Small Sample Bayesian Designs for Complex High-Dimensional Models Based on Information Gained Using Fast Approximations

Jonathan Cumming and Michael Goldstein

Department of Mathematical Sciences

Durham University, U.K.

email: j.a.cumming@durham.ac.uk

June 16, 2009

Abstract

We consider the problem of designing for complex high-dimensional computer models which can be evaluated at different levels of accuracy. Ordinarily, this requires performing many expensive evaluations of the most accurate version of the computer model in order to obtain a reasonable coverage of the design space. In some cases, it is possible to supplement the information from the accurate model evaluations with a large number of evaluations of a cheap, approximate version of the computer model to enable a more informed design choice. We describe an approach which combines the information from both the approximate model and the accurate model into a single multiscale emulator for the computer model. We then propose a design strategy for the selection of a small number of expensive evaluations of the accurate computer model based on our multiscale emulator and a decomposition of the input parameter space. The methodology is illustrated with an example concerning a computer simulation of a hydrocarbon reservoir.

KEYWORDS: Computer experiments; Experimental design; Bayes linear methods; Multiscale emulation; Hydrocarbon reservoir.

1. INTRODUCTION

Computer models are widely used in many areas of scientific research to gain insight into complex physical systems. Evaluating such models for a given choice of input values can be very expensive in time and computation. Therefore, the number of evaluations that can be performed within a fixed budget is limited, resulting in substantial uncertainty associated with the output of the model away from these known values. To represent this uncertainty, we construct a stochastic representation of the simulator, known as an *emulator*, by combining the available model evaluations with appropriate prior knowledge (Sacks, Welch, Mitchell, and Wynn, 1989b; Craig, Goldstein, Rougier, and Seheult, 2001; Santner, Williams, and Notz, 2003; Kennedy and O'Hagan, 2001).

Often, the computer model can be evaluated at different levels of accuracy resulting in different versions of the computer model of the same system (Kennedy and O'Hagan, 2000; Craig, Goldstein, Seheult, and Smith, 1998). For example, this can arise from simplifying the underlying mathematics, by adjusting the model gridding, or by changing the accuracy of the model's numerical solver. Lower accuracy models can often be evaluated for a fraction of the cost of the full computer model, and can share many qualitative features with the original. Often, the coarsened computer model is informative for the accurate computer model, and hence for the physical system itself. By using evaluations of the coarse model in addition to those of the full model, we can construct a single *multiscale emulator* of the computer simulation (Craig et al., 1998; Kennedy and O'Hagan, 2000; Qian and Wu, 2008).

As the accurate computer model is expensive to evaluate, our number of potential evaluations is limited and hence these evaluations must be carefully chosen. Designing for computer experiments and emulators has a long history (Sacks et al., 1989b; Currin, Mitchell, Morris, and Ylvisaker, 1991; Sacks, Schiller, and Welch, 1989a). Popular design methods include space-filling techniques such as Latin hypercubes (Morris and Mitchell, 1995), or criterion-based methods using, for example, maximum entropy or mean square prediction error (Santner et al., 2003, Chapter 6). However, for functions with high-dimensional output spaces, each function evaluation is important for each of a large number of output quantities, and choosing a small collection of design points that are informative for all the outputs is very challenging.

In this paper, we describe a methodology for constructing a multiscale emulator by using many approximate model runs. Secondly, we propose a tractable strategy for making such small-sample designs for high-dimensional computer model output by decomposing the problem into smaller de-

sign problems based on a version of the conditional independence representation of the structure of the design. The structure of this paper is as follows: In Section 2 we describe our general methodology for constructing a multiscale emulator of a complex computer model. Section 3 discusses the structure of the design problem, and introduces our input space decomposition and design construction strategy. The methodology is illustrated by application to a computer simulation of a hydrocarbon reservoir in Section 4, and we conclude with discussion in Section 5.

2. MULTISCALE EMULATION OF COMPLEX COMPUTER MODELS

2.1 Analysis of an Uncertain Multiscale Computer Model

We begin with a complex physical system, represented by a q-vector of system attributes, $\boldsymbol{y}$. The computer simulator, $F(\cdot)$, is a deterministic function which models this system. The inputs, $\boldsymbol{x}$, to the computer model comprise a vector of quantities which influence the value of $F(\cdot)$ (and hence $\boldsymbol{y}$), and must be specified before the simulator can be evaluated. The resulting output, $F(\boldsymbol{x})$, is a q-vector, corresponding to the system attributes $\boldsymbol{y}$. Evaluating the simulator $F(\boldsymbol{x})$ at a given $\boldsymbol{x}$ can be expensive both in time and resources. This severely limits the number of evaluations that we can make, and restricts the available information on $F(\cdot)$. Consequently, there is substantial uncertainty about the simulator at any $\boldsymbol{x}$ which has not yet been evaluated. This uncertainty is characterised by the emulator, $f(\boldsymbol{x})$, obtained from the synthesis of appropriate prior judgement and available model evaluations.

When an approximate version of the simulator is available, we refer to it as the *coarse simulator*, $F^c(\cdot)$, and the original as the *accurate simulator*, $F^a(\cdot)$, to reflect these differences in precision. In such a setting, we can obtain many evaluations of the coarse simulator and use these to construct an informed emulator, $f^c(\cdot)$, for $F^c(\cdot)$. This provides a basis for constructing an informed prior specification for the accurate emulator, $f^a(\cdot)$. We then select and evaluate a small number of accurate model runs to update our emulator for $F^a(\cdot)$. This transfer of beliefs from coarse to accurate emulator is the basis of multiscale emulation.

2.2 Output Dimension Reduction

Suppose that we have a large batch of runs, $\boldsymbol{F}^c$, of the coarse simulator over an appropriate space-filling design, $\boldsymbol{X}^c$, such as a Latin hypercube (McKay, Beckman, and Conover, 1979; Santner et al., 2003). Emulation and design are highly computationally intensive, so it is often infeasible to study every output generated by the computer simulator if the output dimension, q, is large ($q \geq 20$ say). Thus, we first consider reducing the size of this output collection, where possible.

In this paper, we reduce the dimension by identifying a subset of outputs. We assume that we have already identified a set of appropriate outputs of interest from the computer model; we then seek to reduce the size of this collection. Alternatively, we could consider linear combinations or non-linear transformations however the design methods of Section 3 benefit from having emulators which are driven by only a relatively small set of inputs, which may not be the case due to the increased complexity of the relationships between the inputs and the modified outputs. If no suitable reduction via subset selection can be found, then appropriate linear transformations of the outputs based, for example, on canonical correlation or principal components may be used in the following analyses.

To identify the reduced subset of outputs to emulate, we adopt the principal variable selection method of Cumming and Wooff (2007), which operates by scoring each output, y_i, by the value $h_i = \sum_{j=1}^{n(y)} \mathrm{Corr}\,(y_i, y_j)^2$, and selecting the output which maximises this statistic, where $n(y)$ is the number of outputs under consideration at this stage of selection. Subsequent outputs are identified using the partial correlation of the remaining variables, given those already chosen, to eliminate the effects of the selected outputs from subsequent analysis. We calculate this partial correlation by first partitioning the correlation matrix $\boldsymbol{R} = \mathrm{Corr}\,[\boldsymbol{y}]$ into the block form

$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} \end{pmatrix},$$

where $\boldsymbol{R}_{11}$ is the matrix of correlations of the identified principal variables, $\boldsymbol{R}_{22}$ is the sub-matrix of correlations of the remaining variables, and $\boldsymbol{R}_{12}$ and $\boldsymbol{R}_{21}$ contain the correlations between the two groups. Given a non-trivial set of PVs, the matrix over which we calculate the h_i is then given by

$$\boldsymbol{R}_{22\cdot 1} = \boldsymbol{R}_{22} - \boldsymbol{R}_{21}\boldsymbol{R}_{11}^{-1}\boldsymbol{R}_{12}. \tag{1}$$

The process then iterates until sufficient variables are chosen that the standardised partial variance of each remaining output is small. In general, we obtain large values of h_i when output y_i has, on average, high loadings on important principal components of the correlation matrix and thus corresponds to a structurally important variable. Designs which are informative for all the selected principal variables will be informative for the remaining outputs, through the joint correlation structure.

2.3 Emulating the Coarse Simulator

For problems of moderate size and complexity, a fully Bayesian approach based on full probability specifications for the emulator, often using a Gaussian form (Kennedy and O'Hagan, 2001; Santner et al., 2003), can be appropriate and effective. However, when considering problems of high dimension it becomes increasingly complex to specify meaningful prior distributions over high-dimensional spaces and the computations required to learn from data become technically difficult and extremely computationally intensive, particularly in the case of identifying informative designs. For these reasons, we adopt a Bayes linear approach in which we specify and construct Bayes linear emulators. In the Bayes linear formulation (Goldstein and Wooff, 2007), we take expectation as the primitive quantification of uncertainty and require only the specification of the means, variances and covariances for the emulators.

To represent our uncertainty about the high-dimensional coarse computer model $F^c(\boldsymbol{x})$, we build an emulator $f_i^c(\boldsymbol{x})$ for each component i of $F^c(\boldsymbol{x})$. We express this emulator in the following form

$$f_i^c(\boldsymbol{x}) = s_i^c(\boldsymbol{x}) + w_i^c(\boldsymbol{x}), \tag{2}$$

where $s_i^c(\boldsymbol{x})$ is a global trend function which describes the large-scale model effects due to the inputs, and $w_i^c(\boldsymbol{x})$ is a weakly stationary process which expresses the local residual variation.

The global variation $s_i^c(\boldsymbol{x})$ is often driven by a relatively small subset of the inputs, known as the *active inputs* (Craig et al., 2001; Goldstein and Rougier, 2008). Different outputs may have different active inputs; we denote the set of active inputs for $F_i^c(\boldsymbol{x})$ as $\mathcal{X}_{[i]}^A$, and we write $\boldsymbol{x}_{[i]}$ to denote the corresponding 'active' sub-vector of $\boldsymbol{x}$, so that (2) becomes

$$f_i^c(\boldsymbol{x}) = s_i^c(\boldsymbol{x}_{[i]}) + u_i^c(\boldsymbol{x}_{[i]}) + v_i^c(\boldsymbol{x}), \tag{3}$$

where the global trend is now a function of the active inputs only, $u_i^c(\boldsymbol{x}_{[i]})$ is a weakly stationary residual process in $\boldsymbol{x}_{[i]}$, and $v_i^c(\boldsymbol{x})$ is an additional uncorrelated "nugget" process to account for any remaining variation. When the number of active inputs is relatively small and the variation of $v_i^c(\boldsymbol{x})$ is low, this representation substantially reduces the dimension of the design computations, whilst having only a small effect on the accuracy of the final results. Regardless of the form of $s_i^c(\boldsymbol{x}_{[i]})$, the active inputs give structure to our emulator by identifying the important model inputs, simplifying the search for informative designs.

