Surrogate Models for Mixed Discrete-Continuous Variables

Laura P. Swiler[1], Patricia D. Hough[1], Peter Qian[2], Xu Xu[2], Curtis Storlie[3], and Herbert Lee[4]

[1] Sandia National Laboratories, Albuequerque, New Mexico, and Livermore, California
{lpswile,pdhough}@sandia.gov
[2] University of Wisconsin-Madison, Madison, Wisconcin
{peterq,xuxu}@stat.wisc.edu
[3] Los Alamos National Laboratory, Los Alamos, New Mexico
storlie@lanl.gov
[4] Univerity of California, Santa Cruz, Santa Cruz, California
herbie@soe.ucsc.edu

Abstract. Large-scale computational models have become common tools for analyzing complex man-made systems. However, when coupled with optimization or uncertainty quantification methods in order to conduct extensive model exploration and analysis, the computational expense quickly becomes intractable. Furthermore, these models may have both continuous and discrete parameters. One common approach to mitigating the computational expense is the use of response surface approximations. While well developed for models with continuous parameters, they are still new and largely untested for models with both continuous and discrete parameters. In this work, we describe and investigate the performance of three types of response surfaces developed for mixed-variable models: Adaptive Component Selection and Shrinkage Operator, Treed Gaussian Process, and Gaussian Process with Special Correlation Functions. We focus our efforts on test problems with a small number of parameters of interest, a characteristic of many physics-based engineering models. We present the results of our studies and offer some insights regarding the performance of each response surface approximation method.

1 Introduction

Large-scale computational models have become common tools for analyzing complex man-made systems. We are particularly interested in engineering models that have a small number of variables of interest but are characterized by computationally expensive equation solvers such as partial differential equation solvers. Many methods for exploring such models with data and scenario uncertainties are computationally expensive. One key to successfully mitigating the computational expense involves the construction of surrogates for the large-scale models. A surrogate can take many forms, but in this context we mean a meta-model or

M. Ceberio and V. Kreinovich (eds.), *Constraint Programming and Decision Making*, Studies in Computational Intelligence 539,
DOI: 10.1007/978-3-319-04280-0_21,

response surface approximation built from a limited amount of data generated by the computationally expensive model. The purpose of the surrogate model is to increase the efficiency of analyses that require frequent model interrogations such as optimization and uncertainty quantification.

When the models are computationally demanding, meta-model approaches to their analysis have been shown to be very useful. For example, one standard approach in the literature is to develop an emulator that is a stationary smooth Gaussian process [13, 23, 25]. There are many good overview articles which compare various metamodel strategies. For example, Storlie et al. compare various smoothing predictors and nonparametric regression approaches in [30, 31]. Simpson et al. provide an excellent overview not just of various statistical metamodel methods but also approaches which use low-fidelity models as surrogates for high fidelity models [27]. This paper also addresses the use of surrogates in design optimization, which is a popular research area for computationally expensive disciplines such as computational fluid dynamics in aeronautical engineering design. Haftka and his students developed an approach which uses "ensembles" of emulators or hybrid emulators [35, 36]. The advantage of these types of hybrid or ensembles of emulators is that better performance can be obtained. For example, one can select the best surrogate for various features or responses, or one can use weighted model averaging of surrogates.

The particular challenge we address in our work is that of assessing the accuracy relative to computational cost of surrogates for models that have both continuous and discrete variables. While historically the variables of interest in engineering models have been continuous, there is a growing use of discrete variables that represent modeling choices (alternative plausible models) and design choices (e.g. discrete choices of materials, components, or operational settings). The major challenge in using surrogates for mixed variable problems is in handling the discrete variables. Typically, in surrogate models constructed over continuous variables (e.g. polynomial regression, splines, Gaussian process models), there is the assumption of continuity: as a continuous variable varies by a small amount, the response is assumed to vary smoothly. With discrete variables, we do not necessarily have continuous behavior. For example, if a discrete variable representing a design choice varies from choice A to choice B, the system may respond in a fundamentally different manner. Thus, surrogate modeling approaches generally do not explicitly allow for categorical input variables. One option is to order these categorical inputs in some way and treat them as continuous variables when creating a metamodel. In some cases, this can lead to undesirable and misleading results. The other option is categorical regression. In this approach, a separate surrogate model is constructed over the continuous variables for each possible combination of the discrete variable values. This approach has the advantage that the surrogate is only constructed on the continuous variables, conditional on a particular combination of discrete values. However, it quickly becomes infeasible as one increases the number of discrete variables and/or the number of "levels" per variable [18]. It is clear that a more appropriate and efficient treatment of categorical inputs is needed.

In this work, we consider three approaches in the literature for constructing mixed variable surrogates. They have their roots in response surface modeling for continuous problems and tractably incorporate discrete variables in a manner that relies on some simplifications and additional assumptions. Our goal is to empirically evaluate and compare the three methods in order to gain insight into how problem characteristics influence the efficacy of each surrogate approach.

The remaining sections of this paper discuss the three approaches we evaluated and the outcome of our computational experiments. Section 2 outlines three approaches for constructing mixed variable surrogates. Section 3 describes our approach for testing and evaluating the three surrogate methods. Section 4 provides results of the surrogates on several test problems, and Section 5 summarizes the outcome.

2 Mixed Surrogate Approaches

This section describes three classes of methods that we investigated to generate surrogate models for mixed discrete-continuous variable problems. These three classes of methods are:

ACOSSO: ACOSSO, the Adaptive COmponent Selection and Smoothing Operator, is a specialized smoothing spline model [29]. It uses the smoothing spline ANOVA decomposition to separate the underlying function into simpler functional components (i.e., main effects, two-way interactions, etc.) then explicitly estimates these functional components in one optimization. The estimation proceeds by optimizing the likelihood subject to a penalty on each of the functional components. Each component involving continuous predictors has a penalty on its roughness and overall trend, each component involving discrete predictors has a penalty on its magnitude (L^2-norm), while interaction components involving both discrete and continuous predictors receive a combination of these penalties.

Gaussian Processes with Special Correlation Functions: Gaussian process models are powerful emulators for computer models. A Gaussian process model is defined by its mean and covariance function. The covariance function specifies how the response between two points is related: the idea is that points close together in input space will tend to have responses that are similar. Typically, the covariance function is a function of the distance between the points. We investigated a variety of covariance functions that represent the covariance between discrete points and that appropriate for mixed variable problems, all developed in Qian et al. [20, 39]: the exchangeable correlation (EC), the multiplicative correlation (MC), and the unrestricted correlation function (UC). For comparison, we also looked at the Individual Kriging (IK) model which involves constructing a separate Gaussian process surrogate over the continuous variables for each combination of discrete variables. This is similar to categorical regression.

TGP: TGP, the treed Gaussian Process model, is an approach which allows different Gaussian process models (GPs) to be constructed on different partitions of the space [8, 9]. This approach naturally lends itself to discrete

variables, where the partitioning can be done between different values or sets of discrete variables. In TGP, the discrete or categorical variables are converted to a series of binary variables. The binary variables are then what are partitioned upon: they become the "nodes" of the tree [10].

