Statistica Sinica Preprint No: SS-2017-0138	
Title	Surrogate-Assisted Tuning for Computer Experiments with Qualitative and Quantitative Parameters
Manuscript ID	SS-2017-0138
URL	http://www.stat.sinica.edu.tw/statistica/
DOI	10.5705/ss.202017.0138
Complete List of Authors	Ray-Bing Chen Jiahong Chen Akihiro Fujii Reiji Suda and Weichung Wang
Corresponding Author	Ray-Bing Chen
E-mail	rbchen@mail.ncku.edu.tw
Notice: Accepted version subject to English editing.	

Statistica Sinica: Newly accepted Paper
(accepted version subject to English editing)

Surrogate-Assisted Tuning for Computer Experiments with Qualitative and Quantitative Parameters

Jiahong K. Chen[a], Ray-Bing Chen[b], Akihiro Fujii[c], Reiji Suda[d] and Weichung Wang[e]

[a] *Vpon Inc., Taiwan*

[b] *Department of Statistics, National Cheng Kung University, Taiwan*

[c] *Department of Computer Science, Kogakuin University, Japan*

[d] *Department of Computer Science, The University of Tokyo, Japan*

[e] *Institute of Applied Mathematical Sciences, National Taiwan University, Taiwan*

April 11, 2017 (4:57pm)

Abstract

Performance tuning of computer codes is an essential issue in computer experiments. By suitable choosing the values of the tuning parameters, we can optimize the codes in terms of timing, accuracy, robustness, or other performance objectives. As computer software and hardware are becoming more and more complicated, such a tuning process is not an easy task, and there are strong needs for developing efficient and automatic tuning methods. In this article, we consider software auto-tuning problems that involve qualitative and quantitative tuning parameters by solving the resulting optimization problems. Because the performance objective functions in the target optimization problems are usually not explicitly defined, we build up surrogates from the response data and attempt to mimic the true, yet unknown, performance response surfaces. The proposed surrogate-assisted tuning process is an iterative procedure. At each iteration, surrogates are updated and new experimental points are chosen based on the prediction uncertainties provided by the surrogate models until a satisfactory solution is obtained. We propose two surrogate construction methods that adopt two infill criteria for the tuning problems containing qualitative and quantitative parameters. The four variants of the proposed algorithm are used to optimize computational fluid dynamic simulation codes and artificial problems to illustrate the usefulness and strengths of the proposed algorithms.

Keywords: Qualitative and quantitative parameters, computer experiments, performance tuning, surrogates modeling, infill criteria, expected improvement, Gaussian process

1 Introduction

Numerical simulations have been widely used in natural science, engineering, and social sciences under various scenarios. The successes of numerical simulations rely on

the efficiency of computer codes. One key component for producing efficient computer codes is software tuning, which involves finding the optimal parameters within algorithms and codes to achieve optimal performance. Such a tuning process is difficult, if not impossible, to perform manually. It is not trivial because of the increasing complexities of algorithms, software, and computer architectures. Consequently, software automatic performance tuning (auto-tuning in short) plays a key role in computer code efficiency.

Auto-tuning has been studied intensively using various approaches. In early achievements such as ATLAS [1], FFTW [2], SPIRAL [3], and ABCLib_Script [4], manually created models and exhaustive search (with heuristic pruning) were used. More general-purpose auto-tuners such as Active Harmony [5] and OpenTuner [6] have a set of search algorithms, such as exhaustive, random, Hill-Climbing, Nelder-Mead, SA, GA, and PSO. Some researchers used machine learning techniques for auto-tuning, such as regression trees [7], neural networks [8, 9], decision trees [9, 10], PCA plus k-nearest neighbors [11], and kernel canonical correlation analysis [12]. Auto-tuning is also called algorithm configuration in some literature. To solve this type of problems, several iterative algorithms have been developed. At each iteration, one needs to select the configuration candidates, evaluate the performance to compare with the incumbent optimal configuration, and then update the current best parameter setting if necessary. Candidate selection can be performed by Random Online Aggressive Racing [13] or the genetic algorithm [14].

The aforementioned auto-tuning approaches solve the optimization problems (to choose the next testing parameters or configurations) by direct search type methods without relying on performance models. In contrast, statistical model based methods are considered in auto-tuning literature such as [15, 16]. For example, Gramacy et al. [17] have studied computer codes auto-tuning problems involving the qualitative variables (and the ordinal variables). They propose a statistical a surrogate analysis approach that is based on the "dynamic tree" model. Second, the Sequential Model-Based Algorithm Configuration (SMAC) is proposed in [13]. SMAC uses Gaussian process, tree model, random forest, or others are used to form a surrogate of the performance model. The method uses efficient global optimization [18] and the expected improvement criterion to select the next configuration candidate. Furthermore, an intensification step is used to make sure what the best point is. Third, an auto-tuning scheme that searches the parameters based on a discretized spline model is studied in [19]. The idea is to use a "d-spline" function as fitting model function. It assumes continuity of performance value over the parameter space. It increases the sampling points and updates the d-spline function until the estimated optimal point is found.

In this article, we focus on auto-tuning problems that involve quantitative and qualitative (Q&Q) tuning parameters. Such a Q&Q auto-tuning problem is defined as an optimization problem

$$\min_{\mathbf{w}\in\mathbf{\Omega}} y(\mathbf{w}) \tag{1}$$

by using the following notations. Let $\mathbf{x} = (x_1, \ldots, x_I)^t \in \mathbf{\Omega}_{QN} \subset \mathbb{R}^I$ be the vector containing I quantitative tuning parameters and $\mathbf{z} = (z_1, \ldots, z_J)^t \in \mathbf{\Omega}_{QL} \subset \mathbb{R}^J$ be the vector containing J qualitative tuning parameters. $\mathbf{\Omega}_{QN}$ and $\mathbf{\Omega}_{QL}$ are the quantitative and qualitative experimental domains, respectively. We further assume that the qualitative parameters z_j have q_j levels, and thus, a Q&Q auto-tuning problem has M *cases* for $M = \prod_{j=1}^{J} q_j$. Finally, we let $\mathbf{w} = (\mathbf{x}^t, \mathbf{z}^t)^t \in \mathbf{\Omega}$ for $\mathbf{\Omega} = \mathbf{\Omega}_{QN} \times \mathbf{\Omega}_{QL} \subset \mathbb{R}^{I+J}$ and denote the objective function as $y(\mathbf{w})$.

We take the Algebraic Multigrid (AMG) linear system solver [20, 21, 22] as an example to illustrate the optimization problem (1). Linear systems arise in many numerical simulations such as fluid dynamics [23] and elastic analysis [24]. Solving these linear systems is usually the most expensive part of the simulations. To accelerate the running time of an AMG solver, both quantitative and qualitative parameters need to be tuned, and the number of all possible combinations of the tuning parameters can be large. For example, in the AMG implementation introduced in [25], there are 10^6 different combinations of the tuning parameters. It is thus not practical to test all of the combinations to find the shortest running time. Rather than using this exhaustive search approach, we propose a systematic way of finding the suitable combinations of the parameters. Therefore, the execution time of the AMG solver and the associated numerical simulations can be reduced as much as possible. Example 1 is one particular example of the Q&Q auto-tuning optimization problems. We will show how the proposed methods can be applied to solve this problem.

Example 1 (Algebraic Multigrid Linear System Solver) *We assume* $I = J = 2$, $q_1 = 2$, $q_2 = 3$, $M = 6$, *and* $\mathbf{w} = (x_1, x_2, z_1, z_2)^t$ *in this example. The quantitative tuning parameter* x_1 *is the threshold for strong coupling [26], and* x_2 *is the relaxation parameter of smoother [27]. To simplify the tuning procedure, we set a grid over the domain of these two quantitative parameters. In particular,* $x_1 \in \{0.02, 0.025, 0.03, \ldots, 0.08, 0.09, 0.11\}$ *contains* 15 *grid points and* $x_2 \in \{0.6, 0.8, 1.0, \ldots, 1.8\}$ *contains* 7 *grid points. Consequently, the corresponding qualitative experiment domain* $\mathbf{\Omega}_{QN}$ *is a two-dimensional domain with* 15×7 *grid points. The qualitative parameter* z_1 *can be either the Symmetric Gauss-Seidel smoother (SGS) or the Gauss-Seidel smoother (GS) [27]. The qualitative parameter* z_2 *can be the V-, F-, or W-cycle in the AMG solver [27]. The objective function* $y(\mathbf{w})$ *is the total execution time of the AMG solver with the tuning vector* $\mathbf{w}$*. There are* $15 \times 7 \times 2 \times 3 = 630$ *possible combinations of* $\mathbf{w}$ *in total.*

Such Q&Q auto-tuning problems are common in various computer simulation codes. However, to the best of our knowledge, there are few efficient and systematic ways to perform auto-tuning for this type of problem. For example, ppOpen-AT [28], an auto-tuning framework, allows library developers to implement auto-tuning functionality only by inserting tuning directives to the codes. Many functionalities have already been prepared for the directives. For example, d-spline parameter search, loop unrolling, loop splitting, and loop fusion can be implemented only by inserting one or two directive lines. All combination of transformed code patterns and parameter values are checked for the best performance. As it does not distinguish Q&Q parameters, it is difficult to deal with the problems in an efficient way other than the exhaustive searches. Also, a parameter tuning study targeting to a multigrid solver is conducted in [29] by using an auto-tuning language called PetaBricks. The multigrid method is a multi-level method for solving linear equations. The study optimized the combinations of parameter settings at each level for multiple convergence criteria. It succeeded in reducing the parameter space by dynamic programming, was able to choose the optimized parameter setting according to a computing environment, and required convergence levels. However, this study does not distinguish the Q$Q parameters and thus are not recommended to our target problem.

It is worth mentioning that SMAC and other Sequential Model-based Optimization (SMBO) algorithms [13, 14] can solve the Q&Q problem. However, these algorithms are designed to handle different type of problems, and these algorithms are different

with the ones to be proposed in this article. The main difference is that SMBO and SMAC are designed with the assumptions that the responses contain noises. Usually, a large number of runs and extra procedures for updating objective function values are necessary. In contrast, we assume the responses are deterministic and focus on the approaches that can quickly identify a near optimal parameter setting by minimizing the number of the explored points. Note that we assume the responses are deterministic because the performance of the numerical solvers considered in this article has small variations of the response when we repeat running the numerical solver with the same parameter set-up. Thus for a given AMG solver problem, the goal is to efficiently identify a good parameter set-up with fewer computational cost. Second, for the Q&Q modeling, a modified Gaussian Process GP to deal with qualitative factors and the corresponding kernel function is proposed in [13]. In our QQ-GP, the setting of our kernel function is more general, because we also quantify the effect of the different levels of qualitative factors. Third, an expected improvement criterion for log-transformed response is considered in [13]. In our approach, we propose modified expected improvement criterion for the different types of the surrogate modeling approaches.

In short, our target problem has the following challenges:(i) the response functions involve two different types of parameters, (ii) the objective functions are measurable yet not explicitly defined, and (iii) we usually need to determine the optimal parameter by using only a few experimental trials without checking all possible parameter set-ups. To overcome these challenges and solve the Q&Q auto-tuning problem efficiently, we propose a Surrogate-Assisted Tuning (SAT) framework, along with well-developed statistical technologies and the key is to construct the surrogate model as an approximation of the true objective functions based on few experimental trials. The SAT solves the derived optimization problem in an iterative manner. At the beginning, SAT constructs a surrogate based on a set of experimental points. Then, based on the prediction uncertainty, the next explored point is determined via a certain infill criterion. The surrogate is then updated based on the response of the explored point. This process is repeated until the (near) optimal solution is found or the available resources have been consumed. The two key components of SAT are (i) surrogate construction and (ii) infill criteria. For the surrogate constructions, we rely on the Gaussian process [30] and the qualitative and quantitative Gaussian process [31, 32] to construct the surrogates. In the Q&Q setting, we propose two infill criteria that are based on the expected improvement [33]. The expected improvement is designed to search the next experimental point for optimizing the objective function and improving the surrogate. When the problem contains only quantitative parameters, the SAT framework is similar to the Design and Analysis of Computer Experiment (DACE) [30] or the Efficient Global Optimization (EGO) approach [33].