Often the emulator trend is highly-structured and can be represented, for example, in the form

of a regression (Craig, Goldstein, Seheult, and Smith, 1996, 1997). In this case, we write (3) as

$$f_i^c(\boldsymbol{x}) = \sum_{j=1}^{p_i} \beta_{ij}^c g_{ij}(\boldsymbol{x}_{[i]}) + u_i^c(\boldsymbol{x}_{[i]}) + v_i^c(\boldsymbol{x}), \tag{4}$$

where $\boldsymbol{\beta}_i^c = (\beta_{i1}^c, \ldots, \beta_{ip_i}^c)$ are unknown scalars, and $\boldsymbol{g}_i(\boldsymbol{x}_{[i]}) = (g_{i1}(\boldsymbol{x}_{[i]}), \ldots, g_{ip_i}(\boldsymbol{x}_{[i]}))$ are known deterministic functions of the inputs. It is reasonable to assume that any quantitative global effects of the input parameters present on the coarse simulator may persist on the accurate simulator, albeit in a modified form. Writing the emulator trend as a regression form is a natural mechanism for exploiting this relationship. Additionally, the number of available runs will be heavily restricted for the emulation of the accurate simulator, and so the coverage of the input space will be poor. This results in the information contained in the accurate emulator being principally determined by the global model effects, as any local stochastic process components will be only weakly informative away from the few chosen model evaluations. Further, regression forms such as (4) produce enormous computational savings for many of the tasks for which the emulator will be used in practice.

Given the emulator form (4), and following the Bayes linear approach, our prior uncertainty about $F_i^c(\boldsymbol{x})$ can be characterised by the prior mean and variance as follows

$$\begin{aligned}
\mathrm{E}\left[f_i^c(\boldsymbol{x})\right] &= \boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{E}\left[\boldsymbol{\beta}_i^c\right] + \mathrm{E}\left[u_i^c(\boldsymbol{x}_{[i]})\right] + \mathrm{E}\left[v_i^c(\boldsymbol{x})\right], \\
\mathrm{Var}\left[f_i^c(\boldsymbol{x})\right] &= \boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{Var}\left[\boldsymbol{\beta}_i^c\right] \boldsymbol{g}_i(\boldsymbol{x}) + \mathrm{Var}\left[u_i^c(\boldsymbol{x}_{[i]})\right] + \mathrm{Var}\left[v_i^c(\boldsymbol{x})\right],
\end{aligned}$$

where we consider $\{\boldsymbol{\beta}_i^c, u_i^c(\boldsymbol{x}_{[i]}), v_i^c(\boldsymbol{x})\}$ as independent a priori, and we construct emulators separately for each component of $F^c(\boldsymbol{x})$.

The first stage in emulator construction is to determine the form of the global trend function $s_i^c(\boldsymbol{x}_{[i]})$, for each $F_i^c(\boldsymbol{x})$. This requires the identification of the active inputs $\boldsymbol{x}_{[i]}$, the determination of appropriate basis functions $\boldsymbol{g}_i(\cdot)$, and the quantification of $\mathrm{E}\left[\boldsymbol{\beta}_i^c\right]$ and $\mathrm{Var}\left[\boldsymbol{\beta}_i^c\right]$ for each emulator $f_i^c(\boldsymbol{x})$. If appropriate expert information is available on the simulator, then we can specify the global trend directly, and the prior trend function can be updated by evaluations of $F^c(\boldsymbol{x})$. In the absence of expert information, alternative methods for determining the global trend include, for example, using information from earlier versions of the computer simulator, basing the global trend on leading terms from series expansions of simplified versions of the underlying mathematical equations, constructing the trend function by regression modelling, or by using main effects plots based on many coarse model evaluations averaged over sub-collections of inputs (see Craig et al., 1998).

The second component, $u_i^c(\boldsymbol{x}_{[i]})$, is a weakly stationary residual process in $\boldsymbol{x}_{[i]}$ which describes the portion of residual variation of the emulator which is dependent on $\boldsymbol{x}_{[i]}$. We specify a prior covariance structure over $u_i^c(\boldsymbol{x}_{[i]})$. The prior form used for our example analysis is the Gaussian covariance function

$$\mathrm{Cov}\left[u_i^c(\boldsymbol{x}_{[i]}), u_i^c(\boldsymbol{x}'_{[i]})\right] = \sigma_{u_i}^2 \exp\left[-\sum_j \left(\frac{\boldsymbol{x}_{[i]j} - \boldsymbol{x}'_{[i]j}}{\theta_{ij}}\right)^2\right],$$

where $\sigma_{u_i}^2$ is the point variance, and θ_{ij} is the correlation length. There is an extensive literature on appropriate choices for the correlation function (Santner et al., 2003, Chapter 2), and the emulation and design approaches we describe apply equally to all choices of correlation form.

The process $v_i^c(\boldsymbol{x})$, also known as the *nugget*, expresses all remaining variation in $F_i^c(\boldsymbol{x})$ that cannot be explained by $\boldsymbol{x}_{[i]}$ alone. The information contained in v_i^c is typically weak and unstructured compared to s_i^c or u_i^c, and so we model $v_i^c(\boldsymbol{x})$ as uncorrelated noise with $\mathrm{Var}\left[v_i^c\right] = \sigma_{v_i}^2$. This is reasonable at the design stage as there is typically no discernible structure within the remaining inputs that is informative for subsequent design calculations. We consider the variances of the two residual processes to be proportional to the overall residual variance σ_i^2 of the simulator about the emulator trend. Thus, given σ_i^2, we write $\sigma_{u_i}^2 = (1-\delta_i)\sigma_i^2$ and $\sigma_{v_i}^2 = \delta_i\sigma_i^2$, for some typically small value of δ_i.

Finally, we assess the hyper-parameters $\Theta_i = \{\sigma_i^2, \theta_{ij}, \delta_i\}$ of our covariance specifications for u_i^c and v_i^c. Again, there are many methods for making such assessments (see for example Craig et al., 1998).

2.4 Linking the Coarse and Accurate Emulators

We consider that $F^c(\boldsymbol{x})$ is sufficiently informative for $F^a(\boldsymbol{x})$ that it provides a basis for building an informative prior specification for $f^a(\boldsymbol{x})$. We define the emulator residuals as $w_i^c(\boldsymbol{x}) = u_i^c(\boldsymbol{x}_{[i]}) + v_i^c(\boldsymbol{x})$, and similarly for $w_i^a(\boldsymbol{x})$. Our emulator for component i of $F^a(\boldsymbol{x})$ as

$$f_i^a(\boldsymbol{x}) = \sum_{j=1}^{p_i} \beta_{ij}^a g_{ij}(\boldsymbol{x}_{[i]}) + \beta_{w_i}^a w_i^c(\boldsymbol{x}) + w_i^a(\boldsymbol{x}), \tag{5}$$

where the trend has identical structure to that of $f_i^c(\boldsymbol{x})$ but with different coefficients, and we introduce a multiple of the coarse residuals $\beta_{w_i}^a w_i^c(\boldsymbol{x})$, and a new residual term $w_i^a(\boldsymbol{x})$ unique to the accurate emulator which absorbs any additional structure in the active inputs or any additional effects attributable to inputs not previously deemed active. We consider $w_i^a(\boldsymbol{x})$ to be independent of $\{\beta_{ij}^a, \beta_{w_i}^a, w_i^c(\boldsymbol{x})\}$ a priori.

Under this construction, we consider that the global model effects persist across the two simulators. However, the relative magnitudes of these effects may change differentially as we change simulators. Therefore we consider the magnitude of each coefficient in $f_i^a(\boldsymbol{x})$ to be an unknown multiple of the corresponding coarse coefficient, so that $\beta_{ij}^a = \rho_{ij}\beta_{ij}^c$ and $\beta_{w_i}^a = \rho_{w_i}$. This induces the following form for the accurate emulator

$$f_i^a(\boldsymbol{x}) = \sum_{j=1}^{p_i} \rho_{ij}\beta_{ij}^c g_{ij}(\boldsymbol{x}_{[i]}) + \rho_{w_i} w_i^c(\boldsymbol{x}) + w_i^a(\boldsymbol{x}), \tag{6}$$

and we write $\boldsymbol{\rho}_i = (\rho_{i0}, \ldots, \rho_{ip_i}, \rho_{w_i})$ to denote the vector of scaling parameters. Thus we synthesise information from the coarse emulator with judgements about the parameters $\boldsymbol{\rho}_i$ to construct a prior emulator for $F^a(\boldsymbol{x})$. An analogous approach when we are able to construct an appropriate sequence of physical experiments is described in Reese, Wilson, Hamada, Martz, and Ryan (2004).

Kennedy and O'Hagan (2000) propose an alternative formulation where the accurate emulator is linked directly to the coarse simulator

$$f_i^a(\boldsymbol{x}) = \rho_i F_i^c(\boldsymbol{x}) + w_i^a(\boldsymbol{x}), \tag{7}$$

where ρ_i is a single unknown scaling factor. This construction is a special case of (6) obtained by specifying the same prior mean ρ for each element of $\boldsymbol{\rho}_i$ and a correlation of one between each pair of elements. However, the emulator form (6) allows for more prior flexibility by not forcing a single multiplier across the components of the accurate emulator. This single multiplier approach is generalised in Qian, Seepersad, Joseph, Allen, and Wu (2006), and Qian and Wu (2008) by introducing a dependence of ρ on $\boldsymbol{x}$ such that

$$f_i^a(\boldsymbol{x}) = \rho_i(\boldsymbol{x}) F_i^c(\boldsymbol{x}) + w_i^a(\boldsymbol{x}),$$

where the form of $\rho_i(\boldsymbol{x})$ is a linear relationship or a Gaussian process, thus allowing the scaling factor to change throughout the input space. This form is difficult to design for directly. However, the design method we propose in Section 3 explores the main axes of variation for each $F_i^c(\boldsymbol{x})$ and so will be informative for any such modified form.

