf{I}$.

For $\mathcal{Q}_2$, we first have an alternative expression for $\boldsymbol{\Omega}_\alpha$, $\boldsymbol{\Omega}_\alpha = \mathbf{A}^{\mathrm{T}} (\mathbf{P} \otimes \boldsymbol{\Phi}) \mathbf{A}$, where $\otimes$ is the Kronecker product, $\boldsymbol{\Phi}$ is an $n \times n$ matrix satisfies $\Phi_{ii'} = \kappa(r_{ii'}, r_{\max})$, and $\mathbf{A}$ is an $n \times c$ matrix. The ith row of $\mathbf{A}$ has the kth element equal to 1 and the rest $(c-1)$ elements equal to 0, where k is the level of $\boldsymbol{z}_i$ after sorting. Then we have

$$\frac{\partial \mathcal{Q}_2}{\partial \boldsymbol{\rho}} = -\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Omega}_\alpha^{-1}\frac{\partial \boldsymbol{\Omega}_\alpha}{\partial \boldsymbol{\rho}}\right) - \frac{n}{2\widehat{\sigma}_\alpha^2}\frac{\partial \widehat{\sigma}_\alpha^2}{\partial \boldsymbol{\rho}}, \qquad \frac{\partial \boldsymbol{\Omega}_\alpha}{\partial \boldsymbol{\rho}} = \mathbf{A}^{\mathrm{T}}\left(\frac{\partial \mathbf{P}}{\partial \boldsymbol{\rho}} \otimes \boldsymbol{\Phi}\right)\mathbf{A},$$
$$n\frac{\partial \widehat{\sigma}_\alpha^2}{\partial \boldsymbol{\rho}} = \mathrm{tr}\left[\boldsymbol{\Omega}_\alpha^{-1}\frac{\partial \boldsymbol{\Omega}_\alpha}{\partial \boldsymbol{\rho}}\boldsymbol{\Omega}_\alpha^{-1}\boldsymbol{\Sigma}_{\alpha|w}^{(t-1)}\right] + \left[\boldsymbol{\mu}_{\alpha|w}^{(t-1)}\right]^{\mathrm{T}}\boldsymbol{\Omega}_\alpha^{-1}\frac{\partial \boldsymbol{\Omega}_\alpha}{\partial \boldsymbol{\rho}}\boldsymbol{\Omega}_\alpha^{-1}\boldsymbol{\mu}_{\alpha|w}^{(t-1)}.$$

Then $\partial \boldsymbol{\Omega}_\alpha / \partial \boldsymbol{\rho}$ can be expressed using $\partial \mathbf{P} / \partial \boldsymbol{\rho}$.

References

Akkan, H., M. Lang, and L. M. Liebrock (2012). Stepping towards noiseless linux environment. In *Proceedings of the 2nd International Workshop on Runtime and Operating Systems for Supercomputers*, pp. 1–7.

Auder, B., A. De Crecy, B. Iooss, and M. Marquès (2012). Screening and metamodeling of computer experiments with functional outputs. application to thermal–hydraulic computations. *Reliability Engineering and System Safety 107*, 122–131.

Bayarri, M., J. Berger, J. Cafeo, G. Garcia-Donato, F. Liu, J. Palomo, R. Parthasarathy, R. Paulo, J. Sacks, D. Walsh, et al. (2007). Computer model validation with functional output. *The Annals of Statistics 35*, 1874–1906.

Box, G. E. P. and K. B. Wilson (1951). On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society: Series B (Methodological) 13*, 1–38.

Cameron, K. W., A. Anwar, Y. Cheng, L. Xu, B. Li, U. Ananth, J. Bernard, C. Jearls, T. Lux, Y. Hong, L. T. Watson, and A. R. Butt (2019). MOANA: Modeling and analyzing I/O variability in parallel system experimental design. *IEEE Transactions on Parallel and Distributed Systems 30*, 1843–1856.

Capps, D. and W. Norcott (2008). Iozone filesystem benchmark. `https://www.iozone.org`.

Chang, T. H., L. T. Watson, T. C. Lux, J. Bernard, B. Li, L. Xu, G. Back, A. R. Butt, K. W. Cameron, and Y. Hong (2018). Predicting system performance by interpolation using a high-dimensional Delaunay triangulation. In *Proceedings of the High Performance Computing Symposium*, pp. 12.

Chipman, H. A., E. I. George, and R. E. McCulloch (2002). Bayesian treed models. *Machine Learning 48*, 299–320.

Chipman, H. A., E. I. George, and R. E. McCulloch (2010). BART: Bayesian additive regression trees. *The Annals of Applied Statistics 4*, 266–298.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association 74*, 829–836.

Dean, A. and S. Lewis (2006). *Screening: Methods for Experimentation in Industry, Drug Discovery, and Genetics*. New York: Springer.

Deng, X., C. D. Lin, K.-W. Liu, and R. K. Rowe (2017). Additive Gaussian process for computer models with qualitative and quantitative factors. *Technometrics 59*, 283–292.

Drignei, D. (2010). Functional ANOVA in computer models with time series output. *Technometrics 52*, 430–437.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics 19*, 1–67.

Fruth, J., O. Roustant, and S. Kuhnt (2015). Sequential designs for sensitivity analysis of functional inputs in computer experiments. *Reliability Engineering and System Safety 134*, 260–267.

