Combining model calibration and design

Carl Ehrett*
School of Mathematical and Statistical Sciences, Clemson University,
D. Andrew Brown
School of Mathematical and Statistical Sciences, Clemson University,
Evan Chodora
Department of Mechanical Engineering, Clemson University,
Christopher Kitchens
Department of Chemical and Biomolecular Engineering, Clemson University,
and
Sez Atamturktur
Department of Architectural Engineering, Pennsylvania State University

Abstract

The calibration of computer models and the use of those computer models for design are two activities that are traditionally carried out separately. This paper presents a methodology for combining calibration and design under a unified Bayesian framework. This constitutes a generalization of existing Bayesian frameworks for computer model calibration. This unified framework provides a computationally efficient means for undertaking both tasks in a way that quantifies all relevant sources of uncertainty. Specifically, compared with the ordinary approach of undertaking design using estimates made during previous model calibration, this generalized framework includes uncertainty from the calibration process in the design procedure. In addition, we demonstrate how, when adaptive sampling of the phenomenon of interest is possible, the proposed framework is able to select new sampling locations using both the observations made so far of the real phenomenon and the information contained in the computer model. This is especially useful when the model is misspecified in failing to reflect that the parameter to be calibrated is functionally dependent upon the design inputs that are to be optimized.

Keywords: Gaussian processes, optimization, uncertainty quantification, computer model calibration

*The authors gratefully acknowledge grant CMMI-1934438 from the National Science Foundation (NSF). CE was supported by fellowships through Department of Education GAANN grant P200A150310 and NSF NRT grant 1633608. DAB is also supported by NSF grants EEC-1744497 and OIA-1826715.

1 Introduction

This paper connects two distinct areas of research concerning computer models of real phenomena. One area is that of computer model calibration, where the goal is to find a posterior distribution on unknown parameters by calibrating a computer model using real-world observations of the modeled phenomenon. The second area is that of enlisting a computer model for design, using the model to find settings for controllable system inputs such that the resulting system output is optimized with respect to some goal. These two problems are structurally similar, both involving finding estimates or distributions of model inputs to achieve some desired effect on model outputs. In the case of calibration, the desired effect is that the model outputs approximate reality, and in the case of design, the desired effect is that the model outputs approximate the optimal achievable outputs. The similarity of the two problems motivated the development of the design methodology called calibration to target outcomes (CTO; Ehrett et al., 2019), which adapts Bayesian methods for computer model calibration to serve as tools for design. Calibration and design are typically carried out separately, so that CTO and other optimization techniques operate under the assumption that the model is an accurate approximation of the real system of interest. This paper extends the methodology of CTO to include both calibration and design. The result is to provide a unified framework for both problems, which is especially of interest when the two activities are non-trivially intertwined, as in the case when the value of the calibration parameters are functionally dependent upon the design settings.

Bayesian methods for computer model calibration are developed by Kennedy and O'Hagan (2001). Since their seminal paper, the methodology has seen numerous extensions and re-

finements (Higdon et al. 2004; Williams et al. 2006; Bayarri et al. 2007a; Bayarri et al. 2007b; Paulo et al. 2012; Brynjarsdóttir and O'Hagan 2014). Call this approach to calibration KOH. Common to KOH approaches is the Bayesian framework in which one places a prior on the calibration parameters $\boldsymbol{\theta}_c$, often pairing it with a Gaussian process (GP) meta-model of the computer model of interest and a GP prior on the model discrepancy $\delta()$, and using the available observations y_r of the real system to find a posterior distribution $\boldsymbol{\theta}_c, \delta()|y_r$. Such an approach is notable for providing not merely a point estimate of the calibration parameter, but for providing a full posterior distribution quantifying remaining uncertainty about $\boldsymbol{\theta}_c$ and about $\delta()$.

CTO uses the KOH framework to find a posterior distribution, not on unknown model parameters, but rather on controllable design settings. It does this via an approach called counterfactual Bayes. In traditional model calibration, one uses Bayes' rule to discover a posterior distribution on calibration parameters using real observations, so that the real observations are the source of the Bayesian learning. In a design case, there are no relevant observations. One wants to find design settings that induce the system to behave optimally, but one typically has not observed the system doing so, and therefore there seems to be no relevant source of Bayesian learning that could drive the use of Bayes' rule to discover a posterior distribution on optimal design settings. The idea of counterfactual Bayes is to identify artificial observations y_t such that the resulting likelihood is highest in the optimal design region — i.e., artificial observations such that their occurrence is strong evidence that the design settings are optimal. Call y_t "target outcomes". In CTO via counterfactual Bayes, one uses the KOH framework to find a posterior distribution on design settings

given the target outcomes — and given the nature of y_t, this is *de facto* a distribution on optimal design settings for the system. The result retains the benefits of the Bayesian model calibration tools on which it is based, namely the quantification of remaining uncertainty regarding the optimal design settings.

CTO is thus a method for optimizing a computationally expensive black-box function. We may divide optimization approaches in such cases broadly into three camps (Regis and Shoemaker, 2004). Gradient-based approaches (Nocedal and Wright, 2006) are of limited utility when dealing with black-box functions, where we cannot evaluate the objective function's derivative. Approximation of the derivative requires additional function evaluations, rapidly inflating the computational cost when each evaluation involves significant expense. Heuristic approaches (Lee and El-Sharkawi, 2007) such as evolutionary algorithms (Branke et al., 2008; Deb et al., 2002; Kim et al., 2004), particle swarm optimization (Bonyadi and Michalewicz, 2017; Mason et al., 2017), and simulated annealing (Robert and Casella, 2004) avoid the need to know or approximate derivatives, but often require prohibitively many function evaluations. Furthermore, such methods, like gradient-based approaches, do not inherently provide quantification of remaining uncertainty about optimal design settings and the system outputs at those settings. Methods exist for using heuristic approaches while accommodating and quantifying uncertainties (Deb and Gupta, 2006; Zhou et al., 2011), but these come at the cost of even further inflating the number of function evaluations required.