2.1 Adaptive COmponent Selection and Shrinkage Operator (ACOSSO)

The Adaptive COmponent Selection and Shrinkage Operator (ACOSSO) estimate [29] was developed under the smoothing spline ANOVA (SS-ANOVA) modeling framework. As it is a smoothing type method, ACOSSO works best when the underlying function is somewhat smooth. The type of splines we are using involve the minimization of an objective function involving a sum-of-squares error term, similar to regression modeling. However, in the objective function for the splines, there are additional terms which can be viewed as regularization terms: these penalty terms help smooth the function and they also help perform variable selection. In the ACOSSO implementation, there is a penalty on functions of the categorical predictors. This penalty formulation provides a variable selection and automatic model reduction: it encourages some of the terms in the objective function to be zero, removing certain discrete variables or levels of discrete variables from the formulation. To facilitate the description of ACOSSO, we first describe the multiple smoothing spline, then introduce the treatment of categorical predictors and the ACOSSO estimator.

Multivariate Smoothing Splines. Consider a vector of predictors $\mathbf{x} = [x_1, \ldots, x_I]'$. Assume that the unknown function f to be estimated belongs to 2^{nd} order Sobolev space $\mathcal{S}^2 = \{f : f, f' \text{ are absolutely continuous and } f'' \in \mathcal{L}^2[0,1]\}$. The simplest extension of smoothing splines to multiple inputs is the additive model [12]. For instance, assume that

$$f \in \mathcal{F}_{add} = \{f : f(\mathbf{x}) = \sum_{i=1}^{I} g_i(x_i), g_i \in \mathcal{S}^2\}, \tag{1}$$

i.e., $f(\mathbf{x}) = \sum_{i=1}^{I} g_i(x_i)$ is a sum of univariate functions. Let $\mathbf{x}_n = [x_{n,1}, \ldots, x_{n,I}]'$ be the n^{th} observation of a multivariate predictor $\mathbf{x}$, $n = 1, \ldots, N$, and $y_n = f(\mathbf{x}_n) + \varepsilon_n$. The additive smoothing spline estimate of f is the minimizer of

$$\frac{1}{n} \sum_{n=1}^{N} [y_n - f(\mathbf{x}_n)]^2 + \sum_{i=1}^{I} \lambda_i \int_0^1 [g_i''(x_i)]^2 \, dx_i \tag{2}$$

over $f \in \mathcal{F}_{add}$. The minimizer of the expression in Eq. (2), $\hat{f}(\mathbf{x}) = \sum_{i=1}^{I} \hat{g}_i(x_i)$, takes the form of a natural cubic spline for each of the functional components $\hat{g}_i$. Notice that there are I tuning parameters (λ_i) for the additive smoothing spline. These tuning parameters λ_i control the trade-off in the resulting estimate between smoothness and fidelity to the data; large values of λ will result

in smoother functions while smaller values of λ result in rougher functions that more closely match the data. Generally, λ_i is chosen by generalized cross validation (GCV) [6] or m-fold CV [14]. A generalization to two-way and higher order interaction functions can also be achieved in a similar manner; see [29] for the full details of including interactions in the SS-ANOVA framework. The minimizer of the expression in Eq. (2) can be obtained in an efficient manner via matrix algebra using results from reproducing kernel Hilbert space (RKHS) theory; for details see [37] or [11].

Discrete Predictors. A large advantage to the SS-ANOVA framework is the ability to handle categorical predictors with relative ease. To facilitate the discussion, we generalize our notation to the following. Assume that $\mathbf{x} = [x_1, \ldots, x_I]'$ are continuous on $[0, 1]$ as previously in this section, while $\mathbf{z} = [z_1, \ldots, z_J]'$ are unordered discrete variables, and let the collection of the two types of predictors be denoted $\mathbf{w} = [\mathbf{x}', \mathbf{z}']'$. For simplicity, assume $z_j \in \{1, 2, \ldots, b_j\}$ for $j = 1, \ldots, J$ where the ordering of the integers representing the groups for z_j is completely arbitrary. For notational convenience, let $\mathcal{G}_i = \mathcal{S}^2$ for $i = 1, \ldots, I$. Also let the class of $\mathcal{L}^2$ functions on the domain of z_j (i.e., $\{1, 2, \ldots, b_j\}$) be denoted as $\mathcal{H}_j$ for $j = 1, \ldots, J$.

For simplicity, we can once again consider the class of additive functions,

$$\mathcal{F}_{add} = \{f : f(\mathbf{w}) = \sum_{i=1}^{I} g_i(x_i) + \sum_{j=1}^{J} h_j(z_j),\ g_i \in \mathcal{G}_i,\ h_j \in \mathcal{H}_j\}. \tag{3}$$

Let $\mathbf{w}_n = [x_{n,1}, \ldots, x_{n,I}, z_{n,1}, \ldots, z_{n,J}]'$ be the n^{th} observation of a multivariate predictor $\mathbf{w}$. The traditional additive smoothing spline is then the minimizer of

$$\frac{1}{N} \sum_{n=1}^{N} [y_n - f(\mathbf{w}_n)]^2 + \sum_{i=1}^{I} \lambda_i \int_0^1 [g_i''(x_i)]^2 \, dx_i \tag{4}$$

over $f \in \mathcal{F}_{add}$. Notice that in the traditional smoothing spline in (4) there is no penalty term on the functions of the categorical predictors (h_j).

Generalizing to the ACOSSO Estimate. The COmponent Selection and Shrinkage Operator (COSSO) [15] penalizes on the sum of the semi-norms instead of the squared semi-norms as in Eq. (4). A semi-norm is a norm which can assign zero to some nonzero elements of the space. In this case, all functions with zero second derivative (i.e., linear functions) will have zero penalty (i.e., semi-norm equal to zero). For ease of presentation, we will continue to restrict attention to the additive model. However, all of the following discussion applies directly to the two-way (or higher) interaction model as well.

The additive COSSO estimate, $\hat{f}(\mathbf{w}) = \sum \hat{g}_i(x_i) + \sum \hat{h}_j(z_j)$, is given by the function $f \in \mathcal{F}_{add}$ that minimizes

$$\frac{1}{N}\sum_{n=1}^{N}\left[y_n - f(\mathbf{w}_n)\right]^2 + \lambda_1 \sum_{i=1}^{I}\left\{\left[\int_0^1 g_i'(x_i)dx_i\right]^2 + \int_0^1 \left[g_i''(x_i)\right]^2 dx_i\right\}^{1/2} + \lambda_2 \sum_{j=1}^{J}\left\{\sum_{z_j=1}^{b_j} h_j^2(z_j)\right\}^{1/2}. \quad (5)$$

There are four key differences in the penalty term in Eq. (5) relative to the additive smoothing spline of Eq. (4). First, there is an additional term $\left[\int_0^1 g_i'(x_i)dx_i\right]^2$ in the penalty for continuous predictor functional components, which can also be written $[g_i(1) - g_i(0)]^2$, that penalizes the magnitude of the overall trend of the functional components g_i that correspond to continuous predictors. Second, there is now a penalty on the $\mathcal{L}^2$ norm of the h_j that correspond to the categorical predictors. Third, in contrast to the squared semi-norm in the additive smoothing spline, each term in the sum in the penalty in Eq. (5) can be thought of as a semi-norm over functions $g_i \in \mathcal{G}_i$ or $h_j \in \mathcal{H}_j$, respectively, (only constant functions have zero penalty). This encourages some of the terms in the sum to be exactly zero. Fourth, the COSSO penalty only has two tuning parameters (three if two-way interactions are included), which can be chosen via GCV or similar means.