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

Hammouda, A., A. R. Siegel, and S. F. Siegel (2015). Noise-tolerant explicit stencil computations for nonuniform process execution rates. *ACM Transactions on Parallel Computing (TOPC) 2*, 1–33.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media.

Higdon, D., J. Gattiker, B. Williams, and M. Rightley (2008). Computer model calibration using high-dimensional output. *Journal of the American Statistical Association 103*, 570–583.

Hopke, P. K. (2000). A guide to positive matrix factorization. In *Workshop on UNMIX and PMF as Applied to PM2*, Volume 5, pp. 600.

Hung, Y., V. R. Joseph, and S. N. Melkote (2015). Analysis of computer experiments with functional response. *Technometrics 57*, 35–44.

Jiang, F., M. H. Y. Tan, and K.-L. Tsui (2021). Multiple-target robust design with multiple functional outputs. *IISE Transactions 53*, 1052–1066.

Kim, Y., L. K. John, S. Pant, S. Manne, M. Schulte, W. L. Bircher, and M. S. S. Govindan (2012). Audit: Stress testing the automatic way. In *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 212–223.

Koenker, R. (2021). *quantreg: Quantile Regression*. R package version 5.85.

Koenker, R. and G. Bassett (1978). Regression quantiles. *Econometrica 46*, 33–50.

Li, C. and H. Zhang (2021). Tensor quantile regression with application to association between neuroimages and human intelligence. *The Annals of Applied Statistics 15*, 1455–1477.

Lux, T. C., L. T. Watson, T. H. Chang, J. Bernard, B. Li, X. Yu, L. Xu, G. Back, A. R. Butt, K. W. Cameron, D. Yao, and Y. Hong (2018). Novel meshes for multivariate interpolation and approximation. In *Proceedings of the ACMSE 2018 Conference*, pp. 13.

Ma, P., A. Mondal, B. A. Konomi, J. Hobbs, J. J. Song, and E. L. Kang (2022). Computer model emulation with high-dimensional functional output in large-scale observing system uncertainty experiments. *Technometrics 64*, 65–79.

Meyer, M. C. (2008). Inference using shape-restricted regression splines. *The Annals of Applied Statistics 2*, 1013–1033.

Milborrow, S. (2020). *earth: Multivariate Adaptive Regression Splines*. R package version 5.3.0.

Mitchell, T. J. and M. D. Morris (1992). Bayesian design and analysis of computer experiments: Two examples. *Statistica Sinica 2*, 359–379.

Ouyang, J., B. Kocoloski, J. R. Lange, and K. Pedretti (2015). Achieving performance isolation with lightweight co-kernels. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 149–160.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Rahimi, A., D. Cesarini, A. Marongiu, R. K. Gupta, and L. Benini (2015). Task scheduling strategies to mitigate hardware variability in embedded shared memory clusters. In *Proceedings of the 52nd Annual Design Automation Conference*, pp. 1–6.

Ramsay, J. O. (1988). Monotone regression splines in action. *Statistical Science 3*, 425–441.

Rasmussen, C. E. and C. K. I. Williams (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Simonian, J. (2010). The most simple methodology to create a valid correlation matrix for risk management and option pricing purposes. *Applied Economics Letters 17*, 1767–1768.

Swiler, L., M. Gulian, A. Frankel, C. Safta, and J. Jakeman (2020). A survey of constrained Gaussian process regression: Approaches and implementation challenges. *Journal of Machine Learning for Modeling and Computing 1*, 119–156.

Taddy, M. A., R. B. Gramacy, and N. G. Polson (2011). Dynamic trees for learning and design. *Journal of the American Statistical Association 106*, 109–123.

Tan, M. H. Y. (2018). Gaussian process modeling of a functional output with information from boundary and initial conditions and analytical approximations. *Technometrics 60*, 209–221.

Thacker, W. I., J. Zhang, L. T. Watson, J. B. Birch, M. A. Iyer, and M. W. Berry (2010). Algorithm 905: SHEPPACK: Modified Shepard algorithm for interpolation of scattered multivariate data. *ACM Transactions on Mathematical Software 37*, 34:1–34:20.

Wang, Y., L. Xu, Y. Hong, R. Pan, T. Chang, T. Lux, J. Bernard, L. Watson, and K. Cameron (2022). Design strategies and approximation methods for high-performance computing variability management. *Journal of Quality Technology*, doi:10.1080/00224065.2022.2035285.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics 4*, 389–396.

Xu, L., T. Lux, T. Chang, B. Li, Y. Hong, L. Watson, A. Butt, D. Yao, and K. Cameron (2021). Prediction of high-performance computing input/output variability and its application to optimization for system configurations. *Quality Engineering 33*, 318–334.

Xu, L., Y. Wang, T. Lux, T. Chang, J. Bernard, B. Li, Y. Hong, K. Cameron, and L. Watson (2020). Modeling I/O performance variability in high-performance computing systems using mixture distributions. *Journal of Parallel and Distributed Computing 139*, 87–98.

Zhou, Q., P. Z. Qian, and S. Zhou (2011). A simple approach to emulation for computer models with qualitative and quantitative factors. *Technometrics 53*, 266–273.

Zhu, C., R. Byrd, P. Lu, and J. Nocedal (1995). A limited memory algorithm for bound constrained optimisation. *SIAM Journal on Scientific Computing 16*, 1190–1208.