This paper is organized as follows. We first introduce the framework of surrogate-assisted tuning in Section 2. The details of the surrogate constructions and infill criteria are discussed in Section 3 and Section 4, respectively. The proposed algorithms are summarized in Section 2. Numerical studies of the proposed algorithms are presented in Section 5. Finally, the paper is concluded in Section 6. Table 1 presents the notations used in this article.

I	number of quantitative parameters
J	number of qualitative parameters
q_j	number of levels of the jth qualitative parameter
M	No. of all possible combinations in qualitative parameters ($M = \prod_{j=1}^{J} q_j$)
n	number of experimental points
N	number of repetitions in numerical experiments
t	the (tth) iteration number in SAT-QQ
T	maximum iteration numbers allowed in SAT-QQ (as the stopping criterion)
N_{mesh}	number of uniform grid points in each dimension of Ω_{QN}

Table 1: Notations used in this article.

2 Surrogate-Assisted Tuning for Q&Q Parameters (SAT-QQ)

We propose solving the Q&Q auto-tuning optimization problem (1) using the Surrogate-Assisted Tuning framework as outlined in Algorithm 1. This algorithm is called SAT-QQ and is composed of initialization, tuning iteration, and a stopping criterion. The details of the algorithm are discussed as follows.

In the initialization stage of SAT-QQ, we perform the following steps.

- In Step (1.1), we select an initial design $\mathbf{D}$ containing n experimental points $\mathbf{w}^{(i)}$ for $i = 1, \cdots, n$. Note that it is common to apply a space-filling design such as Latin hypercube design (LHD) [34, 35], uniform design [36], or the sliced LHD [37].
- In Step (1.2), we evaluate the corresponding objective functions $y^{(i)} = y(\mathbf{w}^{(i)})$.

In the tuning stage, we (i) construct or update the surrogate, (ii) select a new experimental point based on a certain infill criterion over the current surrogate, (iii) evaluate the function value of the new experimental point, and (iv) update the design and the objective function value vector. These steps are included in the algorithm as shown below.

- In Step (2.1), we need to construct or update the surrogates. We will discuss how we can construct the surrogates in the Q&Q domain $\mathbf{\Omega}$ in Sections 3. These methods are motivated by the Gaussian process (or kriging) [30].
- In Step (2.2), we need to choose the new experimental point. We will discuss how we can find the next experimental point by using the proposed infill criteria that are motivated by the expected improvement (EI) criterion [33].
- In Steps (2.3) and (2.4), we evaluate the corresponding objective functions and update $\mathbf{D}$, $\mathbf{y}$, and $y_{\min}$.

The aforementioned iteration ends until the pre-defined stopping criteria are satisfied. One popular criterion involves the available computing resources such as the total runtime or iteration number.

Algorithm SAT-QQ sketches how we can solve the tuning problems (1). Depending on the choices of surrogate constructions and the infill criteria, the SAT-QQ has four variants: SDGP-SDEI, SDGP-WDEI, WDQQ-SDEI, and WDQQ-WDEI. In the next

sections, we elaborate on how the surrogates are constructed in Section 3 and how the infill criteria are determined in Section 4.

Algorithm 1 Surrogate-Assisted Tuning for Q&Q Parameters (SAT-QQ)

(1) `Initialization`
  (1.1) Select an initial design $\mathbf{D} = \{\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(n)}\}$ in $\mathbf{\Omega}$.
  (1.2) Evaluate the function values $y^{(i)} = y(\mathbf{w}^{(i)})$.
  Set $\mathbf{y} = \left(y^{(1)}, y^{(2)}, \ldots, y^{(n)}\right)^t$ and $y_{\min} = \min_{\mathbf{w}\in\mathbf{D}} y(\mathbf{w})$.
(2) `Tuning Iteration`
  Repeat until stopping criterion is satisfied.
  (2.1) Construct/Update the surrogate based on $\mathbf{D}$ and $\mathbf{y}$ `% surrogate`
  by SDGP (Section 3.2) or WDQQ (Section 3.3).
  (2.2) Determine the next experimental point $\mathbf{w}^+$ `% infill criterion`
  by SDEI (Section 4.2) or WDEI (Section 4.3.)
  (2.3) Evaluate the function value $y^+ = y(\mathbf{w}^+)$.
  (2.4) Update $\mathbf{D} = \mathbf{D} \cup \mathbf{w}^+$ and $\mathbf{y} = (\mathbf{y}; y^+)$ and $y_{\min} = \min(y_{\min}, y^+)$.
(3) `Output`
  (3.1) Output $y_{\min}$ and $\mathbf{w}_{\min} = \arg\min_{\mathbf{w}\in\mathbf{D}} y(\mathbf{w})$.

3 Surrogate Construction

Surrogate construction and updates in Step (2.1) of Algorithm 1 play a key role in the SAT-QQ procedure. A Gaussian process (kriging) is commonly used in computer experiments to construct surrogates based on the stationary assumption [30] of quantitative parameters. For example, Mariani et al. [38] adopted the Gaussian process as the foundation of the OSCAR to deal with a auto-tuning problem. On the other hand, when qualitative parameters are considered, Gramacy et. al. [17] suggested using dynamic tree model. It is worth noting that this surrogate model can also deal with non-stationary surface. Here, we consider the computer experiments involving both of qualitative and quantitative parameters. Based on these qualitative parameters, two models related to Gaussian processes for surrogate constructions are proposed. The first method is called Sub-Domain with Gaussian Process and introduced in Section 3.2. The second method is callsed Whole-Domain with Qualitative and Quantitatibe Gaussian Process and presended in Section 3.3.

3.1 Gaussian Process (GP)

We briefly outline how the Gaussian process is constructed on a quantitative domain $\mathbf{\Omega}_{QN}$. For more details on the GP, see [30, 34], for example. In the Gaussian process, the response function $y(\mathbf{x})$ is characterized as a stochastic process $Y(\mathbf{x})$ in $\mathbf{\Omega}_{QN}$. That is, at point $\mathbf{x}$, the response value $Y(\mathbf{x})$ is a random variable and the observed response value $y^{(k)} = y(\mathbf{x}^k)$ is treated as a realization at the experimental point $\mathbf{x}^{(k)}$, $1 \le k \le n$. Furthermore, we assume that

$$Y(\mathbf{x}) = f(\mathbf{x})^t\beta + \varepsilon(\mathbf{x}), \tag{2}$$

where $f(\mathbf{x}) = (f_1(\mathbf{x}),\ \ldots,\ f_p(\mathbf{x}))^t \in \mathbb{R}^p$ is a pre-specified regression vector, $\beta \in \mathbb{R}^p$ is the corresponding coefficient vector, and $\varepsilon(\mathbf{x})$ is a stationary Gaussian process with mean zero and covariance matrix $\sigma^2 R$. Then, the correlation function of any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ is usually defined as $K_\theta(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\sum_{i=1}^{I} \theta_i (x_{1i} - x_{2i})^2\right)$, where θ_i's are positive parameters and $\theta = (\theta_1, \ldots, \theta_I)$.

Suppose we have n input vectors in the design matrix $X = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\right]^t \in \mathbb{R}^{n \times I}$, and let $\mathbf{y} = \left(y^{(1)}, y^{(2)}, \ldots, y^{(n)}\right)^t \in \mathbb{R}^n$. The unknown parameters, β, σ^2, and θ, are estimated using the maximal likelihood approach. $F = \left[f(\mathbf{x}^{(1)}), f(\mathbf{x}^{(2)}), \ldots, f(\mathbf{x}^{(n)})\right]^t \in \mathbb{R}^{n \times p}$; $R = (R_{ij}) \in \mathbb{R}^{n \times n}$ with $R_{ij} = K_\theta(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$. First, given θ,

$$\widehat{\beta} = (F^t R^{-1} F)^{-1} F^t R^{-1} \mathbf{y}$$

and $\widehat{\sigma}^2 = (\mathbf{y} - F\widehat{\beta})^t R^{-1} (\mathbf{y} - F\widehat{\beta})/n$, and then, θ can be estimated via the following minimization problem,

$$\arg\min_{\theta > 0} \left\{ n \log(\widehat{\sigma}^2) + \log(|R|) \right\}.$$

Thus, the prediction of a particular point $\mathbf{x}$ is

$$\widehat{y}(\mathbf{x}) = f(\mathbf{x})^t \widehat{\beta} + r^t R^{-1} (\mathbf{y} - F\widehat{\beta}), \tag{3}$$

where $r \in \mathbb{R}^n$ is the correlation vector with $r_j = K_{\hat{\theta}}(\mathbf{x}, \mathbf{x}^{(j)})$ and the mean squared error (MSE) of $\widehat{y}(\mathbf{x})$ is

$$\widehat{s}^2(\mathbf{x}) = \widehat{\sigma}^2 \left[1 - r^t R^{-1} r + u^t (F^t R^{-1} F)^{-1} u\right], \tag{4}$$

where $u = F^t R^{-1} r - f(\mathbf{x})$.

We conclude the discussion on the Gaussian process by the following two remarks. (i) The prediction $\widehat{y}(\mathbf{x})$ is the empirical best linear unbiased predictor (EBLUP), and the interpolation property $\widehat{y}(\mathbf{x}^{(k)}) = y(\mathbf{x}^{(k)})$ and $\widehat{s}^2(\mathbf{x}^{(k)}) = 0$ is satisfied [39]. (ii) The prediction at a point $\mathbf{x}$ is a realization of a normal distribution with mean $\widehat{y}(\mathbf{x})$ and variance $\widehat{s}^2(\mathbf{x})$. That is,

$$Y(\mathbf{x}) \sim N(\widehat{y}(\mathbf{x}), \widehat{s}^2(\mathbf{x})). \tag{5}$$

3.2 Sub-Domain with Gaussian Process (SDGP)

The first approach to include the qualitative parameters is to apply the "divide-and-conquer" technique on the experimental domain $\mathbf{w}$. Because the Q&Q auto-tuning optimization problem defined in (1) has M cases, we divide the experimental domain $\mathbf{\Omega}$ into M disjoint sub-domains $\mathbf{\Omega}_m$, $m = 1, \ldots, M$. By doing so, each $\mathbf{\Omega}_m$ involves only the quantitative factors. Consequently, we can apply the Gaussian Process (GP) discussed in Section 3.1 on each sub-domain $\mathbf{\Omega}_m$ independently to construct M GP-based surrogates. That is, all the derivations regarding GP on Ω_{QN} can be applied to Ω_m. For example, the prediction of a certain point $\mathbf{x} \in \Omega_m$ is

$$\widehat{y}_m(\mathbf{x}) = f(\mathbf{x})^t \widehat{\beta}_m + r_m^t R_m^{-1} (\mathbf{y}_m - F_m \widehat{\beta}_m). \tag{6}$$

3.3 Whole-Domain with Qualitative and Quantitative Gaussian Process (WDQQ)

The second one is an "all-in-one" type approach named Whole-Domain Qualitative and Quantitative Gaussian Process (WDQQ). In the SDGP, we need to fit M independent Gaussian processes for each case, and this surrogate modeling approach does not take the correlation effects of the quantitative factors among the different cases into account. In contrast, a modification of the Gaussian process was proposed in [31, 32] to incorporate both qualitative and quantitative variables simultaneously. In other words, this modified Gaussian process can be used to construct a surrogate over the whole experimental domain $\mathbf{\Omega}$ in one shot.

