Statistical Tests for Cross-Validation of Kriging Models

Kleijnen, Jack; van Beers, W.C.M.

Publication date:
2019

Document Version
Early version, also known as pre-print

No. 2019-022

STATISTICAL TESTS FOR CROSS-VALIDATION OF KRIGING MODELS

By

Jack P.C. Kleijnen, Wim C.M. van Beers

3 June 2019

ISSN 0924-7815
ISSN 2213-9532

Statistical tests for cross-validation of Kriging models

Jack P.C. Kleijnen and Wim C.M. van Beers
Department of Management
Tilburg School of Economics and Management (TiSEM)
Tilburg University (TiU)
Postbox 90153, 5000 LE Tilburg, Netherlands
E-mail: kleijnen@tilburguniversity.edu
Orcid iD: https://orcid.org/0000-0001-8413-2366

May 29, 2019

Abstract

Abstract: We derive new statistical tests for leave-one-out cross-validation of Kriging models. Graphically, we present these tests as scatterplots augmented with confidence intervals. We may wish to avoid extrapolation, which we define as prediction of the output for a point that is a vertex of the convex hull of the given input combinations. Moreover, we may use bootstrapping to estimate the true variance of the Kriging predictor. The resulting tests (with or without extrapolation or bootstrapping) have type-I and type-II error probabilities, which we estimate through Monte Carlo experiments. To illustrate the application of our tests, we use an example with two inputs and the popular borehole example with eight inputs.

Keywords: validation, cross-validation, Kriging, Gaussian process, extrapolation, convex hull, Monte Carlo

JEL: C0, C1, C9, C15, C44

1 Introduction

In this publication we derive several variants of a new type of statistical test for *leave-one-out cross-validation* (LOO-CV) of an estimated Kriging model. Graphically, we may present this test through a new type of scatterplot that adds *confidence intervals* (CIs) to the classic scatterplot (readers may take a

peek at the plot in Fig. 8). We focus on Kriging (meta)models that approximate the *input/output* (I/O) functions implicitly defined by the underlying simulation models; however, Kriging may also be applied to real-world data. The advantage of Kriging compared with alternative methods—such as neural nets (NNs), radial basis functions (RBFs), and splines—is that Kriging also quantifies the variance of its predictor—like regression analysis does; our test uses this predictor variance. For a discussion of Kriging within the context of LOO-CV we refer to Section 3; for a general discussion of Kriging we refer to Kleijnen (2019).

LOO-CV for Kriging consists of the following steps (we present an algorithm and variants, in Section 5): (i) Delete one of the n simulated I/O combinations (say) $(\mathbf{x}_i, v_i)$ $(i = 1, ..., n)$. (ii) Fit a Kriging model to the $(n-1)$ remaining I/O combinations $(\mathbf{X}_{-i}, \mathbf{v}_{-i})$. (These combinations are often called the ″training set".*)* (iii) Using the model fitted in step 2, compute $\widehat{y}_i$(predicted output of $\mathbf{x}$, deleted in step 1), and $s^2(\widehat{y}_i)$ (estimated variance of $\widehat{y}_i$). (The "test set" has a single member, in LOO-CV.) (iv) Apply the preceding three steps for all combinations $i = 1, ..., n$.

CV is a very popular method; e.g., our Google search for "cross-validation" delivered 146 million hits (on 14 August 2018). We review selected recent publications in Section 2. Our review suggests that our specific statistical test for LOO-CV in Kriging is a novel test indeed.

We focus on deterministic simulation, but conjecture that extension to random or stochastic simulation is straightforward (also see Section 8, which includes future research). Furthermore, we use a frequentist approach instead of a Bayesian approach.

Note: An advantage of the frequentist approach is that it is not necessary to specify a prior distribution for the Kriging (hyper)parameters; we find it hard to specify such a prior distribution. We discuss the observed values for the estimated Kriging parameters, in Section 6 on our *Monte Carlo* (MC) experiments. Wang et al. (2018) discusses the use of historic data for the specification of a prior distribution. Zou and Zhang (2018) also discusses frequentist and Bayesian approaches for Kriging, and chooses the frequentist approach. For a general discussion of frequentist and Bayesian approaches we refer to Efron (2015). Hasty readers may skip paragraphs that start with "Note:".

It is well known that Kriging may give an inaccurate approximation in case of *extrapolation* (i.e., Kriging is meant for interpolation); see the discussion in Kleijnen (2015, p. 187). If we wish to avoid such extrapolation in LOO-CV, we may require that the left-out input combination $\mathbf{x}_i$ is not a vertex of the *convex hull* (CH) of $\mathbf{X}$, which is the matrix with the n simulation input combinations. An example of a CH is given in Fig. 2, which we shall discuss later In various sections we shall investigate whether extrapolation is indeed a problem in LOO-CV.

Altogether, our LOO-CV test computes the *Studentized* prediction errors for the left-out combinations, and applies Bonferroni's inequality to tests these errors "jointly" or "experimentwise". Actually, we propose several variants of our test, including the CH requirement and the unbiased bootstrapped predictor

variance. We detail these variants in Section 5.

Like any other statistical test, our test has a *type-I* or α *error rate* (or error probability). To estimate this rate, we use a MC experiment guaranteeing that the Kriging metamodel is "perfectly valid"; i.e., the metamodel and the simulation model are identical (practical simulation models imply imperfect metamodels). So, we sample from a stationary *Gaussian process* (GP), which implies a multivariate normal distribution with a given mean—e.g., a constant, as assumed by so-called *ordinary Kriging* (OK)—and a given covariance matrix—e.g., a matrix with an anisotropic Gaussian correlation function. We limit our MC experiment to $d = 2$ inputs. To the best of our knowledge, MC experiments aimed at estimating the α error rate of LOO-CV in Kriging, are new.

Besides the type-I error rate, any test has a *type-II* or β *error rate*. We use several MC experiments that control the magnitude of the approximation error of the Kriging metamodel. More specifically, we fit OK metamodels to I/O data sampled from several GPs that have means that show linear trends with different slopes. (We might imagine a different MC experiment that would fit an OK metamodel to I/O data sampled from a GP with a Matérn—instead of a Gaussian—correlation function.)

Note: In general, MC experiments are more effective and efficient than experiments with realistic simulation models or simplified simulation models like the borehole model in Section 7.2. Indeed, a Kriging metamodel of a given simulation model gives an unknown approximation error, whereas MC experiments enable perfect control of this error—starting with zero error (to estimate the α error rate)—so MC experiments are more effective. Moreover, simulation may require much computer time (unlike the borehole model), whereas MC experimentation does not—so the latter is more efficient.

To illustrate our LOO-CV we apply our validation test to two simple simulation models. The first model is due to Gramacy (2016); it has $d = 2$ inputs and an I/O function with many hilltops and fast-moving changes (see the Figure in Appendix 4). The second model is the popular borehole model with $d = 8$ inputs, which is used in many publications; e.g., Erickson et al. (2018), Gramacy (2016), Gramacy and Apley (2015), Santner et al. (2018. p. 222), and Sun et al. (2018).

Our major contribution is a thorough investigation of several variants of a new statistical tests for LOO-CV in Kriging. We also investigate the estimated OK parameters if either the OK assumptions hold or the data show a linear trend.

We organize the rest of this paper as follows. Section 2 reviews literature, focussing on LOO-CV in Kriging. Section 3 summarizes OK in the context of LOO-CV. Section 4 summarizes the type of *Latin hypercube sampling* (LHS) that we assume for our experiments. Section 5 presents several variants of a new test statistic for LOO-CV. Section 6 details the design and analysis of our MC experiments for estimating the α and β error rates of our LOO-CV variants; these experiments also improve our understanding of the roles of the Kriging parameters. Section 7 applies our LOO-CV to Gramacy (2016)'s example and

the borehole example. Section 8 summarizes conclusions and future research topics.

2 Literature review

We focus on *recent* CV publications, so we do not discuss an old milestone publication such as Stone (1974). LOO-CV is discussed in Santner et al. (2018), which is a classic textbook on Kriging; however, that discussion does not cover the test that we derive in this paper. Bartz-Beielstein (2016) discusses k-fold CV (if $k = 1$, then this equals LOO-CV), to select an "ensemble" (combination) of different types of metamodels including Kriging metamodels. Viúdez-Moreiras (2018) also discusses k-fold CV. Xiao et al. (2018) uses k-fold CV (including LOO-CV) in sequential sampling for reliability problems, focusing on a so-called learning function to select the next point to be simulated; this function is not a measure of the metamodel accuracy (see Xiao et al., p. 409). Shu et al. (2018) also uses LOO-CV to select the next point to be simulated; that LOO-CV measures the metamodel accuracy through either the *root mean squared prediction error* (RMSPE) defined in (11) below or the *maximum absolute error* (MAE) instead of our measure defined in (10); Garbo and German (2019) also uses LOO-CV with the MSPE (the square of the RMSPE) to select the next point. Rasmussen and Williams (2006) discusses LOO-CV within a Bayesian framework. Bhosekar and Ierapetritou (2018) surveys k-fold CV including LOO-CV for Kriging and other types of metamodels in simulation. Strano et al. (2018) and Yin et al. (2018) apply LOO-CV to select the best type of metamodel. Van Steenkiste et al. (2018) uses k-fold CV for sequential *sensitivity analysis* (SA) through metamodels including Kriging metamodels, and the "root relative squared error (CV_{RRSE})"—which is related to the coefficient of determination R^2—or the "Bayesian estimation error quotient"; also see Gorissen et al. (2010). Bacchi et al. (2018) discusses k-fold CV for Kriging metamodels of a tsunami simulation model, including MSPE, R^2, and "residual analysis" where residuals are defined as $w_i - \widehat{y}_i$ ($i = 1, ..., n$). Kleijnen (2015, pp. 114–121) discusses LOO-CV, focussing on linear regression metamodels—but also presenting references and websites for Kriging metamodels. Law (2015) is the most popular textbook in discrete-event simulation, and covers Kriging and linear regression metamodels—but does not discuss CV.

Concerning the *analysis* of the CV results, most publications use visual inspection of a scatterplot with the simulated outputs versus the predicted outputs; e.g., Da Costa et al. (2018), Lupera Calahorrano et al. (2016) and Quirante et al. (2018) use such plots for LOC-CV and Kriging. Parnianifard et al. (2018, p. 4) "follow[s] Kleijnen (2015)", eyeballs the scatterplot, computes the "standardized residuals", and claims that these residuals should fall in the range $[-3, 3]$; however, this standardization uses the RMSPE instead of our measure (Shu et al. (2018) also uses the RMSPE, as we have already mentioned). Many publications use the RMSPE, which should be as small as possible; e.g., Forrester and Keane (2009) considers a Kriging model with a RMSPE smaller

than 2% as a reasonably good model. Ji et al. (2018) uses R^2, and seems to find $R^2 > 0.80$ acceptable. Zhang et al. (2017) uses the "correlation coefficient" (but leaves that coefficient undefined; we conjecture it is R^2) and a threshold (that is subjective?). De Carvalho et al. (2017) applies LOC-CV to Kriging and radial basis function (RBF) metamodels of a car-engine deterministic simulation model, quantifying prediction errors through R^2, the "relative average absolute error" (RAAE), and the "relative maximum absolute error" (RMAE); that article concludes: "Most of the prediction residuals were lower than 3%". The review in Bhosekar and Ierapetritou (2018) includes a table (namely, Table 4) with seven metrics for CV; however, these metrics do not include our statistic. Bastos and O'Hagan (2009) considers validation of Kriging metamodels in a Bayesian framework, but uses a training set and a validation set instead of CV; Jin and Jung (2016) uses Bastos and O'Hagan (2009) and heuristic thresholds for R^2 and RMSPE. Luminari et al. (2018, pp. 73–74) uses k-fold CV, and computes the quadratic difference between the Kriging predictor that uses all n combinations and the Kriging predictor that uses only $n-k$ combinations; next this difference is compared with the predictor that uses all n points, and a value below 6% is deemed acceptable. We point out that the preceding literature review focuses on LOO-CV on Kriging, but LOO-CV is also applied in many other modelling areas; e.g., Lin et al. (2019) uses LOO-CV in simulation via k nearest neighbors.

3 Ordinary Kriging

We present the basics of OK in Section 3.1, and CIs for the OK predictor in Section 3.2.

3.1 Basics of ordinary Kriging

We focus on OK instead of *universal Kriging* (UK). We use the notation in Kleijnen (2018), except that we focus on LOO-CV so we denote the *new* input combination to be predicted by $\mathbf{x}_i$ instead of the usual $\mathbf{x}_0$, and we use the $n-1$ *old* or *training* combinations $\mathbf{X}_{-i}$ instead of the n old combinations $\mathbf{X}$. OK assumes the following (meta)model:

$$y(\mathbf{x}) = \mu + M(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ is a combination of the d simulation inputs x_j ($j = 1, ..., d$), μ is

the constant mean $\mathrm{E}[y(\mathbf{x})]$, and $M(\mathbf{x})$ is a zero-mean stationary GP. We let $\mathbf{X}_{-i}$ denote the $(n-1) \times d$ matrix with the rows $\mathbf{x}_{i'} = (x_{i';1}, ..., x_{i';d})$ with $i \neq i'$. Furthermore, we use the symbol $\mathbf{\Sigma}_M = (\sigma_{i';i''}) = (\mathrm{Cov}(y_{i'}, y_{i''}))$ (with $i', i'' \neq i$) to denote the $(n-1) \times (n-1)$ matrix with the covariances between the metamodel's $(n-1)$ old outputs. Analogously, we let $\sigma_M(\mathbf{x}_i) = (\sigma_{i;i'}) = (\mathrm{Cov}(y_i, y_{i'}))$ denote the $(n-1)$-dimensional vector with the covariances between the new output $y_i(\mathbf{x}_i)$ and the $(n-1)$ old outputs $y_{i'}$. We let $\mathbf{1}_{n-1}$ denote the

$(n-1)$-dimensional vector with all elements equal to 1. Finally, we let $\mathbf{w}_{-i}$ denote the $(n-1)$-dimensional vector with the observed simulation outputs for the $(n-1)$ non-deleted combinations.

If and only if OK assumes a *valid* metamodel of the underlying simulation model, then we may write $y = w$. Assuming a valid metamodel, we determine the *best linear unbiased predictor* (BLUP) $\widehat{y}(\mathbf{x}_i)$ for $\mathbf{x}_i$, using the $n-1$ non-deleted I/O combinations $(\mathbf{X}_{-i}, \mathbf{w}_{-i})$. This BLUP is a weighted average of these $\mathbf{w}_{-i}$:

$$\widehat{y}(\mathbf{x}_i) = \boldsymbol{\lambda}'_{-i}\mathbf{w}_{-i}. \tag{2}$$

If these weights $\boldsymbol{\lambda}_{-i}$ are selected optimally, then the resulting BLUP has minimum variance and is unbiased. We can prove that this BLUP is

$$\widehat{y}(\mathbf{x}_i) = \mu + \sigma_M(\mathbf{x}_i)'\boldsymbol{\Sigma}_M^{-1}(\mathbf{w}_{-i} - \mu\mathbf{1}_{n-1}). \tag{3}$$

We observe that the event $\mathbf{w}_{-i} > \mu\mathbf{1}_{n-1}$ gives $\widehat{y}(\mathbf{x}_i) > \mu$. Furthermore, if we denote $\text{Var}(y)$ by τ^2, then we can prove

$$\text{MSPE}[\widehat{y}(\mathbf{x}_i)] = \tau^2 - \sigma_M(\mathbf{x}_i)'\boldsymbol{\Sigma}_M^{-1}\sigma_M(\mathbf{x}_i) + \frac{[1 - \mathbf{1}'_{n-1}\boldsymbol{\Sigma}_M^{-1}\sigma_M(\mathbf{x}_i)]^2}{\mathbf{1}'_{n-1}\boldsymbol{\Sigma}_M^{-1}\mathbf{1}_{n-1}}. \tag{4}$$

This MSPE equals $\text{Var}[\widehat{y}(\mathbf{x}_i)$ if the metamodel is valid so $\widehat{y}(\mathbf{x}_i)$ is unbiased. (Our test uses an estimator of this $\text{Var}[\widehat{y}(\mathbf{x}_i)]$; see (12).)