In general, our uncertainty judgements about $\boldsymbol{\rho}_i$ and $w_i^a(\boldsymbol{x})$ will be problem-specific. Our design methodology is applicable to any valid specifications. For illustration, we now describe the simplest form that these judgements might take. First, we consider $w_i^a(\boldsymbol{x})$ to be second-order stationary with

$$\begin{aligned} \mathrm{E}\left[w_i^a(\boldsymbol{x})\right] &= 0, \\ \mathrm{Var}\left[w_i^a(\boldsymbol{x})\right] &= \sigma_{w_i^a}^2. \end{aligned} \tag{8}$$

The simplest general parametrisation of prior beliefs derives from considering there to be no systematic biases between the models, a priori, which implies

$$\mathrm{E}\left[\rho_{ij}\right] = 1. \tag{9}$$

We specify the covariance matrix of $\boldsymbol{\rho}_i$ via two constants, σ_{ρ_i} and r_i such that

$$\begin{aligned} \mathrm{Var}\left[\rho_{ij}\right] &= \sigma^2_{\rho_i}, & (10)\\ \mathrm{Corr}\left(\rho_{ij}, \rho_{ik}\right) &= r_i,\ j \neq k, & (11) \end{aligned}$$

where $\sigma_{\rho_i} \geq 0$ and $r_i \in [-1, 1]$. While this is a simple form for $\mathrm{Var}\left[\boldsymbol{\rho}_i\right]$, by increasing the value of $\sigma^2_{\rho_i}$ we can relax the relationship between the two simulators, and by changing the value of r_i we can move from prior beliefs that the differences between the two emulators are independent across components of the global trend, to a model of perfect correlation having only a single effective multiplier corresponding to equation (7). More complex belief specifications can be used when we have prior information relevant to these judgements

Given the emulator form (5), our prior beliefs about $f^a_i(\boldsymbol{x})$ are expressed as

$$\begin{aligned} \mathrm{E}\left[f^a_i(\boldsymbol{x})\right] &= \boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{E}\left[\boldsymbol{\beta}^a_i\right] + \mathrm{E}\left[\beta^a_{w_i} w^c_i(\boldsymbol{x})\right] + \mathrm{E}\left[w^a_i(\boldsymbol{x})\right] \\ &= \sum_{j=1}^{p_i} \left(\mathrm{E}\left[\rho_{ij}\beta^c_{ij}\right] g_{ij}(\boldsymbol{x}_{[i]})\right) + \mathrm{E}\left[\rho_{w_i} w^c_i(\boldsymbol{x})\right] + \mathrm{E}\left[w^a_i(\boldsymbol{x})\right], \end{aligned} \tag{12}$$

$$\begin{aligned} \mathrm{Var}\left[f^a_i(\boldsymbol{x})\right] &= \boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{Var}\left[\boldsymbol{\beta}^a_i\right] \boldsymbol{g}_i(\boldsymbol{x}) + \mathrm{Var}\left[\beta^a_{w_i} w^c_i(\boldsymbol{x})\right] \\ &\quad + \mathrm{Var}\left[w^a_i(\boldsymbol{x})\right] + 2\boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{Cov}\left[\boldsymbol{\beta}^a_i, \beta^a_{w_i} w^c_i(\boldsymbol{x})\right] \\ &= \sum_{j=1}^{p_i}\sum_{k=1}^{p_i} \left(g_{ij}(\boldsymbol{x}_{[i]}) g_{ik}(\boldsymbol{x}_{[i]}) \mathrm{Cov}\left[\rho_{ij}\beta^c_{ij}, \rho_{ik}\beta^c_{ik}\right]\right) \\ &\quad + \mathrm{Var}\left[\rho_{w_i} w^c_i(\boldsymbol{x})\right] + \mathrm{Var}\left[w^a_i(\boldsymbol{x})\right] \\ &\quad + 2\sum_{j=1}^{p_i} \left(g_{ij}(\boldsymbol{x}_{[i]}) \mathrm{Cov}\left[\rho_{ij}\beta^c_{ij}, \rho_{w_i} w^c_i(\boldsymbol{x})\right]\right). \end{aligned} \tag{13}$$

We can also calculate the differences between simulators $d_i(\boldsymbol{x}) = F^a_i(\boldsymbol{x}) - F^c_i(\boldsymbol{x})$, for each point $\boldsymbol{x}$ in a design which is evaluated on both the coarse and accurate simulators. Beliefs about $d_i(\boldsymbol{x})$ are then

$$\begin{aligned} \mathrm{E}\left[d_i(\boldsymbol{x})\right] &= 0, \\ \mathrm{Var}\left[d_i(\boldsymbol{x})\right] &= \boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{Var}\left[\boldsymbol{\beta}^a_i - \boldsymbol{\beta}^c_i\right] \boldsymbol{g}_i(\boldsymbol{x}) + \mathrm{Var}\left[(\beta^a_{w_i} - 1) w^c_i(\boldsymbol{x})\right] \\ &\quad + \mathrm{Var}\left[w^a_i(\boldsymbol{x})\right] + 2\boldsymbol{g}_i(\boldsymbol{x})^T \mathrm{Cov}\left[\boldsymbol{\beta}^a_i - \boldsymbol{\beta}^c_i, (\beta^a_{w_i} - 1) w^c_i(\boldsymbol{x})\right]. \end{aligned} \tag{14}$$

The form of the emulator (6) can be simplified assuming that the $\boldsymbol{\beta}_i^c$ are known and then absorbing their values into the basis functions themselves. Due to the cheapness of $F^c(\boldsymbol{x})$, this assumption is a reasonable action since our choice of design $\boldsymbol{X}^c$ will be approximately orthogonal, $\boldsymbol{X}^c$ will be sufficiently large that our estimates for $\boldsymbol{\beta}_i^c$ can be obtained to a good degree of precision, and values of $\boldsymbol{\beta}_i^c$ will be largely unaffected by alternative choices of $\boldsymbol{X}^c$. Furthermore, given ample model runs over $\boldsymbol{X}^c$ we eliminate much of the uncertainty about $\boldsymbol{\beta}_i^c$ and, pragmatically, any remaining variation in $\boldsymbol{\beta}_i^c$ is negligible in comparison to the other unresolved uncertainties associated with $F^a(\boldsymbol{x})$.

Under this assumption, the values of $w_i^c(\boldsymbol{x})$ are also known at input points taken from $\boldsymbol{X}^c$ since $F_i^c(\boldsymbol{x})$ has been evaluated at these locations. Therefore, we can reduce expressions (12) and (13) to a form where only $\boldsymbol{\rho}_i$ and $w_i^a(\boldsymbol{x})$ are uncertain quantities, giving

$$\begin{aligned} \mathrm{E}\left[f_i^a(\boldsymbol{x})\right] &= \boldsymbol{b}_i(\boldsymbol{x})^T \mathrm{E}\left[\boldsymbol{\rho}_i^+\right] + \mathrm{E}\left[w_i^a(\boldsymbol{x})\right], \qquad (15) \\ \mathrm{Var}\left[f_i^a(\boldsymbol{x})\right] &= \boldsymbol{b}_i(\boldsymbol{x})^T \mathrm{Var}\left[\boldsymbol{\rho}_i^+\right] \boldsymbol{b}_i(\boldsymbol{x}) + \mathrm{Var}\left[w_i^a(\boldsymbol{x})\right] \\ &\quad + 2\boldsymbol{b}_i(\boldsymbol{x})^T \mathrm{Cov}\left[\boldsymbol{\rho}_i^+, w_i^a(\boldsymbol{x})\right] \qquad (16) \end{aligned}$$

where $\boldsymbol{\rho}_i^+ = (\boldsymbol{\rho}_i, \rho_{w_i})$ and we define $\boldsymbol{b}_i(\boldsymbol{x})$ as

$$\boldsymbol{b}_i(\boldsymbol{x}) = (\beta_{i0}^c g_{i0}(\boldsymbol{x}), \ldots, \beta_{ip_i}^c g_{ip_i}(\boldsymbol{x}), w_i^c(x)). \qquad (17)$$

These simplifications substantially reduce the complexity of the emulation and design calculations while, usually, having little effect on the choice of optimal design. Alternatively, if we do not wish to make such assumptions we can perform the subsequent design calculations in generality using equations (12) and (13).

2.5 Tuning the Emulator for the Accurate Simulator

The purpose of tuning the prior accurate emulator is to obtain some preliminary information about the change in behaviour which occurs when we move from the coarse simulator to the accurate. Specifically, we wish to learn about the degree of association between $F^c(\boldsymbol{x})$ and $F^a(\boldsymbol{x})$ via the multilevel parameters $\boldsymbol{\tau}_i = (\sigma_{\rho_i}, r_i, \sigma_{w_i^a})$ as defined in (10), (11), and (8). To do so, we construct a small design via the methods described in Section 3, and evaluate it on both the coarse and accurate simulators. Using these pairs of simulator evaluations, we construct the simulator differences $d_i(\boldsymbol{x}_j)$ for each point $\boldsymbol{x}_j$ in the tuning design. By treating $\boldsymbol{\beta}^c$ as known and constructing the tuning design only from points within the original coarse design as discussed in Section 2.4, we eliminate the

variation in $\boldsymbol{\beta}_i^c$ and $w_i^c(x)$, which allows (14) to be re-expressed as

$$\begin{aligned} \mathrm{Var}\left[d_i(\boldsymbol{x})\right] = \mathrm{E}\left[d_i(\boldsymbol{x})^2\right] &= \boldsymbol{b}_i(\boldsymbol{x})^T \mathrm{Var}\left[\boldsymbol{\rho}_i\right] \boldsymbol{b}_i(\boldsymbol{x}) + \sigma^2_{w_i^a}, \\ &= \sigma^2_{\rho_i} \phi_i(\boldsymbol{x}) + \sigma^2_{\rho_i} r_i \psi_i(\boldsymbol{x}) + \sigma^2_{w_i^a}, \end{aligned} \tag{18}$$

where we define $\boldsymbol{b}_i(\boldsymbol{x})$ as in (17), and where $\phi_i(\boldsymbol{x}) = \sum_j \boldsymbol{b}_{ij}(\boldsymbol{x})^2$, and $\psi_i(\boldsymbol{x}) = \sum_{j \neq k} \boldsymbol{b}_{ij}(\boldsymbol{x}) \boldsymbol{b}_{ik}(\boldsymbol{x})$.