In WDQQ, the response model for an explored point $\mathbf{w}$ is represented as

$$Y(\mathbf{w}) = f(\mathbf{w})^t\beta + \varepsilon(\mathbf{w}), \tag{7}$$

where $f(\mathbf{w}) = (f_1(\mathbf{w}),\ \ldots,\ f_p(\mathbf{w}))^t$ is the vector of the pre-specified functions on $\mathbf{\Omega}$, β is the unknown coefficient vector, and $\varepsilon(\mathbf{w})$ is assumed to be a Gaussian process with mean zero and constant variance σ^2.

The key step in the WDQQ is to define the proper correlation function for the different levels of the qualitative factors. We first denote all of the M cases as $\mathbf{z}_1, \ldots, \mathbf{z}_M$ with $\mathbf{z}_m = (z_{m1}, \ldots, z_{mJ})^t$, and any experimental point can be represented as $\mathbf{w} = (\mathbf{x}^t, \mathbf{z}^t)^t$. Then by following [32], for any two points $\mathbf{w}_i = (\mathbf{x}_i^t, \mathbf{z}_i^t)^t$, $i = 1, 2$, the correlation between $\varepsilon(\mathbf{w}_1)$ and $\varepsilon(\mathbf{w}_2)$ is defined as

$$cor\left(\varepsilon(\mathbf{w}_1), \varepsilon(\mathbf{w}_2)\right) = \tau_{z_1,z_2} K_\theta(\mathbf{x}_1, \mathbf{x}_2), \tag{8}$$

where τ_{z_1,z_2} is the cross-correlation between $\mathbf{z}_1$ and $\mathbf{z}_2$. To estimate the unknown parameters $\tau_{r,s}$ and θ, an efficient maximum likelihood estimation (MLE) approach via the hypersphere parameterization was proposed in [32] to model the correlations of the qualitative factors.

Finally, at an unexplored point $\mathbf{w}$, the prediction $\widehat{y}(\mathbf{w})$ and the mean square error $\widehat{s}^2(\mathbf{w})$ can be derived by replacing $\mathbf{x}$ by $\mathbf{w}$ in the formulas of GP. That is,

$$\widehat{y}(\mathbf{w}) = f(\mathbf{w})^t\widehat{\beta} + r^t R^{-1}(\mathbf{y} - F\widehat{\beta}) \tag{9}$$

and

$$\widehat{s}^2(\mathbf{w}) = \widehat{\sigma}^2\left[1 - r^t R^{-1} r + u^t (F^t R^{-1} F)^{-1} u\right]. \tag{10}$$

Here, $\widehat{\beta} = (F^t R^{-1} F)^{-1} F^t R^{-1}\mathbf{y} \in \mathbb{R}^p$; $F = \left[f(\mathbf{w}^{(1)}), f(\mathbf{w}^{(2)}), \ldots, f(\mathbf{w}^{(n)})\right]^t \in \mathbb{R}^{n\times p}$; $R = (R_{ij}) \in \mathbb{R}^{n\times n}$ with $R_{ij} = cor\left(\varepsilon(\mathbf{w}_i), \varepsilon(\mathbf{w}_j)\right)$, and $r \in \mathbb{R}^n$ is the correlation vector with $r_j = cor\left(\varepsilon(\mathbf{w}), \varepsilon(\mathbf{w}_j)\right)$.

3.4 A Short Summary

The SDGP and WDQQ have their own advantages and disadvantages. We compare these two approaches here theoretically as well as numerically later in Section 5.

First, the WDQQ concerns the correlation structure between the qualitative variables and constructs a global surrogate model containing all of the quantitative and qualitative variables. Unlike the SDGP, information in different sub-domains is integrated for the surrogate construction in WDQQ. From this viewpoint, a global surrogate may fit the true function better. However, compared with one quantitative GP,

the WDQQ contains more parameters, and the correlation matrix in WDQQ may have more chances to become ill-conditioned. In addition, in WDQQ, we have the stationary assumption, but SDGP might deal with some minor non-stationary cases because the variance estimations for the different domains can be different.

Second, the interplay between the modeling accuracy and computational efficiency of SDGP and WDQQ depends on the problem. The WDQQ assumes that the correlation parameters θ_i's are the same in all cases. That is, different sub-domains share the same θ_i's. However, the SDGP constructs the surrogate independently for each sub-domain with different θ_i's. If the true surface in each sub-domain is quite different, one identical θ_i will not fit the correlation structure in each sub-domain properly. While we can modify the correlation structures in the WDQQ accordingly, it would increase the complexity of the covariance matrix and thus increase the computational cost.

Finally, we compare the cost of surrogate construction, particularly the cost for solving the MLE problems, in the SDGP and WDQQ as follows.

- Dimensions of the MLE problems.
  In SDGP, we need to solve the MLE problems with dimension I in each sub-surrogate. In WDQQ, the dimension is $I+M(M-1)/2$, if the product correlation structure (8) is used.
- Complexity for inverting the matrices.
  To compute the prediction value and regression coefficients, a symmetric positive definite linear system $R \cdot x = b$ is solved at the cost of cubic order of the size of R [40]. Let n_m be the experimental point size in $\mathbf{\Omega}_m$ and $n = \sum_{m=1}^{M} n_m$ be the total experimental point size in $\mathbf{\Omega}$. The computational complexity of SDGP and WDQQ is $O(M \cdot \max_{m=1,\ldots,M} n_m^3)$ and $O(n^3)$, respectively.
- Cost of updating the surrogates.
  In the SDGP, when a new experimental point is added, only the corresponding kth sub-surrogate model has to be updated with the cost $O(n_m^3)$. In contrast, in the WDQQ, the global surrogate must be updated with a larger cost $O(n^3)$.

4 Infill Criteria

The second main component of the SAT-QQ is to select the new experimental point using a certain infill criterion. This infill criterion is embedded in the iterative tuning process (i.e., Step (2.2) of Algorithm 1) where we iteratively select the next experimental point to explore the experimental domain and search for the optimal solution. Furthermore, if the stopping criterion has not been reached, we will include the information due to the newly selected point to update the surrogates. To select the next experimental point in Step (2.2), we propose two approaches that are inspired by the expected improvement (EI) criterion [33, 41]. The key part is to integrate these infill criteria with the surrogate constructions proposed in Section 3. We propose these approaches in the following sections.

4.1 Expected Improvement (EI)

The EI selection criterion is proposed in [33] and is widely used in optimization problems with quantitative factors for searching the new experimental point. So that the

experimental space can be exploited locally and explored globally in an efficient way, based on the notations and derivations in Section 3.1 regarding a Gaussian process, we discuss expected improvement (EI) in this section.

First, we consider the experimental region with only the quantitative variable $\mathbf{\Omega}_{QN}$. In this case, we define the improvement function of a point $\mathbf{x} \in \mathbf{\Omega}_{QN}$ as

$$I(\mathbf{x}) = \max(y_{min} - Y(\mathbf{x}),\ 0), \tag{11}$$

where $y_{min} = \min\left\{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\right\}$ is the observed minimum in $\mathbf{\Omega}_{QN}$. This definition of the improvement function can be interpreted as the quantity of improvement at a certain point $\mathbf{x}$, and it is positive only if $Y(\mathbf{x}) < y_{min}$. Based on the GP assumption, the response at $\mathbf{x}$ (i.e. $Y(\mathbf{x})$) satisfies (5). Consequently, instead of measuring the improvement quantity $I(\mathbf{x})$ directly, we compute the expected value of the improvement function

$$EI(\mathbf{x}) = \mathbb{E}[I(\mathbf{x})] \tag{12}$$

as our new experimental point selection criterion. In other words, let $\mathbf{x}^+$ be the next experimental point to be explored, and we define

$$\mathbf{x}^+ = \underset{\mathbf{x} \in \mathbf{\Omega}_{QN}}{\arg\max}\, E(I(\mathbf{x})). \tag{13}$$

Note that it can be shown that

$$\mathbb{E}[I(\mathbf{x})] = (y_{min} - \widehat{y}(\mathbf{x}))\Phi\left(\frac{y_{min} - \widehat{y}(\mathbf{x})}{\widehat{s}(\mathbf{x})}\right) + \widehat{s}(\mathbf{x})\phi\left(\frac{y_{min} - \widehat{y}(\mathbf{x})}{\widehat{s}(\mathbf{x})}\right). \tag{14}$$

$\Phi(\cdot)$ and $\phi(\cdot)$ are the standard normal distribution function and density function, respectively. Equation (14) suggests that the EI infill criterion tries to balance the prediction-based local exploitation $y_{min} - \widehat{y}(\mathbf{x})$ and the error-based global exploration $\widehat{s}(\mathbf{x})$.

The EI criterion aforementioned can be used only for the quantitative variables. To link in the information on the qualitative variables, we propose two variants of this original EI in the next two sections.

4.2 Sub-Domain Expected Improvement (SDEI)

This is a direct extension of EI. Similar to Section 3.2, we can divide the experimental domain $\mathbf{\Omega}$ into M disjoint sub-domains $\mathbf{\Omega}_m$. Because each $\mathbf{\Omega}_m$ involves only the quantitative factors, we can apply the EI criterion to each $\mathbf{\Omega}_m$. First, we choose the sub-domain $\mathbf{\Omega}_{m^*}$ with the minimal prediction value. That is,

$$m^* = \underset{m=1,\ldots,M}{\arg\min}\ (\widetilde{y}_m)\,, \text{ where } \widetilde{y}_m = \min_{\mathbf{x} \in \mathbf{\Omega}_m}\ (\widehat{y}(\mathbf{x}))\,. \tag{15}$$

Then, we apply EI criterion on $\mathbf{\Omega}_{m^*}$ to select the next experimental point

$$\mathbf{x}^+ = \underset{\mathbf{x} \in \mathbf{\Omega}_{m^*}}{\arg\max}\,(E(I(\mathbf{x}))\,. \tag{16}$$

We call this infill criterion the Sub-Domain Expected Improvement (SDEI).

Problem	n_0	M	$n_0 \times M$	T	N	N_{mesh}	Regression functions
AMG solver	6	6	36	90	100	12	$f_1 = 1,\ f_2 = x_1,\ f_3 = x_2$
gabor_lv3	6	3	18	90	100	32	$f_1 = 1$

Table 2: Experiment settings.

4.3 Whole-Domain Expected Improvement (WDEI)

In the Whole-Domain Expected Improvement (WDEI), we link information at different levels of the qualitative factors together to define the WDEI for selecting the new experimental point. Let y_m^* be the minimal observed value on the sub-domain $\mathbf{\Omega}_m$. Define

$$y_g^* = \min_{m=1,\ldots,M} y_m^*$$

to be the currently *global* minimal observed response on the whole domain $\mathbf{\Omega}$. We define the "cross domain improvement function"

$$I_g(\mathbf{w}) = \max(y_g^* - Y(\mathbf{w}),\ 0) \text{ for } \mathbf{w} \in \mathbf{\Omega}, \tag{17}$$

where the prediction $Y(\mathbf{w})$ can be obtained from SDGP or WDQQ. The cross domain expected improvement is defined as

$$EI_g(\mathbf{w}) = \mathbb{E}[I_g(\mathbf{w})]. \tag{18}$$

The next experimental point is then chosen as

$$\mathbf{w}^+ = \arg\max_{\mathbf{w}\in\mathbf{\Omega}} EI_g(\mathbf{w}). \tag{19}$$

Theorem 2 asserts that the WDEI (18) has properties similar to that of the original EI. Because the proof is similar to that for the original EI, we omit the proof here.