Finally, ACOSSO is a weighted version of COSSO, where a rescaled semi-norm is used as the penalty for each of the functional components. Specifically, we select as our estimate the function $f \in \mathcal{F}_{add}$ that minimizes

$$\frac{1}{N}\sum_{n=1}^{N}\left[y_n - f(\mathbf{w}_n)\right]^2 + \lambda_1 \sum_{i=1}^{I} v_i \left\{\left[\int_0^1 g_i'(x_i)dx_i\right]^2 + \int_0^1 \left[g_i''(x_i)\right]^2 dx_i\right\}^{1/2} + \lambda_2 \sum_{j=1}^{J} w_j \left\{\sum_{z_j=1}^{b_j} h_j^2(z_j)\right\}^{1/2}, \quad (6)$$

where the v_i, w_j, $0 < v_i, w_j \leq \infty$, are weights that can depend on an initial estimate of f which we denote $\tilde{f}$. Our implementation of ACOSSO takes $\tilde{f}$ to be the COSSO estimate of Eq. (5), in which λ_1 and λ_2 are chosen by the GCV criterion. We then use

$$v_i = \left[\int_0^1 \tilde{g}_i^2(x_i)dx_i\right]^{-1} \text{ for } i = 1, \ldots, I$$
$$w_j = \left(\frac{1}{b_j}\sum_{z_j=1}^{b_j} \tilde{h}_j^2(z_j)\right)^{-1} \text{ for } j = 1, \ldots, J. \quad (7)$$

This allows for more flexible estimation (less penalty) on the functional components that show more signal in the initial estimate. As shown in [29], this

approach results in better performance on many test cases and more favorable asymptotic properties than COSSO.

The minimizer of the expression in Eq. (6) is obtained using an iterative algorithm and a RKHS framework similar to that used to find the minimizer of Eq. (4), see [29] for more details on the computation of the solution. The two-way interaction model is used in the results of Section 4.

2.2 Gaussian Processes for Models with Quantitative and Qualitative Factors

This section describes a computationally efficient method developed in Zhou, Qian, and Zhou [39] for fitting Gaussian process models with quantitative and qualitative factors proposed in Qian, Wu, and Wu [20]. Consider a computer model with inputs $\mathbf{w} = (\mathbf{x}^t, \mathbf{z}^t)^t$, where $\mathbf{x} = (x_1, \ldots, x_I)^t$ consists of all the quantitative factors and $\mathbf{z} = (z_1, \ldots, z_J)^t$ consists of all the qualitative factors with z_j having b_j levels. The number of the qualitative levels of $\mathbf{z}$ is given by

$$m = \prod_{j=1}^{J} b_j. \tag{8}$$

Throughout, the factors in $\mathbf{z}$ are assumed to be qualitative but not ordinal. Gaussian process models with ordinal qualitative factors can be found in Section 4.4 of [20]. The response of the computer model at an input value $\mathbf{w}$ is modeled as

$$y(\mathbf{w}) = \mathbf{f}^t(\mathbf{w})\boldsymbol{\beta} + \epsilon(\mathbf{w}), \tag{9}$$

where $\mathbf{f}(\mathbf{w}) = [f_1(\mathbf{w}), \ldots, f_p(\mathbf{w})]^t$ is a set of p user-specified regression functions, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^t$ is a vector of unknown coefficients and the residual $\epsilon(\mathbf{w})$ is a stationary Gaussian process with mean 0 and variance σ^2. The model in (9) has a more general form than the standard Gaussian process model with only quantitative factors $\mathbf{x}$ given by

$$y(\mathbf{x}) = \mathbf{f}^t(\mathbf{x})\boldsymbol{\beta} + \epsilon(\mathbf{x}), \tag{10}$$

where $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_p(\mathbf{x})]^t$ is a set of p user-specified regression functions depending on $\mathbf{x}$ only, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^t$ is a vector of unknown coefficients, and the residual $\epsilon(\mathbf{x})$ is a stationary Gaussian process with mean 0, variance σ^2 and a correlation function for $\mathbf{x}$.

For m in (8), let $c_1, \ldots, c_m$ denote the m qualitative levels of $\mathbf{z}$ and let $\mathbf{w} = (\mathbf{x}^t, c_q)^t$ $(q = 1, \ldots, m)$ denote any input value. For two input values $\mathbf{w}_1 = (\mathbf{x}_1^t, c_1)^t$ and $\mathbf{w}_2 = (\mathbf{x}_2^t, c_2)^t$, the correlation between $y(\mathbf{w}_1)$ and $y(\mathbf{w}_2)$ is defined to be

$$\text{cor}\,[\epsilon(\mathbf{w}_1), \epsilon(\mathbf{w}_2)] = \tau_{c_1,c_2}\varphi(\mathbf{x}_1, \mathbf{x}_2), \tag{11}$$

where φ is the correlation function for the quantitative factors $\mathbf{x}$ in the model (9) and τ_{c_1,c_2} is the cross-correlation between the qualitative levels c_1 and c_2. The choice of φ is flexible. We use a *Gaussian correlation function* [25]

$$\varphi(\mathbf{x}_1, \mathbf{x}_2) = \exp\left\{-\sum_{i=1}^{I} \phi_i (x_{1i} - x_{2i})^2\right\} \tag{12}$$

but other correlation functions such as Wendland's *compactly supported correlation function* [38] may also be used.

The unknown roughness parameters ϕ_i in (12) will be collectively denoted as $\mathbf{\Phi} = \{\phi_i\}$. The $m \times m$ matrix $\mathbf{T} = \{\tau_{r,s}\}$, with entries being the cross-correlations among the qualitative levels, must be *positive definite with unit diagonal elements* in order for (11) to be a valid correlation function. This condition can be achieved in two ways. One way is to use the semi-definite programming techniques with positive definiteness constraints [20], which are computationally intensive. [39] provides a more efficient way for modeling $\mathbf{T}$ by using the hypersphere decomposition, originally introduced for modeling correlations in financial applications [21]. This method first applies a Cholesky-type decomposition to $\mathbf{T}$

$$\mathbf{T} = \mathbf{L}\mathbf{L}^t, \tag{13}$$

where $\mathbf{L} = \{l_{r,s}\}$ is a lower triangular matrix with strictly positive diagonal entries. Then, let $l_{1,1} = 1$ and for $r = 2, \ldots, m$, consider a *spherical coordinate system*

$$\begin{cases} l_{r,1} = \cos(\theta_{r,1}), \\ l_{r,s} = \sin(\theta_{r,1}) \cdots \sin(\theta_{r,s-1}) \cos(\theta_{r,s}), \text{ for } s = 2, \ldots, r-1, \\ l_{r,r} = \sin(\theta_{r,1}) \cdots \sin(\theta_{r,r-2}) \sin(\theta_{r,r-1}), \end{cases} \tag{14}$$

where $\theta_{r,s} \in (0, \pi)$. Denote by $\mathbf{\Theta}$ all $\theta_{r,s}$ involved in (14).