The third camp is the diverse collection of response surface methodologies (RSMs; Dean et al., 2017) used for optimization. RSMs operate by fitting a predictive model to

an existing set of model runs, to form a computationally inexpensive meta-model which is then used to explore the model output. CTO is an example of an RSM, using GPs for its meta-model fit. Other popular versions of RSMs include efficient global optimization (EGO; Jones et al., 1998; Brochu et al., 2010) and stepwise uncertainty reduction (SUR; Geman and Jedynak, 1996; Villemonteix et al., 2009; Chevalier et al., 2014; Picheny, 2015; Miguel Hernández-Lobato et al., 2016; Picheny et al., 2019; Binois et al., 2019). These methodologies are both designed to facilitate sequential sampling from the system of interest in a search for the global optimum. They differ in their *acquisition functions*, which determine the location of the next sampling location throughout the optimization process. EGO finds the spot that maximizes the expected improvement (Mockus et al., 1978; Jones et al., 1998), whereas SUR's acquisition function seeks to reduce the volume of excursion sets below the current best known solutions (Chevalier et al., 2014). Such methodologies are designed for the case when one has the ability to sample sequentially from the system of interest. As a result, these methodologies are of limited utility when one is constrained to rely on a pre-existing set of observations, or in general when the observation locations cannot be chosen purely to suit the goal of optimization. In this respect they differ from CTO, which may be used to estimate optimal design settings using a pre-existing set of observations. Furthermore, the acquisition functions employed by EGO and SUR attempt to balance exploitation (proposing a new sample location that optimizes system output) with exploration (proposing a location that promotes learning for subsequent rounds of sampling). As a result, although these acquisition functions constitute distributions of sampling locations, by their nature they are not interpretable as distributions of the *op-*

timum, and hence these distributions do not quantify uncertainty regarding the location of the optimum. By contrast, CTO (understood as a pure-exploitation method) quantifies remaining uncertainty regarding the location of the system optimum.

An example of an RSM more closely resembling CTO is described by Olalotiti-Lawal and Datta-Gupta (2015). Their approach defines a distribution which is designed to lie both on and near the Pareto front (PF) of the objective function. CTO may be used to estimate the Pareto front through repeated applications (Ehrett et al., 2019), or may be used to approximate a given particular set of target outcomes y_t. Both approaches generate a posterior distribution which includes quantified uncertainties, exploring that distribution via Markov chain Monte Carlo (MCMC; Gelfand and Smith, 1990). However, in the case of Olalotiti-Lawal and Datta-Gupta, their posterior distribution is designed by the authors, and does itself come from the model of the system of interest. The uncertainty quantified by this posterior distribution, then, is not dictated by the model itself, and its interpretability is therefore not entirely clear. By contrast, CTO provides a posterior distribution based on the likelihood of the optimal design settings given the (hypothetical) observation of target outcomes y_t, and thus the uncertainty quantified by CTO is model-driven and interpretable as uncertainty regarding the optimal values for the design inputs and the resulting system output.

RSMs are tools for optimization, and do not themselves include methods to accommodate the case in which optimization is performed using a model that stands in need of calibration. In practice, models typically are known or suspected to be biased representations of the phenomenon of interest, and often have inputs that require calibration. The

goal of the work described here is to provide a unified framework for calibration and design by combining KOH and CTO. Call the resulting methodology DCTO, for dual calibration to target outcomes. CTO as previously developed assumes, somewhat idealistically, that the computer model is already perfectly calibrated. DCTO avoids this idealization. Furthermore, when undertaking KOH, some areas of the model range may be of greater interest than others. For example, one may be more interested in calibrating the model to be accurate in the optimal region of some design variable $\boldsymbol{\theta}_d$ than elsewhere. Undertaking dual calibration may allow us to focus our calibration efforts on such regions of interest, prioritizing them over other areas of the model range.

The rest of the paper is organized as follows. Section 2 describes the difficulties involved in combining KOH and CTO, illustrating this by considering the failings of a naïve method for combining the two procedures. Section 3 describes the proposed DCTO framework for combining KOH calibration and CTO design. Here it is shown that DCTO is in fact a generalization of the KOH framework. Section 4 showcases the methodology using a synthetic example, demonstrating that it achieves results comparable to employing KOH followed by CTO, though with more thorough quantification of the relevant uncertainties. Section 5 considers how DCTO may be useful in the case where sequential sampling is possible. In particular, sequential sampling with DCTO is attractive when the calibration parameter is known or suspected to be functionally dependent upon the design settings. Section 6 concludes with discussion of the results and thoughts about future directions.

2 Naïve DCTO

The version of KOH considered here is that which finds a posterior distribution on a parameter of interest, $\boldsymbol{\theta}$, using a GP emulator with hyperparameters $\widehat{\boldsymbol{\phi}_\eta}$ estimated via maximum likelihood using a budget of computer model observations $\boldsymbol{\eta}$. Similarly, one may use a GP prior with hyperparameters $\boldsymbol{\phi}_\delta$ to model discrepancy between the computer model $\eta()$ and the true function $f()$ that it represents. In the work described here, we employ stationary GPs with a Gaussian kernel covariance structure $C(\mathbf{x}, \mathbf{x}') = 1/\lambda \times \exp(-\beta(\mathbf{x} - \mathbf{x}')^2)$, so that $\widehat{\boldsymbol{\phi}_\eta} = \left[\widehat{\beta},\ \widehat{\lambda}\right]$. Setting priors on $\boldsymbol{\theta}$ and on $\boldsymbol{\phi}_\delta$, we train the GP emulator on observations $\boldsymbol{\eta}$ and use MCMC to explore the distribution

$$\pi(\boldsymbol{\theta}, \boldsymbol{\phi}_\delta | \mathcal{D}, \widehat{\boldsymbol{\phi}_\eta}) \propto \pi(\mathcal{D} | \boldsymbol{\theta}, \widehat{\boldsymbol{\phi}_\eta}, \boldsymbol{\phi}_\delta) \times \pi(\boldsymbol{\theta}) \times \pi(\boldsymbol{\phi}_\delta) \tag{1}$$

where $\mathcal{D} = (\boldsymbol{\eta}^T,\ \mathbf{y}^T)^T$ for some observations $\mathbf{y}$.

In a computer calibration problem, $\mathbf{y}$ is a set of observations of the system modeled by $\eta()$. In CTO, by contrast, y is a set of target outcomes – artificial data representing the way that one wishes to induce the system to behave, rather than observations one has made of the system in reality. When one wishes to perform CTO on a system that also requires traditional calibration, then, one obvious idea is to combine the two approaches by using Equation (1) with $\mathbf{y} = (\mathbf{y}_r^T,\ \mathbf{y}_t^T)^T$, an array containing both real observations $\mathbf{y}_r$ (for calibration) and target outcomes $\mathbf{y}_t$ (for CTO). However, this approach will not work, for two reasons. Firstly, the inputs $\boldsymbol{\theta}$ are typically not the same in calibration and in CTO. In calibration, one seeks to estimate the value of an input that either represents some

true unknown quantity, or else which induces the model to represent some true unknown quantity. In CTO, one seeks to find a distribution on an input that is under the researcher's control.