Sometimes, it is convenient to switch from covariances to *correlations*; i.e., sometimes we switch to the correlation matrix $\mathbf{R} = (\rho_{i';i''})$, which equals $\tau^{-2}\boldsymbol{\Sigma}_M$; analogously, $\boldsymbol{\rho}(\mathbf{x}_i) = \tau^{-2}\boldsymbol{\sigma}_M(\mathbf{x}_i)$. There are several types of correlation functions; see (e.g.) Rasmussen and Williams (2006, pp. 80–104). In simulation, however, the most popular function is the *Gaussian correlation function*. To define this function, we define the *distance* vector $\mathbf{h} = (h_j)$ where $h_j = |x_{g;j} - x_{g';j}|$ and $g, g' = 1, ..., n$, and the *correlation factors* $\boldsymbol{\theta} = (\theta_j)$; so, $\mathbf{R} = \mathbf{R}(\mathbf{h},\boldsymbol{\theta})$. These definitions imply that the Gaussian correlation function is

$$\rho(\mathbf{h},\boldsymbol{\theta}) = \prod_{j=1}^{d} \exp\ \left(-\theta_j h_j^2\right) = \exp\ (-\sum_{j=1}^{d} \theta_j h_j^2) \text{ with } \theta_j \geq 0. \tag{5}$$

We notice that this function is *separable*; such a correlation function is called *anisotropic*. If $\theta_j \downarrow 0$, then $\exp\left(-\theta_j h_j^2\right) \uparrow 1$ for any distance h_j; i.e., input j has a strong effect for any h_j. If $\theta_j \uparrow \infty$, then $\exp\left(-\theta_j h_j^2\right) \downarrow 0$, so input j has no effect. We use the symbol $\boldsymbol{\psi}$ to denote the vector with the $(2+d)$ *Kriging (hyper)parameters* $(\mu, \tau^2, \theta_1, ..., \theta_d)'$.

In practice, we must estimate the (nuisance) OK parameters $\boldsymbol{\psi}$. The most popular estimator is the *maximum likelihood estimator* (MLE), which we denote by $\widehat{\boldsymbol{\psi}} = \widehat{\boldsymbol{\psi}}(\mathbf{X}, \mathbf{w})$. This MLE is the solution of

$$\min_{\psi} \ln[\left|\tau^2\mathbf{R}\right|] + (\mathbf{w}-\mu\mathbf{1})'(\tau^2\mathbf{R})^{-1}(\mathbf{w}-\mu\mathbf{1}) \text{ with } \theta \geq \mathbf{0} \tag{6}$$

where $\mathbf{R} = \mathbf{R}(\mathbf{h},\boldsymbol{\theta})$ and $|\mathbf{R}|$ denotes the determinant of $\mathbf{R}$. This MLE implies

the following explicit formula for $\widehat{\mu}$:

$$\widehat{\mu} = (\mathbf{1}'\mathbf{R}^{-1}\mathbf{1})^{-1}\mathbf{1}'\mathbf{R}^{-1}\mathbf{w}, \tag{7}$$

which is a weighted mean of the simulation outputs $\mathbf{w}$. Furthermore, this MLE implies

$$\widehat{\tau}^2 = \frac{1}{n}(\mathbf{w} - \widehat{\mu}\mathbf{1})'\mathbf{R}^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}), \tag{8}$$

so $\widehat{\tau}^2 = \widehat{\tau}^2(\widehat{\mu}, \mathbf{R})$ is a weighted mean of the squared *residuals* $(\mathbf{w} - \widehat{\mu}\mathbf{1})$. However, this MLE does not give an explicit formula for $\widehat{\boldsymbol{\theta}}$, but requires an iterative search for $\widehat{\boldsymbol{\theta}}$. Actually, solving (6) is a mathematical challenge; e.g., different $\widehat{\boldsymbol{\psi}}$ may result from different software packages or from different starting values for the same package; see Erickson et al. (2018) and our MC experiments in Section 6.

We like to program in MATLAB, so we use the MATLAB Kriging toolbox called *DACE* that is documented in Lophaven et al. (2002) (a more recent MATLAB Kriging toolbox is "UQLab " documented in Lataniotis et al. (2015)). Furthermore, we re-estimate $\boldsymbol{\psi}$ when we leave out another I/O combination; i.e., we compute $\widehat{\boldsymbol{\psi}}_{-i} = \widehat{\boldsymbol{\psi}}(\mathbf{X}_{-i}, \mathbf{w}_{-i})$ (whereas Santner et al. (2018) does not re-estimate $\boldsymbol{\psi}$). To initialize the estimation of $\widehat{\boldsymbol{\psi}}_{-i}$, we use $\widehat{\boldsymbol{\psi}}$. Wang and Haaland (2018) also gives a (mathematically advanced) discussion of numeric errors in Kriging.

To obtain $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})$ and $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$, we *plug* the estimator $\widehat{\boldsymbol{\psi}}_{-i}$ into (3) and (4). Most publications ignore the consequences of this plugging-in; i.e., they ignore the fact that $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})$ becomes nonlinear and $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ underestimates the true Kriging variance. We shall discuss bootstrapping to obtain an unbiased variance estimator (see Section 5).

3.2 Confidence intervals for ordinary Kriging predictors

We may combine the preceding $\widehat{y}(\mathbf{x}_i)$ and $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ in a two-sided CI with nominal (prespecified) coverage $1 - \alpha$ based on the standard normal variable $z \sim \mathrm{N}(0, 1)$ so $z_{\alpha/2}$ denotes the $\alpha/2$ quantile of $\mathrm{N}(0, 1)$; i.e., we may assume

$$P[w(\mathbf{x}_i) \in \widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i}) \pm z_{\alpha/2} s[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})] = 1 - \alpha. \tag{9}$$

Actually, the *true coverage rate* may not equal $1 - \alpha$; i.e., the expected type-I error rate $\mathrm{E}(\widehat{\alpha})$ and the nominal α may differ—because of the following three issues (we shall return to these issues, after presenting our basic LOO-CV algorithm):

(i) The nonlinear plug-in predictor $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})$ is biased.

(ii) The plug-in variance estimator $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ underestimates the true $\mathrm{Var}[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$.

(iii) $|z_{\alpha/2}| < |t_{f;\alpha/2}|$—where $t_{f;\alpha/2}$ denotes the $\alpha/2$ quantile of the Student distribution with f $(< \infty)$ degrees of freedom. We assume that $t_{f;\alpha/2}$ with the proper (but unknown) f is the correct factor for a CI that uses an estimated

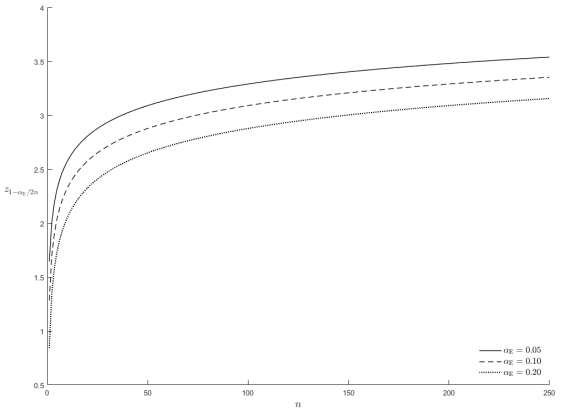

Figure 1: The quantile $z_{1-[\alpha_E/(2n)]}$ of the standard normal z as a function of the sample size n, for the experimentwise error rate $\alpha_E = 0.20$, 0.10, or 0.05

standard deviation. We might use $f = (n-1) - (d+2)$, inspired by the well-known formula for f in linear regression analysis; namely, if q denotes the number of estimated regression parameters and n the number of I/O observations, then $f = n - q$ so our OK implies $q = d+2$ (because this OK estimates θ_1, ..., θ_d, τ^2, and μ) and LOO-CV uses $n-1$ observations. (Following a Bayesian approach, Bastos and O'Hagan (2009) also use a t_f with a specific f-value.)

Whereas (9) uses the *per comparison* type-I error rate α, we wish to realize a prespecified *experimentwise* type-I error rate (say) α_E in LOO-CV; i.e., we wish that all n CIs hold "simultaneously" (or "jointly"). A problem is that as n increases, the expected number of (correlated) CIs that do not cover the corresponding true values also increases so we get more *false alarms*. We use a simple solution; i.e., we apply *Bonferroni's inequality*, which implies that α is divided by n so $z_{\alpha/2}$ in (9) becomes $z_{\alpha/(2n)}$ (or $z_{1-\alpha/(2n)}$ with $|z_{\alpha/(2n)}| = z_{1-\alpha/(2n)}$). Obviously, this replacement increases the n individual halfwidths $z_{\alpha/(2n)} s[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ ($i = 1, ..., n$); so it indeed reduces the probability of a false alarm. Unfortunately, Bonferroni's inequality is known to be *conservative* so $E(\widehat{\alpha}_E) \leq \alpha_E$. Altogether we hypothesize that $\alpha_E = \alpha/n$ gives acceptable results. We shall test this hypothesis in several MC experiments; if we reject this hypothesis, then we investigate possible explanations and solutions (see Section 6).

Note: Our use of Bonferroni's inequality implies that our validation criterion is the *maximum* (instead of the average) of the n Studentized prediction errors (see (13) below); for further discussion of various validation criteria we refer to Kleijnen (2015, p. 120) and Gorissen et al. (2010).

Usually Bonferroni's inequality is applied to only "a few" correlated statistics. In our LOO-CV, however, we have n CIs. In Fig. 1 we display $z_{1-[\alpha_E/(2n)]}$

as a function of n with $n = 1, ..., 250$, for $\alpha_{\text{E}} = 0.20, 0.10$, or 0.05. We select this range for n, because we shall present examples with $n = 10d$ where d is 2 or 8 so the highest n is 80; moreover, realistic applications of Kriging may have up to (say) 25 inputs, so $n = 250$. This Figure shows that the combination $\alpha_{\text{E}} = 0.10$ and $n = 1$ implies $z_{1-[\alpha_{\text{E}}/(2n)]} = z_{0.95} = 1.64$ (a familiar classic value); $n = 80$ (as in the borehole example) implies $z_{1-[\alpha_{\text{E}}/(2n)]} \approx z_{0.9994} = 2.50$; $n = 250$ implies $z_{1-[\alpha_{\text{E}}/(2n)]} \approx z_{0.9998} = 3.54$. So, practical d-values imply that as n increases, $z_{1-[\alpha_{\text{E}}/(2n)]}$ shows a decreasing increase. (Simes (1986) gives a (simple) variant of Bonferroni's inequality, which may make joint tests less conservative; however, this improvement may be negligible.)

If we impose the *CH condition* (to avoid extrapolation), then n becomes $n - n_{\text{CH}}$ where n_{CH} denotes the number of vertices (corner points) of the CH of $\mathbf{X}_{n\times d}$. Hence, there are fewer CIs—which implies a *lower* probability of false alarms—and a lower $\left|z_{\alpha_{\text{E}}/(2n)}\right|$ so shorter CIs—which imply a higher probability of false alarms. We shall investigate ; the *net* effect of these two effects in our various MC experiments and our two applications. In practice, we may decide *apriori* to impose the CH condition, so we determine which input combinations determine the CH; next we apply LOO-CV only to $n - n_{\text{CH}}$ I/O combinations instead of all n combinations, which saves computer time.

Note: The CI in (9) implies that the *standardized residuals* or *Studentized residuals* should have he same distribution as z (standard normal variable):

$$\frac{w(\mathbf{x}_i) - \widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})}{s[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]} = z. \tag{10}$$

Our standardization uses $s[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$, whereas other publications (e.g., Da Costa et al. (2018) and Parnianifard et al. (2018)) use

$$\widehat{\text{RMSPE}} = \sqrt{\frac{\sum_{i=1}^{n}[w(\mathbf{x}_i) - \widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]^2}{n}}. \tag{11}$$

Obviously, this RMSPE does not vary with i, whereas our $s[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ does. Furthermore, RMSPE uses the average of all n points; i.e., RMSPE estimates the square root of the *integrated MSE* (IMSE) where the integral is computed over the whole design space.

4 Latin hypercube designs for Kriging

OK gives a *global* metamodel. Most experiments that use OK to analyze simulation models use LHS to select the design matrix $\mathbf{X}_{n\times d}$—or briefly $\mathbf{X}$.

Note: Besides LHS there are alternative space-filling designs. Examples are orthogonal array, uniform, maximum entropy, minimax, maximin, integrated mean squared prediction error, and "optimal" designs; see Kleijnen (2015, p. 198). Details on LHS are given in Kleijnen (2015, pp. 198–203).

We focus on LHS with d inputs that have *uniform* (symbol U) marginal distributions in the interval $[0, 1]$, so $x_j \sim \mathrm{U}(0, 1)$. Moreover, this LHS samples n *midpoints*, which are equispaced with distance $1/n$ over the interval $[0, 1]$; so, these midpoints are $0.5/n$, $1.5/n$, ..., $1-0.5/n$. In general, LHS samples *without replacement*; so, each midpoint is sampled only once in the sample of size n. Altogether, $\mathbf{x}_j = (x_{i;j})$ (with $i = 1, ..., n$ and $j = 1, ..., d$) is a permutation of the n midpoints; i.e., $\mathbf{X}_{n \times d}$ is a stratified sample of n points from a d-dimensional grid.

Note: We claim that *sampling* of points within subintervals (instead of midpoints) has the disadvantage that it may give two values—in two neighboring subintervals—that are very close together. So, the two resulting outputs w are close together (assuming a smooth I/O function, as Kriging does) and give little new information (because these outputs have a high positive correlation). Furthermore, sampling of midpoints ensures that we obtain n realizations of x_j that are "wide apart"—which we conjecture gives better estimates of the correlation parameters θ_j in the correlation function (5). Sampling midpoints never gives $x \downarrow 0$ or $x \uparrow 1$ ($x = 0$ or $x = 1$ is impossible because x is continuous; $x = \epsilon$ or $x = 1$ - ϵ is also undesirable); we wish to avoid these two extreme values because Kriging can use the output near $x = 0$ only to predict the output at $x > 0$ (not at $x < 0$), so Kriging can use $x = 0$ only in one direction; a similar argument holds for $x \uparrow 1$.

LHS implies that projection of the n points (in the d-dimensional input space) onto the d individual axes gives n *non-collapsing* values per axis. Consequently, it is relatively easy to estimate the d individual separable correlation functions in (5). (The usual argument in favor of noncollapsing designs is that some of the d inputs may be unimportant, so—unlike factorial designs—all n input combinations are still informative for the remaining important inputs; we, however, expect that all d inputs are important in Kriging with a low d-value.)

LHS does not impose a strict mathematical relationship between the sample size n and the dimensionality d (whereas a grid with s subintervals implies $n = s^d$; e.g., a grid with 10 values per input has 10^d points). Nevertheless, if LHS uses a "small" n and a "large" d, then LHS covers $[0, 1]^d$ *sparsely*. For example, if $d = 8$ (as in the borehole example, presented in Section 7.2) and $n = 80$, then we sample only 80 gridpoints from the total of $80^8 = 1.7 \times 10^{15}$ gridpoints This sparsity implies that there are only a few old points close to the new point, so the Kriging predictor may be inadequate. For LHS in Kriging aimed at SA, Loeppky et al. (2009) gives the *rule-of-thumb* $n = 10d$, which implies $n \geq 10$ if $d \geq 1$. (If $n \geq 10$, then we expect to estimate θ_j quite accurately.) An example of a LHS design with $d = 2$ and $n = 20$ (and its CH) will be presented in Fig. 2; these $n = 20$ observations cover the experimental area relatively *sparsely*. In general, consider the original inputs z_j with $j = 1, ..., d$ (z_j should not be confused with the standard normal variable z). The volume of the original experimental area increases exponentially with d, whereas Loeppky et al. (2009) implies that n increases only linearly with d. This sparsity implies that there are only a few points among the $n - 1$ points in LOO-CV (namely, the $n - 1$ observations $(\mathbf{X}_{-i}, \mathbf{w}_{-i})$) that are *relatively close* to $\mathbf{x}_i$ (point to be

predicted). Mathematically, there are several popular distance measures; we focus on the Euclidean measure. The Euclidean distance between two points $\mathbf{x}_j$ and $\mathbf{x}'_j$ in a d-dimensional space is $[\sum_{j=1}^{d}(\mathbf{x}_j - \mathbf{x}'_j)^2]^{1/2}$, which indeed increases as d increases.

Note; There is much software for LHS. For example, Microsoft's Excel spreadsheet software has add-ins that include LHS; also see Oracle's Crystal Ball, Palisade's @Risk, and Frontline Systems' Risk Solver. LHS is also available in the MATLAB Statistics toolbox, the R package, the Open TURNS software, and Sandia's DAKOTA software. Various LHS algorithms are referenced in Kleijnen (2015, p. 200); recent algorithms are detailed in Dong and Nakayama (2017), and Le Guiban et al. (2018). Panagiotopoulos et al. (2018) uses LHS variants for RBFs—instead of Kriging—metamodels for optimization through genetic algorithms.

To implement LHS, we use MATLAB's function *lhsdesign*. This function has a parameter called "'smooth" that can be turned "off" or "on" where "off" produces points at the midpoints of the n subintervals; the default is "on". The *random* permutations in LHS may give a "bad" $\mathbf{X}$. To decide on a "good" $\mathbf{X}$, we need a criterion. We decide to use the *maximin* criterion, which maximizes the minimum Euclidean distance between the n d-dimensional points in $[0,1]^d$ (there are $n(n-1)/2$ distances; some distances may have the same value). This criterion is the default in MATLAB's lhsdesign. This criterion means that lhsdesign generates (say) M permutations, and selects the design among these M permutations that maximizes the minimum distance between any two points among the n points. We use MATLAB's default $M = 5$.