Suppose that the computer model under consideration is conducted at n different input values, $D_{\mathbf{w}} = (\mathbf{w}_1^0, \ldots, \mathbf{w}_n^0)$, with the corresponding response values denoted by $\mathbf{y} = (y_1, \ldots, y_n)^t$. The parameters in model (9) to be estimated are σ^2, $\boldsymbol{\beta}$, $\mathbf{\Phi}$ and $\mathbf{\Theta}$. The *maximum likelihood estimators* of these parameters are denoted by $\hat{\sigma}^2$, $\hat{\boldsymbol{\beta}}$, $\hat{\mathbf{\Phi}}$ and $\hat{\mathbf{\Theta}}$, respectively. The log-likelihood function of $\mathbf{y}$, up to an additive constant, is

$$-\frac{1}{2}\left[n \log(\sigma^2) + \log(|\mathbf{R}|) + (\mathbf{y} - \mathbf{F}\beta)^t \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\beta)/\sigma^2\right], \tag{15}$$

where $\mathbf{F} = \left[\mathbf{f}(\mathbf{w}_1^0), \ldots, \mathbf{f}(\mathbf{w}_n^0)\right]^t$ is an $n \times p$ matrix and $\mathbf{R}$ is the correlation matrix with (i,j)th entry $\text{cor}\left[\epsilon(\mathbf{w}_i^0), \epsilon(\mathbf{w}_j^0)\right]$ defined in (11). Given $\mathbf{\Phi}$ and $\mathbf{\Theta}$, $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ are

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^t \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^t \mathbf{R}^{-1} \mathbf{y}, \tag{16}$$

and

$$\hat{\sigma}^2 = (\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}})^t \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}})/n. \tag{17}$$

Plugging (16) and (17) into (15), $\hat{\mathbf{\Phi}}$ and $\hat{\mathbf{\Theta}}$ can be obtained as

$$(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Theta}}) = \arg\min_{\mathbf{\Phi},\mathbf{\Theta}} \{n \log(\hat{\sigma}^2) + \log(|\mathbf{R}|)\}. \tag{18}$$

The optimization problem in (18) only involves the constraints that $\theta_{r,s} \in (0, \pi)$ for $\hat{\mathbf{\Theta}}$ and $\phi_i \geq 0$ for $\hat{\mathbf{\Phi}}$. It can be solved by modifying the DACE toolbox in Matlab [16] to incorporate the reparameterization in (14). A small nugget term is added to the diagonals of $\mathbf{R}$ to mitigate potential singularity. The fitted model can be used to predict the response value y at any untried input value. Given $\hat{\sigma}^2$, $\hat{\boldsymbol{\beta}}$, $\hat{\mathbf{\Phi}}$ and $\hat{\mathbf{\Theta}}$, the *empirical best linear unbiased predictor* (EBLUP) of y at any input value $\mathbf{w}_0$ is

$$\hat{y}(\mathbf{w}_0) = \mathbf{f}^t(\mathbf{w}_0)\hat{\boldsymbol{\beta}} + \hat{\mathbf{r}}_0^t \hat{\mathbf{R}}^{-1}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}), \tag{19}$$

where $\hat{\mathbf{r}}_0 = \left\{\text{cor}\left[\epsilon(\mathbf{w}_0^0), \epsilon(\mathbf{w}_1^0)\right], \ldots, \text{cor}\left[\epsilon(\mathbf{w}_0^0), \epsilon(\mathbf{w}_n^0)\right]\right\}^t$ and $\hat{\mathbf{R}}$ is the estimated correlation matrix of $\mathbf{y}$. Similarly to its counterpart for the Gaussian process model in (10) with quantitative factors, the EBLUP in (19) smoothly interpolates all the observed data points. Features of the function $\hat{y}(\mathbf{w})$ can be visualized by plotting the estimated functional main effects and interactions.

In this work, we consider four methods for building Gaussian process models for a computer experiment with qualitative and quantitative factors.

- The individual Kriging method, denoted by IK. This method fits data associated with different qualitative levels separately using distinct Gaussian process models for the quantitative variables in (10).
- The *exchangeable correlation* method for the qualitative factors, denoted by EC. It assumes the cross-correlation $\tau_{r,s}$ in (11) to be

$$\tau_{r,s} = c \; (0 < c < 1) \quad \text{for } r \neq s.$$

- The *multiplicative correlation* method for the qualitative factors, denoted by MC. It assumes the cross-correlation $\tau_{r,s}$ in (11) to be

$$\tau_{r,s} = \exp\{-(\theta_r + \theta_s) I[r \neq s]\} \; (\theta_r, \theta_s > 0).$$

- The method proposed in (13) and (14) with an *unrestricted* correlation function for the qualitative factors, denoted by UC.

2.3 Treed Gaussian Processes (TGP)

In practice, many situations involving the emulation of computer models call for more flexibility than is reasonable under the common assumption of stationarity. However, a fully nonstationary model may be undesirable as well, because of the vastly increased difficulty of performing inference due to a nonstationary model's complexity. A good compromise can be local stationarity. A treed Gaussian process (TGP) [8] is designed to take advantage of local stationarity. It defines a treed partitioning process on the predictor space and fits distinct,

but hierarchically related, stationary GPs to separate regions at the leaves. The treed form of the partition makes the model easily interpretable: having the treed partitions with separate GPs makes it easy to identify the GP model in each branch. At the same time, the partitioning results in smaller matrices for inversion than would be required under a standard GP model and thereby provides a nonstationary model that actually facilitates faster inference. Using a fully Bayesian approach allows for model averaging across the tree space, resulting in smooth and continuous fits when the data are not naturally partitioned. The partitions are fit simultaneously with the individual GP parameters using reversible jump Markov chain Monte Carlo, so that all parts of the model can be learned automatically from the data. The posterior predictive distribution thus takes into account uncertainty from the data, from the fitted parameters, and from the fitted partitions.

TGP inherits its partitioning scheme from simpler treed models such as CART [3] and BCART (for Bayesian CART) [5] . Each uses recursive binary splits so that each branch of the tree in any of these models divides the predictor space in two, with multiple splits allowed on the same variable for full flexibility. Consider predictors $x \in R^P$ for some split dimension $p \in \{1, ..., P\}$ and split value v, points with $x_p \leq v$ are assigned to the left branch, and points with $x_p > v$ are assigned to the right branch. Partitioning is recursive and may occur on any input dimension p, so arbitrary axis-aligned regions in the predictor space may be defined. Conditional on a treed partition, models are fit in each of the leaf regions. In CART the underlying models are "constant" in that only the mean and standard deviation of the real-valued outputs are inferred. TGP fits a Gaussian process Z_ν in each leaf ν using the following hierarchical model:

$$\mathbf{Z}_\nu|\boldsymbol{\beta}_\nu, \sigma_\nu^2, \mathbf{K}_\nu \sim N_{n_\nu}(\mathbf{F}_\nu\boldsymbol{\beta}_\nu, \sigma_\nu^2\mathbf{K}_\nu) \quad \boldsymbol{\beta}_0 \sim N_m(\mu, \mathbf{B})$$

$$\sigma_\nu^2 \sim IG(\alpha_\sigma/2, q_\sigma/2)$$

$$\boldsymbol{\beta}_\nu|\sigma_\nu^2, \tau_\nu^2, \mathbf{W}, \boldsymbol{\beta}_0 \sim N_m(\boldsymbol{\beta}_0, \sigma_\nu^2\tau_\nu^2\mathbf{W}) \quad \mathbf{W}^{-1} \sim W((\rho\mathbf{V})^{-1}, \rho)$$

$$\tau_\nu^2 \sim IG(\alpha_\tau/2, q_\tau/2) \tag{20}$$

where $\mathbf{F}_\nu = (\mathbf{1}, \mathbf{X}_\nu)$ contains the data in that leaf. N, IG, and W are the Multivariate Normal, Inverse–Gamma, and Wishart distributions, respectively. $\mathbf{K}_\nu$ is the separable power family covariance matrix with a nugget.