Thus the expected squared simulator difference can be characterised entirely by the multilevel parameters $\boldsymbol{\tau}_i = \{\sigma_{\rho_i}, r_i, \sigma_{w_i^a}\}$. To tune our beliefs about these quantities, we use the values of $d_i(\boldsymbol{x})^2$ observed from the tuning runs to learn about $\boldsymbol{\tau}_i$. Two possible approaches to learning about $\boldsymbol{\tau}_i$ are weighted least squares fitting or Bayes linear variance adjustment.

For the weighted least squares approach, give the observed values of $d_i(\boldsymbol{x}_j)^2$ and their corresponding input values the tuning of beliefs about $\boldsymbol{\tau}_i$ reduces to the linear regression problem of (18) in the unknown coefficients $\boldsymbol{\tau}_i$. We can then estimate $\boldsymbol{\tau}_i$ via weighted least squares, weighting each observation by $1/\mathrm{Var}\left[d_i(\boldsymbol{x}_j)^2\right]$ giving the estimates $\hat{\boldsymbol{\tau}}_i$ for the tuning parameters. The Bayes linear approach is appropriate when we are able to make informed second-order prior specifications for $\boldsymbol{\tau}_i$ and $d_i(\boldsymbol{x})^2$. We then use the observed $d_i(\boldsymbol{x}_j)^2$ to perform a Bayes linear adjustment of the prior values for each of the variance terms (Goldstein and Wooff, 2007, Chapter 8). Both approaches provide a means of obtaining order-of-magnitude assessments for the values of $\boldsymbol{\tau}_i$. The results of the tuning process can also serve as a validation of the structure of the specification for $\boldsymbol{\tau}_i$; for example, if we determine that r_i is close to one, then the simpler single multiplier model of (7) may be appropriate.

In addition to tuning our beliefs about $\boldsymbol{\tau}_i$, the model differences also provide qualitative information about the simulator changes as we move from coarse to accurate. For instance, we can identify outputs whose variance change substantially. These outputs may not lend themselves to a simple multilevel emulation treatment since their behaviour changes substantially across the two simulators, making it difficult to use $F^c(\boldsymbol{x})$ to construct informative emulators and designs for $F^a(\boldsymbol{x})$. Any such outputs will be screened out of the subsequent analysis, and emulators for these outputs will be built individually from the accurate simulator alone. Alternatively, this may motivate the search for alternative coarsenings of the accurate simulator.

2.6 Updating the Emulator for the Accurate Simulator

Finally, we obtain a design over the simulator inputs appropriate for learning about the accurate emulator itself. We evaluate the accurate simulator at these design points and combine them

with the accurate simulator evaluations made during the tuning process, to obtain the collection of accurate simulator runs $\boldsymbol{F}^a$. We update the tuned prior accurate emulator, $f^a(\boldsymbol{x})$, by this collection of simulator runs to obtain the adjusted accurate emulator $f^a_{\boldsymbol{F}^a}(\boldsymbol{x})$.

Since our beliefs about $F^a(\boldsymbol{x})$ are expressed via mean and variance specifications for $f^a(\boldsymbol{x})$, we update these beliefs by Bayes linear adjustment (see Goldstein and Wooff, 2007). Using only mean, variance and covariance specifications, the adjusted expectation and variance for the random vector $\boldsymbol{y}$, given random vector $\boldsymbol{z}$, are as follows

$$\begin{aligned} \mathrm{E}_{\boldsymbol{z}}[\boldsymbol{y}] &= \mathrm{E}[\boldsymbol{y}] + \mathrm{Cov}[\boldsymbol{y},\boldsymbol{z}]\,\mathrm{Var}[\boldsymbol{z}]^{-1}(\boldsymbol{z}-\mathrm{E}[\boldsymbol{z}]), \qquad (19)\\ \mathrm{Var}_{\boldsymbol{z}}[\boldsymbol{y}] &= \mathrm{Var}[\boldsymbol{y}] - \mathrm{Cov}[\boldsymbol{y},\boldsymbol{z}]\,\mathrm{Var}[\boldsymbol{z}]^{-1}\,\mathrm{Cov}[\boldsymbol{z},\boldsymbol{y}]. \qquad (20)\end{aligned}$$

Applying the Bayes linear adjustment formulae, we obtain the following general expressions for the adjusted expectation and variance of $f^a_i(\boldsymbol{x})$ given the accurate simulator runs $\boldsymbol{F}^a$,

$$\begin{aligned} \mathrm{E}_{\boldsymbol{F}^a}[f^a_i(\boldsymbol{x})] &= \boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{E}_{\boldsymbol{F}^a}[\boldsymbol{\beta}^a_i] + \mathrm{E}_{\boldsymbol{F}^a}\left[\beta^a_{w_i}w^c_i(\boldsymbol{x})\right] + \mathrm{E}_{\boldsymbol{F}^a}[w^a_i(\boldsymbol{x})],\\ \mathrm{Var}_{\boldsymbol{F}^a}[f^a_i(\boldsymbol{x})] &= \boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{Var}_{\boldsymbol{F}^a}[\boldsymbol{\beta}^a_i]\,\boldsymbol{g}_i(\boldsymbol{x}) + \mathrm{Var}_{\boldsymbol{F}^a}\left[\beta^a_{w_i}w^c_i(\boldsymbol{x})\right] + \mathrm{Var}_{\boldsymbol{F}^a}[w^a_i(\boldsymbol{x})]\\ &\quad +2\boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{Cov}_{\boldsymbol{F}^a}\left[\boldsymbol{\beta}^a_i,\beta^a_{w_i}w^c_i(\boldsymbol{x})\right] + 2\boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{Cov}_{\boldsymbol{F}^a}[\boldsymbol{\beta}^a_i,w^a_i(\boldsymbol{x})]\\ &\quad +2\mathrm{Cov}_{\boldsymbol{F}^a}\left[\beta^a_{w_i}w^c_i(\boldsymbol{x}),w^a_i(\boldsymbol{x})\right] \qquad (21)\end{aligned}$$

Thus the adjusted accurate emulator can be expressed in terms of the adjusted accurate coefficients and residual processes, all of which can be obtained from application of (19) and (20) to the prior belief statements and the specifications described in Sections 2.4.

If we choose to adopt the simplifying assumptions and use the simplified forms (16) and (15), then the adjusted expectation and variance of the accurate emulator is written as

$$\begin{aligned} \mathrm{E}_{\boldsymbol{F}^a}[f^a_i(\boldsymbol{x})] &= \boldsymbol{b}_i(\boldsymbol{x})^T\mathrm{E}_{\boldsymbol{F}^a}\left[\boldsymbol{\rho}^+_i\right] + \mathrm{E}_{\boldsymbol{F}^a}[w^a_i(\boldsymbol{x})],\\ \mathrm{Var}_{\boldsymbol{F}^a}[f^a_i(\boldsymbol{x})] &= \boldsymbol{b}_i(\boldsymbol{x})^T\mathrm{Var}_{\boldsymbol{F}^a}\left[\boldsymbol{\rho}^+_i\right]\boldsymbol{b}_i(\boldsymbol{x}) + \mathrm{Var}_{\boldsymbol{F}^a}[w^a_i(\boldsymbol{x})]\\ &\quad +2\boldsymbol{b}_i(\boldsymbol{x})^T\mathrm{Cov}_{\boldsymbol{F}^a}\left[\boldsymbol{\rho}^+_i,w^a_i(\boldsymbol{x})\right]. \qquad (22)\end{aligned}$$

3. BORDER-BLOCK DESIGNS FOR MULTISCALE EMULATORS

3.1 Border-block structure in the design problem

Our aim is to design a set of n runs of the accurate simulator by selecting input choices $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ at which to evaluate $F^a(\boldsymbol{x})$ where n will be small since $F^a(\boldsymbol{x})$ is expensive to evaluate. To identify an effective design, we use structural information from the coarse emulators to decompose

the design problem into several smaller and simpler problems. For example, suppose that the simulator has only two outputs, y_1 and y_2 say, with coarse emulators $f_1^c(\boldsymbol{x})$ and $f_2^c(\boldsymbol{x})$ which are expressed as $f_1^c(\boldsymbol{x}) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(\boldsymbol{x})$ and $f_2^c(\boldsymbol{x}) = s_2^c(x_1, x_2, x_5, x_6) + w_2^c(\boldsymbol{x})$ where $s_i^c(\cdot)$ is a polynomial trend function. Thus $\{x_1, x_2, x_3, x_4\}$ and $\{x_1, x_2, x_5, x_6\}$ are the active input sets for f_1^c and f_2^c respectively.

The structure exposed by the sets of active inputs shows that the design decomposes into the smaller active subspaces. Reducing the dimensionality of the design space is beneficial as we can obtain better coverage of lower-dimensional spaces given a fixed design size, and we can more easily search over lower-dimensional spaces to identify efficient designs. By further examination of the active sets $\{x_1, x_2, x_3, x_4\}$ and $\{x_1, x_2, x_5, x_6\}$, we see that the inputs $\{x_1, x_2\}$ are active in both emulators, whilst $\{x_3, x_4\}$ and $\{x_5, x_6\}$ are unique to f_1^c and f_2^c respectively. Using the terminology of Bates, Buck, Riccomagno, and Wynn (1996), $\mathcal{X}_B^A = \{x_1, x_2\}$ is known as the set of *border* inputs, which can potentially interact freely with any other active inputs, whereas $\mathcal{X}_1^A = \{x_3, x_4\}$ and $\mathcal{X}_2^A = \{x_5, x_6\}$ are sets of *block* inputs which can potentially interact only with inputs within the same block and with the inputs in the border. Thus $\mathcal{X}_B^A$ and the $\mathcal{X}_i^A$, for $i = 1, \ldots, b$, are disjoint and exhaustive and thus partition the set of active inputs $\mathcal{X}^A$. Introducing $\mathcal{X}^U$ as the set of inactive inputs across all outputs, allows us to partition the entire set of inputs, $\mathcal{X}$, to the computer simulator. Furthermore, for any suggested set of design choices for $\mathcal{X}_B^A = \{x_1, x_2\}$, we can consider the efficient selection of design choices for $\mathcal{X}_1^A = \{x_3, x_4\}$ as a separate problem to that of the efficient selection for $\mathcal{X}_2^A = \{x_5, x_6\}$. This decomposition can be depicted in a graph such as that in Figure 1(a), where inputs are connected if they appear in the same active set.