5 Leave-one-out cross-validation

We present the basic variant of LOO-CV in Section 5.1, the CH variant in Section 5.2, and the bootstrap variant in Section 5.3.

5.1 Basic variant of LOO-CV

In this section we present Algorithm 1 for LOO-CV. This algorithm adjusts the algorithm in Kleijnen (2015, pp. 116–117), which is based on Kleijnen (1983) and assumes a linear regression metamodel of a random simulation model—instead of a Kriging metamodel of a deterministic simulation model. Our algorithm starts with the *original* I/O simulation data $(\mathbf{X}_{n\times d}, \mathbf{w}_n)$—or briefly $(\mathbf{X}, \mathbf{w})$—and computes the corresponding $\widehat{\boldsymbol{\psi}}(\mathbf{X}, \mathbf{w})$—or briefly $\widehat{\boldsymbol{\psi}}$. This $\widehat{\boldsymbol{\psi}}$ is used to initialize the search for $\widehat{\boldsymbol{\psi}}(\mathbf{X}_{-i}, \mathbf{w}_{-i})$—or $\widehat{\boldsymbol{\psi}}_{-i}$ (Garbo and German (2019)—and other publications discussed in our literature review in Section 2—do not re-estimate $\boldsymbol{\psi}$, but use the original $\widehat{\boldsymbol{\psi}}$ to compute $\widehat{y}$ in LOO-CV; next they use the RMSPE). This $\widehat{\boldsymbol{\psi}}_{-i}$—together with $\mathbf{x}_i$—gives the predictor $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})$ or briefly $\widehat{y}_{-i}$. This $\widehat{y}_{-i}$—together with w_i—gives the *prediction error*

$$\mathrm{PE}_i = w_i - \widehat{y}_{-i}.$$

This PE is normally distributed because Kriging assumes a GP for $\mathbf{w}_n$; so the marginal distribution of w_i is normal, and—ignoring that $\widehat{y}_{-i}$ uses $\widehat{\boldsymbol{\psi}}_{-i}$ instead of $\boldsymbol{\psi}$—this GP implies that $\widehat{y}_{-i}$ is normally distributed. Because we have already defined SPE as the squared prediction error, we now introduce the term *prediction error standardized* (PES) for the Studentized PE (so PES is scale-independent):

$$\text{PES}_i = \frac{w_i - \widehat{y}_{-i}}{s(\widehat{y}_{-i})} \tag{12}$$

where $s(\widehat{y}_{-i}) = s[\widehat{y}(\mathbf{x}_{-i}, \widehat{\boldsymbol{\psi}}_{-i})]$. LOO-CV gives the n variables $|\text{PES}_i|$ ($i = 1, ..., n$), which are *statistically dependent* because they use common data; e.g., if one of the observed simulation outputs w_i is relatively high, then the $(n-1)$ predictions for all other points $\mathbf{x}_{i'}$ are relatively high (also see (3)). So we should not analyze $|\text{PES}_i|$ as if these n variables were *independently and identically distributed* (IID). The exact analysis is difficult, so we resort to the following simple approach: we do not reject the metamodel if *all* n PES-values are nonsignificant. This approach implies that we do reject the metamodel if one or more of the n PES-values are significant; this implies that the maximum of these n values is significant. Bonferroni's inequality implies that the nominal *experimentwise* type-I error rate α_{E} is divided by n, when we compute the significance of each of the n individual PES-values. (Various steps in the following algorithm may be executed simultaneously, so LOO-CV suits parallel computers.)

Algorithm 1

1. Read $(\mathbf{X}_{n\times d}, \mathbf{w}_n)$, and compute $\widehat{\boldsymbol{\psi}} = \widehat{\boldsymbol{\psi}}(\mathbf{X}_{n\times d}, \mathbf{w}_n)$.
2. Initialize: $i = 1$.
3. Delete $(\mathbf{x}_i, w_i)$ from $(\mathbf{X}_{n\times d}, \mathbf{w}_n)$, to obtain $(\mathbf{X}_{-i}, \mathbf{w}_{-i})$.
4. Use $(\mathbf{X}_{-i}, \mathbf{w}_{-i})$ and $\widehat{\boldsymbol{\psi}}$ to compute $\widehat{\boldsymbol{\psi}}_{-i} = \widehat{\boldsymbol{\psi}}(\mathbf{X}_{-i}, \mathbf{w}_{-i})$.
5. Compute $\widehat{y}_{-i} = \widehat{y}(\mathbf{x}_{-i}, \widehat{\boldsymbol{\psi}}_{-i})$ and $s(\widehat{y}_{-i}) = s[\widehat{y}(\mathbf{x}_{-i}, \widehat{\boldsymbol{\psi}}_{-i})]$.
6. Compute PES for combination i.
7. If $i < n$ then $i = i + 1$ and return to step 3; else go to the next step.
8. Reject the Kriging metamodel if

$$max_{1\le i\le n}\ |\text{PES}_i| > z_{1-[\alpha_{\text{E}}/(2n)]}. \tag{13}$$

If the metamodel is not valid, then $|w_i - \widehat{y}_{-i}|$ increases (by definition) so the numerator in (12) increases. Moreover, $s(\widehat{y}_{-i})$ increases because $\text{MSPE}[\widehat{y}(\mathbf{x})] = \text{Var}[\widehat{y}(\mathbf{x})] + (w_i - \widehat{y}_{-i})^2$**;** see the discussion of (4). Altogether, a non-valid metamodel implies that both the numerator and the denominator in (12) increase. The net effect on our test statistic $max_i\ |\text{PES}_i|$ is unknown so we shall use MC experiment to quantify this effect (see Section 6).

5.2 CH variant of LOO-CV

We may further adjust Kleijnen (1983), and require that the left-out point $\mathbf{x}_i$ not be a *vertex* of the CH of the n points in $\mathbf{X}$; i.e., if $\mathbf{x}_i$ does not satisfy this requirement, then we do not include $\mathbf{x}_i$ in LOO-CV. We then replace n by $n - n_{\text{CH}}$ in Bonferroni's inequality. To find this CH, we can use one of the following two methods (method (ii) is much faster if d is high so n is high, as we shall see below).

(i) The MATLAB function $\mathbf{K} = convhulln(\mathbf{X})$ returns the matrix $\mathbf{K}$ with the indices of the points in $\mathbf{X}$ that determine the *facets* of the CH of $\mathbf{X}$; this function is based on Barber et al. (1996). If this CH has p facets, then $\mathbf{K}$ is a $p \times d$ matrix. We notice that if the index of a specific point in $\mathbf{X}$ occurs in $\mathbf{K}$, then that index occurs more than once in $\mathbf{K}$. We can use the function "convhulln" to identify those points in $\mathbf{X}$ that are the same as the points corresponding with an index in $\mathbf{K}$; i.e., we can check whether the point $\mathbf{x}_i$ (to be predicted from the reduced set $\mathbf{X}_{-i}$) gives the same indices in $\mathbf{K}(\mathbf{X})$ and in $\mathbf{K}(\mathbf{X}_{-i})$ (for this check we use the MATLAB function "equal"). An example is Fig. 2 for $\mathbf{X}_{20\times 2}$ (generated by our LHS). This Figure displays $n = 20$ input combinations $\mathbf{x}_i$ (i = 1, ..., 20)—identified by the subscript i—and the CH formed by n_{CH} = 6 combinations connected by (dashed) lines (these lines are colored red in the PDF file). This Figure implies that $\mathbf{K}$ includes the index "4", so a CH point is $\mathbf{x}_4 = (1.7, 0.7)'$. Altogether, this Figure gives $n - n_{\text{CH}}$ = 20 - 6 = 14; these 14 points include the combination $\mathbf{x}_1 = (-0.5, 0.9)'$. The results of this MATLAB function agree with the results of human pattern recognition; however, such recognition fails when there are $d = 8$ inputs (also see the borehole example in Section 7.2), which we discuss now.

$\mathbf{K}(\mathbf{X}_{80\times 8})$ (with $\mathbf{X}_{80\times 8}$ generated by our LHS) shows that the CH consists of *all* $n = 80$ points $\mathbf{x}_i$ (i = 1, ..., 80). We may explain this (surprising?) finding as follows: $\mathbf{X}_{80\times 8}$ fills the high-dimensional input space only *sparsely*, so none of the 80 points is a linear combination of the remaining 79 points.

Fig. 3 shows n_{CH} (number of vertices) of $\mathbf{X}_{n\times d}$ for different combinations of n and d (instead of $n = 10d$; see again Loeppky et al. (2009)). This Figure shows that $d = 1$ implies that the CH has only two vertices; namely, $\min(x_i)$ and $\max(x_i)$ (MATLAB requires $d \geq 2$, but we do not need this function for d = 1). For $d = 2$ and increasing n the CH has more vertices so n_{CH} increases (Fig. 2 showed that $n_{\text{CH}} = 6$ for our $\mathbf{X}_{20\times 2}$; Fig. 3 does not show $n = 20$). In general, the Figure shows that the higher n is, the higher n_{CH} is. We notice that $n = 30$ and d is 5 or 6 give n_{CH} that does not increase, but equals 29 and 28, respectively; this is caused by the randomness of LHS.

(ii) We formulate the following *linear programming* (LP) problem where the coefficients of the objective function $f_{i'}$ ($i' \neq i$) are irrelevant (we select $f_{i'} = 1$), because the question is whether this problem has a feasible solution:

$$\min_{a_{i'}} \sum_{i \neq i'} f_{i'} a_{i'} \text{ such that } \sum_{i' \neq i} a_{i'} \mathbf{x}_{i'} = \mathbf{x}_i, \sum_{i' \neq i} a_{i'} = 1, a_{i'} \geq 0 \text{ with } i' \neq i. \quad (14)$$

To solve this problem, we may use the MATLAB function *linprog*. If this

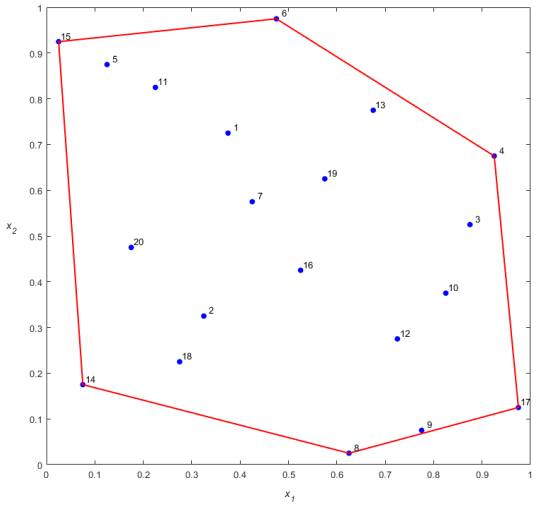

Figure 2: LHS design with $n = 20$ and $d = 2$; CH denoted by (dashed) lines

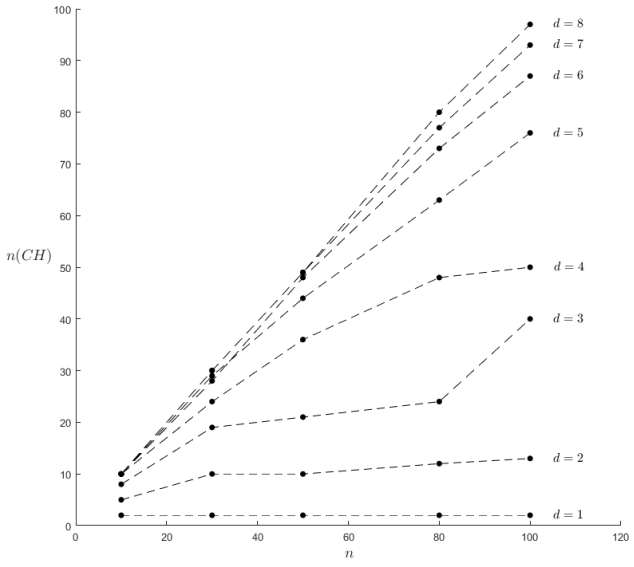

Figure 3: n_{CH} versus n for various d in $\mathbf{X}_{n\times d}$

n	10		30		50	
method	CH	LP	CH	LP	CH	LP
$d = 1$	N/A	0.17	N/A	1.83	N/A	0.79
$d = 2$	0.01	0.17	0.01	0.46	0.03	0.75
$d = 3$	0.01	0.12	0.01	0.45	0.03	0.70
$d = 4$	0.01	0.12	0.02	0.39	0.04	0.63
$d = 5$	0.01	0.11	0.03	0.39	0.09	0.61
$d = 6$	0.01	0.11	0.06	0.37	0.36	0.67
$d = 7$	0.01	0.11	0.17	0.37	1.02	0.60
$d = 8$	0.01	0.11	0.45	0.35	4.34	0.56

n	80		100		500	
method	CH	LP	CH	LP	CH	LP
$d = 1$	NA	0.95	NA	1.75	NA	10.00
$d = 2$	0.03	1.29	0.03	1.57	0.21	9.54
$d = 3$	0.02	1.30	0.04	1.50	0.31	9.72
$d = 4$	0.10	1.13	0.08	1.43	1.12	9.45
$d = 5$	0.25	1.07	0.40	1.29	7.74	8.87
$d = 6$	1.21	1.05	1.74	1.29	72.47	8.45
$d = 7$	5.28	0.96	8.81	1.20	643.98	8.12
$d = 8$	21.67	0.96	63.29	1.25	5616.84	7.86

Table 1: CPU times for convhulln method versus LP method, for various (d, n) combinations

function finds no feasible point, then we exclude $\mathbf{x}_i$ from LOO-CV with the CH constraint.

Sub (i) and (ii): Table 1 shows the CPU times (in seconds) needed by method (i) and method (ii), for various combinations of d and n. If we apply Loeppky et al.'s rule ($n \geq 10d$), then not all these combinations are practically relevant. Interesting combinations are $d = 8$ and $n = 80$ (which we shall use in the borehole example, which gives 21.67 seconds for the convexhulln method and 0.96 seconds for the LP method (we conjecture that *convhulln* is so slow because it uses the Delaunay triangularization.) We conclude that the LP method is much faster if d is not very small; else both methods are so fast that their difference is practically irrelevant.

5.3 Bootstrap variant of LOO-CV

Whether we do or do not impose the CH condition, we know that $s(\widehat{y}_{-i})$ (denominator of PES_i) underestimates the true Kriging variance; see the many references in Kleijnen (2015, p. 189). Whereas in some applications (e.g., sequential sampling for estimating the optimum combination) it suffices to estimate the *relative* magnitude of $s(\widehat{y}_i)$ for different $\mathbf{x}_i$, in LOO-CV we may need

to correct the biased value of $s(\widehat{y}_{-i})$ in order to obtain a valid CI. To obtain such an unbiased estimator, we apply *parametric bootstrapping*. Several variants of this bootstrapping are given in Kleijnen (2015, pp. 191-197).

Note: Den Hertog et al. (2006) details three variants of this bootstrapping. However, our next algorithm gives an adaptation of the first variant only, because only this variant is relevant in LOO-CV. Moreover, our adaptation is useful, because we know that LOO-CV uses $n-1$ "old" combinations and 1 "new" combination. Kleijnen (2015, pp. 194–197) discusses a bootstrap variant called "conditional simulation" (CS), which may also be implemented through the R software package called "DiceKriging"; see Roustant et al. (2012). However, Mehdad and Kleijnen (2015, p. 1808) proves that CS gives an estimate that is smaller (albeit not significantly smaller) than the bootstrap estimate. We prefer the bigger variance estimate, because it makes |PES| (which is the ratio of |PE| and $s(\widehat{y}_{-i})$) smaller, so it decreases the type-I error rate.

We point out that MSPE $[\widehat{y}(\mathbf{x}_i)]$ (defined in (4)) implies that if $\widehat{y}(\mathbf{x}_i)$ is biased (because the Kriging metamodel is only an approximation of the underlying simulation model with output $w(\mathbf{x}_i)$), then $\text{MSPE}[\widehat{y}(\mathbf{x}_i)] > \text{Var}[\widehat{y}(\mathbf{x}_i)]$. We call this *metamodel bias*. Moreover, the classic plug-in estimator $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ (defined above (9)) has bias, because this estimator ignores the fact that $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})$ is a nonlinear estimator. We call this *plug-in bias*. Den Hertog et al. (2006, pp. 405–408) compares $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ with the bootstrapped estimator, in several examples. We observe that in each example $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ is biased—not only because of plug-in bias (as Den Hertog et al. mentions)—but also because of metamodel bias (the examples are not GPs but they use GPs to approximate given deterministic functions). More specifically, one example (namely, Fig. 9 in Den Hertog et al.) shows that the value of the maximal classic estimate is roughly 2.8, whereas the corresponding maximal bootstrapped value is 6.5 (obviously, the minimum estimates are zero, at "old" input combinations). Actually our LOO-CV uses the standard deviation instead of the variance, so the difference between the classic and the bootstrapped estimates is smaller. We shall give more numerical results in Section 6 on our MC experiments.

In Algorithm 2 we use the notation that is also used in the bootstrap literature; namely, the bootstrap observations are denoted by the superscript $*$, and the bootstrap sample size by B (a classic value for B is 100).