Classical treed methods, such as CART, can cope quite naturally with categorical, binary, and ordinal inputs. For example, categorical inputs can be encoded in binary, and splits can be proposed with rules such as $x_p < 1$. Once a split is made on a binary input, no further process is needed, marginally, in that dimension. Ordinal inputs can also be coded in binary, and thus treated as categorical, or treated as real-valued and handled in a default way. This formulation presents an alternative to that of Section 2.2. While that formulation allows a powerful and flexible representation of qualitative inputs in the model, it does not allow for nonstationarity. TGP allows the combination of qualitative inputs and nonstationary modeling.

Rather than manipulate the GP correlation to handle categorical inputs, the tree presents a more natural mechanism for such binary indicators. That is, they can be included as candidates for treed partitioning but ignored when it comes to fitting the models at the leaves of the tree. The benefits of removing the Booleans from the GP model(s) go beyond producing full-rank design matrices at the leaves of the tree. Loosely speaking, removing the Boolean indicators from the GP part of the treed GP gives a more parsimonious model. The tree is able to capture all of the dependence in the response as a function of the indicator input, and the GP is the appropriate nonlinear model for accounting for the remaining relationship between the real-valued inputs and outputs. Further advantages to this approach include speed (a partitioned model gives smaller covariance matrices to invert) and improved mixing in the Markov chain when a separable covariance function is used. Finally, the treed model allows the practitioner to immediately ascertain whether the response is sensitive to a particular categorical input by tallying the proportion of time the Markov chain visited trees with splits on the corresponding binary indicator. A much more involved Monte Carlo technique (e.g., following [24]) would otherwise be required in the absence of the tree. Here we use the implementation developed by Broderick and Gramacy [4].

3 Testing and Assessment Approach

In order to evaluate and compare the three mixed-variable surrogate modeling approaches, we established a common experimental strategy that can be consistently applied to all of them. There are three primary components to which we paid particular attention. They are the test functions, the sample design used for surrogate construction, and the comparison metrics. Each is described in the following subsections.

3.1 Test Functions

In order to consolidate a common set of test functions on which to evaluate the different surrogate approaches, we developed a generic C++ testbed. It was designed and developed to meet the following requirements:

- ability to control the number of discrete variables and the number of levels per discrete variable in order to test method scalability with respect to these features,
- ability to control problem complexity in order to evaluate performance on a variety of problems,
- extendable in order to easily add new functions, and
- easy to use in multiple computing environments across all surrogate software packages.

The testbed include both defined hard-wired functions and randomly-generated polynomial functions. The latter is based on work by McDaniel and Ankenman [17]. We refer the reader to [33] for more details on the implementation.

3.2 Sample Design

The accuracy of a response surface surrogate can be affected by the number of data points used to build it as well as how those points are chosen. Therefore, we vary the number and design of build points in our numerical experiments. All designs are based on Latin Hypercube designs (LHD) of the parameter space. We define n to be the number of LHD runs per qualitative level of the categorical variables and m to be the number of discrete levels (or combinations of levels). The total number of points used to build each surrogate is mn. We consider $n = 10, 20, 40, 80$, and the sample design for each training set is constructed in three different ways.

Standard Latin Hypercube. In this approach, one Latin Hypercube design of size mn is generated over all of the continuous parameters. It is then randomly split it into m groups of n samples, and each group is assigned a qualitative level of the categorical variables.

k Latin Hypercube. In this approach, a separate Latin Hypercube design is generated for every given level of categorical variables. That is, we generate m independent Latin hypercube designs, each of size n and corresponding to one qualitative level.

Sliced Latin Hypercube. This approach is based on recent work by Qian [19]. This design is a Latin Hypercube for the continuous factors and is sliced into groups of smaller Latin Hypercube designs associated with different categorical levels. In this case, we generate a sliced Latin hypercube design with m slices, where each slice of n runs corresponds to one qualitative level.

Because of the randomness associated with the LHS samples, we generate 10 replicate training sets for each combination of n and design type.

3.3 Comparison Metric

Evaluating the performance of computational methods can be challenging, particularly with regard to the accuracy of the method. This is because the accuracy required for different applications of the method can vary. In this study, our primary focus is on gaining an understanding of the accuracy of mixed variable surrogates relative to each other, so we use a relatively fine-grained metric. In particular, we use mean squared error between surrogate predictions and true function values over a set of given points. For every replication of a given n and training design type, the mean squared errors (MSE) are calculated based on a testing set using a Latin hypercube design with 200 samples for each qualitative level. We then compare the mean and spread of the errors. Lower values of these quantities constitute better performance.

4 Results

We present selected results of our evaluation of the surrogate approaches discussed in Section 2. Specifically, we compared the results of TGP, ACOSSO,

and the Gaussian process model with the various special correlation functions. Generally, we found that sample design type (standard Latin Hypercube, kLHD, or sliced LHD) did not have a large effect on the MSE. Thus, we show only the results using the sliced LHD. We applied these different surrogate methods to a set of test functions to be described and compared the results over different training set sizes.

4.1 Test Function 2

The first function we considered in our numerical experiments has one categorical variable with five levels. It also two continuous variables, both of which fall between the values of 0 and 1. This function has regions where the responses at the different categorical levels are very similar. This will allow us to evaluate how well the different surrogate approaches can resolve the different discrete levels.

$$f(x) = \begin{cases} \sin(2\pi x_3 - \pi) + 7\sin^2(2\pi x_2 - \pi) & \text{if } x_1 = 1 \\ \sin(2\pi x_3 - \pi) + 7\sin^2(2\pi x_2 - \pi) + 12.0\sin(2\pi x_3 - \pi) & \text{if } x_1 = 2 \\ \sin(2\pi x_3 - \pi) + 7\sin^2(2\pi x_2 - \pi) + 0.5\sin(2\pi x_3 - \pi) & \text{if } x_1 = 3 \\ \sin(2\pi x_3 - \pi) + 7\sin^2(2\pi x_2 - \pi) + 8.0\sin(2\pi x_3 - \pi) & \text{if } x_1 = 4 \\ \sin(2\pi x_3 - \pi) + 7\sin^2(2\pi x_2 - \pi) + 3.5\sin(2\pi x_3 - \pi) & \text{if } x_1 = 5 \end{cases}$$

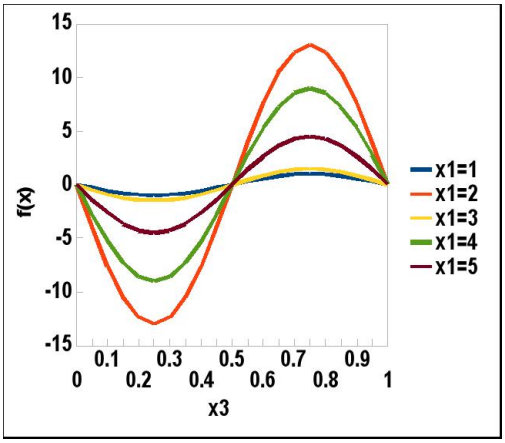

Fig. 1. Test Function 2

Figures 2-3 give the boxplots of the MSEs of the four methods for $n = 10, 20$. The Y axis is the mean squared error (MSE) of the surrogate construction. The Gaussian correlation function in (12) for the quantitative factors is used in the IK, EC, MC and UC methods. Note that the Y-axis scale is different on these figures. Ideally, it would be nice to see the MSE plotted on the same scale so

that it is easy to see the decrease in error as a function of the number of training samples. However, the MSE varied so dramatically for some of these results that keeping an MSE scale to allow for plotting maximum MSE values would result in the reader not seeing the differences in situations where the MSE was low.