Relatedly, by including target outcomes in one's observations $\mathbf{y}$ used in KOH calibration, one compromises the integrity of the calibration. KOH calibration, after all, aims to use one's observations of the real system to estimate the value of $\boldsymbol{\theta}$. If one's observations $\mathbf{y}$ do not reflect the behavior of the real system, then they will constitute a poor source of information for bringing the model into alignment with reality. Simply put, to represent reality one must train one's model on observations of reality, not on unobserved targets.

3 Dual calibration to target outcomes

Given that naïve DCTO cannot accomplish the dual tasks of calibration and design, a different approach must be taken. The two tasks must be separated. An obvious choice here is to perform KOH calibration first, without involving any target outcomes, and then to use the calibrated model in order to perform CTO. Under this approach, with observations $\mathbf{y}_r$ of the system of interest, one would employ the model described in Equation (1) with $\boldsymbol{\theta} = \boldsymbol{\theta}_c$ (the parameters to be calibrated) and with $\mathcal{D} = \mathcal{D}_c = (\boldsymbol{\eta}^T, \mathbf{y}_r^T)^T$. The result would be a posterior distribution of $\boldsymbol{\theta}_c$ and of $\delta(\cdot)$, the systematic discrepancy between the computer model $\eta(\cdot, \cdot)$ and the true system $f(\cdot)$. These can be used to produce estimates $\widehat{\boldsymbol{\theta}}_c$ and $\widehat{\delta}(\cdot)$ such that $f(\mathbf{z}) \approx \eta(\mathbf{z}, \widehat{\boldsymbol{\theta}}_c) + \widehat{\delta}(\mathbf{z})$ for all $\mathbf{z}$ in the domain of f. The result is a calibrated model $\eta_c(\mathbf{z}) = \eta(\mathbf{z}, \widehat{\boldsymbol{\theta}}_c) + \widehat{\delta}(\mathbf{z})$ which can be used for CTO.

With η_c in hand, one can partition $\mathbf{z}$ into $(\mathbf{x}, \boldsymbol{\theta}_d)$ where $\boldsymbol{\theta}_d$ is the set of inputs over

which one wishes to optimize, and $\mathbf{x}$ are all other inputs to the calibrated model. We can write $\eta_c(\mathbf{z})$ as $\eta_c(\mathbf{x}, \boldsymbol{\theta}_d)$. Then one can perform CTO again using Equation (1), this time with $\boldsymbol{\theta} = \boldsymbol{\theta}_d$ and $\mathcal{D} = \mathcal{D}_t = (\boldsymbol{\eta}_c^T, \mathbf{y}_t^T)^T$ where $\boldsymbol{\eta}_c = \boldsymbol{\eta} + \widehat{\boldsymbol{\delta}} = \boldsymbol{\eta} + (\widehat{\delta}(\mathbf{z}_1), \cdots, \widehat{\delta}(\mathbf{z}_n))^T$. Notice that a single set of simulator runs $\boldsymbol{\eta}$ can be used both for KOH and for subsequent CTO. A crucial difference between KOH and CTO is that for the CTO step one would not attempt to model any systematic discrepancy between η_c and f, since an estimate of that discrepancy is already included in η_c. For the purposes of Equation (1), this amounts to setting a degenerate prior on ϕ_δ at 0.

Above, the use of CTO following KOH relies on two separate implementations of Equation (1). It will be useful to produce an integrated model which describes the use of both procedures, and which makes clear the relationship between them. For this purpose, consider η as having three inputs $(\mathbf{x}, \mathbf{t}_c, \mathbf{t}_d)$ where $\mathbf{t}_c$ denotes the parameters targeted for KOH calibration, $\mathbf{t}_d$ denotes the input settings targeted for design via CTO, and $\mathbf{x}$ denotes the remaining known and/or controllable inputs. If η can be run quickly, then we use it directly in MCMC. However, if it is computationally expensive, we employ a surrogate by setting a Gaussian process (GP) prior on η with mean $m_\eta(\mathbf{x}, \mathbf{t}_c, \mathbf{t}_d)$ and covariance function $C_\eta((\mathbf{x}, \mathbf{t}_c, \mathbf{t}_d), (\mathbf{x}', \mathbf{t}_c', \mathbf{t}_d'))$. From here on in this discussion, assume that a GP surrogate is used for η. Model the systematic discrepancy between η and f at the true value of $\mathbf{t}_c = \boldsymbol{\theta}_c$ with another GP prior $\delta(\cdot, \cdot)$ having mean $m_\delta(\mathbf{x}, \mathbf{t}_d)$ and covariance function $C_\delta((\mathbf{x}, \mathbf{t}_d), (\mathbf{x}', \mathbf{t}_d'))$. In addition to systematic discrepancy between η and reality, measurement error ϵ_r may be included in the model for real observations $\mathbf{y}_r$, and additional Gaussian observation error ϵ_d may be included for target outcomes $\mathbf{y}_t$. The purpose of additional

observation error ϵ_d is twofold. Depending on the distribution of ϵ_c, the target outcomes $\mathbf{y}_t$ may or may not be within the support of a model that lacks ϵ_d. Including ϵ_d ensures that the targets are compatible with the model. Secondly, including ϵ_d and estimating its variance σ_d^2 provides computational benefits. For example, even if the target outcomes are compatible with a model that does not include ϵ_d, they may be extreme outliers to the extent that the relevant likelihoods are small enough to generate significant numerical errors during MCMC. In terms of the interpretation of the model, adding ϵ_d amounts to supposing that the counterfactual target outcomes were observed with greater than usual observation error, where that additional error is distributed as $N(0, \sigma_d^2)$. Though it is not necessary to assume that ϵ_c is Gaussian, for simplicity of presentation we assume here that it is distributed as $N(0, \sigma_c^2)$. Finally, we assume that η, δ, ϵ_c and ϵ_d are all mutually independent.