Algorithm 2

1. Read sample size B, simulation I/O data $(\mathbf{X}, \mathbf{w})$, estimated Kriging parameters $\widehat{\boldsymbol{\psi}} = (\widehat{\mu}, \widehat{\tau}^2, \widehat{\boldsymbol{\theta}}')'$, and row index of I/O data to be deleted i (= 1, ..., n).

2. Initialize bootstrap sample: $b = 1$

3. Using $\mathbf{X}$ and $\widehat{\boldsymbol{\psi}}$ (of step 1), sample "old" and "new" bootstrap outputs $\mathbf{w}_b^* = \mathbf{w}_b^*(\mathbf{X}, \widehat{\boldsymbol{\psi}})$ from $\text{N}_n(\widehat{\mu}\mathbf{1}_n, \widehat{\boldsymbol{\Sigma}})$ with $\widehat{\boldsymbol{\Sigma}} = \widehat{\tau}^2\mathbf{R}(\mathbf{X}, \widehat{\boldsymbol{\theta}})$.

4. Using $\mathbf{w}^*_{-i;b}$ (with i of step 1 and $\mathbf{w}^*_b$ of step 3) and $\mathbf{X}_{-i}$ (of step 1), compute the bootstrapped MLE $\widehat{\boldsymbol{\psi}}^*_{-i;b}= (\widehat{\mu}_{-i;b}, \widehat{\tau}^2_{-i;b}, \widehat{\boldsymbol{\theta}}'_{-i;b})'.'$

5. Using $\mathbf{w}^*_{-i;b}$ and $\widehat{\boldsymbol{\psi}}^*_{-i;b}$ (of step 4), compute $\widehat{y}^*_{-i} = \widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}^*_{-i;b})$, which is the bootstrapped predictor for the deleted combination i that uses the general predictor formula (3) and the definitions $\widehat{\boldsymbol{\sigma}}^*_{-i;b} = \widehat{\boldsymbol{\sigma}}_M(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}^*_{-i;b})$, and $\widehat{\boldsymbol{\Sigma}}^{*-1}_{-i;b} = \widehat{\boldsymbol{\Sigma}}^{-1}_{-i}(\widehat{\boldsymbol{\psi}}^*_{-i;b})$:

$$\widehat{y}^*_{-i;b} = \widehat{\mu}^*_{-i;b} + \widehat{\boldsymbol{\sigma}}^{*\prime}_{-i;b} \widehat{\boldsymbol{\Sigma}}^{*-1}_{-i;b}(\mathbf{w}^*_{-i;b} - \widehat{\mu}^*_{-i;b}\mathbf{1}_{n-1}). \tag{15}$$

6. Using $\widehat{y}^*_{-i;b}$ (of step 5) and $w^*_{i;b}$ (of step 3), compute the bootstrap estimator of the *squared prediction error* (SPE):

$$\text{SPE}^*_{-i;b} = (\widehat{y}^*_{-i;b} - w^*_{i;b})^2.$$

7. If $b < B$ then $b = b + 1$ and return to step 3; else go to the next step.

8. Using $\text{SPE}^*_{-i;b}$ (of step 6) with $b = 1$, ..., B (see step 7), compute the bootstrap estimator of MSPE:

$$\text{MSPE}^*_{-i} = \frac{\sum_{b=1}^{B} \text{SPE}^*_{-i;b}}{B}. \tag{16}$$

   If we ignore the bias of the predictor $\widehat{y}^*_{-i}$, then $\left(\text{MSPE}^*_{-i}\right)^{1/2}$ equals $\widehat{\sigma}(\widehat{y}^*_{-i})$ which is the bootstrap estimator of $\sigma[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}})]$.

If $\widehat{\sigma}(\widehat{y}^*_{-i})$ makes $max_{1\leq i\leq n}$ $|\text{PES}_i|$ in (13) nonsignificant, then another combination may give a significant $max_{1\leq i\leq n}$ |PES|, so this combination needs bootstrapping. To limit the required computer time, we might apply bootstrapping only to those combinations that give a *significant* |PES|. The remaining combinations will remain nonsignificant if we replace $s(\widehat{y}_{-i})$ in (12) by $\widehat{\sigma}(\widehat{y}^*_{-i})$ and—as expected—$s(\widehat{y}_{-i}) < \widehat{\sigma}(\widehat{y}^*_{-i})$.

6 Monte Carlo experiments

We use MC experiments to estimate the α error rate of our test in Section 6.1. and the β error rate or power function of this test in Section 6.2.

6.1 Estimating the α error rate in MC experiments

In the Introduction (Section 1) we mentioned that (in general) the type-I error rate with prespecified value α is defined as $\text{P}(H_0 \text{ rejected}|H_0)$ where the *null-hypothesis* H_0 is rejected if the test statistic exceeds its critical value that is determined by α (also see (13)). In Section 3 we specified that OK assumes

the metamodel defined in (1). This (1) implies an *n-variate normal distribution* N_n, so our specific H_0 is

$$H_0 : \mathbf{w}(\mathbf{X}_{n\times d}) \sim \mathrm{N}_n(\mu \mathbf{1}_n, \mathbf{\Sigma}_M(\mathbf{X}_{n\times d})). \tag{17}$$

To sample from multivariate normal distributions—such as N_n defined in (17)—we use the MATLAB function *mvnrnd*. Regarding the $n \times n$ matrix $\mathbf{\Sigma}_M(\mathbf{X}_{n\times d})$ in (17), we assumed a Gaussian correlation function (with parameters $\boldsymbol{\theta}$) so $\mathbf{\Sigma}_M(\mathbf{X}_{n\times d}) = \tau^2 \mathbf{R}(\boldsymbol{\theta}, \mathbf{X}_{n\times d})$ where $\mathbf{R}(\boldsymbol{\theta}, \mathbf{X}_{n\times d})$ has all main-diagonal components equal to 1, and components above this diagonal that decrease as $|x_{i;j} - x_{i';j}|$ with $i' > i$ increases ($j = 1, ..., d$); $\boldsymbol{\theta}$ determines the rate of this decrease. Altogether, (17) implies $(2 + d)$ parameters, collected in $\boldsymbol{\psi} = (\mu, \tau^2, \theta_1, ..., \theta_d)'$.

Note: To implement this sampling in MATLAB, we use output of DACE. This output includes the Cholesky triangular matrixes for the estimated $\mathbf{R}$. Furthermore, DACE gives the estimated τ^2. Finally, DACE gives the estimated μ for the normalized output and inputs. For details we refer to Lophaven et al. (2002, p. 14). We shall return to the effects of this normalization, in Appendix 4.

To select *specific* values for this $\boldsymbol{\psi}$, we use the estimated Kriging parameters ($\widehat{\boldsymbol{\psi}}_v$) that we shall compute for the example with $d = 2$ in Gramacy (2016); we detail this example in Section 7.1. This $\boldsymbol{\psi}$ combined with the input $\mathbf{X}_{n\times d}$ gives the I/O data $(\mathbf{X}_{n\times d}, \mathbf{w}_n)$ where $\mathbf{w}$ is defined in (17). To this $(\mathbf{X}, \mathbf{w})$ we fit an OK metamodel. For this fitting, we use the DACE software, which gives $\widehat{\boldsymbol{\psi}}_w$ or briefly $\widehat{\boldsymbol{\psi}}$. (We use the MATLAB function *full* to fill $\mathbf{R}$ from the sparse (lower-triangular) matrix in the Cholesky decomposition.)

Note; We have already observed that LHS guarantees that $\mathbf{X}_{n\times d}$ is *non-collapsing*; i.e., the projection of a point $\mathbf{x}_i$ in the d-dimensional input space onto one of the d axes gives n values that are equidistant midpoints (see again Section 4). However, the distance between two points $\mathbf{x}_i$ and $\mathbf{x}_{i'}$ depends on the specific realization of the LHS design. We limit our MC experiments to the realization of $\mathbf{X}_{n\times d}$ that is the best among $M = 5$ realizations; see Section 4.

In our MC experiments we sample (say) m times from N_n defined in (17) with specific $\boldsymbol{\psi} = (\mu, \tau^2, \boldsymbol{\theta}')'$. The m observations from N_n are IID. These observations are called *macroreplications* or (briefly) *replication*s. These replications are IID, because they use nonoverlapping *pseudo-random number* (PRN) streams. (We focus on deterministic simulation; in random simulation we would distinguish between macroreplications and replications, because each input combination may use multiple replications.)

These m observations on $\widehat{\boldsymbol{\psi}}_r$ ($r = 1, ..., m$) enable us to make *boxplots with whiskers* (or briefly "boxplots") that summarize the marginal distributions of the $2 + d$ components of $\widehat{\boldsymbol{\psi}}$. Actually, we use the MATLAB function *boxplot*, which creates a boxplot where the central mark indicates the median, the bottom and top edges indicate the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and the outliers use the '+' symbol; e.g., the MC experiment with $d = 2$ gives Fig. 4. This Figure is based on $\mathbf{X}_{20\times 2}$ generated by LHS (see Section 4). The exact coordinates of the $n = 20$

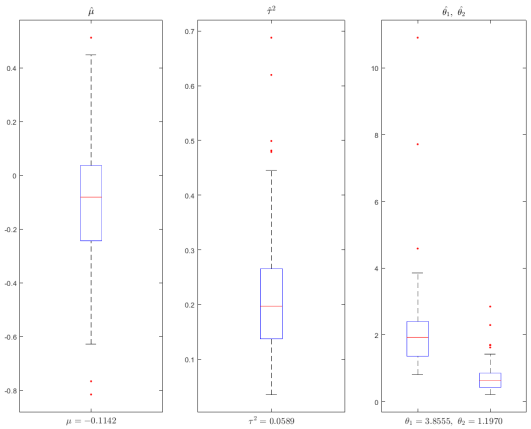

Figure 4: Boxplots for $\widehat{\mu}_r$, $\widehat{\tau}_r^2$, $\widehat{\theta}_{1;r}$ and $\widehat{\theta}_{2;r}$ with $r = 1, ..., 100$ in MC experiment with input $\mathbf{X}_{20\times 2}$ and constant mean output μ

points are given in Appendix 1. This gives the boxplots for $\widehat{\mu}_r$, $\widehat{\tau}_r^2$, $\widehat{\theta}_{1;r}$, and $\widehat{\theta}_{2;r}$ computed through (7), (8), and (6) with $\mathbf{w}$ replaced by $\mathbf{w}_r$. This Figure should give $m = 100$ estimates that do not deviate importantly from the true values μ = -0.1142, τ^2 = 0.0589, θ_1 = 3.8555, and θ_2 = 1.1970. We may check this Figure visually. This inspection shows that the sample median of $\widehat{\mu}_r$ lies close to the true mean μ = -0.1142. However, the sample median of $\widehat{\tau}_r^2$ is approximately 0.2, whereas $\tau^2 = 0.0589$. The sample median of $\widehat{\theta}_{1;r} \approx 2$, whereas $\theta_1 = 3.8555$; the sample median of $\widehat{\theta}_{2;r} \approx 0.5$, whereas $\theta_2 = 1.1970$. This inspection confirms that it is indeed difficult to estimate the Kriging parameters—as Erickson et al. (2018) observes when comparing software for computing the Kriging predictor and its estimated variance, which depend on $\widehat{\boldsymbol{\psi}}$, besides $\mathbf{X}_{20\times 2}$ and $\mathbf{w}_{20}$ (see again our text below (6)).

Our basic variant of LOO-CV implies that we predict n new points, using $\widehat{\boldsymbol{\psi}}_{-i;r}$ ($i = 1, ..., n$). Upon projection of $\mathbf{x}_i$ onto the d individual axes, 50% of these new points lies in the interval $0.0 < x_j < 0.5$ ($j = 1, ..., d$) because we use the type of LHS defined in Section 4; the other 50% lies in $0.5 < x_j < 1.0$. In Fig. 5 we shall show PES_i (defined in (12)) for only two of the $m = 100$ replications; namely, the first replication that gives a *nonsignificant* value for max_i $|\text{PES}_i|$ (defined in (13), and the first replication that gives a *significant* value.

Finally, we use all m replications to estimate α (type-I error-rate). Therefore we define the *binary* variable b_r (with $r = 1, ..., m$) such that if—in replication r— max_i $|\text{PES}_i|$ is significantly high, then $b_r = 1$; else $b_r = 0$:

$$b_r = 1 \text{ if } max_{1\leq i\leq n} \left|\frac{w_{i;r} - \widehat{y}_{-i;r}}{s(\widehat{y}_{-i;r})}\right| > z_{1-[\alpha_{\text{E}}/(2n)]}; \text{ else } b_r = 0. \qquad (18)$$

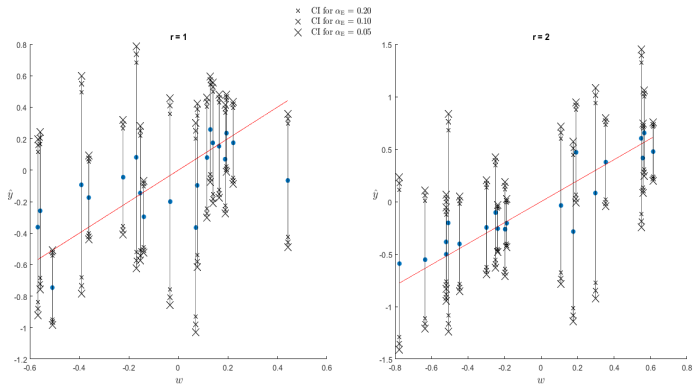

Figure 5: Scatterplot for $(w_i, \widehat{y}_i)$ augmented with $\widehat{y}_i \pm z_{\alpha_E/(2n)} s(\widehat{y}_{-i})$ and $\widehat{y}_i = w_i$ in MC experiment with $\alpha_E = 0.20$, 0.10, 0.05, $n = 20$, $d = 2$, and constant mean output

Hence the unbiased estimator of α is

$$\widehat{\alpha} = \overline{b} = \frac{\sum_{r=1}^{m} b_r}{m}. \tag{19}$$

To obtain this estimate $\widehat{\alpha}$, we sample $\mathbf{w}$ from the multivariate normal distribution with parameter vector $\boldsymbol{\psi}$; see (17). To select specific values for the components of this $\boldsymbol{\psi}$., we use the MLE $\widehat{\boldsymbol{\psi}} = (\widehat{\mu}, \widehat{\tau}^2, \widehat{\theta}_1, \widehat{\theta}_2)' = (-0.1142, 0.0589, 3.8555, 1.1970)'$ that we find when we fit an OK metamodel to the I/O data of Gramacy (2016)'s example (which we shall detail in Section 7.1). We obtain $m = 100$ replications, which gives $(\mathbf{X}_{20\times 2}, \mathbf{w}_{20;r})$ with $r = 1$, ..., 100. For each replication we use DACE to obtain $\widehat{\boldsymbol{\psi}}_r = (\widehat{\mu}_r, \widehat{\tau}_r^2, \widehat{\theta}_{1;r}, \widehat{\theta}_{2;r})'$. After preliminary experimentation, we let DACE search for $\widehat{\theta}_{1;r}$ and $\widehat{\theta}_{2;r}$ in the range $[0.01, 50.00]$. The resulting $\widehat{\boldsymbol{\psi}}_r$ have already been displayed in Fig. 4. This $\widehat{\boldsymbol{\psi}}_r$ gives the OK predictor $\widehat{y}_r$ and its standard deviation $s(\widehat{y}_r)$. Our LOO-CV gives $\text{PES}_{i;r}$ $(= (w_{i;r} - \widehat{y}_{-i;r})/s(\widehat{y}_{-i;r}))$ and the validation statistic $\max|\text{PES}_{i;r}|$, which defines b_r in (18) and gives $\widehat{\alpha} = \overline{b}$ defined in (19). This LOO-CV gives the following specific results for our MC experiment.

LOO-CV with $n = 20$ uses the quantile $z_{1-[\alpha_E/(2n)]} = z_{1-\alpha_E/40}$ for three $\alpha_E$values. For $\alpha_E = 0.20$ this quantile is $z_{0.995} = 2.58$; likewise, $\alpha_E = 0.10$ gives $z_{0.9975} = 2.81$, and $\alpha_E = 0.05$ gives $z_{0.99875} = 3.02$ (these increasing values are also implied by Fig. 1) We do not present $m = 100$ tables, each with $n = 20$ rows (for $i = 1$, ..., n) and four columns (for w, $\widehat{y}$, $s(\widehat{y})$, and |PES|). Instead, we present Fig. 5, which displays two *augmented scatterplots* with the $n = 20$ pairs $(w_i, \widehat{y}_i)$ $(i = 1, ..., n)$, the n CIs $\widehat{y}_i \pm z_{\alpha_E/(2n)} s(\widehat{y}_{-i})$ for our three α_E-values, and the (45°) line $\widehat{y}_i = w$. These two plots correspond with two of the $m = 100$ replications; namely, the first replication with a significant result and the first replication with a nonsignificant result, for any of our three $\alpha_E$values, respectively. Obviously, a replication gives a nonsignificant result if *all* its n

CIs intersect the 45° line; else the replication is significant. These definitions are equivalent to the definitions that use max_i $|\text{PES}_i|$. In replication $r = 2$ all 20 CIs cover w_i, even for the shortest CI (with $\alpha_{\text{E}} = 0.20$). In replication $r = 1$ even the longest CI (using $\alpha_{\text{E}} = 0.05$) for the highest w_i does not cover the true output.