[Figure 1 about here.]

Of course, with such a small example it is trivial to identify which inputs form the border and which the blocks. With larger problems, involving tens or hundreds of inputs, making such an identification by eye becomes substantially more complex. However, methods such as the algorithm of Zečevič and Šiljak (1994) can determine the necessary partitions automatically. The algorithm operates on the adjacency matrix of the graph associated with the active inputs and permutes the columns and rows of the matrix until it obtains a bordered block diagonal structure – see Figure 1(b) – the input partitions can then be read directly from the permuted matrix. For large sets of inputs, blocks may contain tens or hundreds of inputs and we may wish to repeat the decomposition algorithm on each block to further break it into a sub-border and sub-blocks.

3.2 Design generation

Given the partition of the inputs into active and inactive inputs, we generate the design over $\mathcal{X}$ by combining an informed design over the active inputs, $\mathcal{X}^A$, column-wise with an appropriate space-filling design over the inactive inputs, $\mathcal{X}^U$. We compare alternative choices of design for $\mathcal{X}^A$ with respect to a design criterion $C(\cdot)$. The choice of $C(\cdot)$ depends upon the purpose of the design and the future goals of the analysis. Each criterion is the sum of the expected gain in information, appropriately defined, for each selected output on the accurate simulator. Design criteria are discussed in Section 3.3.

We propose a method for the construction of a design over $\mathcal{X}^A$ that exploits the border-block structure. By partitioning the active input space, we construct the design for $\mathcal{X}^A$ by combining smaller sub-designs for each of the border and the blocks. In the example, we decomposed the input space into three subspaces $\{x_1, x_2\}$, $\{x_3, x_4\}$ and $\{x_5, x_6\}$. Therefore, we construct an appropriate design for each of these 2-dimensional spaces and combine them column-wise to form our final design. In order to apply this methodology, we need to be able to construct larger designs by augmenting an existing design with additional columns. For this purpose, Latin hypercubes are useful since they can be augmented in such a way and yet still preserve their properties.

To begin the process, we construct an initial design for the border inputs. In our example, this is the set $\mathcal{X}_B^A = \{x_1, x_2\}$. For our initial design, we generate many possible Latin hypercube designs over $\mathcal{X}_B^A$ and then choose the design which maximises the minimum inter-point distance (Morris and Mitchell, 1995). Given the design for $\mathcal{X}_B^A$, we now consider designs for each $\mathcal{X}_i^A$. Since we can design over each $\mathcal{X}_i^A$ independently once the design for $\mathcal{X}_B^A$ is fixed, the order in which we consider the $\mathcal{X}_i^A$ is unimportant. With a design for $\mathcal{X}_B^A$, any design over $\mathcal{X}_i^A$ will provide sufficient information to determine the expected information for selected simulator outputs with active inputs in $\mathcal{X}_i^A \cup \mathcal{X}_B^A$ according to criterion $C(\cdot)$. In the example, having fixed a design over $\mathcal{X}_B^A = \{x_1, x_2\}$, then, for any choice of design over $\mathcal{X}_1^A = \{x_3, x_4\}$, we can determine the gain in information for $f_1^a(\boldsymbol{x})$, and so the block designs can be evaluated with respect to criterion $C(\cdot)$. For each block $\mathcal{X}_i^A$, we generate a number of Latin hypercube candidate designs over $\mathcal{X}_i^A$ each of which is then augmented column-wise by our chosen design for $\mathcal{X}_B^A$ and is evaluated with respect to $C(\cdot)$. The design which maximises this criterion value is then selected and the process is repeated for each of the remaining blocks.

With an initial design over $\mathcal{X}^A$, we iteratively refine the design to improve its overall perfor-

mance. For each of the sub-designs over $\mathcal{X}_B^A$ and the $\mathcal{X}_i^A$, we generate many new potential candidate sub-designs by the above methods. We then consider the effect of replacing the existing sub-design with any of the candidates. If the replacement results in an improvement in performance with respect to $C(\cdot)$, then we accept the change. We move through all of the design subspaces, each time seeking preferable sub-designs, with each iteration attempting to swap out the sub-designs in pursuit of a better overall design. To prevent confounding of the inputs by accidental selection of similar designs for different blocks we check the orthogonality of the resulting design before accepting a change in sub-design, and reject a potential change if it would introduce pairwise correlations above a threshold level of 0.85. The iterative refinement stops when our search procedures result in little or no further improvement in $C(\cdot)$. If each block has been further reduced into a sub-border and sub-blocks, then optimisation for each block is also structured according to this procedure.

3.3 Design criteria

3.3.1. Designing to tune $f^a(\boldsymbol{x})$. Tuning is described in Section 2.5 and is the process by which we learn about the values of the multilevel parameters $\boldsymbol{\tau}_i = (\sigma_{\rho_i}, r_i, \sigma_{w_i^a})$ from a small number of observed simulator differences. Designing for this goal is focussed on selecting the input points that result in simulator differences which are effective at reducing the uncertainty associated with $\boldsymbol{\tau}_i$. We proposed two possible methods for tuning the prior accurate emulator – weighted least squares, and a Bayes linear variance update. We now describe appropriate design criteria for each method.

Weighted Least Squares tuning. Given the observed values of the squared simulator differences, $k_{ij} = d_i(\boldsymbol{x}_j)^2$, over the design points $\boldsymbol{x}_j$ for $j = 1, \ldots, n$, the tuning problem reduces to the linear regression of (18) with coefficients $\boldsymbol{\tau}_i$ which are then estimated via weighted least squares giving the estimates $\hat{\boldsymbol{\tau}}_i$. A good tuning design is therefore one which substantially reduces the variance in $\hat{\boldsymbol{\tau}}_i$, suggesting a criterion of the form

$$C_{\text{WLS}} = \mathbf{tr}\{\text{Var}\,[\hat{\boldsymbol{\tau}}_i]\} = \mathbf{tr}\{\boldsymbol{A}_i \boldsymbol{W}_i \boldsymbol{A}_i^T\}, \tag{23}$$

averaged over all outputs. We define $\hat{\boldsymbol{\tau}}_i$ as the estimates of $\boldsymbol{\tau}_i$, $\boldsymbol{W}_i$ as the diagonal matrix of weights, $\boldsymbol{A}_i = (\boldsymbol{V}_i \boldsymbol{W}_i \boldsymbol{V}_i^T)^{-1} \boldsymbol{V}_i \boldsymbol{W}_i$, and $\boldsymbol{V}_i$ as the model matrix with jth row equal to $(1, \phi_i(\boldsymbol{x}_j), \psi_i(\boldsymbol{x}_j))$.

Bayes Linear tuning. The Bayes linear approach to tuning requires a second order prior specification for $\boldsymbol{\tau}_i$ and k_{ij}, and a specification of their covariance. Given this information, we use the observations k_{ij} to perform a Bayes linear variance adjustment of the prior values for each of the

multilevel parameters. An appropriate criterion related to the effectiveness of the update is the resolution of the Bayes linear adjustment of $\boldsymbol{\tau}_i$ by k_{ij} (Goldstein and Wooff, 2007). The resolution is a scale-free measure which quantifies the magnitude of the effect of an adjustment in reducing our uncertainties. The resolution value lies in the interval $[0, 1]$, where larger values indicate a greater proportion of the uncertainties have been resolved. The resolution of the adjustment of the random vector $\boldsymbol{X}$ by the random vector $\boldsymbol{D}$ is defined as

$$\mathrm{R}_{\boldsymbol{D}}(\boldsymbol{X}) = \frac{1}{\mathrm{r}_{\boldsymbol{X}}}\mathbf{tr}\{\mathrm{Var}\left[\boldsymbol{X}\right]^{-1}\mathrm{Cov}\left[\boldsymbol{X}, \boldsymbol{D}\right]\mathrm{Var}\left[\boldsymbol{D}\right]^{-1}\mathrm{Cov}\left[\boldsymbol{D}, \boldsymbol{X}\right]\},$$

where $\mathrm{r}_{\boldsymbol{X}}$ is the rank of the matrix $\mathrm{Var}\left[\boldsymbol{X}\right]$. Thus, our design criterion is

$$C_{\mathrm{BL}} = \mathrm{R}_{\boldsymbol{k}_i}(\boldsymbol{\tau}_i),$$

averaged over all chosen outputs where $\boldsymbol{k}_i$ is the vector of the k_{ij}.

3.3.2. Designing to learn about $F^a(\boldsymbol{x})$. At the final stage in the emulation process, we seek a design which has the greatest effect in reducing uncertainty about $F^a(\boldsymbol{x})$ as assessed by the adjusted variance of the accurate emulator given the simulator evaluations. This variance does not depend on the values of the model evaluations and follows from (21).