Theorem 2 (Properties of WDEI) *The improvement function defined in (17) has the following properties. (i)* $\mathbb{E}[I_g(\mathbf{w})] > 0$ *if and only if* $\widehat{s}^2(\mathbf{w}) > 0$. *(ii) If* $\widehat{s}^2(\mathbf{w}) > 0$, *there is a closed form for the expected improvement*

$$\mathbb{E}[I_g(\mathbf{w})] = \left(y_g^* - \widehat{y}(\mathbf{w})\right) \Phi\left(\frac{y_g^* - \widehat{y}(\mathbf{w})}{\widehat{s}(\mathbf{w})}\right) + \widehat{s}(\mathbf{w})\phi\left(\frac{y_g^* - \widehat{y}(\mathbf{w})}{\widehat{s}(\mathbf{w})}\right), \tag{20}$$

where $\Phi(\cdot)$ *and* $\phi(\cdot)$ *are the standard normal distribution function and density function, respectively.*

5 Numerical Experiments

We conduct numerical experiments to study the performance of the proposed algorithms. The algorithms are implemented in MATLAB, and the numerical experiments are conducted on a workstation with Intel Xeon X5570 CPU (8 cores) and 48 GB of main memory. To generate the initial design in Step (1.1) of Algorithm 1, we simply use Latin hypercube design (LHD) [42] for the quantitative variables in each case independently. The sizes of LHD in each case are all the same. We let n_0 denote the number of the initial design points in each sub-domain $\mathbf{\Omega}_k$ and let $n_0 M$ denote

Qualitative level	λ	ϑ	ψ	σ	γ	Function
1	1.0	$\pi/4$	2	0.5	2.0	g_{re}
2	1.2	$\pi/4$	3	0.5	1.8	g_{re}
3	0.8	$\pi/4$	4	0.5	2.2	g_{im}

Table 3: Definitions of the three categories of the qualitative parameter in garbor_lv3.

the total number of initial design points. We partition the quantitative domain $\mathbf{\Omega}_{QN}$ into uniform grids with N_{mesh} points for each dimension. The stopping criterion in the Repeat-loop is the maximal number of iterations, which is denoted as T. Other experimental settings are listed in Table 2.

We have tested the proposed algorithms for the following two types of tuning problems.

- **TP1.** In the first test problem (TP1), we consider the AMG solver introduced in Example 1. The AMG solver is used to solve anisotropic (ani) or isotropic (iso) problems in physical simulations with Conjugate Gradient (cg) or Biconjugate gradient stabilized (bicgstab) linear system solvers [22, 25]. That is, we consider four AMG solvers that are denoted as amg_ani_cg, amg_ani_bicgstab, amg_iso_cg, and amg_iso_bicgstab.
- **TP2.** The second type of test problems (TP2) involve artificial oscillatory surfaces. The surface, named gabor_lv3, is constructed using the Gabor functions

$$g_{re} = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda} + \psi\right)$$

and

$$g_{im} = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\sin\left(2\pi\frac{x'}{\lambda} + \psi\right),$$

where $x' = x\cos\vartheta + y\sin\vartheta$ and $y' = -x\sin\vartheta + y\cos\vartheta$. In this test problem, we consider one qualitative parameter with three qualitative levels. See Table 3 for the definitions of the three qualitative levels. In short, we assume $\mathbf{\Omega} = [-3, 3]^2 \times \{1, 2, 3\}$.

5.1 Numerical Results

We study the performance of the four variants of SAT-QQ in this section in terms of relative errors, hit frequency, quantile curves, execution time, and comparison with random selection.

- **Relative errors.** Let y^*_{min} denote the minimal objective function value out of all possible parameter combinations and $y^{(t)}_{min}$ denote the observed minimum at the tth iteration. To measure the solution qualities after SAT-QQ has been iterated T times, we compute the relative error

$$e_R = \frac{\mid y^{(T)}_{min} - y^*_{min} \mid}{\mid y^*_{min} \mid}. \tag{21}$$

Table 4: Results of TP1

Problem	Meas.	SDGP		WDQQ	
		SDEI	WDEI	SDEI	WDEI
(a)	f_{hit} (%)	85	60	**97**	87
amg_ani_cg	mean(e_R)	2.1238E-02	7.7617E-02	**1.0194E-02**	3.9869E-02
(b)	f_{hit} (%)	27	**28**	13	14
amg_ani_bicgstab	mean(e_R)	3.3310E-02	**2.2610E-02**	4.4969E-02	4.9670E-02
(c)	f_{hit} (%)	58	73	75	**76**
amg_iso_cg	mean(e_R)	1.6217E-02	**2.1621E-03**	7.5348E-03	8.5935E-03
(d)	f_{hit} (%)	60	75	**78**	75
amg_iso_bicgstab	mean(e_R)	2.5353E-02	1.6276E-02	**1.4383E-02**	1.6332E-02

Table 5: Results of TP2

Problem	Meas.	SDGP		WDQQ	
		SDEI	WDEI	SDEI	WDEI
(a)	f_{hit} (%)	16	18	33	**96**
gabor_lv3	mean(e_R)	1.6619E-01	1.7340E-01	1.3472E-01	**1.6012E-02**

Tables 4 and 5 show that the four methods SDGP-SDEI, SDGP-WDEI, WDQQ-SDEI, and WDQQ-WDEI achieve similar relative errors after $T = 90$ iterations in the four test problems. Most of the relative errors are in the same order, while there are a couple of exceptions. One such exception is TP2-(a). In this problem, the infill criterion WDEI significantly outperforms the SDEI by two orders.

- **Sub-domain hit frequency.** To check whether the algorithms can find the *right* sub-domain (i.e., case), we define the sub-domain hit frequency

$$f_{hit} = \frac{N_{hit}}{N}. \tag{22}$$

N_{hit} is the number of SAT-QQ search processes that locate the sub-domain containing the optimal point, and N is the number of repetitions. In other words, the hit frequency measures the chance that $\mathbf{w}_{min}^{(T)}$ and $\mathbf{w}^*$ belong to the same sub-domain. $\mathbf{w}_{min}^{(T)}$ is the argument of the minimal found by the SAT-QQ in the Tth iteration, and $\mathbf{w}^*$ is the argument attaining the minimal (i.e., $\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbf{\Omega}} y(\mathbf{w})$). In our numerical experiments, $\mathbf{w}^*$ is identified by searching all the possible arguments in Ω. It is clear that the higher f_{hit} is, the better the result is.

Table 4 suggests that WDQQ-SDEI and WDQQ-WDEI have similar performances for TP1-(b) and TP1-(c). WDQQ-SDEI is slightly better for TP1-(a) and TP-(d). For TP1, SDGP-WDEI significantly outperforms the other approaches. The reasons behind these results can be understood from the shapes of the response surfaces. From all the possible parameter combinations, we plot

the "true" response surfaces as shown in Figure 1 for the problem amg_ani_cg. The surfaces in each of the sub-domains are somewhat simple, as there are certain main trends of some small vibrations. In contrast, the true surface of gabor_lv3 shown in Figure 2 is oscillatory. As the true surfaces of TP1 are simple, the surrogates can capture the main trends and therefore search the optimal points on the "right" sub-domain with a small number of experimental points. In contrast, as shown in Table 5, the true surfaces of TP2 are oscillatory, and WDQQ considers both the qualitative and quantitative factors and evaluates the WDEI of all of the sub-domains, which can lead to a better understanding of the surface properties.

In the problem TP1-(b), f_{hit} are low for all four approaches. This is because the minimal values in each sub-domain are very close in value to each other. It is thus hard to distinguish these sub-domains in terms of the objective function values even though we emphasize that the relative errors are small in these problems. In other words, all four variants of SAT-QQ can still find good experimental points whose objective function values are close to the optimal one.

- **Quantile curves.** Because the performance of the SAT-QQ depends on the initial designs and number of iterations, we repeat the numerical experiments N times with different initial space-filling designs and then plot the quantile curves to analyze how $y_{min}^{(t)}$ converges with the iterations. The quantile curves show the 5-percentile, 95-percentile, and median of the N values of $y_{min}^{(t)}$ in the tth iteration.

  In TP1, SDGP has a better chance of achieving smaller objective functions with fewer iterations, as shown in the median and 5% quantile curves presented in Figures 3, 4, 5, and 6. However, SDGP may stick in a "wrong" sub-domain due to the lack of a global view across the sub-domains. In such cases, the surrogates of the other sub-domains may not be updated. Consequently, the objective function values may not decrease as shown in the 95% quantile curves. In contrast, the 95% quantile curves produced by WDQQ decrease if the number of iterations is large enough.

  In TP2, the quantile curves due to WDQQ-WDEI decrease faster than other approaches as shown in Figures 7. This observation is parallel to the results and explanations in the discussions of relative errors mentioned above.

- **Timing.** Table 6 lists the computational cost in time (seconds) for constructing the surrogates of all $N = 100$ repetitions. The table shows that WDQQ is much more expensive than SDGP, regardless of whether SDEI or WDEI is used. The main cost of each surrogate construction is the optimization problem for parameter estimation in SDGP and WDQQ, respectively. Another reason is that because WDQQ integrates all the information from all of the sub-domains, the size of the correlation matrix, R, is larger than that of SDGP. Therefore, a greater computational cost is borne due to the computation of the inverse of R. Note that the computational complexity is shown in Section 3.4. Consequently, if the available tuning resource in time is limited, the SDGP is preferred for surrogate construction.

- **Results of a random method.** We attempted to solve the target Q&Q auto-tuning problem (1) by randomly selecting experiment points. In each of the test problems, we let the total number of experimental points be n_0M, $n_0M + T/3$, $n_0M + 2T/3$, or $n_0M + T$. These experimental points are distributed into the M

Table 6: Total surrogae construction time (seconds) of $N = 100$ repetations

Problem	SDGP		WDQQ	
	SDEI	WDEI	SDEI	WDEI
gabor_lv3	195.49	248.38	1252.90	4039.98
amg_ani_cg	100.26	104.09	1125.21	1148.83
amg_ani_bicgstab	101.91	105.37	1142.38	1176.22
amg_iso_cg	97.79	103.57	1162.95	1162.90
amg_iso_bicgstab	97.75	103.86	1162.02	1169.26

sub-domains evenly, and the experimental points are chosen as the LHD points in each sub-domain. One hundred repetitions (i.e., $N = 100$) are performed in each numerical experiments. The mean and standard deviations of the corresponding e_R are listed in Table 7. While more experimental points yield better results, we can see that the proposed methods outperform the random results by comparing Tables 4, 5, and 7. In short, this random method is not recommended.