Furthermore, the n pairs $(w_i, \widehat{y}_i)$ in Fig. 5 are also used in the *classic scatterplot*, as follows. These $(w_i, \widehat{y}_i)$ determine the line (say) $\widehat{y} = a + bw$ with a and b determined through the *least squares* (LS) criterion (or L_2-norm). This line gives R^2. Actually, Fig. 5 implies $R^2 = 0.43$ for $r = 1$ and $R^2 = 0.22$ for $r = 2$. Intuitively we reject very low R^2-values. However, in this MC experiment the OK model is valid, because we sample the I/O data from (17) (this validity is confirmed by our LOO-CV statistic $\max |\text{PES}_i|$). In general, scatterplots and R^2-values are mathematical instead of statistical plots and measures, so they can also be used for alternative models such as NNs, RBFs, and splines—but they lack no statistical critical values.

Next we proceed from these two replications to all $m = 100$ replications, and compute $\widehat{\alpha}$ (defined in (19)). The value $\alpha_{\text{E}} = 0.20$ gives $\widehat{\alpha} = 45/100 = 0.45$; $\alpha_{\text{E}} = 0.10$ gives $\widehat{\alpha} = 0.32$, and $\alpha_{\text{E}} = 0.05$ gives $\widehat{\alpha} = 0.23$. Obviously, these $\widehat{\alpha}_{\text{E}}$-values are too high. To solve this problem, we try various solutions.

We may impose the *CH condition*. When we impose this condition, we do not have to run the MC experiment again, but we apply LOO-CV to only $n - n_{\text{CH}}$ of the n combinations. To identify the n_{CH} vertices of the CH, we apply the MATLAB function *convhulln*($\mathbf{X}$) to our specific $\mathbf{X}_{20\times 2}$. This function has already given Fig. 2, which implies $n - n_{\text{CH}} = 20$ - 6 = 14. We apply LOO-CV to these 14 combinations, using $\alpha_{\text{E}}/[2(n - n_{\text{CH}})] = \alpha_{\text{E}}/28$ (instead of $\alpha_{\text{E}}/(2n) = \alpha_{\text{E}}/40$). Hence, $\alpha_{\text{E}} = 0.20$ implies $\alpha_{\text{E}}/28 \approx 0.007$ (instead of $\alpha_{\text{E}}/40 = 0.005$), so $z_{1-\alpha_{\text{E}}/(2n_{\text{CH}})} \approx z_{0.993} = 2.45$ (instead of $z_{1-\alpha_{\text{E}}/(2n)} \approx z_{0.995} = 2.58$). Likewise, $\alpha_{\text{E}} = 0.10$ implies $\alpha_{\text{E}}/28 \approx 0.0036$ (instead of $\alpha_{\text{E}}/40 = 0.0025$), so $z_{1-\alpha_{\text{E}}/(2n_{\text{CH}})} \approx z_{0.9964} = 2.691$ (instead of $z_{0.9975} = 2.81$); $\alpha_{\text{E}} = 0.05$ implies $\alpha_{\text{E}}/28 \approx 0.0018$ (instead of $\alpha_{\text{E}}/40 = 0.0013$), so $z_{1-\alpha_{\text{E}}/(2n_{\text{CH}})} \approx z_{0.9982} = 2.91$ (instead of $z_{0.99875} = 3.02$). Altogether, the CH-condition implies fewer CIs that are shorter. Moreover, we select those $n - n_{\text{CH}}$ combinations that we conjecture to be relatively easy to predict (because these combinations avoid extrapolation). We do use all $n-1$ combinations to estimate the Kriging model for the combination that is left out in our LOO-CV with the CH-condition.) Imposing the CH-condition, we obtain the following results: $\alpha_{\text{E}} = 0.20$ gives $\widehat{\alpha} = 38/100 = 0.38$ (without the CH-condition $\widehat{\alpha}$ was 0.45); $\alpha_{\text{E}} = 0.10$ gives $\widehat{\alpha} = 0.26$ (was 0.32), and $\alpha_{\text{E}} = 0.05$ gives $\widehat{\alpha} = 0.17$ (was 0.23. We conclude that in this example the CH requirement decreases $\widehat{\alpha}$—for all three nominal rates α_{E}—but $\widehat{\alpha}$ is still significantly high.

Because the CH-constraint does not give acceptable $\widehat{\alpha}$-values, we investigate the three issues listed below (9).

(i) The MC experiment with H_0 defined in (17) enables us to estimate $E(\widehat{y}_{-i;r}|\mathbf{w}_{-i;r}) - E(w_{i;r}|\mathbf{w}_{-i;r})$, which is the *bias* of the (nonlinear plug-in predictor) $\widehat{y}_{-i;r} = \widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i;r})$ with $\widehat{\boldsymbol{\psi}}_{-i;r}$ estimated from $(\mathbf{X}_{-i}, \mathbf{w}_{-i;r})$. We empha-

Point	Inside CH	On CH
i	$t_{m-1}^{(i)}$	$t_{m-1}^{(i)}$
2	0.84	
3	1.35	
4		2.31
5	0.86	
6		1.58
10	1.12	

Table 2: Significant bias for points inside the CH and on the CH, in MC experiment with two inputs and constant mean output

size that the definition of this bias is slightly complicated because $E(w_{i;r}|\mathbf{w}_{-i;r})$ also varies with r (and i) (whereas $E(w_{i;r}) = E(w_i) = \mu$) where $i = 1, ..., n$ and $r = 1, ..., m$. We define the (optimistic) null-hypothesis of no bias:

$$H_0^{(\text{PE})} : E(\text{PE}_{i;r}) = 0 \text{ with } \text{PE}_{i;r} = \widehat{y}_{-i;r} - w_{i;r} \tag{20}$$

where $\text{PE}_{i;r}$ are m IID variables for combination i. To estimate the magnitude of the bias, we compute the average of these m variables: $\overline{\text{PE}}_i = \sum_{r=1}^{m} \text{PE}_{i;r}/m$. To test $H_0^{(\text{PE})}$ (defined in (20)), we compute $t_{m-1}^{(i)} = \overline{\text{PE}}_i / s(\overline{\text{PE}}_i)$ with $s(\overline{\text{PE}}_i)$ $= \sqrt{\sum_{r=1}^{m} (\text{PE}_{i;r} - \overline{\text{PE}}_i)^2/[(m-1)m]}$. Because we have n combinations, we may (again) apply Bonferroni's inequality with experimentwise error rate (say) α:

$$\text{Reject } H_0^{(\text{PE})} \text{ if } \max_i |t_{m-1}^{(i)}| > t_{m-1;1-\alpha/(2n)}. \tag{21}$$

Furthermore we can investigate whether—the absolute value of—the bias is higher for the n_{CH} combinations of the CH. We then apply the test in (21) to the n_{CH} combinations of the CH and the $n - n_{\text{CH}}$ combinations inside this CH, respectively.

Our MC experiment with H_0 defined in (17) gives Table 2, which shows those combinations i that give $\left|t_{m-1}^{(i)}\right| > t_{m-1;1-\alpha/(2n)}$ where $t_{m-1;1-\alpha/(2n)} = 0.84$ for $\alpha = 0.20$, 0.10, or 0.05 and the 14 combinations inside the CH and $t_{m-1;1-\alpha/(2n)}$ $= 0.81$ for the 6 combinations of the CH and $\alpha = 0.20$ and $t_{m-1;1-\alpha/(2n)} = 0.82$ for $\alpha = 0.10$ or 0.05 (these 14 + 6 combinations have already been displayed in Fig. 2). This table suggests that some combinations give significant bias. More specifically, the highest estimated bias occurs for 2 of the 6 combinations in the CH; namely, the combinations 4 and 6. However, (less) significant bias also occurs for 4 of the 14 combinations inside the CH; namely, the combinations 2, 3, 5, and 10. Nevertheless, this bias is so small that we decide to ignore this bias—as is usual in OK.

(ii) A simple solution is to replace z in (13) by t_f with $f = (n-1) - (d+2)$. This replacement indeed decreases $\widehat{\alpha}$, but only slightly; i.e., if we impose the CH

requirement (so $n - n_{\rm CH} = 14$) and $\alpha_{\rm E}$ is 0.20, 0.10, or 0.05, then $\widehat{\alpha}$ is 0.35, 0.25, 0.15 (whereas z gives 0.38, 0.26, 0.17). So we conclude that this replacement does not give an acceptable solution.

(iii) A computationally more demanding solution uses bootstrapping—through Algorithm 2—to obtain an unbiased estimator of $\sigma^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$. This bootstrapping gives the following results: $\alpha_{\rm E} = 0.20$ gives $\widehat{\alpha}_{\rm E} = 10/100 = 0.10$ (was $\widehat{\alpha}_{\rm E} = 0.45$), $\alpha_{\rm E} = 0.10$ gives $\widehat{\alpha}_{\rm E} = 0.05$ (was $\widehat{\alpha}_{\rm E} = 0.32$), and $\alpha_{\rm E} = 0.05$ gives $\widehat{\alpha}_{\rm E} = 0.04$ (was $\widehat{\alpha}_{\rm E} = 0.23$). Because these $\widehat{\alpha}_{\rm E}$-values are too low, we may replace t_f by z; however, this replacement gives negligible increases of the $\widehat{\alpha}_{\rm E}$-values. Altogether, our LOO-CV (which uses Bonferroni's inequality) combined with bootstrapping is *conservative.*

6.2 Estimating the power function in MC experiments

Whereas there is a single null-hypothesis H_0 defined in (17), there are infinitely many *alternative* hypotheses H_1. Actually, we choose to replace $\mu\mathbf{1}_n$ in H_0 by a first-order polynomial with coefficients $\boldsymbol{\beta}_d = (\beta_1, ..., \beta_d)'$ and zero intercept, which gives

$$H_1 : \mathbf{w}(\mathbf{X}_{n\times d}) \sim \mathrm{N}(\mathbf{X}_{n\times d}\boldsymbol{\beta}_d, \boldsymbol{\Sigma}_M(\mathbf{X}_{n\times d})). \tag{22}$$

We use $\boldsymbol{\beta}_d$ to control the magnitude of the *type-II* (or β) error rate, which we denote by $\beta_{\rm II}$ (the subscript II distinguishes this rate from the parameters β_j (j = 1, ..., d) in the polynomial in (22)). Then $1 - \beta_{\rm II}$ is the *power* of the test (the test's alarms are true instead of false). We define $\gamma = 1 - \beta_{\rm II}$, so

$$\gamma = P(H_0 \text{ rejected}|H_1). \tag{23}$$

For simplicity's sake we assign the same positive value β to each β_j, so $\beta_j = \beta \geq 0$ and $\boldsymbol{\beta}_d = \beta\mathbf{1}_d$. Next we experiment with several values for β. The smallest β-value is zero; obviously, if $\beta = 0$, then $\gamma = \alpha$. Besides $\beta = 0$ we select more values for β. Actually, we obtain MC results for various values for the *signal-to-noise* ratio (say) g (so $\beta = g\tau$); namely, g = 0, 1, 5, and 25. This experiment estimates the *power curve* $\gamma(\beta)$.

Note: To estimate $\gamma(\beta)$, we again use b_r (defined in (18)). So the right-hand side of (19) gives the estimator $\widehat{\gamma}(\beta)$.. To improve the statistical accuracy of our comparisons among $\widehat{\gamma}(\beta)$ for the various values of g in $\beta = g\tau$, we use *common random numbers* (CRN). To implement CRN, we might use the following two options: (i) We sample m times from N_n with zero means; (so, $\mu = 0$ in (17)); we store the results in an $n \times m$ table; to estimate γ, we add $\mathbf{x}_i'\boldsymbol{\beta}$ to each element in this table. (ii) Instead of creating such a table, we initialize the PRN stream with the same *seed* for each value of β (so i =1 and $r = 1$ use the same seed for each β value). Actually, we select option (i).

We continue to use OK (which erroneously assumes $E(\mathbf{w}) = \mu\mathbf{1}$, whereas UK might correctly assume $E(\mathbf{w}) = \mathbf{X}\boldsymbol{\beta}$). We expect that the MLE of the OK parameters is biased: $E(\widehat{\boldsymbol{\psi}}) \neq \boldsymbol{\psi}$. More specifically, we use LHS, so $E(\mathbf{x}_i')\boldsymbol{\beta} = (0.5d)\boldsymbol{\beta}$, which gives $E(\widehat{\mu}) = 0.5d\beta$. Furthermore, the output w varies not only because of τ^2, but also because $E(w)$ is not a constant but varies with $\mathbf{x}$;

g	$\widehat{\mu}_r$	$\widehat{\tau}_r^2$	$\widehat{\theta}_{1;r}$	$\widehat{\theta}_{2;r}$
0	-0.0060	0.1972	1.9278	0.6388
1	0.0008	0.3251	1.6210	0.4232
5	0.1155	5.9077	0.9639	0.1058
25	1.3144	230.3007	0.4819	0.0529

Table 3: Sample medians of estimated OK parameters for trend g and two inputs in MC experiment

i.e., $E(\widehat{\tau}^2) > \tau^2$. This increase of $\widehat{\tau}^2$ implies that $s^2[\widehat{y}(\mathbf{x}_i)]$ increases; see the first term in (4). Originally, we conjectured expected that $\widehat{\theta}$ is unbiased (so, $E(\widehat{\theta}_j) = \theta_j$), but we shall return to this conjecture in our discussion of Table 3 below. If H_1 in (22) holds, then the OK predictor $\widehat{y}$ is not optimal (actually, the UK predictor—with correctly specified mean—would be optimal). In general, however, OK is known to give a *robust* metamodel. So we expect that γ remains relatively low, and increases as β increases.

Now we estimate the *power* γ of our LOO-CV test for I/O data with linear trends and different slopes. We proceed as discussed around (23), so, we obtain $m = 100$ samples from N_{20} defined in (22) with mean $\beta_1 x_1 + \beta_2 x_2$.

To understand the effects of g (trend slope) on the power curve $\gamma(\beta)$, we first investigate the effects of g on $\widehat{\boldsymbol{\psi}}$ (estimated OK parameters). Fig. 4 in Appendix 2 gives the boxplots for all four g-values. We summarize these boxplots in Table 3, which displays the sample medians of the 100 estimated OK parameters in the MC experiment with trend g and $d = 2$ inputs (so we have $\widehat{\theta}_{1;r}$ and $\widehat{\theta}_{2;r}$) (sample medians are more robust estimators than sample means). We analyze this Table as follows.

Table 3 clearly shows that the sample median of $\widehat{\mu}_r$ increases as g increases. This increase makes sense because $E(\widehat{\mu}) = 0.5\beta_1 + 0.5\beta_2 = \beta = g\tau$. The sample median of $\widehat{\tau}_r^2$ clearly increases as g increases; this makes sense, because the output shows more spread as g increases (this increase of $\widehat{\tau}_r^2$ implies that $s^2[\widehat{y}(\mathbf{x}_i)]$ increases, which decreases |PES|). The sample medians of $\widehat{\theta}_{1;r}$ and $\widehat{\theta}_{2;r}$ clearly decrease as g increases; i.e., a stronger trend implies that OK assigns higher weights to observations that are closer to the combination to be predicted (higher weights imply higher correlations, which imply lower θ; see (5)).

Finally, Table 4 shows the estimated power $\widehat{\gamma}$. Part (a) shows that $\widehat{\gamma}$ increases as g increases—as is to be expected. If $g = 0$, then (by definition) $\widehat{\gamma} = \widehat{\alpha}$; in this part of the table, $\widehat{\alpha}$ is much too high ($\widehat{\alpha} \gg \alpha_{\mathrm{E}}$). Therefore we add the *CH requirement*, and obtain part (b). This part shows that—for any of the three α_{E}-values—$\widehat{\gamma}$ tends to increase as g increases. If $g = 0$, then we still have $\widehat{\gamma} = \widehat{\alpha} \gg \alpha_{\mathrm{E}}$ ($g = 25$ gives $\widehat{\gamma}$ slightly smaller than $\widehat{\gamma}$ without the CH condition). We conclude that the CH constraint improves our results, but does not give good results. In part (c) we *bootstrap* the predictor variance, using a bootstrap sample size $B = 100$. This part shows that—for any of the three α_{E}-values—$\widehat{\gamma}$

g	$\alpha_{\mathrm{E}} = 0.20$	$\alpha_{\mathrm{E}} = 0.10$	$\alpha_{\mathrm{E}} = 0.05$
	(a) Basic variant		
0	0.45	0.32	0.23
1	0.50	0.38	0.24
5	0.83	0.81	0.78
25	0.93	0.88	0.87
	(b) CH variant		
0	0.35	0.25	0.14
1	0.35	0.25	0.19
5	0.77	0.73	0.61
25	0.89	0.84	0.81
	(c) Bootstrap variant		
0	0.10	0.05	0.04
1	0.12	0.06	0.03
5	0.71	0.61	0.50
25	0.97	0.95	0.93

Table 4: Estimated power in MC experiment with trend g and two inputs

tends to increase as g increases. If $g = 0$, then $\widehat{\gamma} = \widehat{\alpha} \ll \alpha_{\mathrm{E}}$, so Bonferroni's inequality is indeed conservative. If $g = 25$, then the estimated power is slightly larger when the variance estimator is bootstrapped. We conclude that in this MC example it is worthwhile to bootstrap the variance.