A collection of simulation runs is needed to train the GP code surrogate. Let $(\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}})$ be the design matrix for these simulation runs, and let $\mathbf{y_s}$ denote the output of these runs. Similarly, let $\mathbf{y_r}$ be observations made at $\mathbf{x_r}, \mathbf{t_{dr}}$, and let $\mathbf{y_t}$ be target outcomes "observed" at $\mathbf{x_t}$. Finally, let $\mathbf{y} = (\mathbf{y_s}^T, \mathbf{y_r}^T, \mathbf{y_t}^T)^T$. Then it follows that $\mathbf{y} \sim \mathrm{N}(\mathbf{m}, \mathbf{C})$, where

$$\mathbf{m} = \begin{pmatrix} m_s(\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}) \\ m_s(\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}) + m_\delta(\mathbf{x_r}, \mathbf{t_{dr}}) \\ m_s(\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T) + m_\delta(\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T) \end{pmatrix},$$

$$\mathbf{C} = \begin{pmatrix} \mathbf{C_{11}} & \mathbf{C_{12}} & \mathbf{C_{13}} \\ \mathbf{C_{21}} & \mathbf{C_{22}} & \mathbf{C_{23}} \\ \mathbf{C_{31}} & \mathbf{C_{32}} & \mathbf{C_{33}} \end{pmatrix},$$

$$\mathbf{C_{11}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}})\right)$$

$$\mathbf{C_{21}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}})\right)$$

$$\mathbf{C_{31}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right)$$

$$\mathbf{C_{12}} = \mathbf{C_{21}}^T$$

$$\mathbf{C_{22}} = C_\eta\left((\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}), (\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}})\right) + C_\delta\left((\mathbf{x_r}, \mathbf{t_{dr}}), (\mathbf{x_r}, \mathbf{t_{dr}})\right) + \sigma_c^2\mathbf{I}$$

$$\mathbf{C_{32}} = C_\eta\left((\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + C_\delta\left((\mathbf{x_r}, \mathbf{t_{dr}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T)\right)$$

$$\mathbf{C_{13}} = \mathbf{C_{31}}^T$$

$$\mathbf{C_{23}} = \mathbf{C_{32}}^T$$

$$\mathbf{C_{33}} = C_\eta\left((\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + C_\delta\left((\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + \sigma_c^2\mathbf{I} + \sigma_d^2\mathbf{I}$$

Note that when $\mathbf{y_t}$ and $\mathbf{x_t}$ are empty and $\mathbf{m}, \mathbf{C}$ reduce respectively to their first two and upper two-by-two block elements, this is simply the KOH framework. Thus, DCTO is a combination of KOH and CTO that generalizes the KOH framework.

This inclusive framework allows for KOH and CTO to be undertaken simultaneously. A primary benefit of DCTO is that under this combined approach, the results of CTO include quantification of all sources of uncertainty. By performing KOH and then subsequently undertaking CTO using static estimates $\widehat{\boldsymbol{\theta}_c}$ and $\widehat{\delta}$ from KOH, uncertainty surrounding

those estimates is not included in the results of CTO. Another benefit of the combined approach appears in cases in which $\boldsymbol{\theta}_c$ is suspected to be a function of $\boldsymbol{\theta}_d$. In such cases, one may be interested only or primarily in the value of $\boldsymbol{\theta}_c$ at the optimal value of $\boldsymbol{\theta}_d$. If one has the freedom to sample adaptively from the true system, then this freedom can be applied in DCTO to concentrate samples disproportionately in the region of interest. This idea is explored further in Section 5.

Another crucial difference between performing CTO after KOH and performing DCTO is the role of the targets $\mathbf{y}_t$ in the calibration of $\boldsymbol{\theta}_c$. In DCTO as described above, the likelihood of $\boldsymbol{\theta}_c$ is affected by $\mathbf{y}_t$, whereas in KOH $\mathbf{y}_t$ is not included in the model and hence cannot affect the distribution of $\boldsymbol{\theta}_c$. This is a point in favor of performing KOH separate from CTO, as $\mathbf{y}_t$ is artificial data and cannot plausibly serve as a source of information about the correct value of $\boldsymbol{\theta}_c$. A similar issue affects CTO when an emulator is used. In KOH, the hyperparameters of an emulator may be estimated prior to calibration (e.g. via maximum likelihood), or else one may set priors for these hyperparameters and sample from their posteriors during the calibration process. If the latter route is chosen for CTO, then one would again face a situation in which learning about real quantities from artificial data. The hyperparameters of the GP emulating η should be estimated using observations of η and of the real process that η simulates. The solution in the case of CTO is to employ some form of modularity (Liu et al., 2009). A modular analysis intentionally falls short of being a full Bayesian analysis, either for computational benefits, or to quarantine "suspect" aspects of the model. The target outcomes $\mathbf{y}_t$ are precisely such a suspect source of Bayesian learning—they are by their nature extreme outliers, and hence are a poor

guide both for estimating the hyperparameters of the GP emulator and for estimating the parameter $\boldsymbol{\theta}_c$. In CTO, modularization with respect to the GP hyperparameters typically takes the form of producing maximum likelihood estimates of the GP hyperparameters and using those in lieu of setting priors and exploring a posterior distribution. Modularization with respect to $\boldsymbol{\theta}_c$ is implicit in the fact that an estimate $\widehat{\boldsymbol{\theta}_c}$ is used in CTO after KOH. DCTO can similarly modularize simply by refraining from including $\mathbf{y}_t$ in the updates of $\boldsymbol{\theta}_c$ during MCMC. That is, rather than calculating the likelihood of a proposed sample $t_c^{(i+1)}$ at step i of the MCMC using $\mathbf{y} = (\mathbf{y}_s^T, \mathbf{y}_r^T, \mathbf{y}_t^T)^T$, one can instead calculate its likelihood using only $\mathbf{y} = (\mathbf{y}_s^T, \mathbf{y}_r^T) \sim N(\mathbf{m}_r, \mathbf{C}_r)$, where $\mathbf{m}_r$ and $\mathbf{C}_r$ are respectively the upper two and upper-left two-by-two components of $\mathbf{m}$ and $\mathbf{C}$. Such modularization ensures that all Bayesian learning of $\boldsymbol{\theta}_c$ is based upon the real observations rather than upon $\mathbf{y}_t$.

4 Example with simulated data

Consider the function of three inputs $\eta(x, t_c, t_d) = x/(t_d^{t_c-1}\exp(-0.75t_d) + 1)$. Figure 1 shows the output of this function for $x = 1$ over the range $(t_c, t_d) \in [1.5, 4.5] \times [0, 5]$. We arbitrarily set $\theta_c = 2$ to be the "true" value of t_c. For any value of x and t_c, the optimal (minimizing) value of t_d is $(4/3)(t_c - 1)$, so we have $\theta_d = 4/3$. Figure 2 shows the locations of the true and optimal values (respectively) of θ_c and θ_d. There it is clear that the true value of θ_c is far from optimal – if this value were within our control, its optimal value would be at the upper end of its support, at 4.5. Thus η showcases the ability of DCTO to perform simultaneously both calibration and design in the case when our "truth-seeking" goals and our design goals are in tension.