For small-sample designs, it will not be possible to reduce much of the uncertainty associated with the residuals from the accurate model over most of the input space, and most small-sample designs will not differ greatly in their performance at this task. Therefore, we focus our attention on variance reduction for $(\boldsymbol{\beta}_i^a, \beta_{w_i}^a w_i^c(\boldsymbol{x}))$, and design using a criterion based on

$$\begin{aligned}\mathrm{Var}_{\boldsymbol{F}_i^a}\left[f_i^a(\boldsymbol{x})\right] \;\simeq\; & \boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{Var}_{\boldsymbol{F}^a}\left[\boldsymbol{\beta}_i^a\right]\boldsymbol{g}_i(\boldsymbol{x}) + \mathrm{Var}_{\boldsymbol{F}^a}\left[\beta_{w_i}^a w_i^c(\boldsymbol{x})\right] \\ & +2\boldsymbol{g}_i(\boldsymbol{x})^T\mathrm{Cov}_{\boldsymbol{F}^a}\left[\boldsymbol{\beta}_i^a, \beta_{w_i}^a w_i^c(\boldsymbol{x})\right]. \end{aligned} \tag{24}$$

As we are concerned with reducing uncertainty in $F^a(\boldsymbol{x})$ across the whole input space, we choose a collection of input points, $\boldsymbol{X}$, which covers the space and evaluate equation (24) over $\boldsymbol{X}$. Thus we get an adjusted variance matrix which we collapse into a scalar using the trace for use as a design criterion

$$\begin{aligned} C_{\mathrm{L}} \;=\; & \mathbf{tr}\{\boldsymbol{G}_i^T\mathrm{Var}_{\boldsymbol{F}^a}\left[\boldsymbol{\beta}_i^a\right]\boldsymbol{G}_i\} + \mathbf{tr}\{\mathrm{Var}_{\boldsymbol{F}^a}\left[\beta_{w_i}^a w_i^c(\boldsymbol{X})\right]\} \\ & +2\,\mathbf{tr}\{\boldsymbol{G}_i^T\mathrm{Cov}_{\boldsymbol{F}^a}\left[\boldsymbol{\beta}_i^a, \beta_{w_i}^a w_i^c(\boldsymbol{x})\right]\}, \end{aligned} \tag{25}$$

where $\boldsymbol{X}$ is the grid of evaluation points, $\boldsymbol{G}_i$ is the matrix with jth row $\boldsymbol{g}_i(\boldsymbol{x}_j)$ corresponding to the jth design point $\boldsymbol{x}_j$.

The criterion (25) can be simplified when we have eliminated all variation in $\boldsymbol{\beta}_i^c$, evaluating the criterion over the coarse design at $\boldsymbol{X}^c$ and consequently further eliminating variation in $w^c(\boldsymbol{x})$ as discussed in Section 2.4. Under these assumptions, we can use the simplified form (22) to write the design criterion C_{L} as

$$C_{\mathrm{L}2} = \mathbf{tr}\{\mathrm{Var}_{\boldsymbol{F}^a}\left[\boldsymbol{\rho}_i^+\right] \boldsymbol{B}_i^*\}, \tag{26}$$

where $\boldsymbol{\rho}_i^+ = (\boldsymbol{\rho}_i, \rho_{w_i})$, and $\boldsymbol{B}_i^* = \boldsymbol{B}_i \boldsymbol{B}_i^T$ where $\boldsymbol{B}_i$ is the matrix with jth row $\boldsymbol{b}_i(\boldsymbol{x}_j)$ as defined in (17) for design point $\boldsymbol{x}_j$. We favour designs which give the greatest reduction in these criteria.

The criteria (25) and (26) are defined univariately for each selected output. To obtain an aggregate score, we sum or average the design criterion values over the individual outputs. As an extension, we could allow for different outputs to have different contributions to the overall combined criterion value. Possible weightings could include the principal variable loadings h_i to reflect the intrinsic importance of the outputs and their representativeness of outputs not included in the design calculation. Other choices could include measures of the quality of available observational data if we intend to calibrate the emulator, or utility scores to measure the value of accurate predictions for each output variable if prediction is our ultimate goal.

4. EXAMPLE: HYDROCARBON RESERVOIR MODEL

4.1 Model description

We illustrate our approach with a simulation of a hydrocarbon reservoir provided by Energy SciTech Ltd. The model is based on a $48 \times 26 \times 25$ grid. Each cell represents a region of subterranean rock with particular geological properties and containing varying proportions of oil, water and gas. The reservoir contains four producing wells which pump fluids out of the reservoir under fixed operating conditions, and a single injection well which injects gas into the reservoir to maintain pressure and production levels at the extraction wells.

The outputs of the model are time series of aspects of well behaviour such as pressures, production rates of oil, water and gas, cumulative production of oil, water, and gas, and ratio quantities including water cut and gas-oil ratio. For the purposes of this example, we focus on a single time point approximately eighteen months into the operation of the simulation. At this point there are a total of 65 outputs for consideration. The inputs to the model are a collection of twenty scalars which affect various geological properties. These inputs control quantities including permeability, porosity, transmissibility of the geological faults, critical oil saturation properties, and properties of the reservoir's aquifer.

4.2 Results

4.2.1. Coarse evaluations. To obtain a fast approximation we coarsened the vertical gridding of the model to just two layers and we used the 25-layer grid for the full simulator. The evaluation time for the coarse simulator was approximately 30 seconds compared with 35 minutes for the accurate simulator. The coarse model was then evaluated over a 1500 point Latin hypercube in all 20 inputs to provide us with our initial batch of coarse model runs.

4.2.2. Screen the outputs. Application of principal variable methods to the coarse simulator evaluations identified those outputs which accounted for the majority of the variation in the model runs. As there is substantial correlation among the outputs, the principal variable screening proved to be effective and the output collection could be well-represented by a relatively small subset. Overall, we consider that we can obtain a good representation of all 65 outputs using a subset of 15, as the proportion of variation explained by our chosen principal variables is at least 0.947 for each remaining output. The names of the chosen 15 variables are given in the first column of Table 1. The prefixes 'W1', 'W2', 'W3', 'W4' indicate that the variable corresponds to correspondingly numbered production well, whereas 'G1' corresponds to the single gas injector. The quantities measured at each of those wells are given by the variable suffix: 'pre' and 'bhp' correspond to two measures of pressure; 'wct' is the water cut; 'gor' is the gas-oil ratio; variables such as 'wat.tot' and 'oil.rt' correspond to the total water production, and the rate of oil production respectively; and finally 'gasinj.tot' is the total amount of gas injected.

4.2.3. Coarse emulation. Prior to emulation, the design was scaled so all inputs took the range $[-1, 1]$, and the outputs were scaled to have mean 0 and sample variance 1 over the design. The first stage in constructing an emulator of the form (4) is to identify the collection of active inputs $\mathcal{X}_{[i]}^{A}$ for each $F_i^c(\boldsymbol{x})$. The active inputs for each emulator were determined from the simulator runs, $\boldsymbol{F}^c$, by a stepwise model search using simple linear regression. At each stage, all terms involving a given input were removed from the model then the new model was fitted and the proportion of variation explained was evaluated and compared to that of the original model via the adjusted R^2 coefficient. The input which explained the least amount of output variation was then removed, and the process iterates until we have identified the best model using three to six inputs, which we consider to be $\mathcal{X}_{[i]}^{A}$. We chose this method for model selection as our aim here is to drive down the residual variance; there are a number of alternative choices for selection methods and criteria

(see for example Hocking, 1976), and our design and emulation methodologies are not tied to any particular approach.

The next stage in emulation is to choose the basis functions $\boldsymbol{g}_i(\cdot)$ for each F_i^c. Our design methodology is not tied to any specific form of $\boldsymbol{g}_i(\cdot)$, and so possible bases could include simple monomials, orthogonal polynomials, or more general basis function such as those found via generic screening methods (Welch, Buck, Sacks, Wynn, Mitchell, and Morris, 1992). For this example, we consider the maximal set of basis functions to be the monomial terms comprising an intercept, linear, quadratic, cubic and pairwise interaction terms in the active inputs. To select an appropriate representation of $F_i^c(\boldsymbol{x})$ using this basis, the saturated linear regression model over these terms is fitted to the coarse simulator runs, and we perform a stepwise selection removing a single term at each stage whose impact on the proportion of variation explained by the model was less than 1%. Adding any more active inputs to the emulators did not provide any substantial improvement in the quality of the OLS fit.

To quantify $\mathrm{E}\left[\boldsymbol{\beta}_i^c\right]$ and $\mathrm{Var}\left[\boldsymbol{\beta}_i^c\right]$, we fit the linear regression model in the basis functions $\boldsymbol{g}_i(\boldsymbol{x}_{[i]})$ by OLS. We then assign $\mathrm{E}\left[\boldsymbol{\beta}_i^c\right]$ to be the least squares regression estimate of the coefficients, and we consider $\mathrm{Var}\left[\boldsymbol{\beta}_i^c\right]$ to be the least squares variance of the corresponding estimates. Given a large number of evaluations of $F^c(\boldsymbol{x})$ over an approximately orthogonal design, the associated estimation errors will be negligible and these assessments will be reasonable order of magnitude posterior specifications.

Finally, we extract the emulator trend residuals and then apply the robust variogram methods of Cressie (1991) to determine appropriate values for Θ_i. We then use these values as plug-in estimates and adjust the residual processes in light of the observed residuals.

[Table 1 about here.]

A summary of the emulation is given in Table 1. The adequacy of the fit variesbetween the outputs, but most appear to have a moderate to good level of adjusted R^2. Well 1 oil production rate (W1.oil.rt) fared relatively poorly, and further investigation showed that we are unable to obtain a better fit using any number or combination of available inputs as the output behaviour could not be well-described by a simple functional form. The emulation revealed that of the 20 inputs only 8 were identified as being active for the outputs under consideration. Those inputs deemed inactive included all multipliers pertaining to the aquifer, four multipliers for fault transmissibilities and all multipliers for oil saturation properties.

4.2.4. Linking coarse and accurate emulator. In order to design for the tuning runs, we wish to construct prior accurate emulators of the form given in (6). Therefore, we need to make specifications for the prior expectations of the multilevel parameters $\boldsymbol{\tau}_i = (\sigma_{\rho_i}, r_i, \sigma_{w_i^a})$ as defined in (10), (11) and (8). At this stage we have no knowledge about how the two simulators may differ, so we make simple pragmatic choices. For σ_{ρ_i}, we assign the prior expectation to be $\mathrm{E}[\sigma_{\rho_i}] = 0.5$, which when combined with (9) corresponds to a prior belief that the model coefficients are unlikely to change sign between simulators. We assign a small correlation between the $\boldsymbol{\rho}_i$ by setting $\mathrm{E}[r_i] = 0.5$, and finally in the absence of any other information, we consider that $\mathrm{Var}[w_i^a(\boldsymbol{x})] = \mathrm{Var}[w_i^c(\boldsymbol{x})]$ resulting in $\mathrm{E}\left[\sigma_{w_i^a}\right] = \sigma_i$, the residual variance of the emulator trend.

[Table 2 about here.]