7 Two examples

We apply our LOO-CV to Gramacy (2016)'s example (with $d = 2$) and the borehole model (with $d = 8$). Each example gives the I/O data $(\mathbf{Z}_{n\times d}, \mathbf{v}_n)$ where we use the symbol $\mathbf{v}_n$—or briefly $\mathbf{v}$—to denote the output vector ($\mathbf{v}$ resembles $\mathbf{w}$, used in preceding MC experiments). The Kriging method treats the underlying simulation model as a *black box*, so OK uses only the simulation I/O data $(\mathbf{Z}_{n\times d}, \mathbf{v}_n)$. For this Kriging, we again use the DACE software, which also gives $\widehat{\boldsymbol{\psi}}_v$ (MLE of the parameters of the OK model for v). Finally, we apply our LOO-CV to these two examples. In general, OK gives a *robust* metamodel, so we expect that in these two examples OK gives a metamodel that is not rejected by our LOO-CV.

7.1 Gramacy (2016)'s example with two inputs

Gramacy (2016) presents an example with $d = 2$ original dimensionless inputs z_j ($j = 1, 2$) with the ranges $[l_j, u_j] = [-2, 2]$. To select n, we again follow the rule-of-thumb in Loeppky et al. (2009), so $n = 10d = 20$ (see Section 4). We generate a single LHS design $\mathbf{X}_{20\times 2}$ (using the midpoints of the n

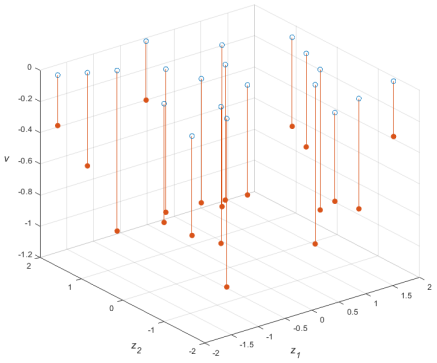

Figure 6: I/O data $(z_{1;i}, z_{2;i}, v_i)$ with $i = 1, ..., 20$ of example in Gramacy (2016)

subintervals for input x_j with $j = 1, 2$; also see the LHS in Section 4, and the MC experiment with $d = 2$ inputs in Section 6). This $\mathbf{X}_{20\times 2}$ gives $\mathbf{Z}_{20\times 2}$, using the linear transformation $z_j = -2 + 4x_j$. Using this $\mathbf{Z}_{20\times 2}$ as input for the simulation model gives the I/O data $(z_{1;i}, z_{2;i}, v_i)$ (with $i = 1, ..., 20$). We plot these I/O data that have negative v_i-values, in Fig. 6 which is a 3D plot with 20 vertical lines that start at $(z_{1;i}, z_{2;i}, 0)$ (empty circles) and end at $(z_{1;i}, z_{2;i}, v_i)$ (solid circles).

Actually, this simulation model is a single explicit mathematical function; because OK does not know the *white box* inside the black box, we present this function in Appendix 3. Moreover, we present the $d = 2$ scatterplots $(z_{j;i}, v_i)$ that use the I/O data in Fig. 6; such scatterplots may be used for SA (as Kleijnen and Helton (1999) does, albeit not in a Kriging context). Neither Fig. 6 (presented above) nor the Figure in Appendix 3 gives much insight into the behavior of the black box. Nevertheless, we apply OK to the I/O data $(\mathbf{Z}_{20n\times d}, \mathbf{v}_i)$ (LOO-CV focusses on prediction, not SA).

Note: We could have investigated many $\mathbf{X}_{20\times 2}$-matrixes (sampled through LHS or specified through some other type of space-filling design). However, we consider the $\mathbf{X}_{20\times 2}$ specified in Fig. 2 to be representative for applications of our LOO-CV to the example in Gramacy (2016).

Using these I/O data $(\mathbf{X}_{20\times 2}, \mathbf{v}_{20})$, DACE gives $\widehat{\boldsymbol{\psi}}_v$. Appendix 3 includes comments on this $\widehat{\boldsymbol{\psi}}_v$. For example, because $f(z_1) = f(z_2)$ if $z_1 = z_2$, we conjecture that $\widehat{\theta}_1/\widehat{\theta}_2 \approx 1$. The results in this appendix do not contradict this conjecture.

Using these $(\mathbf{X}, \mathbf{v})$ and $\widehat{\boldsymbol{\psi}}_v$, DASE gives $\widehat{y}$ (Kriging predictor) and $s^2(\widehat{y}_{-i})$ (estimated variance). In the upper pane of Fig. 7 we display the resulting augmented scatterplot, which implies that LOO-CV does not reject the OK metamodel. The corresponding $\max|\text{PES}_i|$ is 2.33, which is not significant (because our basic LOO-CV with $n = 20$ uses $z_{1-[\alpha_E/(2n)]} = z_{1-\alpha_E/40}$ so $\alpha_E =$

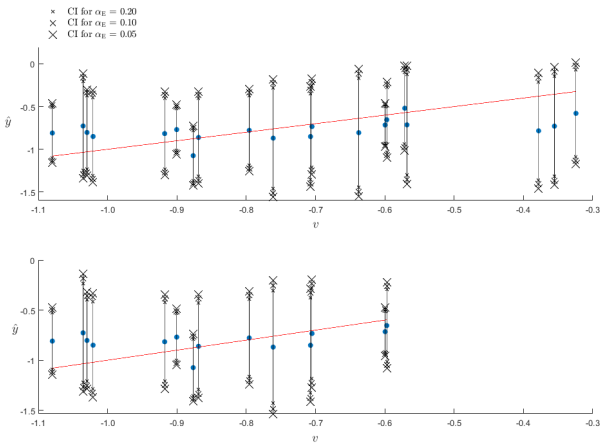

Figure 7: Scatterplot for $(v_i, \widehat{y}_i)$ augmented with $\widehat{y}_i \pm z_{\alpha_E/(2n)} s(\widehat{y}_{-i})$ and $\widehat{y}_i = v_i$, in Gramcy (2016)'s example with $d = 2$, $n = 20$ (upper pane) and $n - n_{CH} = 14$ (lower pane)

0.20 gives $z_{0.995} = 2.58$; obviously, $\alpha_E < 0.20$ gives $z_{1-\alpha_E/40} > 2.58$).

Given these results, we do not need to impose the CH condition. Nevertheless it may be interesting to check if $\max |\text{PES}_i| = |\text{PES}_{18}|$ occurs at a point in the CH. This CH was shown in Fig. 2, using the standardized input $\mathbf{x}$. So we find that $\max |\text{PES}_i|$ occurs at $\mathbf{z}_{18} = (-0.9, -1.1)'$, which lies inside the CH. (The next highest $|\text{PES}_i|$ is 1.82 and occurs at $\mathbf{z}_6 = (-0.1, 1.9)'$, which is one of the CH vertices.)

Note: The *classic* scatterplot gives $R^2 = 0.26$ without the CH condition, and $R^2 = 0.10$ with the CH condition, so these plots suggest that the OK metamodel is inadequate! (Actually, we conjectured that R^2 with the CH condition would be higher than R^2 without the CH condition.)

Furthermore these results imply that we do not need bootstrapping to estimate $Var(\widehat{y}_{-i})$. We know that the OK metamodel is not a perfect metamodel so the OK metamodel has some bias. This bias implies that MSPE $[\widehat{y}(\mathbf{x}_i, \boldsymbol{\psi})]$ exceeds $Var[\widehat{y}(\mathbf{x}_i, \boldsymbol{\psi})]$. This property makes the estimator based on (4) overestimate $Var(\widehat{y}_{-i})$. On the other hand, $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$ used in (9) ignores the plug-in character of the OK predictor, so it underestimates $Var(\widehat{y}_{-i})$. We do not know the net result. In practice we may ignore these complications, and use classic Kriging software to compute $s^2[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$; only if LOO-CV rejects the fitted OK metamodel, we may decide to apply bootstrapping to estimate $Var[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$.

7.2 Borehole example with eight inputs

Kleijnen and Van Beers (2019) gives details on the borehole model, but we summarize this model as follows. The borehole model has the output (say) v

j	Name of original input z_j, and measurement unit	Symbol	Range $[l_j, u_j]$
1	Radius of borehole, in meters (m)	r_w	0.05, 0.15
2	Radius of influence in m	r	100, 50000
3	Transmissivity of upper aquifer in m^2/year	T_u	63070, 115600
4	Potentiometric head of upper aquifer in m	H_u	990, 1110
5	Transmissivity of lower aquifer in m^2/year	T_l	63.1, 116
6	Potentiometric head of lower aquifer in m	H_l	700, 820
7	Length of borehole in m	L	1120, 1680
8	Hydraulic conductivity of borehole in m/year	K_w	9855, 12045

Table 5: Borehole inputs

and the $d = 8$ original inputs z_j (so $j = 1, ..., 8$). These z_j and their ranges $[l_j, u_j]$ are listed in Table 5 (the subscript w in r_w or z_1 has nothing to do with our output w in N_n). The output v denotes the water flow rate, measured in m^3 per year.

Analogously to our approach to the example in Gramacy (2016) presented in Section 7.1, we use the *standardized* inputs $0 \leq x_j \leq 1$; i.e., we use the ranges $[l_j, u_j]$ of z in Table 5 and the linear transformations $x_j = (z_j - l_j)/(u_j - l_j)$ (this standardization is also used in DASE; see Lophaven et al. (2002, eq. (2.1)). We use LHS to sample $x_{i;j}$ with $i = 1, ..., n$, and transform $x_{i;j}$ into $z_{i;j}$. We again follow Loeppky et al. (2009) so $n = 10d = 80$ (Gramacy (2016) uses $n = 50$ or $n = 200$ and Kleijnen and Van Beers (2019) uses $n = 60$). Next we obtain the I/O data $(\mathbf{Z}_{n\times d}, \mathbf{v}_n)$. Because $d = 8$ we cannot plot the analogue of Fig. 6 with $d = 2$. We can make the $d = 8$ scatterplots $(z_{j;i}, v_i)$; see the Figure in Appendix 4. Our interpretation of these plots is that z_1 (or r_w) has the strongest (positive) effect, whereas the other seven effects have no clear (main or first-order) effects. (This interpretation suggests UK with a first-order polynomial in r_w, but we limit our investigation to OK.) We use the I/O data $(\mathbf{Z}, \mathbf{v})$ of the borehole model to compute the OK metamodel. Appendix 4 includes the (white box) function $v(z_1, ..., z_8)$ and a detailed SA.

Note: We use the MATLAB function *lhsdesign* with $M = 5$ random permutations (see Section 4); to enable other researchers to reproduce our results, we mention that we initialize the PRN-stream with the MATLAB function "rng('default')".

Using $\mathbf{X}_{-i}$ and $\widehat{\boldsymbol{\psi}}_{-i}$, DACE computes $\widehat{y}_{-i}$ and $s(\widehat{y}_{-i})$ for $i = 1, ..., n = 80$. Using these DACE results, we compute $|\text{PES}_i|$ We find that $max_{1\leq i\leq n} |\text{PES}_i| = |\text{PES}_{71}| = 0.33$, which is not significant at all, for any of the three critical values $z_{1-\alpha_E/(2n)}$ and $t_{f;1-\alpha_E/(2n)}$ with $f = (n - 1) - (d + 2)$. More precisely, $\alpha_E = 0.20$ gives $z_{0.9988} = 3.0233$ and $t_{69;0.9988} = 3.1383$; $\alpha_E = 0.10$ gives $z_{0.9994} = 3.2272$ and $t_{69,0.9994} = 3.3659$; and $\alpha_E = 0.05$ gives $z_{0.9997} = 3.4205$ and $t_{69;0.9997} = 3.5847$. To summarize our LOO-CV, we make an augmented scatterplot with 80 CIs with $\alpha_E = 0.20$; see Fig. 8. This Figure implies that all 80 CIs intersect the 45° line, so LOO-CV does not reject the validity of the OK metamodel.

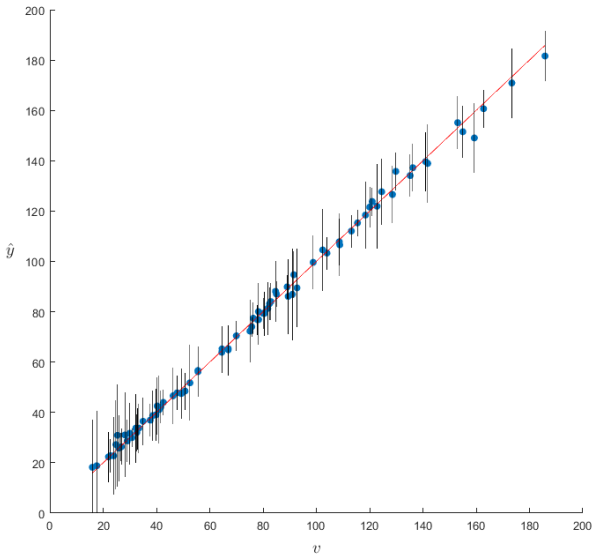

Figure 8: Scatterplot for $(v_i, \widehat{y}_i)$ augmented with $\widehat{y}_i \pm z_{\alpha_E/(2n)} s(\widehat{y}_{-i})$ and $\widehat{y}_i = v_i$, in borehole example with $d = 8$, $\alpha_E = 0.20$, $n = 80$

Obviously, scatterplots with $\alpha_E = 0.10$ or $\alpha_E = 0.05$ give longer CIs, so these CIs also overlap the 45° line. (The classic scatterplot gives $R^2 = 0.997$ with intercept $a = 1.276$ and slope $b = 0.983$, so R^2 is very high, but we do not know whether the fitted line significantly deviates from the 45° line so the OK metamodel is not adequate.)

Given these results, we do not need to impose the CH condition for our $\mathbf{X}_{80\times 8}$. Nevertheless, it is interesting that we find that the CH has $n_{CH} = n = 80$ vertices, so zero points remain to apply LOO-CV. If we increased n from 80 to 100, then we would find $n - n_{CH} = 100$ - 97 so we could apply LOO-CV to these 3 points (while computing $\widehat{y}$ from 99 points). Anyhow, a high d-value may imply that many points of the available n points require extrapolation.

Moreover, we may decide not to bootstrap the predictor variances $\sigma[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$, because we assume that such bootstrapping increases these variances which makes max_i $|\text{PES}_i|$ even less significant. Nevertheless, for $\mathbf{x}_{71}$ (which gave $max_{1\leq i\leq n} |\text{PES}_i| = 0.33$) we do obtain the bootstrapped estimator of $\sigma[\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}}_{-i})]$, which gives max_i $|\text{PES}_i| = 0.25$ (was 0.33).

Altogether, our results may explain why many researchers use OK for the borehole example.

8 Conclusions and future research

Our main conclusions are: (i) In practice, LOO-CV often does not reject the OK metamodel. (ii) However, if the basic variant of LOO-CV rejects the OK model, then imposing the CH constraint hardly improves LOO-CV. (iii) Replacing the normal quantile by a Student quantile hardly affects LOO-CV. (iv) Bootstrapping the predictor variance makes LOO-CV—which uses Bonferroni's inequality—conservative.

Furthermore we conclude that our MC experiments explain why OK is *robust*; i.e., OK gives valid approximations—tested through LOO-CV—even if the I/O data show a linear trend. This trend implies that the MLEs of the Kriging parameters increase the variance of the OK predictor; these MLEs change the correlation coefficients such that OK assigns higher weights to outputs of nearby points. (if the OK assumptions hold, then these MLEs may give outliers.)

Future research may address the following topics: (i) LHS may use non-uniform distributions, which is the case in uncertainty analysis. Actually, triangular distributions are discussed in Kleijnen and Van Beers (2019), using either midpoints or points sampled within subintervals. (ii) If LOO-CV rejects the OK (meta)model, then we may either apply an alternative Kriging model (e.g., UK) or collect additional I/O data. (iii) We may extend LOO-CV to *random* (instead of deterministic) simulation. In such a simulation we distinguish between the extrinsic *noise* $M(\mathbf{x})$ and the intrinsic noise caused by PRNs; this intrinsic noise may have either a homogeneous (constant) variance or heterogeneous variances. (iv) We may investigate k-fold CV with $k > 1$.