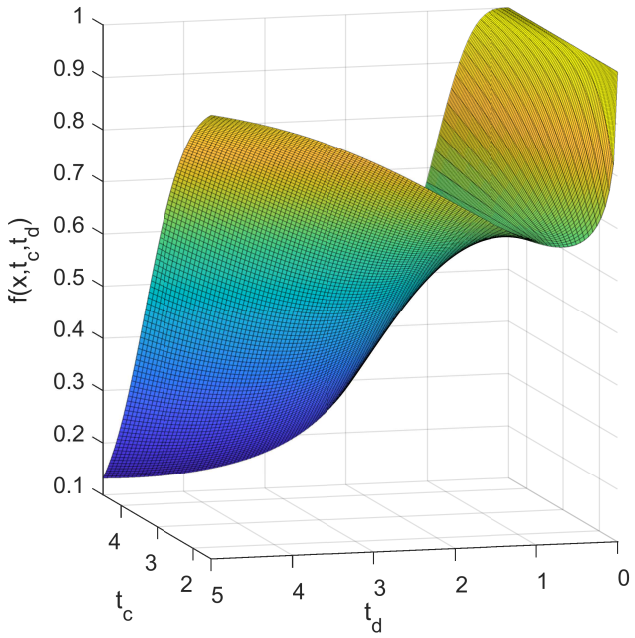

Figure 1: Example computer model output over the support of the calibration parameter t_c and the design parameter t_d.

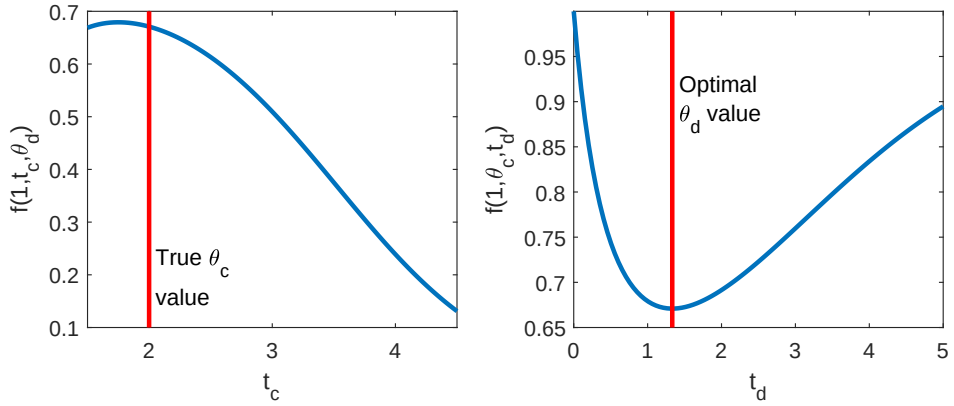

Figure 2: The lefthand plot shows the computer model output at $x = 1$ and optimal θ_d for each value of the calibration parameter t_c. The righthand plot show the model output at $x = 1, t_c = \theta_c$ for each value of the design parameter t_d.

4.1 Results

We used DCTO on four versions of the problem. First, we assumed that η is free from discrepancy – i.e. that $\eta(x, \theta_c, t_d)$ is an unbiased estimator of the "true" system $f(x, t_d)$. The other three versions each assume that η suffers from some form of discrepancy. Let f_1, f_2, f_3 denote the "true" systems in these three cases. We set

$$f_1(x, t_d) = \eta(x, \theta_c, t_d)\,(1 - a(x - .5)(x - 1)/x))$$
$$f_2(x, t_d) = \eta(x, \theta_c, t_d) - a(x - .5)(x - 1)\left(t_d - \frac{4}{3}\right)^2 + b$$
$$f_3(x, t_d) = \eta(x, \theta_c, t_d) + axt_d + b$$

Where a, b are constants which determine how severe the discrepancy is in each case. The function f_1 has a multiplicative discrepancy dependent only on x. This discrepancy does not affect the optimal value of t_d. The discrepancy of f_2 is additive, and is dependent upon both x and θ_c. Though this discrepancy can affect the optimal value of t_d, in the case that $\theta_c = 2$ (which is what we assume to be the truth) it does not. Thus under f_1 and f_2, it remains the case that the optimal value of t_d is $\theta_d = 4/3$. By contrast, f_3 has an additive discrepancy which does affect the optimal setting for t_d. For f_3, optimal t_d is dependent upon both the true value of θ_c and upon the value of a. For example, for $\theta_c = 2$ and $a = 0.055$, the optimal t_d is $\theta_d \approx 1$. Figure 3 shows the discrepancies for two different versions (corresponding to different settings of (a, b)) of each f_i.

We applied DCTO to each of seven cases: the non-discrepancy case, and the two different versions of each f_i shown in Figure 3. We found that in these cases, no appreciable