Table 7: Results of randomly selected points

Test Problems	Experimental Points	mean(e_R)	std(e_R)
gabor_lv3	n_0M	0.9997	0.0015
	$n_0M + T/3$	0.4860	0.2257
	$n_0M + 2T/3$	0.3642	0.1377
	$n_0M + T$	0.2778	0.1470
amg_ani_cg	n_0M	1.3126	1.0689
	$n_0M + T/3$	0.1850	0.1060
	$n_0M + 2T/3$	0.1481	0.0898
	$n_0M + T$	0.1481	0.0898
amg_ani_bicgstab	n_0M	0.8426	1.1412
	$n_0M + T/3$	0.1603	0.0984
	$n_0M + 2T/3$	0.1135	0.1048
	$n_0M + T$	0.1135	0.1048
amg_iso_cg	n_0M	1.4279	0.7150
	$n_0M + T/3$	0.3155	0.1880
	$n_0M + 2T/3$	0.1833	0.1297
	$n_0M + T$	0.0794	0.0869
amg_iso_bicgstab	n_0M	0.9754	0.7068
	$n_0M + T/3$	0.1166	0.1179
	$n_0M + 2T/3$	0.0687	0.0293
	$n_0M + T$	0.0559	0.0238

- **Results of a straightforward combination of GP and EI.** To solve the target Q&Q auto-tuning problem, we can simply apply the GP and EI on each

subdomain and then select the next exploring point as the point with the maximal EI value among all sub-domains. Note that (i) this approach is different from SDEI introduced in Section 4.2 and (ii) this infill approach can be treated as a straightforward generalization of EI over multiple sub-domains. We use this straightforward GP-EI based method to solve the test problem TP1 and report the numerical results in Table 8. Comparing Tables 4 and 8, the performance of this GP-EI-based method is mixing. It performs reasonably well in term of f_{hit} for TP1-(b,c,d). However, for TP1-(a), it performs much worse than the four variants proposed in the two previous sections. Furthermore, this straightforward method does not perform well in term of e_R for all four cases associated with TP1. The numerical results suggest that more information associated the qualitative factors can improve the computational performance. For example, the information of the qualitative factors is involved in WDQQ-SDEI and the criterion links the individual GP for each sub-domain in SDGP-WDEI. In contrast, this straightforward method considers only the information associated with the qualitative factors.

- **Results of treed type methods.** Treed type methods such as classification and regression trees (CART) [43, 44] can be used to model the qualitative variables. In such methods, the qualitative variables are represented by a set of indicator variables. For example, we can represent the V-, F-, or W-cycle of the qualitative parameter z_2 as $(0, 0)$, $(1, 0)$, and $(0, 1)$, respectively. Note that, for another qualitative parameter z_1, we need to define another indicator variable because z_1 contains two classes. After variable transformation, we use these indicator variables to build the tree and apply these quantitative variables for model fitting in the leaves of the tree [45]. For the infill criterion, the EI criterion is adopted to select the next explored point. We revisit the first test problem TP1 to illustrate the performance of the treed based methods by comparing the results shown in Table 4.

  First, a linear model is used to describe the relationship between the quantitative factors in each leaf of the tree model. Here the function `btlm` in the `tgp` R-package [46] is adopted. We repeat this treed method 30 times by randomly generating a Latin hypercube design. In each replication, we iterate this treed method 90 times. The results are summarized in Table 9. Comparing Table 9, we see the treed method performs quite well in the measurement of f_{hit} for all four cases. For the mean values of e_R, however, the proposed approaches outperform the first three cases TP1-(a), (b), (c). For TP1-(d), the treed method performs better by obtaining lower e_R mean value.

  Second, the Bayesian treed Gaussian process model is used in the leaf of the tree model. This approach is implemented by the `tgp` R-package [46]. In the latest version, the `tgp` extends the treed Gaussian process models for Bayesian nonstationary and semiparametric nonlinear regressions and designs. The tree structure can be constructed based on the qualitative factors. The stationary or Bayesian non-stationary Gaussian process models in the nodes are used to describe the relationship between the quantitative factors. That is, the latest `tgp` can be used to solve the tuning problems considered in this article. In `tgp`, the function `btgpllm` is used to implement the tree construction based on the qualitative factors and the surrogate fitting via the treed Gaussian process for the quantitative factors in each leaf. The results obtained by 30 replications are summarized in Table 10. For TP1-(a) and (b), this treed Gaussian process can

Table 8: Results of TP1 by a straightforward combination of GP and EI.

Means	(a) amg_ani_cg	(b) amg_ani_bicgstab	(c) amg_iso_cg	(d) amg_iso_bicgstab
f_{hit} (%)	55	25	81	77
mean(e_R)	1.420E-01	3.690E-02	5.900E-03	1.490E-0.2

Table 9: Results of TP1 by a treed method with EI criterion based on 30 replications.

Means	(a) amg_ani_cg	(b) amg_ani_bicgstab	(c) amg_iso_cg	(d) amg_iso_bicgstab
f_{hit} (%)	67	53	73	90
mean(e_R)	1.724E-01	7.900E-02	1.280E-02	6.730E-0.3

capture these true model quite well and achieves extremely high f_{hit} rate and lower e_R. However, the proposed SAT-QQ have better performances in TP1-(c) and (d). The table also shows the execution time taken by the `tgp` on a PC with Intel Core i7-2600 CPU (faster than the Intel Xeon X5570 CPU used in other numerical experiments) and 8GB of memory. Comparing Tables 6, we see that the treed Gaussian process approach takes longer CPU time for the same number of replications.

6 Conclusion

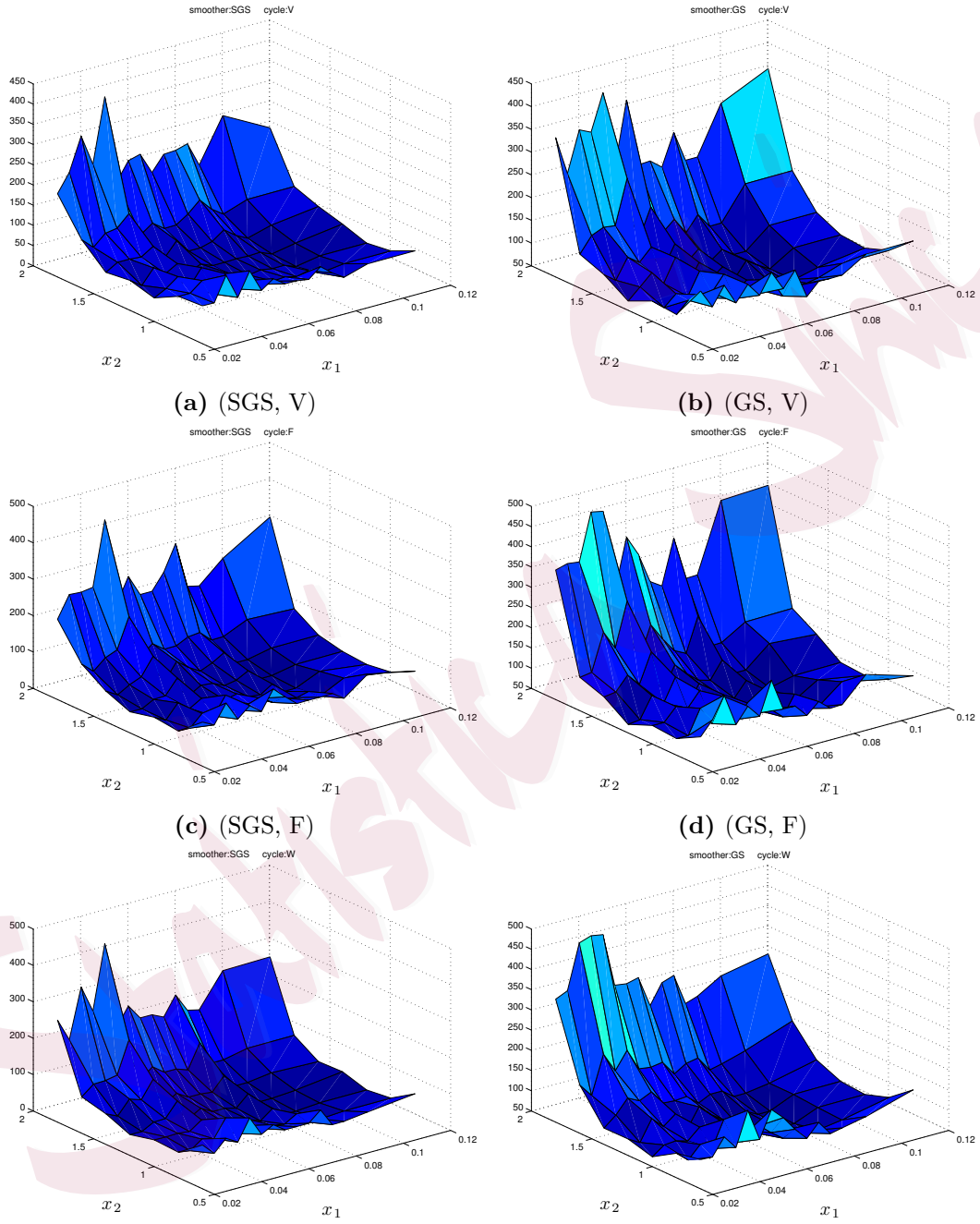
We consider computer experiment performance auto-tuning problems involving quantitative and qualitative parameters in this article. To solve the resulting optimization problems, we propose a surrogate assistant tuning algorithm called SAT-QQ. SAT-QQ is an iterative procedure. At each iteration, the main ingredients of SAT-QQ are surrogate constructions and infill criteria that take both quantitative and qualitative parameters into consideration.

Two surrogate construction approaches, SDGP and WDQQ, based on Gaussian processes are proposed to approximate the true yet unknown response surfaces. In the Sub-Domain with Gaussian Process (SDGP), we divide the experimental domain into M sub-domains along the qualitative parameters and then construct M surrogate models individually using a Gaussian process. In the Whole-Domain with Qualitative and Quantitative Gaussian Process (WDQQ), we construct the surrogates over the whole experimental domain by modifying the correlation structure for the qualitative variables suggested in [31, 32].

We select new experimental points using two proposed infill criteria to explore the experimental domain for identifying the optimal parameters. In Sub-Domain Ex-

Table 10: Results of TP1 by a treed Gaussian process method with EI criterion based on 30 replications.

Means	(a) amg_ani_cg	(b) amg_ani_bicgstab	(c) amg_iso_cg	(d) amg_iso_bicgstab
f_{hit} (%)	100	90	57	60
mean(e_R)	0.000E-00	8.116E-03	2.896E-02	2.516E-0.2
CPU time (s)	1541.08	1677.38	2042.08	2039.19

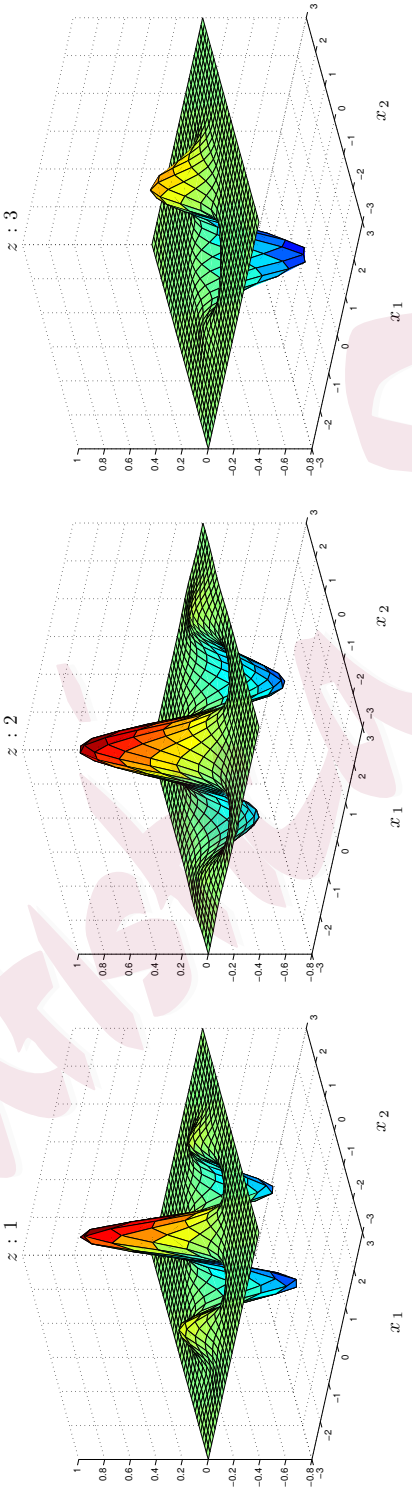


(a) (SGS, V) **(b)** (GS, V)