References

Bacchi, V., E. Antoshchenkova, H. Jomard, L. Bardet, C-M. Duluc, O. Scotti, and H. Hebert (2018), Development of a methodological framework for the assessment of seismic induced tsunami hazard through uncertainty quantification: application to the Azores-Gibraltar Fracture Zone. *Natural Hazards and Earth System Sciences* (under review)

Barber, C.B., D.P. Dobkin, and H. Huhdanpaa (1996), The Quickhull algorithm for complex hulls. *ACM Transactions on Mathematical Software*, 22, no. 4, pp. 469–483

Bartz-Beielstein, T. (2016), Stacked generalization of surrogate models; a practical approach. TH Köln

Bastos,L.S. and A. O'Hagan (2009), Diagnostics for Gaussian process emulators. *Technometrics*, 51, no. 4, pp. 425–438

Bhosekar, A. and M. Ierapetritou (2018), Advances in surrogate based modeling, feasibility analysis, and optimization: a review. *Computers and Chemical Engineering*, 108, pp. 250–267

Da Costa, J.J., F. Chainet, B. Celse,M. Lacoue-Negre, C. Ruckebusch, N. Caillol, and D. Espinat (2018), Kriging modeling to predict viscosity index of base oils. *Energy Fuels*, 32, pp. 2588–2597

de Carvalho, R.N., G.B. Machado, and M.J. Colaço (2017), Estimating gasoline performance in internal combustion engines with simulation metamodels. *Fuel*, 93, pp. 230–240

Den Hertog, D., J.P.C. Kleijnen, and A.Y.D. Siem (2006) The correct Kriging variance estimated by bootstrapping. *Journal Operational Research Society.* 57, no. 4, pp. 400–409

Dong, H. and M.K. Nakayama (2017), Quantile estimation with Latin hypercube sampling. *Operations Research*, 65, no. 6, pp. 1678–1695

Efron, B. (2015), Frequentist accuracy of Bayesian estimates. *Royal Statistical Society, Series B*, 77, no. 3, pp. 617–646

Erickson, C.B., B.E. Ankenman, S.M. Sanchez (2018), Comparison of Gaussian process modeling software. *European Journal of Operational Research*, 266, pp. 179–192

Forrester, A. and A. Keane (2009), Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45, no. 1–3, pp. 50–79

Garbo, A. and B.J. German (2019), Performance assessment of a cross-validation sampling strategy with active surrogate model selection. *Structural and Multidisciplinary Optimization*, in press

Gorissen, D., I. Couckuyt, E. Laermans, and T. Dhaene (2010), Multiobjective global surrogate modeling, dealing with the 5-percent problem. *Engineering with Computer*s, 26, pp. 81–98

Gramacy, R.B. (2016), laGP: large-scale spatial modeling via local approximate Gaussian processes in R. *Journal of Statistical Software*, 72, no. 1, pp. 1–46

Gramacy, R. B. and D. W. Apley (2015), Local Gaussian process approximation for large computer experiments. *Journal of Computational and Graphical Statistics*, 24, no. 2, pp. 561–578

Ji, D., W. Dong, T. Hong, T. Dai, Z. Zheng, S. Yang, and X. Zhu (2018), Assessing parameter importance of the weather research and forecasting model based on global sensitivity analysis methods. *Journal of Geophysical Research: Atmospheres*, 123, in press

Jin, S-S and H-J Jung (2016), Self-adaptive sampling for sequential surrogate modeling of time-consuming finite element analysis. *Smart Structures and Systems*, 17, no. 4,

Kleijnen, J.P.C. (1983), Cross-validation using the t statistic. *European Journal of Operational Research*, 13, no. 2, pp. 133–141

Kleijnen, J.P.C. (2015), *Design and analysis of simulation experiments; second edition.* Springer

Kleijnen, J. P. C. (2019), Kriging: methods and applications. *Handbook on Model Order Reduction. Volume I: Methods & Algorithms*, edited by P. Benner, G. Rozza, S. Grivet-Talocia, W.H.A. Schilders, A. Quarteroni, and L.M. Silveira, Walter De Gruyter, Berlin (a preprint is available on

https://sites.google.com/site/kleijnenjackpc/home/news)

Kleijnen, J.P.C. and J.C. Helton (1999), Statistical analyses of scatter plots to identify important factors in large-scale simulations, 1: review and compar-

ison of techniques. *Reliability Engineering & System Safety*, 65, no. 2, pp. 147–185

Kleijnen, J.P.C. and W. van Beers (2019), Prediction for big data through Kriging: small sequential and one-shot designs. *American Journal of Mathematical and Management Sciences*, accepted

Lataniotis, C., S. Marelli, and B. Sudret (2015), *UQLab user manual – Kriging (Gaussian process modelling)*. Report UQLab-V0.9-105, Chair of Risk, Safety & Uncertainty Quantification, ETH Zurich

Law, A.M. (2015), *Simulation modeling and analysis; fifth edition*. McGraw-Hill, Boston

Le Guiban, K., A. Rimmel, M-A. Weisser, and J. Tomasik (2018), The first approximation algorithm for the maximin Latin hypercube design problem. *Operations Research*, 66, no. 1, pp. 253–266

Lin, Y., B.L. Nelson, and L. Pei (2019), Virtual statistics in simulation via k nearest neighbors. *INFORMS Journal on Computing*, in press

Luminari, N., C. Airiau, and A. Bottaro (2018), Effects of porosity and inertia on the apparent permeability tensor in fibrous media. *International Journal of Multiphase Flow*, 106, pp. 60–74

Loeppky, J.L., J. Sacks, and W. Welch (2009) Choosing the sample size of a computer experiment: a practical guide. *Technometrics*, 51, no. 4, pp. 366–376

Lophaven, S.N., H.B. Nielsen, and J. Sondergaard (2002), *DACE: a Matlab Kriging toolbox, version 2.0*. IMM Technical University of Denmark, Kongens Lyngby

Lupera Calahorrano, G.J., A. Shokry, G. Campanya, and A. Espuña (2016),Application of the meta-multiparametric methodology to the control of emissions in the industry under continuous and discrete uncertain parameters. *Chemical Engineering Research and Design*, 115, pp. 365–373

Mehdad, E. and J.P.C. Kleijnen (2015) Classic Kriging versus Kriging with bootstrapping or conditional simulation: classic Kriging's robust confidence intervals and optimization. *Journal Operational Research Society,* 11, pp. 1804-1814

Panagiotopoulos, D., O. Iqbal, Z. Mourelatos, and D. Papadimitriou (2018), Optimal water jacket flow distribution using a new group-based space-filling design of experiments algorithm. SAE Technical Paper 2018-01-1017, 2018, doi:10.4271/2018-01-1017

Parnianifard, A., A.S. Azfanizam, M.K.A. Ariffin, and M.I.S. Ismail (2018), Kriging-assisted robust black-box simulation optimization in direct speed control of DC motor under uncertainty. *IEEE Transactions on Magnetics*, pp. 1–10

Quirante, N., J. Javaloyes-Antón, and J.A. Caballero (2018), Hybrid simulation-equation based synthesis of chemical processes. *Chemical Engineering Research and Design*, 132, pp. 766–784

Rasmussen, C.E. and C.K.I. Williams (2006), *Gaussian processes for machine learning*. The MIT Press,

http://www.gaussianprocess.org/gpml/chapters/RW.pdf

Roustant, O., D. Ginsbourger, and Y. Deville (2012). DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization. *Journal of Statistical Software*, 51, no. 1, pp. 1—55

Santner, T.J., B.J. Williams, and W.I. Notz (2018), *The design and analysis of computer experiments; second revised edition.* Springer, New York

Shu, L., P. Jiang, L. Wan, Q. Zhou, X. Shao, and Y. Zhang, (2017), Metamodel-based design optimization employing a novel sequential sampling strategy .*Engineering Computations*, 34, no. 8, pp.2547–2564

Simes, R.J. (1986), An improved Bonferroni procedure for multiple tests of significance. *Biometrika*, 73, no. 3, pp. 751–754

Song, E., B.L. Nelson, and J. Staum (2016), Shapley effects for global sensitivity analysis: theory and computation. *SIAM-ASA Journal on Uncertainty Quantification*, 4, pp. 1060–1083

Stone, M. (1974), Cross-validatory choice and assessment of statistical predictions. *Journal Royal Statistical Society, Series B*, 36, no. 2, pp. 111–147

Strano, M., Q. Semeraro, L. Iorio, and R. Sofia (2018), Hierarchical metamodeling of the air bending process . *Journal of Manufacturing Science and Engineering*, 140, pp. 1–19

Sun, F., R.B. Gramacy, B. Haaland, E. Lawrence , and A. Walker (2018), Evaluating satellite drag from large simulation experiments. arXiv:1712.00182v2 [stat.CO] 21 Aug 2018

Van Steenkiste, T., J. van der Herten, I. Couckuyt, and T. Dhaene (2018), Sequential sensitivity analysis of expensive black-box simulators with metamodelling. *Applied Mathematical Modelling,* 61, pp. 66 8–6 81

Viúdez-Moreiras, D. (2018), Performance influences on metamodelling for aerodynamic surrogate-based optimization of an aerofoil. *Engineering Optimization*, in press

Wang, W. and B. Haaland (2018), Controlling sources of inaccuracy in stochastic kriging, *Technometrics* (in press)

Wang, X., D.J. Nott, C.C. Drovandi, K. Mengersen, and M.Evans (2018): Using history matching for prior choice, *Technometrics*, 60, no. 4, pp. 445–460

Xiao, N-C. M.J. Zuo,and W. Guo (2018), Efficient reliability analysis based on adaptive sequential sampling design and cross-validation. *Applied Mathematical Modelling*, 58, pp. 404–420

Xiong, Y., W. Chen, D. Apley, and X. Ding. 2007. A non-stationary covariance-based kriging method for metamodelling in engineering design. *International Journal for Numerical Methods in Engineering*, 71, no. 6, pp. 733–75

Yin, s., Z. Wang, Z. Zhu, X. Zou, and W. Wang (2018), Using Kriging with a heterogeneous measurement error to improve the accuracy of extreme precipitation return level estimation. *Journal of Hydrology*, 562, pp. 518–529

Zhang, J., A.A. Taflanidis, and J. C. Medina (2017), Sequential approximate optimization for design under uncertainty problems utilizing Kriging metamodeling in augmented input space. *Computer Methods in Applied Mechanics and Engineering*, 315, pp. 369–395

i	1	2	3	4	5	6	7	8	9	10
x_1	-0.5	-0.7	1.5	1.7	-1.5	-0.1	-0.3	0.5	1.1	1.3
x_2	0.9	-0.7	0.1	0.7	1.5	1.9	0.3	-1.9	-1.7	-0.5

i	11	12	13	14	15	16	17	18	19	20
x_1	-1.1	0.9	0.7	-1.7	-1.9	0.1	1.9	-0.9	0.3	-1.3
x_2	1.3	-0.9	1.1	-1.3	1.7	-0.3	-1.5	-1.1	0.5	-0.1

Table 6: A LHS design for twenty combinations of two inputs

Zou L. and X. Zhang (2018), Stochastic Kriging for inadequate simulation models. arXiv:1802.00677v2 [stat.ME] 13 Feb 2018

Acknowledgement: Dick den Hertog (Tilburg University) suggested the LP solution for our CH problem. Ruud Brekelmans (Tilburg University) suggested the use of the MATLAB function "full".

Appendix 1: LHS design for twenty combinations of two inputs

Table 6 displays our LHS design with $n = 20$ combinations of the $d = 2$ standardized inputs $x_{1;i}$ and $x_{2;i}$ with $i = 1, ..., n$.

Appendix 2: Boxplots with estimated Kriging parameters in MC experiment with four slopes of linear trend

Fig. 9 displays boxplots for the Kriging parameters $\widehat{\mu}_r$, $\widehat{\tau}_r^2$, $\widehat{\theta}_{1;r}$ and $\widehat{\theta}_{2;r}$ with $r = 1, ..., 100$ in the MC experiment with input $\mathbf{X}_{20\times 2}$ and a linear trend with slope $\beta = g\tau$ with $g = 0$, 1, 5, and 25, respectively. We observe that $g = 25$ gives a box for $\widehat{\theta}_{1;r}$ with zero length; the first, second, and third quantiles have the same value (namely, 0.4819). Furthermore, $\widehat{\theta}_{2;r}$ varies between 0.0602 and 0.8839, but 65 (out of 100) estimates coincide with the sample median.

Appendix 3: Details of Gramacy (2016) example

Fig. 11 displays the two scatterplots $(z_{j;i}, v_i)$ with $j = 1$, 2 for Gramacy's example.

Actually, the I/O data of the black box in Gramacy's example are determined by the following (white box) explicit mathematical function:

$$v(z_1, z_2) = -f(z_1)f(z_2) \text{ with } -2 \leq z_j \leq 2 \text{ and} \tag{24}$$

$$f(z_j) = e^{-(z_j-1)^2} + e^{-0.8(z_j+1)^2} - 0.05\sin(8(z_j + 0.1)). \tag{25}$$

Fig. 12 gives the 3D plot of this function using as many as $N = 40{,}401$ input combinations defined by the 201×201 grid in $[-2, 2]^2$. This plot shows that this function is very "wiggly", which agrees with our interpretation of the plots with only $n = 20$ I/O data. (Fig. 11 and Fig. 12 show that v is extreme if z_j is extreme.) OK assumes a constant mean, which seems a reasonable

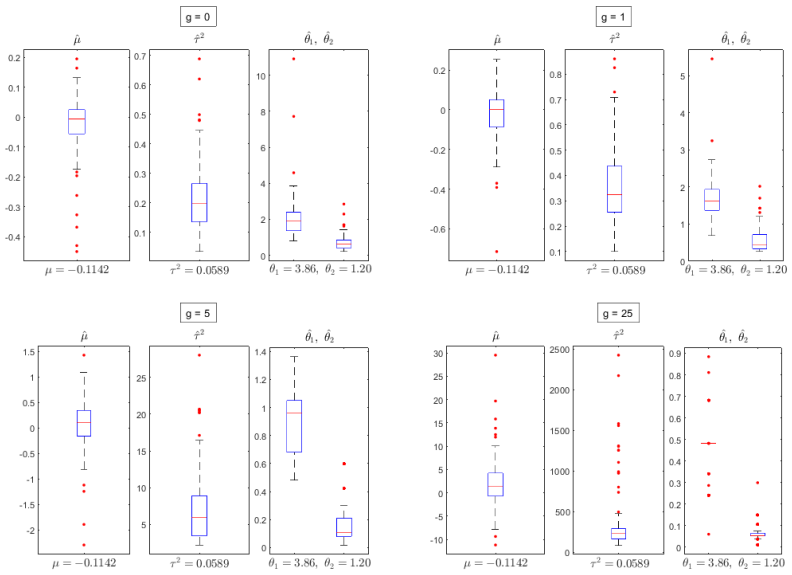

Figure 9: Boxplots for OK parameters in MC experiment with 100 replications, 20 combinations of two inputs and slope g = 0, 1, 5, and 25 of linear trend

assumption (unlike the borehole example which has a monotonically increasing I/O function).

Note: Gramacy (2016) considers $N = 40{,}401$ I/O points. Because such a high N value gives computational problems when computing the OK predictor and its variance, Gramacy uses only $n = 50$ points—among these N points—that lie close to a given new input combination for which the output has to be predicted. Whereas Gramacy uses a rather complicated procedure to select these n points, Kleijnen and Van Beers (2019) simply uses the n nearest neighbors of this new input combination. Anyhow, we expect that a local set of I/O data gives an adequate OK model, because there are relatively many neighboring points to obtain an accurate interpolator. We, however, are not interested in local prediction, but in prediction over the whole area $[-2, 2]^2$.