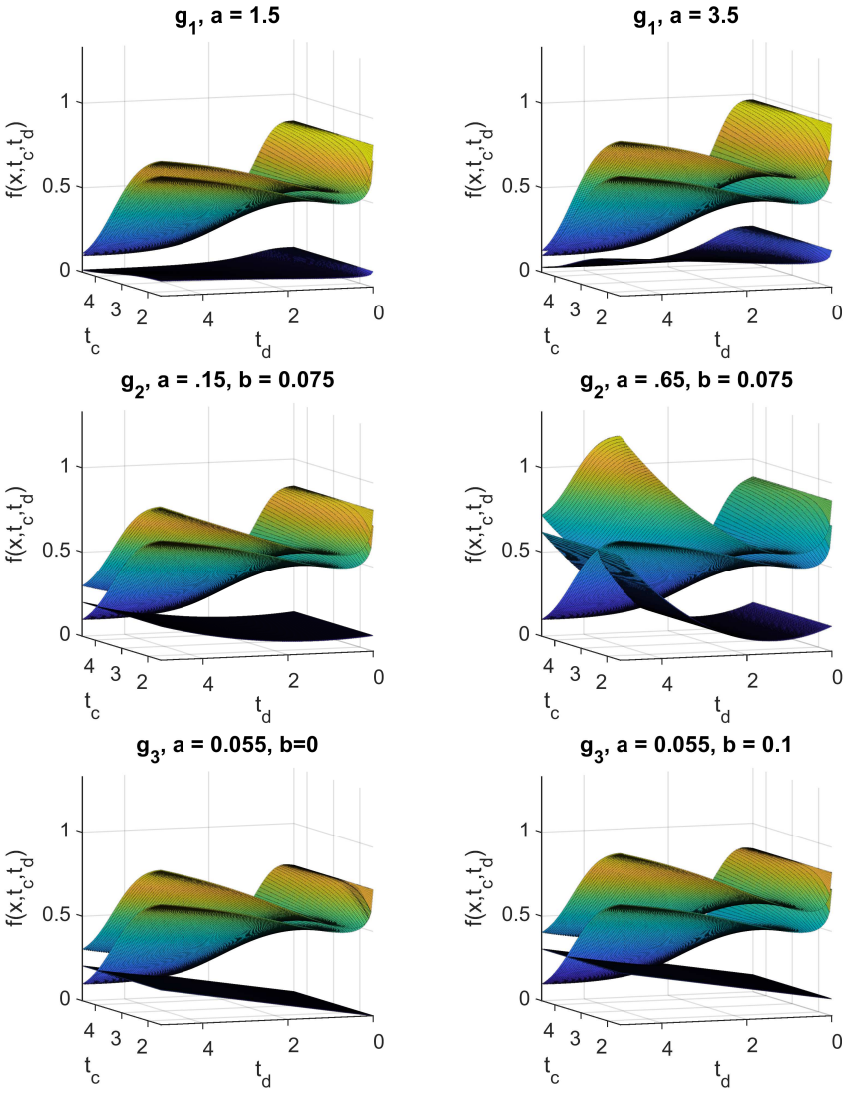

Figure 3: The i^{th} row shows f_i (the objective function with discrepancy), η (the computer model), and the discrepancy $f_i - \eta$, all at $x = 0.75$. In each row, a less aggressive version of the discrepancy appears on the left, and a more aggressive on the right. In each plot, the topmost surface is f_i, the middle surface is η, and the bottom surface is the discrepancy $f_i - \eta$.

difference resulted from the decision of whether or not to use an emulator (where the emulator was trained on a latin hypercube design of 250 points on the space of model inputs). Therefore, the results reported here do not employ an emulator. In each case, we gathered 30 "observations" of f_i on a latin hypercube design over the supports of x and θ_d, setting θ_c equal to its "true" value of 2. After standardizing the response to have mean 0 and standard deviation 1, we added i.i.d. N(0,0.05) noise to the response. We then carried out DCTO using Metropolis-Hastings-within-Gibbs MCMC, drawing 8000 samples each of $t_c, t_d, \boldsymbol{\rho}_\delta, \lambda_\delta, \sigma_d^2$, where $\boldsymbol{\phi}_\delta = (\boldsymbol{\rho}_\delta^T, \lambda_\delta)^T$. We modularized the analysis by drawing each of $\boldsymbol{\theta}_c, \boldsymbol{\rho}_\delta, \lambda_\delta$ using the likelihood based only on $(\mathbf{y}_s^T, \mathbf{y}_r^T)^T$ rather than on $(\mathbf{y}_s^T, \mathbf{y}_r^T, \mathbf{y}_t^T)^T$.

In order to evaluate the success of the calibration component of DCTO, we also carried out a two-step procedure of using traditional KOH calibration of θ_c, followed by a second step using CTO to obtain a distribution of θ_d. The first step is essentially DCTO with $\mathbf{x_t}, \mathbf{y_t}$ as empty (null) vectors, and the second step uses the distributions obtained in the first step to estimate θ_d. Thus, the comparison between DCTO and KOH+CTO shows the difference between DCTO and performing CTO on a system which has been calibrated using traditional methods. Figure 4 shows the results of DCTO and KOH+CTO for f_0, the case of no discrepancy. The two methods deliver comparable results, illustrating that combining KOH calibration and CTO design into DCTO does not undermine the performance of either task. Strong Bayesian learning has occurred for both parameters, in that the posterior distributions of θ_c, θ_d are peaked around their true and optimal values, respectively. KOH gives a similar posterior for θ_c, showing that the expansion of DCTO to undertake design has not interfered with its calibration performance. The skewness apparent in the posterior

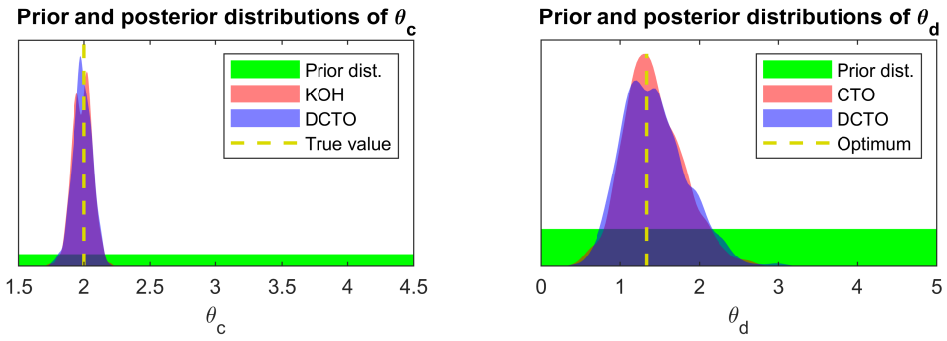

Figure 4: Prior and posterior distributions of the calibration parameter θ_c and design parameter θ_d, along with their true/optimal values, for DCTO and KOH+CTO carried out when there is no discrepancy between the true system and the computer model.

distributions of θ_d occur in all of the results gathered here, and is likely due to the shape of the objective function f, which increases sharply for $t_d < \theta_d$ and increases much more gently for $t_d > \theta_d$.

We performed each procedure 30 times on each of the seven different discrepancy situations (no discrepancy, and a large and small version of each of three discrepancies). The results are summarized in Table 2. The upper table gives the sample mean, over the thirty runs, of the marginal posterior variance of each of θ_c and θ_d. The two procedures generate extremely similar outcomes with respect to θ_c. However, the posterior variance for θ_d under DCTO is slightly higher than that under CTO in each of the seven cases considered. This is due to the fact that DCTO includes remaining uncertainty about the values of the hyperparameters of the discrepancy GP $\delta(\cdot)$. By contrast, CTO after KOH uses point estimates of those hyperparameters and thus achieves narrower posterior distributions due to excluding this source of uncertainty. The lower table gives the root mean square errors (RMSEs) for the posterior means of θ_c and θ_d, using their true value of 2 for θ_c and optimal value $4/3$ for θ_d in discrepancy cases $0, 1, 2$ and optimal value 1 for discrepancy 3. Again

	Posterior θ_c var.		Posterior θ_d var.	
Discrepancy	DCTO	KOH+CTO	DCTO	KOH+CTO
0	0.00633	0.00619	0.145	0.129
1, small	0.0140	0.0137	0.149	0.129
1, large	0.0141	0.0140	0.149	0.131
2, small	0.0143	0.0141	0.135	0.116
2, large	0.0609	0.0608	0.0804	0.0731
3, small	0.0134	0.0135	0.0882	0.0743
3, large	0.0143	0.0142	0.0945	0.0814