(c) (SGS, F) **(d)** (GS, F)

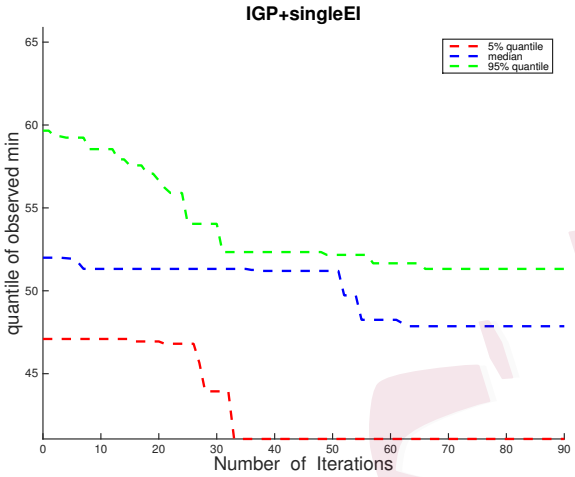
(e) (SGS, W) **(f)** (GS, W)

Figure 1: True surface of amg_ani_cg of each (Smoother, Cycle)

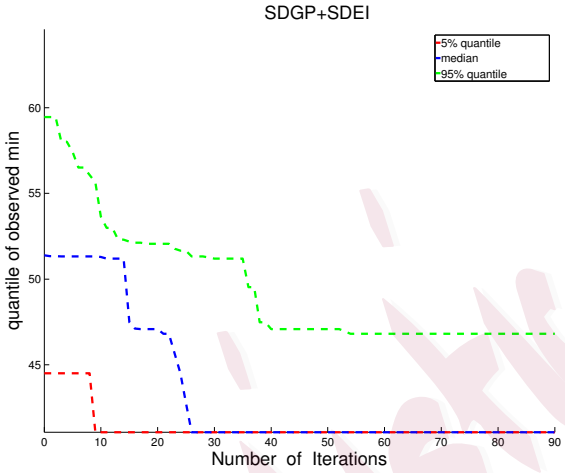


(a) $z = 1$ (b) $z = 2$ (c) $z = 3$

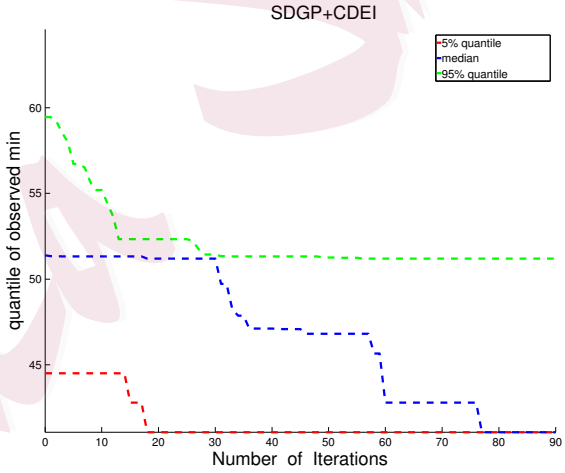
Figure 2: True surface of gabor_lv3 with each qualitative level



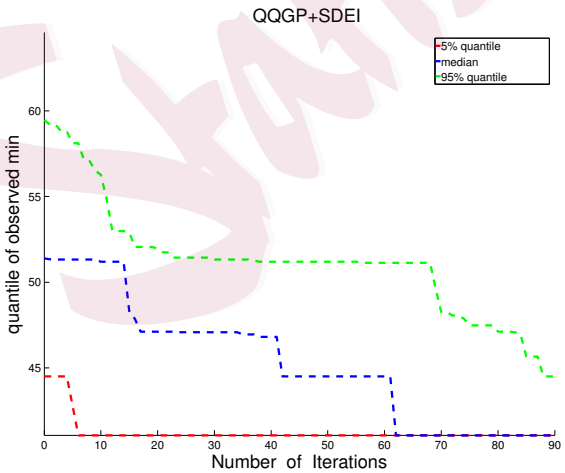
(a) SDGP+SIEI



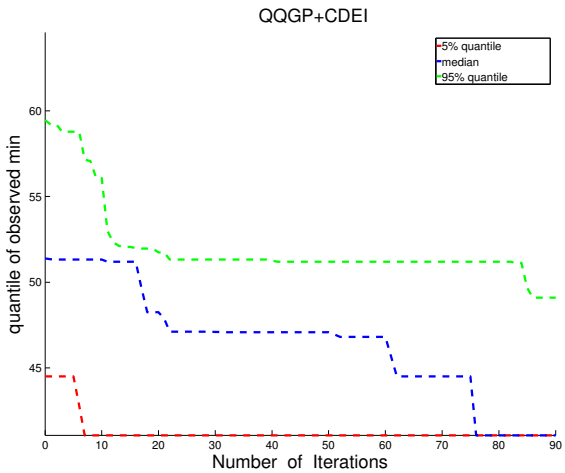
(b) SDGP+SDEI



(c) SDGP+WDEI

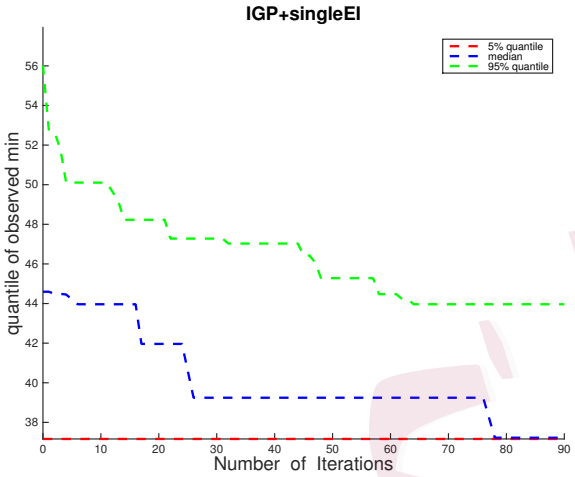


(d) WDQQ+SDEI

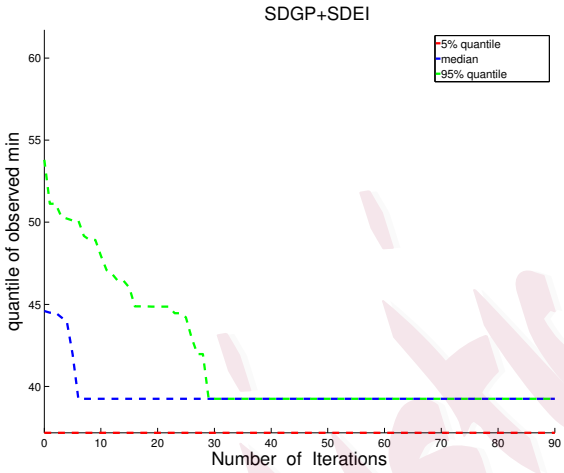


(e) WDQQ+WDEI

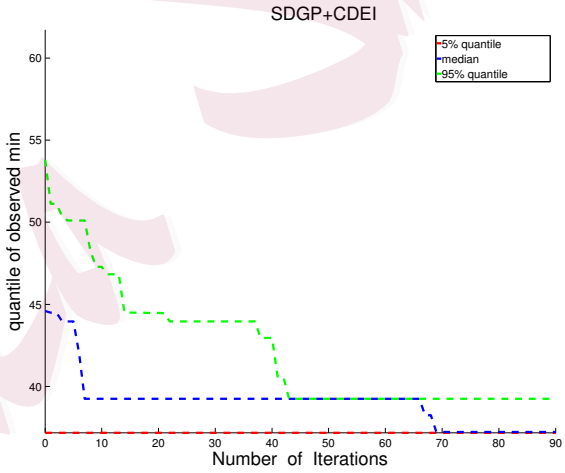
Figure 3: Quantile curves of amg_ani_cg



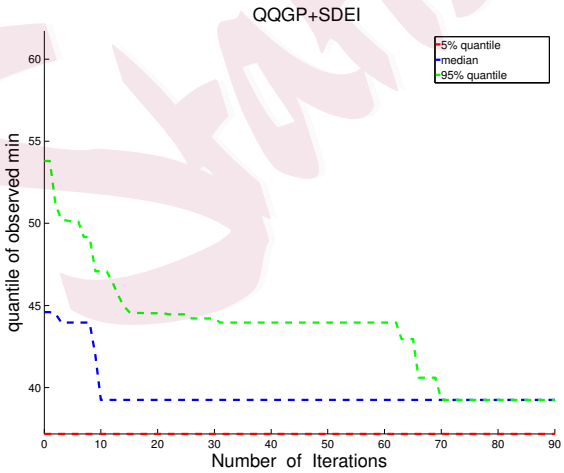
(a) SDGP+SIEI



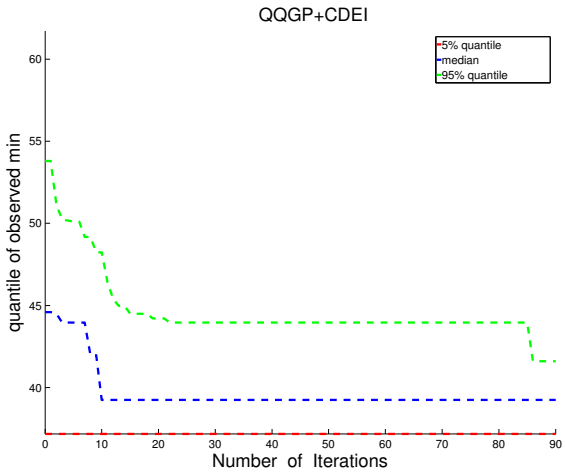
(b) SDGP+SDEI



(c) SDGP+WDEI

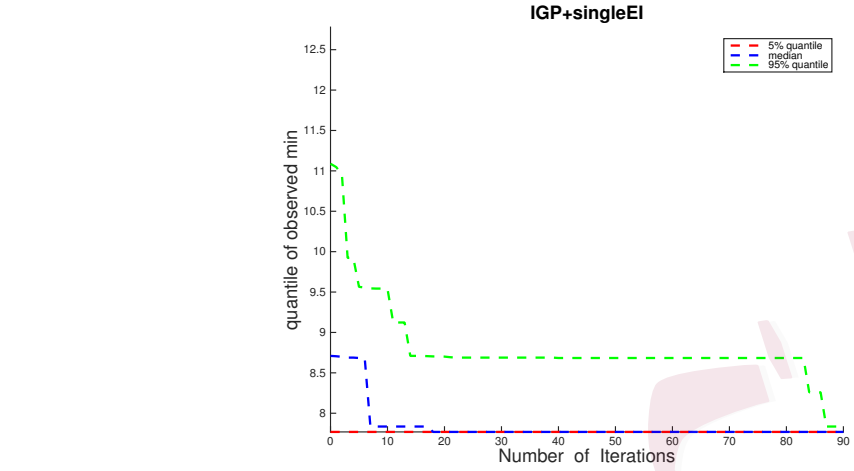


(d) WDQQ+SDEI

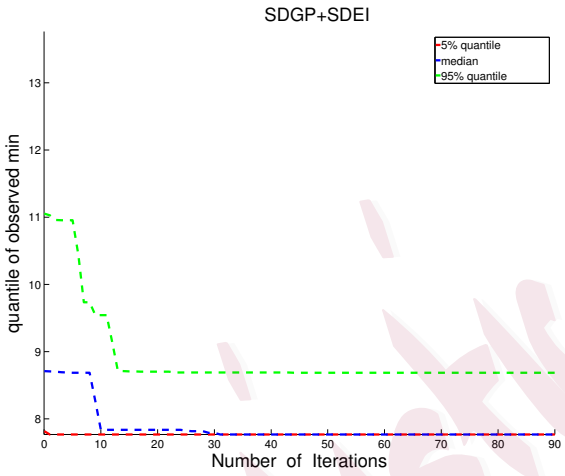


(e) WDQQ+WDEI

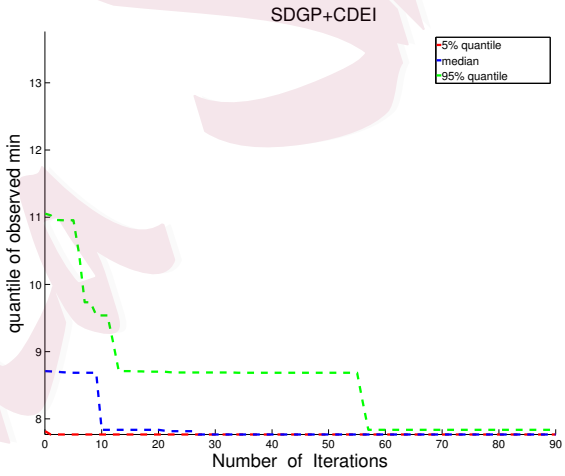
Figure 4: Quantile curves of amg_ani_bicgstab