Overall, ACOSSO does very well on this function and outperforms the other methods, especially at the smaller sample levels of $n = 10$ and $n = 20$. We expect this is because the structure of the ACOSSO surrogate maps naturally to separable functions such as this one. For the four GP correlation schemes, the EC, MC and UC methods outperform the IK method.

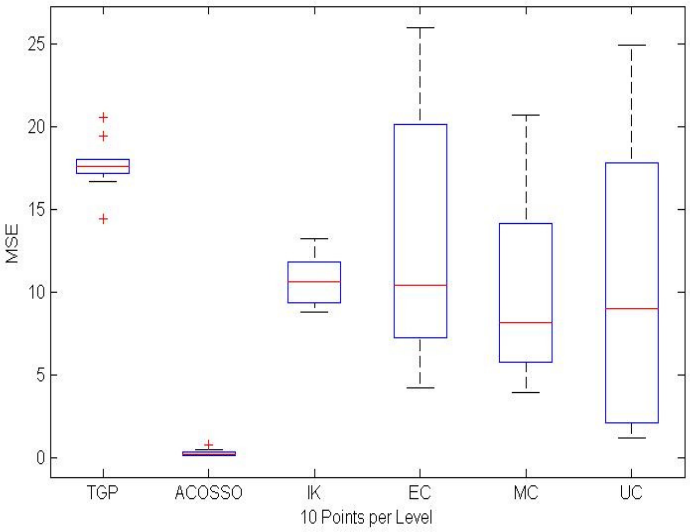

Fig. 2. Test Function 2. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 10$ using the sliced LHD scheme.

In summary for Test Function 2: ACOSSO performed the best overall, the GP variations with IK, EC, MC and UC methods also performed well.

4.2 Goldstein-Price

The second function we considered is the Goldstein-Price function. It has one continuous variable and one discrete variable. The discrete variable, x_1, can take on the values of $-2, 0$, and 2. The continuous variable, x_2, ranges between the values of -2 and 2. This function varies by five orders of magnitude over the domain we chose, so we performed the surrogate construction in log space and the error is presented in log space.

$$f(x) = (1 + (x_1 + x_2 + 1)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \cdot \\ (30 + (2x_1 - 3x_2)^2 \cdot (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$$

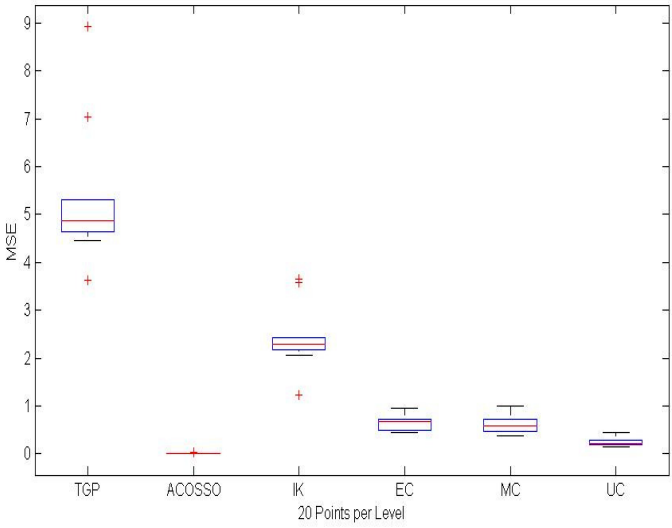

Fig. 3. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 20$ using the sliced LHD scheme

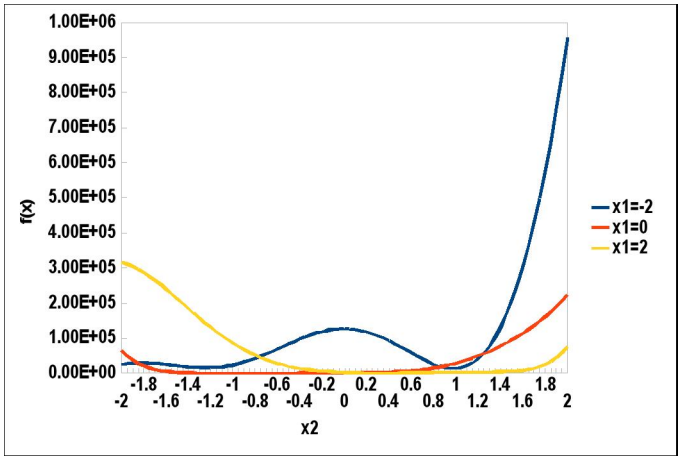

Fig. 4. Goldstein Price Function

Figures 5-6 give the boxplots of the MSEs of the four methods for $n = 10, 20, 40$. In these figures, the Y axis is the mean squared error of the surrogate in log space. For the Gaussian process model, the four variations of IK, EC, MC and UC methods all used the compact support Wendland correlation function. For the Goldstein-Price function, the compactly supported correlation performed better than the Gaussian correlation function.

Overall, the variations of the Gaussian process model do very well on this function. ACOSSO also performs well, and the mean MSE from ACOSSO is close to the mean from the various GP methods. However, the variability of the ACOSSO results is slightly larger, as shown in Figures 5-6. Note that TGP has larger MSE at all sample levels. However, when we performed the surrogate construction in the original space without taking the logarithm of the Goldstein-Price function, TGP outperformed the other methods. This is likely due to the ability of TGP to identify different regions of the space with different properties (e.g. the scale of the Goldstein-Price function is much smaller in the center of the domain than at the edges of the domain we are using for this case study).

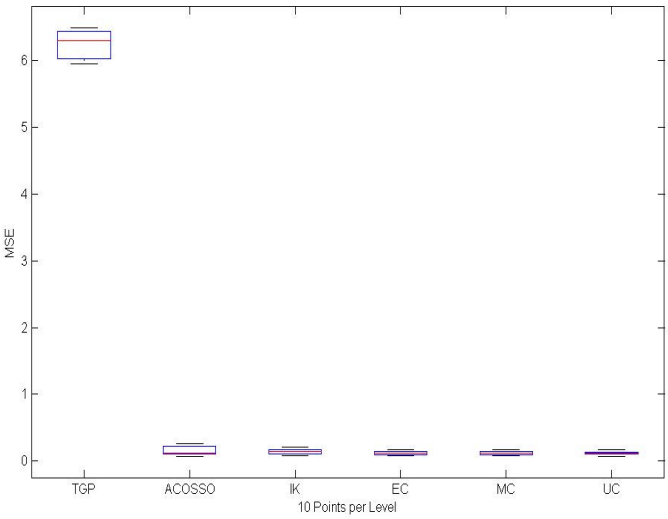

Fig. 5. Goldstein-Price. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 10$ using the sliced LHD scheme.