	$\widehat{\theta}_c$ RMSE		$\widehat{\theta}_d$ RMSE	
Discrepancy	DCTO	KOH+CTO	DCTO	KOH+CTO
0	0.0790	0.0795	0.120	0.121
1, small	0.0955	0.0956	0.149	0.144
1, large	0.137	0.139	0.209	0.209
2, small	0.109	0.106	0.130	0.127
2, large	0.158	0.155	0.123	0.121
3, small	0.294	0.292	0.0919	0.0919
3, large	0.279	0.281	0.0995	0.0990

Table 1: Posterior variance and root mean square error (RMSE) for the calibration variable θ_c and the design variable θ_d under both DCTO and KOH+CTO. The estimator $\widehat{\theta}_i$ is the posterior mean of t_i for $i = c, d$.

we see very similar outcomes in the two procedures for both parameters. In all cases but one, DCTO has slightly higher RMSE for θ_d than does CTO. This is to be expected given the above-mentioned wider posterior distributions of θ_d under DCTO.

5 Dependence of θ_c on θ_d

In many cases of computer model calibration, it is known or suspected that the value of one or more calibration parameters are functionally dependent upon the values of other model inputs (Atamturktur and Brown, 2015; Atamturktur et al., 2017; Ezzat et al., 2018). If one is interested to understand the functional form of the calibration parameter, then state-aware methods can be used to arrive at such an estimate (Atamturktur and Brown, 2015; Atamturktur et al., 2017). However, in a case of simultaneous calibration and design in which the calibration parameter is functionally dependent upon the design settings, one might be interested only to know the value of the calibration parameter in the optimal design region. In this case, the machinery of state-aware calibration is not needed, and effort is better spent focusing on estimating the fixed calibration parameter value in the region of interest. In such a case, it is preferable that one's calibration be founded on observations for which the design settings are in the optimal design region. This will allow one to calibrate the model using observations taken from the region of interest, so that the calibration takes on values from that region.

When observations may be made adaptively, other RSM approaches such as EGO (Jones et al., 1998; Brochu et al., 2010) or SUR (Geman and Jedynak, 1996; Villemonteix et al., 2009; Chevalier et al., 2014; Picheny, 2015; Miguel Hernández-Lobato et al., 2016; Picheny

et al., 2019; Binois et al., 2019) may be more efficient than CTO for estimating optimal design settings, though CTO offers more interpretable and model-driven uncertainty quantification. However, RSM approaches in general do not include tools to accommodate the case in which a model stands in need of calibration as well as optimization. DCTO provides such a framework for combined calibration and design. Therefore we now consider under the lens of DCTO the case in which the design settings of the observations of the true system may be chosen adaptively.

An obvious approach when both calibration and design are undertaken and the calibration parameter is suspected to be functionally dependent on the design settings is to perform the design optimization first, and then the calibration. In this way, one would gather observations in service of finding the optimal design settings, and then use these observations to learn the value of the calibration parameter in the associated optimal design region. For this purpose, any optimization method could be used to select the design settings for the observations. If a limited budget of observations is available and quantification of remaining uncertainty surrounding the true and optimal values (respectively) of the calibration parameter and design settings is desired, then RSMs are a good choice. In such a situation, CTO retains its advantage of interpretable and model-driven uncertainty quantification, and adds another advantage: namely, that DCTO constitutes a method for combining both the observations of the true system and the existing uncalibrated computer model to select new evaluation points.

The use of DCTO with adaptive sampling is potentially of greatest use when it is known or suspected that the calibration parameter is a function $\theta_c(t_d)$ of the design setting

t_d, and particularly when interest focuses on learning the optimal design setting θ_d and the corresponding value $\theta_c(\theta_d)$ of the calibration parameter. To implement DCTO with adaptive sampling in such a case, we perform DCTO beginning with an empty vector of true observations y_r. During the burn-in period, every (e.g.) 100 iterations of the MCMC, we sample draw a sample $\widehat{\theta}_d$ from the posterior distribution of the design settings t_d, as a means of estimating the optimal design settings θ_d. We then perform a new observation of the system at $(\mathbf{x}_i, \widehat{\theta}_d)$ where $i \in \{1, \cdots, m\}$ and m is the total budget of observations to be made. The non-design input settings $\mathbf{x}_i$ can be chosen to maximize distance from previous observations, or these locations can be predetermined according to a space-filling design over the domain of non-design inputs. This process is continued until the total budget of observations is reached, after which burn-in continues until the MCMC chains converge for all parameters of interest. The result is that observations are concentrated around the design settings of interest, so that the unknown calibration parameter values in those observations are concentrated around the value $\theta_c(\theta_d)$.

To demonstrate the use of DCTO with adaptive sampling in a case of functional dependence of the calibration parameter on design settings, we use the same objective function and discrepancy cases as described in Section 4. This time, instead of setting $\theta_c = 2$, we set

$$\theta_c(t) = 2.25 - .75\frac{\exp\left(40\left(\frac{t-1.5}{.75} - .5\right)\right)}{1 + \exp\left(40\left(\frac{t-1.5}{.75} - .5\right)\right)}.$$

The resulting optimal design settings and calibration parameter value at the optimum vary in the discrepancy cases, though $\theta_c(\theta_d)$ is near 2.16 in each case. Representative results from performing DCTO with adaptive sampling in each discrepancy case appear in Figure

	$\widehat{\theta}_c$ RMSE		$\widehat{\theta}_d$ RMSE	
Discrepancy	AS	SFD	AS	SFD
0	0.188	0.433	0.163	0.479
1, small	0.233	0.32	0.243	0.414
1, large	0.188	0.247	0.213	0.393
2, small	0.221	0.263	0.187	0.348
2, large	0.228	0.16	0.183	0.206
3, small	0.452	0.506	0.182	0.329
3, large	0.448	0.468	0.167	0.292

Table 2: Posterior root mean square error (RMSE) for the calibration variable θ_c and the design variable θ_d, for DCTO with adaptive sampling (AS) and a predetermined space-filling design (SFD). The estimator $\widehat{\theta}_i$ is the posterior mean of t_i for $i = c, d$.

5, along with results from applying DCTO non-adaptively (using a space-filling set of observations). A summary of the results of thirty applications of DCTO both with and without adaptive sampling, for each of the discrepancy cases, appears in Table 2.