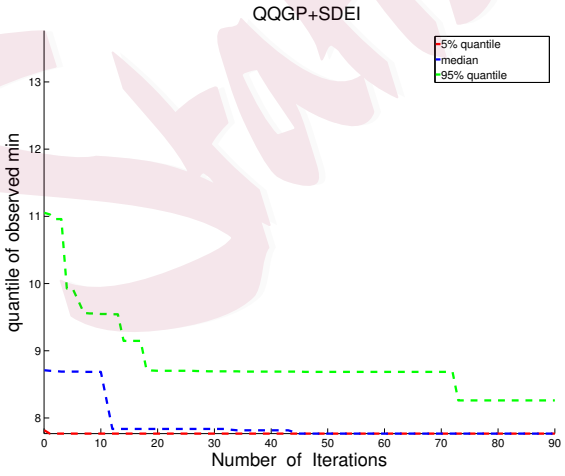
(a) SDGP+SIEI



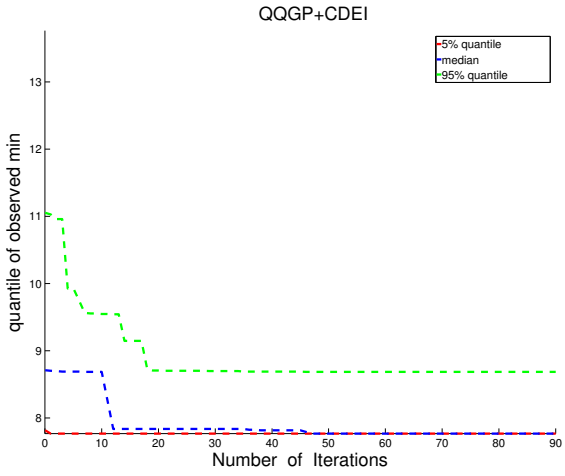
(b) SDGP+SDEI



(c) SDGP+WDEI

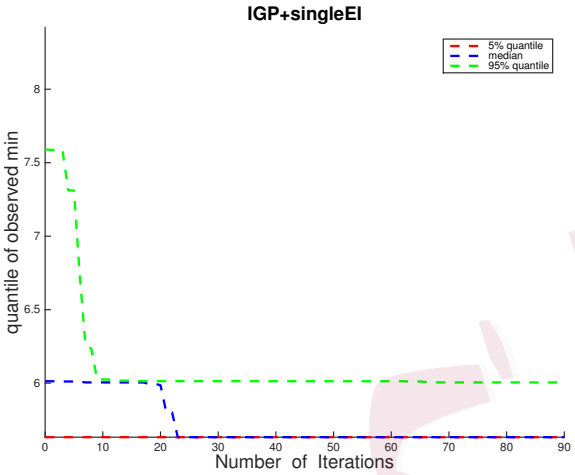


(d) WDQQ+SDEI

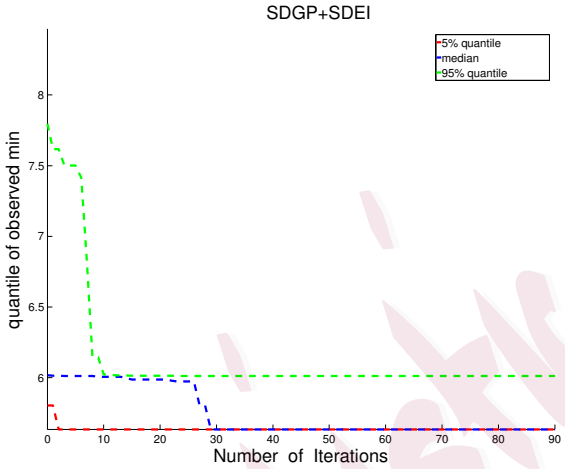


(e) WDQQ+WDEI

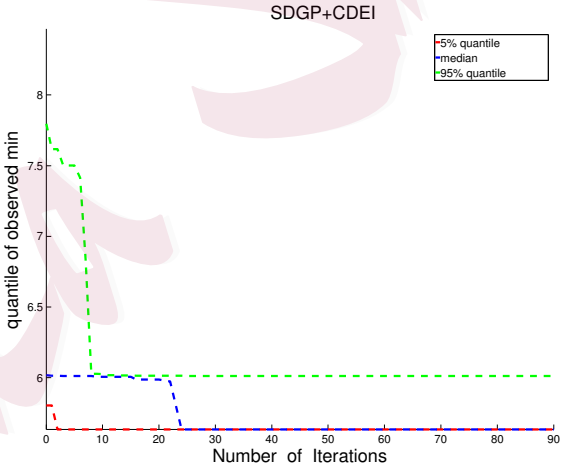
Figure 5: Quantile curves of amg_iso_cg



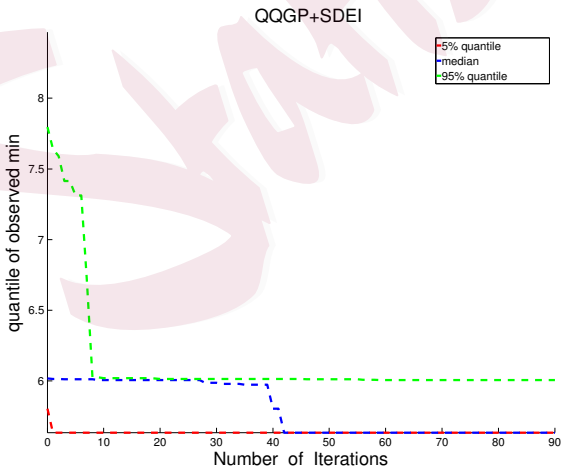
(a) SDGP+SIEI



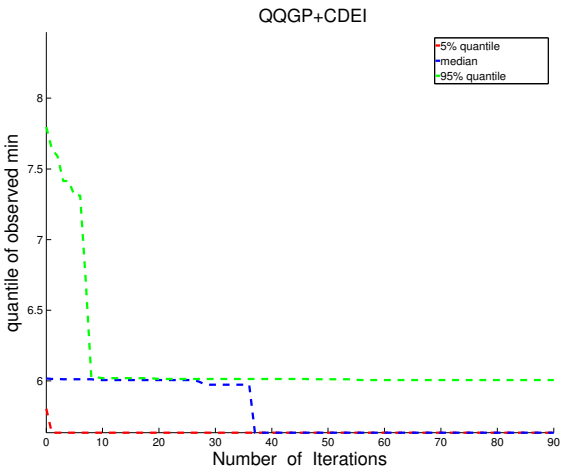
(b) SDGP+SDEI



(c) SDGP+WDEI

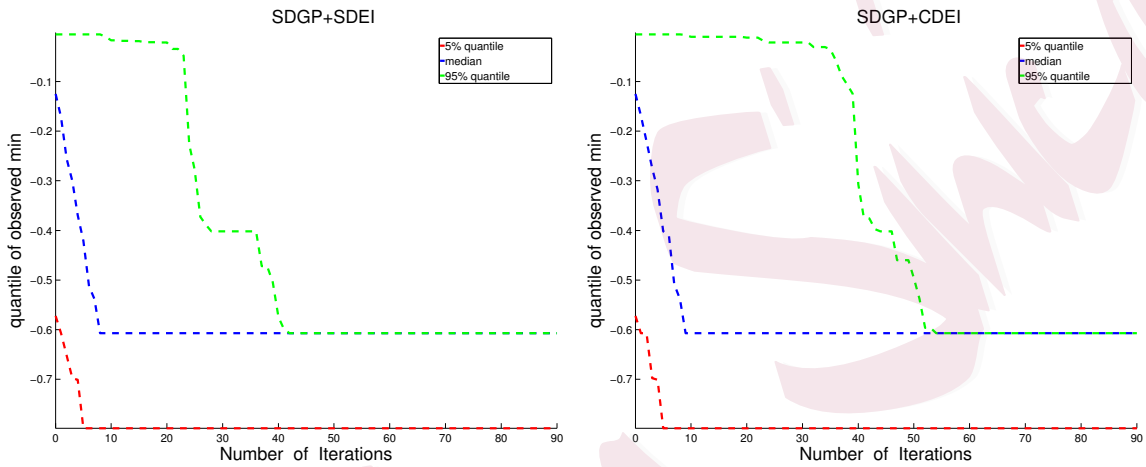


(d) WDQQ+SDEI



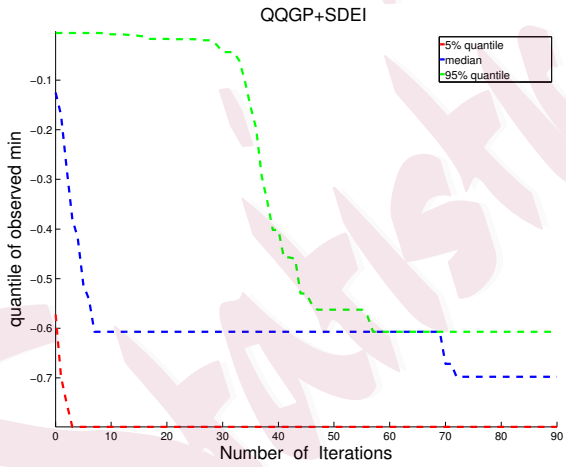
(e) WDQQ+WDEI

Figure 6: Quantile curves of amg_iso_bicgstab

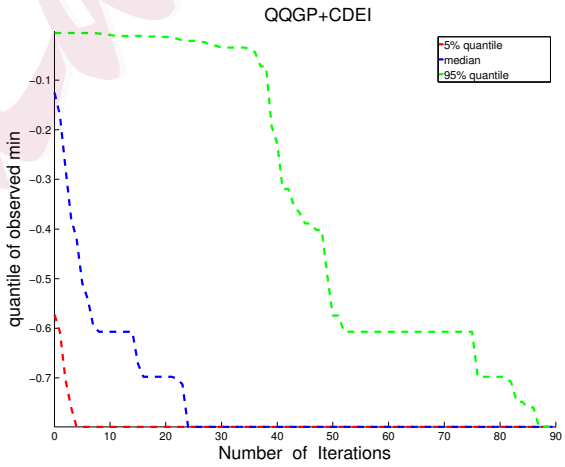


(a) SDGP+SDEI

(b) SDGP+WDEI



(c) WDQQ+SDEI



(d) WDQQ+WDEI

Figure 7: Quantile curves of gabor_lv3

pected Improvement (SDEI), we find the sub-domain that has the minimal prediction value first and then use expected improvement in this sub-domain to select the next experimental point. In the second infill criterion, Whole Domain Expected Improvement (WDEI), we identify the best point within the whole domain that is associated with the maximum expected improvement value.