Since this is an illustrative example rather than an actual case study investigated in real time, a batch of 200 accurate model runs were available to allow us to check the validity of the prior emulator form (6). Fitting the model terms in the coarse emulators to these accurate simulator runs, we observe that the coefficients in the model trends do indeed change differentially rather than scaling uniformly (see for example Table 2) indicating that the extra prior flexibility provided by (6) is valuable. It should be noted that the problem of model choice, i.e. determining $\mathcal{X}^A$ and the $\boldsymbol{g}_i(\cdot)$, requires a substantially larger number of simulator runs than for estimation of the coefficients, when the form of model is given. Therefore, we require a large batch of available simulator evaluations to determine the form of the emulators, but far fewer are required to learn about the $\boldsymbol{\beta}^a$. This is a crucial benefit of the multilevel approach as the coarse simulator provides us with ample information to determine the emulator form, while the estimation of $\boldsymbol{\beta}^a$ can be performed from fewer carefully-chosen, accurate simulator runs.

[Figure 2 about here.]

4.2.5. Design for tuning. Partitioning the inputs into the border and blocks by application of the algorithm discussed in Section 3.1 gives the decomposition in Figure 2(a). The structure of the emulators is such that there is a three-dimensional border $\{k_x, k_y, P\}$, corresponding to permeability in the x and y directions, and porosity, and the remaining inputs fall into either a larger block containing three inputs, or two blocks containing a single input each. Since the sizes of these subsets are small relative to the number of overall active inputs, there are clear advantages to decomposing the design problem in this way.

We have three multilevel parameters requiring estimation to tune the emulators, and so we require at least 3 tuning runs. Evaluations of the accurate computer model are expensive and we do not wish to expend a substantial amount of resources at this stage in order to allow for ample evaluations for learning about the accurate emulator. We therefore choose to construct a tuning design of size 6 to allow an additional degree of freedom for each parameter. Following the general design procedure discussed in Section 3.2, we generated a design with a criterion value of 0.893 using C_{WLS} from (23). We used the WLS method for tuning rather than the Bayes linear approach as we did not want to confound the question of our ability to make informed prior judgements about the physical issues involved in coarsening reservoir simulators with the general question about the effectiveness of our methodology. The chosen design is presented in Table 3. We are free to make any choices in the remaining inputs, for example using a Latin hypercube. This design was then evaluated on both the coarse and accurate computer simulators.

[Table 3 about here.]

4.2.6. Tuning. Using the two sets of model runs, we construct the simulator differences and tune the multilevel parameters $\boldsymbol{\tau}_i$ using the WLS method of Section 2.5, weighting each of the model differences by its prior variance. We now examine our tuned values for $\boldsymbol{\tau}_i$, and perform a simple data analysis of the model differences to investigate the behaviour of the two functions. This was particularly insightful as it revealed that the outputs could be divided into three groups. The first group (labelled $(*)$ in Table 1) had coarse-accurate differences that were well-behaved and conformed reasonably well to the multilevel model. The second group (labelled $(+)$) consisted of variables which displayed substantially greater variation on the accurate simulator than on the coarse simulator – interestingly this group contains outputs which are all concerned with the production of water from the reservoir. The final group (labelled $(-)$) behaved in an opposite manner, having far less or even no variation on the accurate simulator compared to the coarse model.

The behaviour of the $(+)$ and $(-)$ variables is likely an artefact of the reservoir coarsening disrupting meaningful relationships between the two simulators for these variables. In these cases, it is likely that the coarsening of $F^c(\boldsymbol{x})$ was too severe, and that using a model at an intermediate level of accuracy would remedy this problem. Further investigation after the analysis suggested that coarsening to five layers appeared to eliminate this problem. To simplify our account, we suppose that we choose to stay with our original coarsening.

Therefore, for design purposes, we chose to screen out the (+) and (−) variables and to focus on the remainder. Since the coarse simulator is uninformative about the accurate simulator for these quantities, they convey little or no useful information relevant to the construction of informative designs for $F^a(\boldsymbol{x})$. These inputs are not discarded from the analysis, but will be further investigated using standard emulation techniques once runs of the accurate simulator have been performed. For the remaining quantities, the new choices for values of $\boldsymbol{\tau}_i$ were not substantially different from our prior specifications, therefore we shall treat our original choices as acceptable prior values at the next stage. The number of outputs is now further reduced from 15 to 7. However, the remaining 7 outputs explain at least 50% of the variation of the remainder, at least 75% of the variation of 52, and at least 95% of the variation of 38 of the remaining 58 outputs.

4.2.7. Design for learning. The change in the collection of outputs impacts the composition of the collection of active inputs. The border-block partition now has the structure shown in Figure 2(b) which includes only five active inputs. For the seven outputs that we have retained, we now develop a design to learn about the accurate emulators. For each output we seek to maximise the reduction in the variance via minimising C_{CF} from (26). We construct this design as a combination of the tuning design as given in Table 3 (since this has not yet been used to perform an update of the emulator), with an additional design generated by the border-block methods of Section 3, evaluating the design criterion over the combination of the two designs. For this stage, we constructed new designs adding from 2 to 26 additional points. To simplify the search process, we proceed by adding two new observations at each stage. This makes it easier to compare different designs and speeds up the iterative search algorithm. As the size of the problem increases, such simplifications become important.

[Table 4 about here.]

[Table 5 about here.]

The value of the design criterion for the new designs is given in Table 4, and the first six points of this additional design are given in Table 5. The criterion value for the tuning design alone is 92.130. Adding extra points to the design and optimising for the appropriate criterion produces a noticeable improvement in design performance, with the performance increasing as the design size increases albeit with generally diminishing returns. We observe a similar relationship in Figure 3 which displays the performance of the accurate emulator in terms of the average proportion of

variation in the accurate simulator outputs explained by the accurate emulator as we progressively increase the size of the final design from the initial 6-point tuning design. Again, we observe steep learning for the first six design points (corresponding to the points in the tuning design) followed by progressively diminishing returns. As the size of the improvement in design quality and simulator variation explained tends to level off as the design size increases, if the design budget is very tight for this stage then we might settle for only 6 additional runs. However, if more runs were possible then we might perform a further one or two batches of 6 runs, supporting the formal design calculations with careful data analysis given the results of each batch.

[Figure 3 about here.]

Finally, we compared the design obtained from combining the tuning design with a design generated by border-bock methods using (25), with that obtained from combining the tuning design with designs constructed from maximin Latin hypercubes. The Latin hypercube designs were generated in a similar fashion, adding 2 points each iteration in a manner which preserves the Latin hypercube property of the design and preferring designs which maximised the minimum inter-point distance. 100 such comparison designs were generated and their performance according to criterion (25) was calculated. The results are presented in Figure 4. We can see from the plot that, even with this relatively low-dimensional space, the border-block method clearly outperforms the Latin hypercubes and provides consistently good performance, whereas the Latin hypercube designs suffer from substantial internal variability.

[Figure 4 about here.]

5. DISCUSSION

Learning about complex high dimensional functions with many outputs from limited numbers of evaluations is a very challenging problem. We have presented a method to tackle this problem which exploits fast approximations to the simulator and uses valuable structural information about the model parameters. We have decomposed the problem into a number of stages each of which can be performed in different ways depending on the requirements of the problem. Each stage may be carried out in a more or less thorough way depending on how carefully we make our belief specifications, how focused we are on particular subsequent practical uses for the analysis, and how much time and computing power we are prepared to expend on seeking efficient designs. These are highly problem specific issues. The modular structure of our approach, and the computational

simplifications within the Bayes linear formulation, give plenty of scope for upgrading any individual step which is particularly important in a specific problem. We view the framework that we have suggested as giving a sensible set of guidelines for organising the tasks that we should carry out when planning to use a coarse simulator as the basis for small sample designs, and our suggestions for implementing each stage supply a tractable starting baseline for carrying out each task.

We have restricted our attention to the role of the coarse simulator in creating informative designs. We therefore finish our account at the point at which we have sufficiently exploited information from the coarse simulator that we are ready to move on to the second design stage, namely capturing the information in the full simulator which cannot be approximated by information in the coarse version. We have emphasised the requirement for small sample designs for the first design stage, in our approach, as this releases the maximum budget for the second stage of the design process. This process starts with a diagnostic analysis as to the ways in which our representation breaks down, for example by "leave one out" diagnostics, more detailed comparisons of the covariance structure of predicted against observed outcomes or more open-ended investigations of the information contained in all of the simulator evaluations to that point. Predictive discrepancies, in combination with expert judgement, should form the basis of new exploratory designs which can be used to uncover systematic features that were represented in the first stage of the analysis simply as unstructured variation. The additional structure uncovered on the full simulator may be then investigated, where relevant, on the coarse approximation, leading to an iterative version of the design cycle that we have described.

6. ACKNOWLEDGEMENTS

This paper was produced with the support of the Basic Technology initiative as part of the "Managing Uncertainty for Complex Models" project. We are grateful to Energy SciTech Limited for providing the model and reservoir simulator software and the formulation of the example. We are also grateful for the detailed comments of the referees, which have lead to improvements in our account.

REFERENCES

Bates, R. A., Buck, R. J., Riccomagno, E., and Wynn, H. P. (1996), "Experimental Design and Observation for Large Systems," *Journal of the Royal Statistical Society, Series B*, 58, 77–94.

Craig, P. S., Goldstein, M., Rougier, J. C., and Seheult, A. H. (2001), "Bayesian forecasting for

complex systems using computer simulators," *Journal of the American Statistical Association*, 96, 717–729.

Craig, P. S., Goldstein, M., Seheult, A. H., and Smith, J. A. (1996), "Bayes Linear Strategies for History matching of hydrocarbon reservoirs," in *Bayesian Statistics 5*, eds. Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., Oxford, UK: Clarendon Press, pp. 69–95.

— (1997), "Pressure matching for hydrocarbon reservoirs: a case study in the use of Bayes linear strategies for large computer experiments," in *Case Studies in Bayesian Statistics*, eds. Gatsonis, C., Hodges, J. S., Kass, R. E., McCulloch, R., Rossi, P., and Singpurwalla, N. D., New York: Springer-Verlag, vol. 3, pp. 36–93.

— (1998), "Constructing partial prior specifications for models of complex physical systems," *Applied Statistics*, 47, 37–53.

Cressie, N. (1991), *Statistics for Spatial Data*, Wiley.