We find it hard to explain the specific values for $\widehat{\theta}_j$ (also see our discussion of the prior distribution for θ_j in the Bayesian approach to GP, in the Introduction). We may compare the *relative* values of $\widehat{\theta}_j$ ($j = 1, 2$) in this example (with $f(z_1) = f(z_2)$ if $z_1 = z_2$; see (25)), so we conjecture $\theta_1/\theta_2 = 1$ or $\widehat{\theta}_1/\widehat{\theta}_2 \approx 1$. However, our specific LHS design gives $\widehat{\theta}_1/\widehat{\theta}_2 = 3.8555/1.1970 = 3.2$. To investigate this difference between $\widehat{\theta}_1$ and $\widehat{\theta}_2$, we sample 100 LHS designs—all with the same $n = 20$ and the same range $[0, 50]$ for $\widehat{\theta}_1$ and $\widehat{\theta}_2$ in DACE's search for the MLEs. This sample gives Fig. 13, which is the bivariate plot of the resulting pairs $(\widehat{\theta}_1, \widehat{\theta}_2)$ with $r = 1, ..., 100$. Actually, some points in this plot coincide;

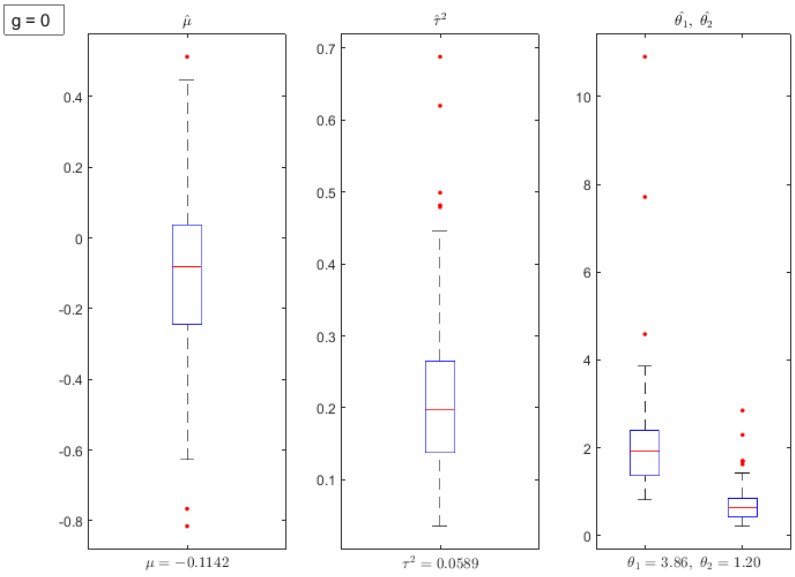

Figure 10: Boxplots for OK parameters in MC experiment with 100 replications. 20 combinations of two inputs and slope g = 0, 1, 5, and 25 of linear trend

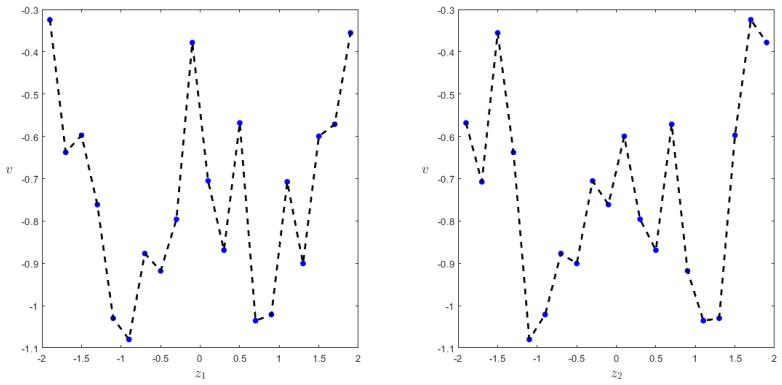

Figure 11: Scatterplots $(z_{1;i}, v_i)$ and $(z_{2;i}, v_i)$ using I/O data of example in Gramacy (2016)

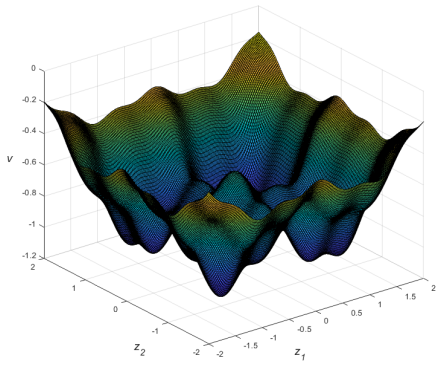

Figure 12: Gramacy (2016)' s example with I/O data at $N = 40{,}401$ points

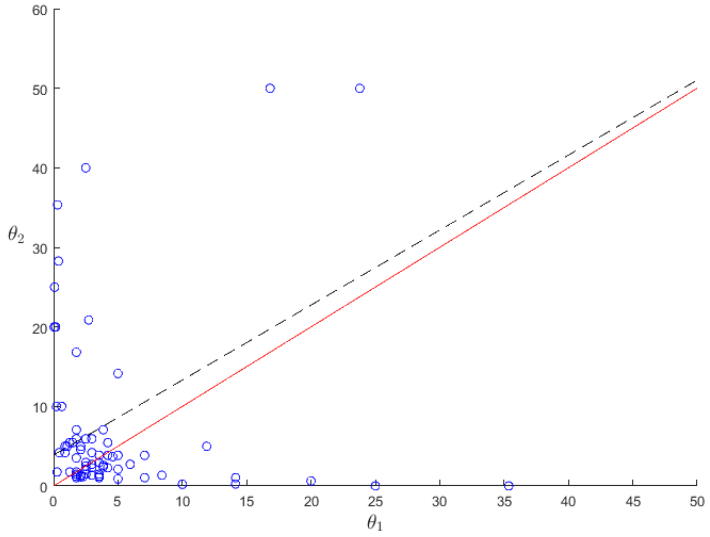

Figure 13: Bivariate plot of pairs $(\widehat{\theta}_{1;r}, \widehat{\theta}_{2;r})$ $(r = 1, ..., 100)$ with $0 \leq \widehat{\theta}_{j;r} \leq 50$ $(j = 1, 2)$ in 100 LHS designs for Gramacy (2016)'s example

i.e., (0.078, 25) occurs two times, and (0.63, 10) occurs four times. Two points are very close; namely, (0.04, 20) and (0.16, 20). Furthermore, this plot shows that two replications give $\widehat{\theta}_2 = 50$, which is the upper limit in DACE's search for the MLEs (in practice, we would relax this limit, and continue DACE's search). More important, the plot shows that 53 of these 100 designs give $\widehat{\theta}_2 > \widehat{\theta}_1$; i.e., 53 points lie above the line with unit slope and zero intercept (see the solid, non-dashed line). This result does not contradict our conjecture (namely, $\theta_{v;1}$ $= \theta_{v;2}$). The plot also shows the first-order polynomial fitted through LS, which turns out to have intercept 3.89 and slope 0.94 (see dashed line). This intercept is close to the "true" value $\theta_{v;1} = \theta_{v;2} \approx 3$ (the value 3 is explained in the next Note). Most of the 100 (estimated) points cluster around the (true) point $(3, 3)$; i.e., DACE's search does give "reasonable" estimates. If we use $\rho_{\boldsymbol{\theta}}$ to denote Pearson's correlation coefficient for the pair $(\widehat{\theta}_1, \widehat{\theta}_2)$, then this fitted line gives $\widehat{\rho}_{\boldsymbol{\theta}} = 0.49$ (the relationship between $\rho^2_{\boldsymbol{\theta}}$ for a two-dimensional random variable and R^2 for a model with one or more independent variables or predictors and one dependent variable or response is discussed in Kleijnen (2015, p. 113)). We know that the LS criterion gives results that are sensitive to outliers, so in the plot we might remove outliers such as the two points with $\widehat{\theta}_2 = 50$. However, we do not further refine our analysis of the fitted line, because this line does not contradict our conjecture (namely, $\theta_{v;1} = \theta_{v;2}$). The rather low value of $\widehat{\rho}_{\boldsymbol{\theta}}$ (namely, 0.49) is explained by the rather high standard deviations of $\widehat{\theta}_1$ and $\widehat{\theta}_2$; namely, 6.98 and 13.47, while the sample means are 5.08 and 8.68 and the sample medians are 2.50 and 3.54.

Note: We do obtain $\widehat{\theta}_1 \approx \widehat{\theta}_2$ if we use a grid (instead of LHS); e.g., $n \times n$ grids with n is 4, 5, 10, 25 give (0.3125 0.2210), (2.5000 2.4148), (2.9730 2.9730), (3.2421 3.3856). Obviously, the estimates do not equal the true values $\theta_{v;j}$ ($j =$ 1, 2), which implies that the estimates $\widehat{\boldsymbol{\lambda}}_{-i}$ of the Kriging weights do not equal the true values $\boldsymbol{\lambda}_{-i}$, so the plug-in Kriging predictor is not the BLUP. To further investigate Gramacy's example (in which z_1 and z_2 play the same role),we replace the Kriging metamodel by a simpler metamodel; namely, the *second-degree polynomial*. We conjecture that in this polynomial the first-order effects of z_1 and z_2 are the same, and so are their purely quadratic effects. Indeed, when we fit this polynomial to the I/O data of our LHS design with $n = 20$, then we obtain estimated first-order effects that are nearly the same (namely, 0.03), and purely quadratic effects that are also nearly the same (namely, 0.09) (the two-factor interaction is 0.00, and the intercept is -0.98). The 4×4 grid (which is the grid with the minimum size when fitting the Kriging metamodel) gives estimated first-order effects that are exactly the same (namely, 0.005), and purely quadratic effects that are also exactly the same (namely, 0.104) (the two-factor interaction is -0.000, and the intercept is -0.963). Altogether, the LHS and the grid designs confirm our conjecture; i.e., z_1 and z_2 have the same first-order and purely quadratic effects in the second-degree polynomial metamodel.

Appendix 4: Details of the borehole example

Fig. 14 displays the scatterplots $(z_{j;i}, v_i)$ with $j = 1$, ..., 8 and $i = 1$, ..., 80

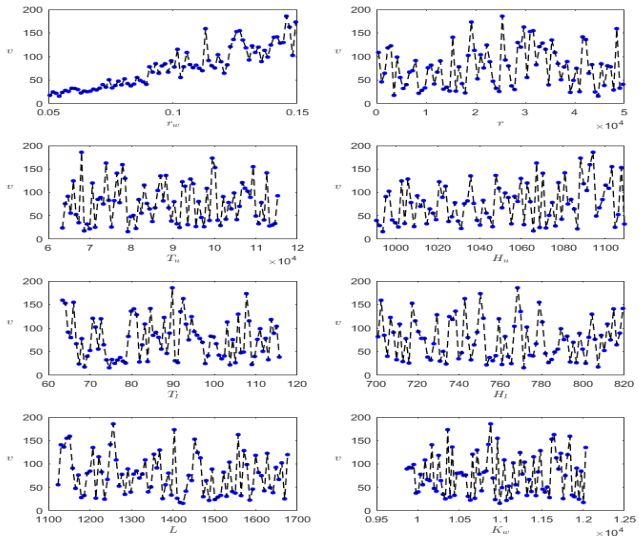

Figure 14: Scatterplots $(z_{j;i}, v_i)$ with $j = 1, ..., 8$ and $i = 1, ..., 80$ using I/O data of borehole example

for all inputs. Actually, we compute v from the following (white box) function:

$$v = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)(1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 d_w} + \frac{T_u}{T_l})}. \tag{26}$$

Computer codes in MATLAB and R for this model are available on

https://www.sfu.ca/~ssurjano/borehole.html.

Obviously, (26) is nonlinear and has non-additive effects. The *interpretation* of the simulation experiment should use the original I/O data, as we explain now.

We make 3D plots (like Fig. 12); i.e., we plot the output v_i $(i = 1, ..., n)$ versus two original inputs while keeping the other six original inputs constant at the midpoints of their ranges. Santner et al. (2018) estimates that r_w is the most important input and that the three inputs L, H_l, and H_u have approximately equally important effects. In the left-hand pane of Fig. 15 we use a grid of $N = 21 \times 21$ input combinations (r_w, L) while keeping the other six inputs constant; the right-hand pane gives a similar plot for (r, T_u). We point out that the range of w is relatively small in the right-hand side, so the effect of r is relatively small (the effect of $T_u$is unimportant). We also make such 3D plots for all other combinations of two inputs; these plots are very similar, so we do not display them. Moreover, we make contourplots. These 3D plots and contourplots confirm Santner et al.'s conclusion that r_w has the most important

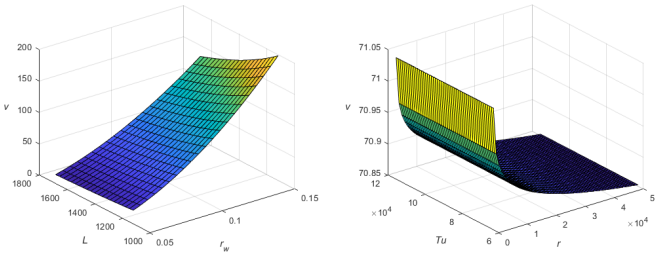

Figure 15: Borehole example: I/O function with $N = 21 \times 21$ input combinations (r_w, L) (left pane) and (r, T_u) (right pane) versus output w

main effect and has important interactions. Moreover, these plots suggest that the I/O function is *monotonic*; i.e., the borehole model looks complicated, but seems to give a simpler I/O function than Gramacy's example in Appendix 3.

Santner et al. (2018, Fig 7.10 and Table 7.11) quantifies the importance of z_j through the estimated "main effect" and the "total sensitivity index"; this index measures the contribution to the total output variance due to z_j including all variances caused by all the interactions between z_j and any other inputs (we shall further discuss main effects and indices below).

Using $(\mathbf{X}_{80\times 8}, \mathbf{v}_{80})$, DACE gives $\widehat{\boldsymbol{\psi}}$. This MLE includes $\widehat{\mu} = 76.74$ (which agrees with the mean output in the left-hand pane of Fig. 15). Furthermore, $\widehat{\tau} = 40.10$ (which also agrees with the spread in the outputs observed in this pane). When searching for the values of $\widehat{\theta}_j$ that maximize its log-likelihood function, we did some preliminary tests which suggest that the interval $[0.001, 5]$ is appropriate for this search. This gives the following $\widehat{\theta}_j$-values (for $j = 1$, ..., 8), where we display the first three decimal units because the lower limit of our search (namely, 0.001) also uses three decimals: 0.310, 0.067, 0.015, 0.017, 0.001, 0.007, 0.011, 0.002. So $\widehat{\theta}_5 = .001$, which equals the lower limit; i.e., a better estimate of θ_5 might have a lower value than 0.001. We now discuss the use of these $\widehat{\theta}_j$ for SA of the borehole example.

Originally we conjectured that $\widehat{\theta}_1$ would be the smallest of the $\widehat{\theta}_j$-values, because r_w (or z_1) seems the most important input (see Fig. 15). The magnitudes of $1/\widehat{\theta}_j$ (not $\widehat{\theta}_j$) are also shown in the boxplot in Sun (2018, p. 12), but not in a CV context; that boxplot confirms our results. More precisely, our definition of the Gaussian correlation function equals the definition in Lophaven et al. (2002, p. 6). Our definition equals the definition in Sun et al. (2018, p. 6) provided we replace our $\theta_j h_j^2$ by h_j^2/θ_j. Moreover, in Section 3 we mentioned that different software may give different $\widehat{\theta}$.

Now, however, we revisit our original conjecture; i.e., now we notice that the Gaussian correlation function is determined not only by $\widehat{\theta}_j$, but also by h_j^2 where $h_j = |x_{g;j} - x_{g';j}|$ $(g, g' = 1, ..., n)$. If we use the original inputs, then θ_j becomes (say) $\theta_{z;j}$, and h_j becomes $h_{z;j}$ which depends on the range

$[l_j, u_j]$. In $\mathbf{Z}_{n\times d}$ (determined by $\mathbf{X}_{n\times d}$ sampled by LHS) the common length of the n subranges of z_j is $(u_j - l_j)/n$. So the observed distances between two outputs at the midpoints of these subranges are a multiple of this length; e.g., there are $n-1$ observations on outputs with inputs at the distance $(u_j - l_j)/n$ and there is a single observation on the two outputs with the longest distance $(u_j - l_j)(n-1)/n$. In general, there are i observations on outputs with inputs that are $(u_j - l_j)(n-i)/n$ apart (with $i = 1, ..., n$). The range of r_w is very small compared with the ranges of the other seven inputs. We observe that Gramacy (2016)'s example in Appendix 3 has inputs with the same range (namely, $[-2, 2]$). Similar scaling effects occur in linear regression, as we explain next.

In general, low-order polynomial linear-regression gives a good *local* prediction (Taylor-series argument), whereas Kriging gives a good *global* prediction. More precisely, linear regression gives $\widehat{y} = \mathbf{x}'\widehat{\boldsymbol{\beta}} = \mathbf{z}'\widehat{\boldsymbol{\beta}}_{\mathbf{z}}$ where $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}$ and $\widehat{\boldsymbol{\beta}}_{\mathbf{z}} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}$; however, numerical inaccuracies affect $\widehat{\boldsymbol{\beta}}_{\mathbf{z}}$ more than $\widehat{\boldsymbol{\beta}}$. A low-order polynomial helps SA, because this polynomial implies d main effects, $d(d-1)/2$ two-factor interactions, and d—diminishing or increasing—rates of return. Standardization of the inputs such that the standardized values range between -1 and +1 immediately shows which input has the most important estimated main effect; namely, the input with he highest absolute value of the estimated main effect (classic "response surface methodology" or RSM does not standardize, so its steepest-ascent direction is scale-dependent). *Analysis of variance* (ANOVA) estimates higher-order interactions including the interactions among all d inputs. Kriging may speed-up *functional ANOVA* (FANOVA) or *global SA* (GSA), which uses Sobol's indices including "total sensitivity indices". (Sobol's indices—and Shapley's value in game theory—can also estimate interactions; see Kleijnen (2015, pp. 216–218) and Song et al. (2016).). Because Kriging is primarily predictive, most analysts do not pay much attention to the individual parameters within $\boldsymbol{\psi}$ (vector with $2 + d$ Kriging parameters), but focus on the Kriging predictor $\widehat{y}(\mathbf{x}_i, \widehat{\boldsymbol{\psi}})$—as we also do in LOO-CV. For more discussion we refer to Kleijnen (2015).