Combining the two surrogate construction methods and two infill criteria, we have four variants of SAT-QQ. The performances of these four methods have been studied by solving problems arising in real simulation codes and artificial functions. The numerical results suggest the following findings. WDQQ-WDEI leads to good solutions in general. However, if computational resources are limited, we suggest using SDGP-WDEI. If we have prior knowledge that the true response surfaces are "simple and smooth" (rather than oscillatory), SDGP type methods are recommended. Otherwise, WDQQ would be a more proper choice because it considers the correlations between sub-domains.

We have proposed four variants of SAT-QQ based on the stationary assumption. Such stationary information of a data set is usually unknown a priori and may be costly to verify. For example, in [17], the non-stationary of the responses was confirmed based on large training point set. However, the proposed methods remain applicable in some cases. One of such examples is the test problem TP1. While it is arguable whether the testing data satisfies the stationary assumption as shown in Figure 1, the response surfaces clearly demonstrate main trends that are likely stationary. Consequently, the proposed methods can identify the region of interests that are associated with the lower timing results.

There are several further research directions. In our study, the number of all possible cases (M) is small. However, for large M, WDQQ might be problematic in terms of the parameter estimation in its correlation matrix. We need to simplify the correlation structure based on some prior knowledge. SDGP can be used to fit the surrogates for all sub-domains, but it ignores the relations between sub-domains. Numerical results show that the performance of the treed based methods and the proposed methods is mixing. Further studies will deepen the understanding of these methods and provide more choices for the auto-tuning problems involving qualitative and quantitative parameters.

Furthermore, the SAT-QQ can be viewed as a sequential design procedure, because the next point is selected based on an EI criterion from unexplored point set. The proposed methods share a similarity with the active learning. Active learning is a semi-supervised learning approach and is also a sequential procedure that iteratively adds un-label points to update the learning model The point selection criterion is a key in active learning. For example, in [47], the selection approach is a combination of stochastic approximation and D-optimal criterion. In particular, this selection method identifies one D-optimal design point from a small candidate based on the learning model fitting. Consequently, it is worth investigating whether we can integrate some selection approaches in active learning with the proposed EI-type criterion to improve the computational efficiency, especially for the case with many candidate experimental points.

Last, but not the least, a possible further study could explore modifying SAT-QQ for on-line tuning problems. Due to the lack of the training set, the key component is still on the surrogate construction. Since the stationary assumption might not be held, what type of the surrogate model should be used is a challenge problem. In addition, How to efficiently update the surrogate in each SAT-QQ iteration will be an

issue, especially for large experimental domains.

Acknowledgments

The authors are grateful to the Associate Editor and the referees for the helpful comments and suggestions. This specific work was supported in part by the Ministry of Science and Technology, the Taida Institute of Mathematical Sciences, the Center for Advanced Study in Theoretical Sciences, and the Mathematics Division of the National Center for Theoretical Sciences in Taiwan.

References

[1] R C. Whaley, A. Petitet, and J. J. Dongarra. Automated empirical optimizations of software and the atlas project. *Parallel Computing*, 27(1):3–35, 2001.

[2] M. Frigo and S. G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005.

[3] F. Franchetti, M. Püschel, Y. Voronenko, S. Chellappa, and J. M. F. Moura. Discrete Fourier transform on multicore. *Signal Processing Magazine, IEEE*, 26(6):90–102, 2009.

[4] T. Katagiri, K. Kise, H. Honda, and T. Yuba. Abclibscript: A directive to support specification of an auto-tuning facility for numerical software. *Parallel Computing*, 32(1):92–112, 2006.

[5] C. Ţăpuş, I-H. Chung, J. K Hollingsworth, et al. Active harmony: Towards automated performance tuning. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–11. IEEE Computer Society Press, 2002.

[6] J. Ansel, S. Kamil, K. Veeramachaneni, J. Ragan-Kelley, J. Bosboom, U.-M. O'Reilly, and S. Amarasinghe. Opentuner: An extensible framework for program autotuning. In *Proceedings of the 23rd International Conference on Parallel Architectures and Compilation*, pages 303–316. ACM, 2014.

[7] J. Bergstra, N. Pinto, and D. Cox. Machine learning for predictive auto-tuning with boosted regression trees. In *Innovative Parallel Computing (InPar), 2012*, pages 1–9. IEEE, 2012.

[8] T. L. Falch and A. C. Elster. Machine learning based auto-tuning for enhanced OpenCL performance portability. In *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*, pages 1231–1240. IEEE, 2015.

[9] S. Pellegrini, T. Fahringer, H. Jordan, and H. Moritsch. Automatic tuning of MPI runtime parameter settings by using machine learning. In *Proceedings of the 7th ACM International Conference on Computing Frontiers*, pages 115–116. ACM, 2010.

[10] A. Monsifrot, F. Bodin, and R. Quiniou. A machine learning approach to automatic production of compiler heuristics. In *Artificial Intelligence: Methodology, Systems, and Applications*, pages 41–50. Springer, 2002.

[11] A. Collins, C. Fensch, H. Leather, and M. Cole. Masif: Machine learning guided auto-tuning of parallel skeletons. In *High Performance Computing (HiPC), 2013 20th International Conference on*, pages 186–195. IEEE, 2013.

[12] A. Ganapathi, K. Datta, A. Fox, and D. Patterson. A case for machine learning to optimize multicore performance. In *First USENIX Workshop on Hot Topics in Parallelism*, 2009.

[13] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.

[14] Frank Hutter, Holger H Hoos, Kevin Leyton-Brown, and Kevin P Murphy. An experimental investigation of model-based parameter optimisation: Spo and beyond. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 271–278. ACM, 2009.

[15] B. Ylvisaker and S. Hauck. Probabilistic auto-tuning for architectures with complex constraints. In *Proceedings of the 1st International Workshop on Adaptive Self-Tuning Computing Systems for the Exaflop Era*, pages 22–33. ACM, 2011.

[16] R. Suda. A Bayesian method of online automatic tuning. In *Software Automatic Tuning*, pages 275–293. Springer, 2011.

[17] R. B. Gramacy, M. Taddy, and Wild S. M. Variable selection and sesitivity analysis using dynamic trees, with an application to computer code performance tuning. *The Annals of Applied Statistics*, 7(1):51–80, 2013.

[18] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[19] Riku Murata, Jun Irie, Akihiro Fujii, Teruo Tanaka, and Takahiro Katagiri. Enhancement of incremental performance parameter estimation on ppOpen-AT. In *Embedded Multicore/Many-core Systems-on-Chip (MCSoC), 2015 IEEE 9th International Symposium on*, pages 203–210. IEEE, 2015.

[20] A. Fujii, A. Nishida, and Y. Oyanagi. Evaluation of parallel aggregate creation orders: Smoothed aggregation algebraic multigrid method. In *High Performance Computational Science and Engineering*, pages 99–122. Springer, 2005.

[21] A. Fujii and O. Marques. Axis communication method for algebraic multigrid solver. *IEICE Transactions on Information and Systems*, 97(11):2955–2958, 2014.

[22] A. Fujii. AMGS: Algebraic multigrid solvers. http://hpcl.info.kogakuin.ac.jp/lab/software/amgs, 2014.

[23] M. Darwish, I. Sraj, and F. Moukalled. A coupled finite volume solver for the solution of incompressible flows on unstructured grids. *Journal of Computational Physics*, 228(1):180–201, 2009.

[24] M. F. Adams, H. H. Bayraktar, T. M. Keaveny, and P. Papadopoulos. Ultrascalable implicit finite element analyses in solid mechanics with over a half a billion degrees of freedom. In *Proceedings of the 2004 ACM/IEEE Conference on Supercomputing*, page 34. IEEE Computer Society, 2004.

[25] A. Fujii, A. Nishida, and Y. Oyanagi. A parallel AMG algorithm based on domain decomposition. *IPSJ Transactions on Advanced Computing System*, 44:9–17, 2003.

[26] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56(3):179–196, 1996.

[27] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial: Second Edition*. SIAM, Philadelphia, PA, USA, 2000.

[28] Takahiro Katagiri, Satoshi Ohshima, and Masaharu Matsumoto. Auto-tuning of computation kernels from an fdm code with ppOpen-AT. In *Embedded Multicore/Manycore SoCs (MCSoc), 2014 IEEE 8th International Symposium on*, pages 91–98. IEEE, 2014.

[29] Cy Chan, Jason Ansel, Yee Lok Wong, Saman Amarasinghe, and Alan Edelman. Autotuning multigrid with petabricks. In *High Performance Computing Networking, Storage and Analysis, Proceedings of the Conference on*, pages 1–12. IEEE, 2009.

[30] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and Analysis of Computer Experiments. *Statistical Science*, 4(4):409–423, 1989.

[31] P. Z. G Qian, H. Wu, and C. F. J. Wu. Gaussian process models for computer experiments with qualitative and quantitative factors. *Technometrics*, 50(3):383–396, 2008.

[32] Q. Zhou, P. Z. G. Qian, and S. Zhou. A simple approach to emulation for computer models with qualitative and quantitative factors. *Technometrics*, 53(3):266–273, 2011.

[33] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4):455–492, December 1998.

[34] T. J. Santner, Williams B., and Notz W. *The Design and Analysis of Computer Experiments*. Springer-Verlag, 2003.

[35] R.-B. Chen, D.-N. Hsieh, Y. Hung, and W. Wang. Optimizing latin hypercube designs by particle swarm. *Statistics and Computing*, 23(5):663–676, 2013.

[36] K.-T. Fang, D. K. J. Lin, P. Winker, and Y. Zhang. Uniform design: Theory and application. *Technometrics*, 42:237–248, 2000.

[37] P. Z. G. Qian. Sliced latin hypercube designs. *Journal of the American Statistical Association*, 107(479):393–399, 2012.

[38] G. Mariani, G. Palermo, V. Zaccaria, and C. Silvano. Oscar: An optimization methodology exploiting spatial correlation in multicore design spaces. *IEEE Trans. Comput.-Aided Des. Intergr. Circuits*, 31(5):740–753, 2012.

[39] S. N. Lophaven, H. B. Nielsen, and J. Sondergaard. Aspects of the Matlab toolbox DACE. Technical Report IMM-REP-2002-13, Technical University of Denmark, 2002.

[40] L. N. Trefethen and D. Bau III. *Numerical Linear Algebra*. SIAM, 1997.

[41] A. Forrester, A. Sobester, and A. Keane. *Engineering Design via Surrogate Modelling: a Practical Guide*. John Wiley & Sons, 2008.

[42] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239 – 245, 1979.

[43] L. Breiman, J. H. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[44] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian cart model search (with discussion and a rejoinder by the authors). *Journal of the American Statistical Association*, page 935V960.

[45] R. B. Gramacy and M. A. Taddy. Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an r package for treed gaussian process models. *Journal of Statistical Software*, page 1V48.

[46] R. B. Gramacy and M. A. Taddy. *tgp: Bayesian Treed Gaussian Process Models*. R Foundation for Statistical Computing, Vienna, Austria, 2016. https://cran.r-project.org/web/packages/tgp/index.html.

[47] V. R. Deng, X. W. andJoseph, A. Sudjianto, and C. F. J. Wu. Active learning through sequential design, with applications to detection of money laundering. *Journal of the American Statistical Association*, 104:969–981, 2009.

Statistica Sinica