4.3 Fourth Order Polynomial

Using the polynomial generator, we randomly generated a 19-term fourth order polynomial. It has four parameters, two of which are continuous and two of which are discrete. The x_3 and x_4 are continuous variables that fall between 0 and 100, and x_1 and x_2 are discrete variables that have three levels, namely 20, 50, and 80. The polynomial is given by the following:

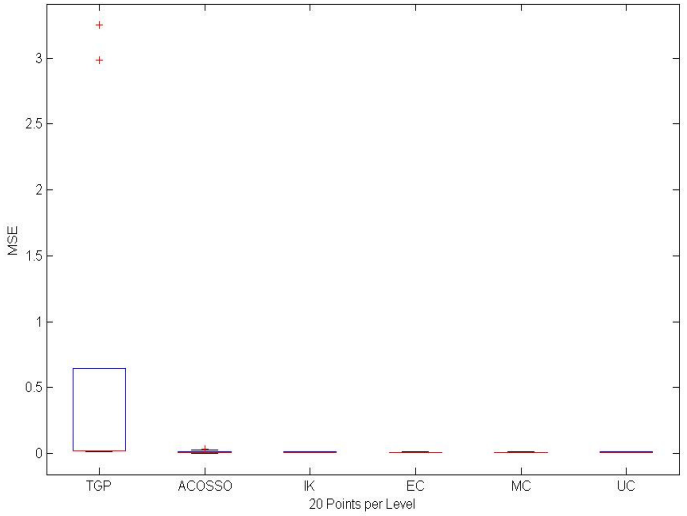

Fig. 6. Goldstein-Price. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 20$ using the sliced LHD scheme.

$$
\begin{aligned}
f(x) = {} & 53.3108 + 0.184901x_1 - 5.02914 \cdot 10^{-6}x_1^3 + 7.72522 \cdot 10^{-8}x_1^4 - \\
& 0.0870775x_2 - 0.106959x_3 + 7.98772 \cdot 10^{-6}x_3^3 + 0.00242482x_4 + \\
& 1.32851 \cdot 10^{-6}x_4^3 - 0.00146393x_1x_2 - 0.00301588x_1x_3 - \\
& 0.00272291x_1x_4 + 0.0017004x_2x_3 + 0.0038428x_2x_4 - 0.000198969x_3x_4 + \\
& 1.86025 \cdot 10^{-5}x_1x_2x_3 - 1.88719 \cdot 10^{-6}x_1x_2x_4 + 2.50923 \cdot 10^{-5}x_1x_3x_4 - \\
& 5.62199 \cdot 10^{-5}x_2x_3x_4
\end{aligned}
$$

The results for the fourth order polynomial are shown in Figures 7-8. These figures show that the Gaussian processes with the various correlation functions such as EC, MC, etc. perform well. Interestingly, ACOSSO does not seem to improve, even as the size of training set increases. That is, the average MSE for ACOSSO with $n = 10$ is 1.5, while the average MSE for ACOSSO with $n = 80$ is 1.4. In contrast, the other approaches all improve the MSE by eight orders of magnitude. We hypothesize that ACOSSO is struggling when there is significant interaction between variables. In particular, it is trying to construct its response as the aggregation of separable functions which may not capture the interactions well.

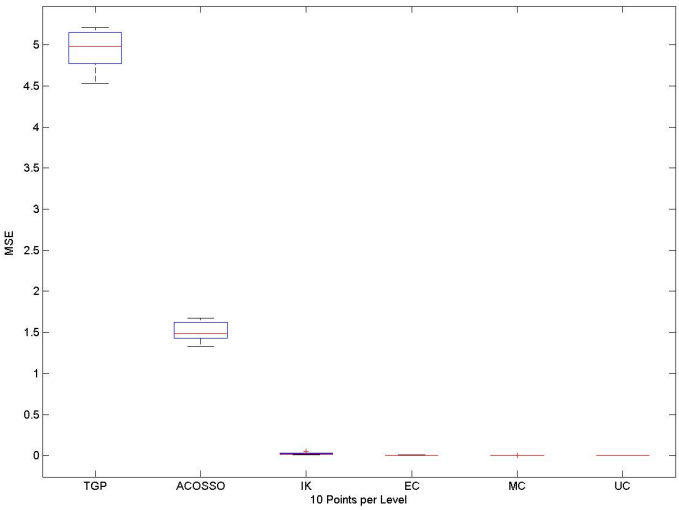

Fig. 7. Fourth-order Polynomial. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 10$ using the sliced LHD scheme.

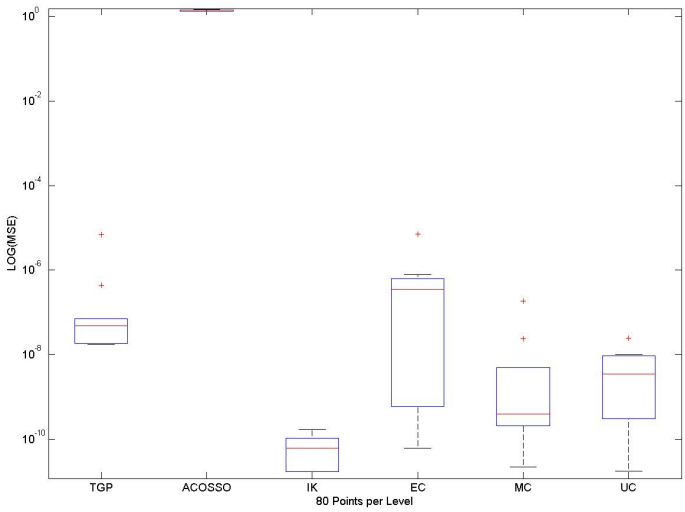

Fig. 8. Fourth-order Polynomial. Boxplots of the MSEs for the TGP, ACOSSO, IK, EC, MC and UC methods with $n = 80$ using the sliced LHD scheme.

5 Summary

This paper investigated three main classes of surrogate methods which can handle "mixed" discrete and continuous variables: adaptive smoothing splines, Gaussian processes with special correlation functions, and Treed Gaussian processes. We chose test problems which were challenging but tractable for repeated comparison runs. The results presented are representative of the extensive comparisons we performed, varying the number of build points used in the surrogate construction, varying the sample designs used, and building multiple surrogates of a given type so that we could compute statistics of the response to give fair comparisons (e.g. so we would not be misled by constructing only one surrogate on one set of build points).

Overall, all methods appear viable for small numbers of categorical variables with a few levels, and we were able to gain some general insights across the wide range of studies performed. ACOSSO and the Gaussian processes with special correlation functions generally performed well. ACOSSO performed best for separable functions, especially at small training set sizes. This is a particularly valuable trait, as computational expense usually prevents large training set sizes. When there are significant interactions between discrete and continuous parameters, as in the fourth-order polynomial, ACOSSO performs poorly even with a larger number of training points (40 or 80). Both results are expected because ACOSSO is constructed over separable functions, and its performance may degrade somewhat when significant interactions between variables are present. The GP with special correlation functions appears the most consistent of all the methods. However, that approach was the most sensitive to build design and did not perform as well with a plain LHD design, whereas ACOSSO and TGP were not significantly affected by the design. TGP success depends on being able to identify splits where individual GPs work well in separate parts of the domain. TGP performs well on poorly scaled functions, but we found it does not perform well when the continuous variables are not predictive for certain combinations of categorical variable levels. These insights will allow us to move forward with applying these surrogate methods to computational analysis problems for which they are best suited.