Cumming, J. A. and Wooff, D. A. (2007), "Dimension reduction via principal variables," *Computational Statistics & Data Analysis*, 52, 550–565.

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1991), "Bayesian Prediction of Deterministic Functions With Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, 86, 953–963.

Goldstein, M. and Rougier, J. C. (2008), "Reified Bayesian Modelling and Inference for Physical Systems," *Journal of Statistical Planning and Inference*, 139.

Goldstein, M. and Wooff, D. A. (2007), *Bayes Linear Statistics*, Chichester, England: Wiley.

Hocking, R. R. (1976), "The Analysis and selection of variables in linear regression," *Biometrics*, 32, 1–49.

Kennedy, M. C. and O'Hagan, A. (2000), "Predicting the Output from a Complex Computer Code when Fast Approximations are Available," *Biometrika*, 87, 1–13.

— (2001), "Bayesian Calibration of computer models," *Journal of the Royal Statistical Society, Series B*, 63, 425–464.

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

Morris, M. D. and Mitchell, T. J. (1995), "Exploratory designs for computational experiments," *Journal of Statistical Planning and Inference*, 43, 381–402.

Qian, Z., Seepersad, C. C., Joseph, V. R., Allen, J. K., and Wu, C. F. J. (2006), "Building surrogate models based on detailed and approximate simulations," *ASME Journal of Mechanical Design*, 128, 668–677.

Qian, Z. and Wu, C. F. J. (2008), "Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments," *Technometrics*, 50, 192–204.

Reese, C. S., Wilson, A. G., Hamada, M., Martz, H. F., and Ryan, K. J. (2004), "Integrated Analysis of Computer and Physical Experiments," *Technometrics*, 46, 153–164.

Sacks, J., Schiller, S. B., and Welch, W. J. (1989a), "Designs for computer experiments," *Technometrics*, 31, 41–47.

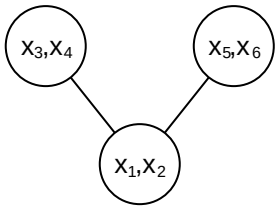
Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989b), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409–435.

Santner, T. J., Williams, B. J., and Notz, W. I. (2003), *The Design and Analysis of Computer Experiments*, New York: Springer-Verlag.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. (1992), "Screening, Predicting, and Computer Experiments," *Technometrics*, 34, 15–25.

Zečevič, A. I. and Šiljak, D. D. (1994), "Balanced decomposition of sparse systems for multilevel parallel processing," *IEEE Trans. Circ. Syst. Fund. Theory Applic.*, 41, 220–233.

List of Figures

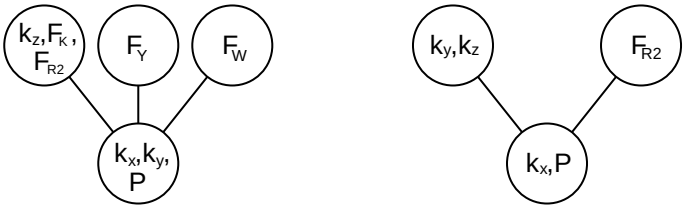


(a) Interaction graph of inputs

$$\left(\begin{array}{cc|ccc|cc} x_1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x_2 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & x_3 & 1 & 0 & 0 \\ 1 & 1 & 1 & x_4 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & x_5 & 1 \\ 1 & 1 & 0 & 0 & 1 & x_6 \end{array}\right)$$

(b) Matrix representation

Figure 1: Decomposition of active input structure within the emulators $f_1^c(\boldsymbol{x}) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(\boldsymbol{x})$ and $f_2^c(\boldsymbol{x}) = s_2^c(x_1, x_2, x_5, x_6) + w_2^c(\boldsymbol{x})$



(a) Initial screening (b) Second screening

Figure 2: Border block structure of the inputs (a) after the initial screening, and (b) after the second screening following the tuning runs

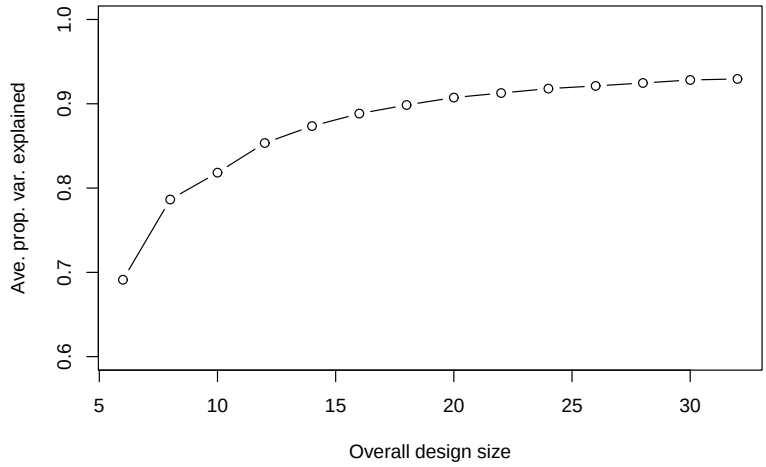


Figure 3: Average proportion of variance of the accurate computer model outputs explained by its emulator as a function of the size of the design beginning with the initial 6-point tuning design

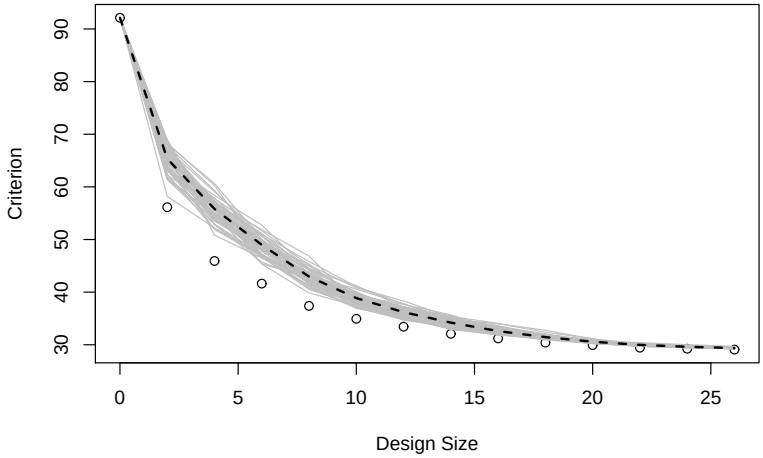


Figure 4: Comparison of design performance between the design constructed by the border-block method using criterion (25) (indicated by $\circ$), the performance of 100 maximin Latin hypercubes (grey lines), and the average performance of the 100 hypercubes (dashed line).

List of Tables

Output	Active inputs	Emulator trend adj. R^2	No. Model Terms	Classification
W4.pre	k_x, P, F_{R2}	0.90	11	$(*)$
W1.oil.rt	k_x, k_y, P	0.63	12	$(*)$
W3.bhp	k_x, k_y, P, F_K	0.75	16	$(-)$
W2.wat.tot	k_x, k_y, P, F_{R2}	0.76	17	$(+)$
W3.wct	$k_x, k_y, k_z, P, F_K, F_{R2}$	0.71	25	$(-)$
W2.gor	k_x, k_y, P	0.86	11	$(-)$
W4.wct	k_x, P, F_W	0.85	12	$(+)$
G1.gasinj.tot	k_x, k_y, k_z, P	0.93	12	$(*)$
W2.liqrt	k_x, k_y, P	0.70	11	$(*)$
W3.gor	k_x, k_y, P	0.77	11	$(*)$
W2.wat.rt	k_y, P, F_{R2}	0.71	12	$(+)$
W4.wat.tot	k_x, k_y, P, F_Y	0.87	15	$(+)$
W4.gor	k_x, k_y, P, F_W	0.89	16	$(-)$
W1.oil.tot	k_x, k_y, P	0.81	8	$(*)$
W4.oil.rt	k_x, k_y, P	0.74	10	$(*)$

Table 1: Emulation summary for each model output

	Well 2 Liquid Rate		Well 1 Oil Total	
Coefficient	Coarse	Accurate	Coarse	Accurate
(Intercept)	499.8	-3424.5	566.4	-654.0
k_x	-1614.1	-1353.4	22.9	56.9
k_y	216.9	467.1	50.3	230.4
P	2824.7	9681.3	1418.4	5044.6
k_x^2	1594.9	2643.6		
P^2	-3504.9	-6448.4	-1936.9	-5406.9
k_x^3	-536.9	-1050.0		
P^3	1325.7	1472.2	787.5	1879.9
$k_x : k_y$	-66.0	84.85	-42.9	-45.33
$k_x : P$	57.7	-819.2		
$k_y : P$	-182.6	-457.7	-42.5	-223.1
R^2	0.70	0.86	0.81	0.72

Table 2: Coefficients for the emulator trend of Well 2 Liquid Rate and Well 1 Oil Total fitted to the coarse model runs, and to 200 accurate model runs

k_x	k_y	k_z	P	F_Y	F_W	F_K	F_{R2}
-0.85	-0.85	-0.95	-0.96	-0.93	-0.99	-0.84	-0.82
0.78	0.64	-0.33	0.51	0.14	0.20	-0.63	-0.14
-0.53	-0.29	0.91	-0.22	0.76	0.64	0.39	0.40
0.66	-0.36	0.55	-0.35	0.61	-0.65	-0.11	0.26
-0.10	0.74	0.26	0.05	-0.23	0.70	0.15	-0.61
0.23	0.19	-0.54	0.86	-0.52	-0.03	0.88	0.73

Table 3: The tuning design generated for $\mathcal{X}^A$

Additional Design Size	Criterion Value	Additional Design Size	Criterion Value
0	92.130	14	31.958
2	56.500	16	31.072
4	46.760	18	30.594
6	42.142	20	30.040
8	37.289	22	29.524
10	34.792	24	29.237
12	32.861	26	29.167

Table 4: Values of design criterion (25) for the second stage design

k_x	k_y	k_z	P	F_{R2}
0.26	0.34	0.02	-0.57	0.78
-0.28	-0.74	-0.09	0.46	0.24
-0.63	0.44	0.63	0.76	-0.59
0.80	-0.41	-0.80	-0.69	0.94
-0.83	0.16	-0.35	0.19	0.07
0.92	0.89	0.47	-0.06	-0.84

Table 5: The design for the final stage of the analysis