Acknowledgements. The authors would like to thank a number of colleagues for many fruitful discussions on this topic: William Hart, John Siirola, Jean-Paul Watson, Genetha Gray, Cynthia Phillips, Ali Pinar, and David Woodruff. We also thank Sandia management, specifically M. Daniel Rintoul, and the Laboratory Directed Research and Development office at Sandia for supporting this project. Finally, we would like to thank Ken Perano for writing the C++ software that encodes the test functions we used to evaluate the surrogate approaches.

References

1. Adams, B.M., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Eldred, M.S., Gay, D.M., Hough, P.D., Swiler, L.P.: DAKOTA, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 5.1 user's manual. Technical Report SAND2010-2183, Sandia National Laboratories, Albuquerque, NM (2010), `http://dakota.sandia.gov/documentation.html`
2. Berlinet, A., Thomas-Agnan, C.: Reproducing Kernel Hilbert Spaces in Probability and Statistics. Kluwer Academic Publishers, Norwell (2004)
3. Breiman, L., Friedman, J.H., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth, Belmont (1984)
4. Broderick, T., Gramacy, R.B.: Classification and categorical inputs with treed Gaussian process models. Journal of Classification 28(2), 244–270 (2011)
5. Chipman, H., George, E., McCulloch, R.: Bayesian CART Model Search (with discussion). Journal of the American Statistical Association 93, 930–960
6. Craven, P., Wahba, G.: Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. Numerical Mathematics 31, 377–403 (1979)
7. Eubank, R.L.: Nonparametric Regression and Spline Smoothing. CRC Press, London (1999)
8. Gramacy, R.B., Lee, H.K.H.: Bayesian treed gaussian process models with an application to computer modeling. Journal of the American Statistical Association 103, 1119–1130 (2008)
9. Gramacy, R.B., Lee, H.K.H.: Gaussian processes and limiting linear models. Computational Statistics and Data Analysis 53, 123–136 (2008)
10. Gramacy, R.B., Taddy, M.: Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an r package for treed gaussian process models. Technical report, R manual (2009), `http://cran.r-project.org`
11. Gu, C.: Smoothing Spline ANOVA Models. Springer, New York (2002)
12. Hastie, T., Tibshirani, R.J.: Generalized Additive Models. Chapman & Hall/CRC, London (1990)
13. Kennedy, M.C., O'Hagan, A.: Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society B 63, 425–464 (2001)
14. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, vol. 2(12), pp. 1137–1143 (1995)
15. Lin, Y., Zhang, H.: Component selection and smoothing in smoothing spline analysis of variance models. Annals of Statistics 34(5), 2272–2297 (2006)
16. Lophaven, S.N., Neilson, H.B., Sondergaard, J.: Dace - a matlab kriging toolbox. Technical report (2009), `http://www2.imm.dtu.dk/~hbn/dace/`
17. McDaniel, W.R., Ankenman, B.E.: A response surface test bed. Quality and Reliability Engineering International 16, 363–372 (2000)
18. Neter, J., Wasserman, W., Kutner, M.H.: Applied Linear Statistical Models, 2nd edn. Irwin (1985)
19. Qian, P.Z.G.: Sliced latin hypercube designs (2011) (submitted)

20. Qian, P.Z.G., Wu, H., Wu, C.F.J.: Gaussian process models for computer experiments with qualitative and quantitative factors. Technometrics 50(3), 383–396 (2008)
21. Rebonato, R., Jackel, P.: The most general methodology for creating a valid correlation matrix for risk management and option pricing purposes. The Journal of Risk 2, 17–27 (1999)
22. Reich, B.J., Storlie, C.B., Bondell, H.D.: Variable selection in bayesian smoothing spline anova models: Application to deterministic computer codes. Technometrics 51(2), 110–120 (2009)
23. Sacks, J., Welch, W.J., Mitchel, T.J., Wynn, H.P.: Design and analysis of computer experiments. Statistical Science 4(4), 409–435 (1989)
24. Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., Tarantola, S.: Global sensitivity analysis, The Primer. Wiley and Sons (2008)
25. Santner, T., Williams, B., Notz, W.: The Design and Analysis of Computer Experiments. Springer, New York (2003)
26. Schimek, M. (ed.): Smoothing and Regression: Approaches, Computation, and Application. John Wiley, New York (2000)
27. Simpson, T.W., Toropov, V., Balabanov, V., Viana, F.A.C.: Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come – or not. In: Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia, Canada (September 2008) AIAA Paper 2008-5802
28. Storlie, C.B., Bondell, H.D., Reich, B.J.: A locally adaptive penalty for estimation of functions with varying roughness. Journal of Computational and Graphical Statistics 19(3), 569–589 (2010)
29. Storlie, C.B., Bondell, H.D., Reich, B.J., Zhang, H.H.: Surface estimation, variable selection, and the nonparametric oracle property. Statistica Sinica 21(2), 679–705 (2010)
30. Storlie, C.B., Helton, J.C.: Multiple predictor smoothing methods for sensitivity analysis: Description of techniques. Reliabililty Engineering and System Safety 93(1), 28–54 (2008)
31. Storlie, C.B., Swiler, L.P., Helton, J.C., Sallaberry, C.J.: Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models. Reliability Engineering and System Safety 94(11), 1735–1763 (2009)
32. Swiler, L.P., Wyss, G.D.: A user's guide to Sandia's latin hypercube sampling software: LHS UNIX library and standalone version. Technical Report SAND04-2439, Sandia National Laboratories, Albuquerque, NM (July 2004)
33. Swiler, L.P., Hough, P.D., Qian, P., Xu, X., Storlie, C.B., Lee, H.: Surrogate models for mixed discrete-continuous variables. Technical Report SAND2012-0491, Sandia National Laboratories, Albuquerque, NM (August 2012)
34. Tibshirani, R.J.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society B 58, 267–288 (1996)
35. Viana, F.A.C., Haftka, R.T., Steffan Jr., V.: Multiple surrogates: How cross-validation errors can help us obtain the best predictor. Structural and Multidisciplinary Optimization 39(4), 439–457 (2009)

36. Viana, F.A.C., Haftka, R.T., Steffan Jr., V., Butkewitsch, S., Leal, M.F.: Ensemble of surrogates: a framework based on minimization of the mean integrated square error. In: Proceedings of the 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Schaumburg, IL (April 2008) AIAA Paper 2008–1885
37. Wahba, G.: Spline Models for Observational Data. CBMS-NSF Regional Conference Series in Applied Mathematics (1990)
38. Wu, Z.: Multivariate compactly supported positive definite radial functions. Advances in Computational Mathematics 4, 283–292 (1995)
39. Zhou, Q., Qian, P.Z.G., Zhou, S.: A simple approach to emulation for computer models with qualitative and quantitative factors. Technometrics 53, 266–273 (2011)