The results show superior performance for the adaptive sampling DCTO over DCTO using a space-filling design. The adaptive DCTO posterior means have lower RMSEs in all cases for θ_d, and in all cases except one for θ_c. This demonstrates a useful robustness of adaptive DCTO to model misspecification, specifically in the case that the model treats as constant a calibration parameter that is more properly understood as functionally dependent upon other model inputs. By using the CTO-driven estimate $\widehat{\theta}_d$ to sample from the region of interest, DCTO learns from observations such that $\theta_c(\widehat{\theta}_d)$ is near to the value $\theta_c(\theta_d)$. This promotes better calibration with respect to the region of interest, and thereby better estimation of the optimal design settings. By relying on DCTO rather than on performing KOH using samples gathered using heuristic optimization methods, or other RSM approaches, we achieve these estimates with quantification of all relevant model-driven uncertainty with respect to the values of θ_c and θ_d.

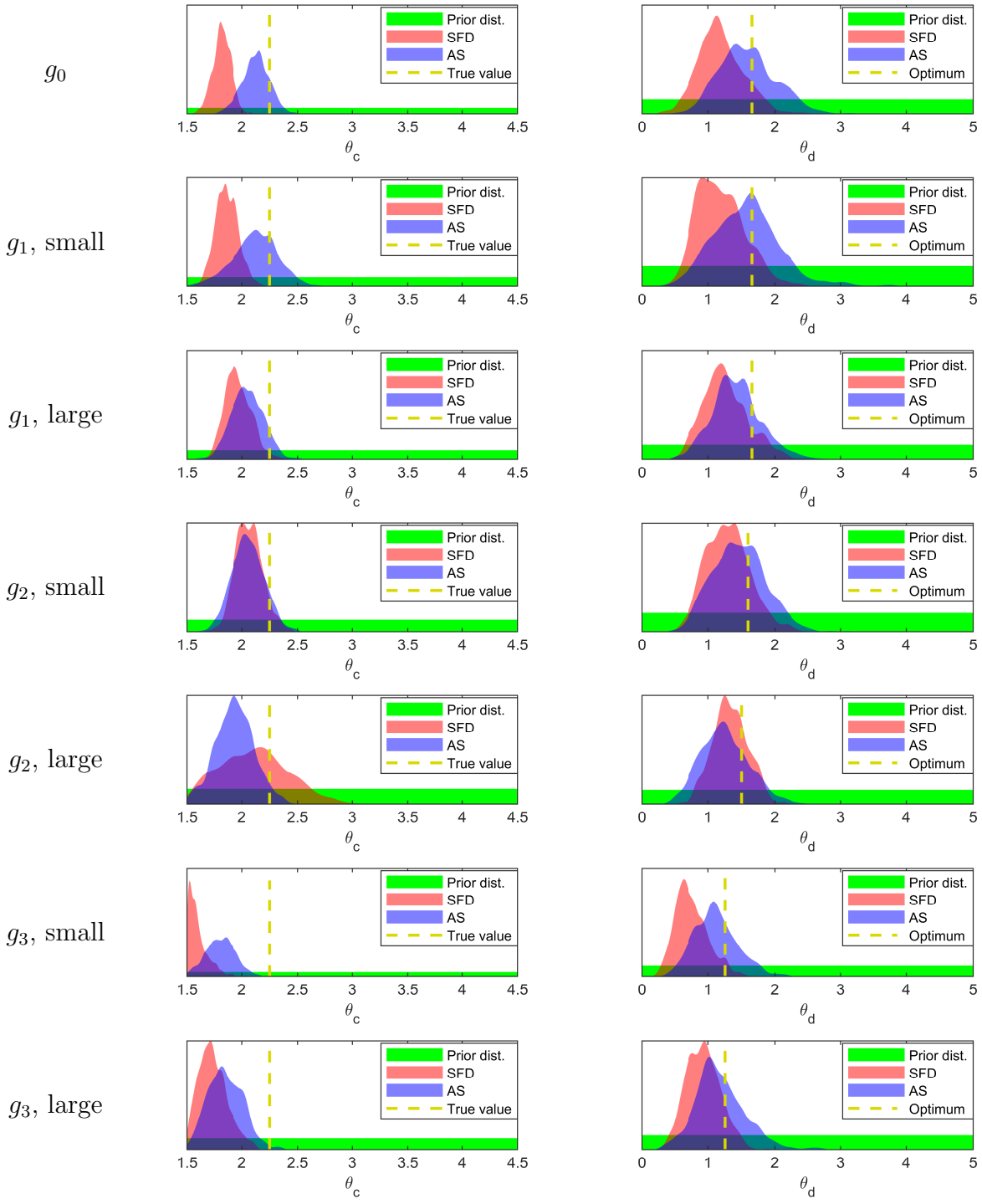

Figure 5: Prior and posterior distributions of the calibration parameter θ_c and design parameter θ_d, along with their true/optimal values, for DCTO with adaptive sampling (AS) and with predetermined space-filling design (SFD).

6 Conclusion

DCTO provides a method for combining calibration (under a KOH framework) with design (under a CTO framework). The result is a generalization of the KOH framework, which secures the benefits of KOH both for calibration and for design. This includes the ability to quantify uncertainty remaining in the true value of the calibration parameter, the optimal settings for the design input, and the resulting model output. DCTO achieves similar results to performing KOH calibration followed by CTO optimization, but provides a computationally efficient method of propogating the uncertainties remaining from KOH calibration through the CTO procedure. In the case when observations of the real system can be carried out sequentially at adaptively chosen locations, DCTO is robust to model misspecification where the calibration parameter is functionally dependent on the value of the design input, and the model fails to reflect this. In such a case, if the functional form of the dependence of $\boldsymbol{\theta}_c$ on $\boldsymbol{\theta}_d$ is of interest, then state-aware calibration should be used. However, if one only wishes to estimate the calibration parameter at the optimal design settings, then DCTO provides a means of doing so. In this application, DCTO with adaptive sampling uses information from both the sequentially-performed observations of the real system and from the existing computer model to identify new sampling locations. Future work on this subject will include pairing adaptive sampling DCTO with other methodologies for selecting new sampling locations, such as EGO and SUR.

References

Atamturktur, S. and D. A. Brown (2015). State-aware calibration for inferring systematic bias in computer models of complex systems. *NAFEMS World Congress Proceedings, June 21-24*.

Atamturktur, S., J. Hegenderfer, B. Williams, M. Egeberg, R. A. Lebensohn, and C. Unal (2017). Mechanics of advanced materials and structures: a resource allocation framework for experiment-based validation of numerical models. *Mechanics of Advanced Materials and Structures 22*(8), 641–654.

Bayarri, M. J., J. O. Berger, and J. Cafeo (2007a). Computer model validation with functional output. *The Annals of Statistics 35*, 1874–1906.

Bayarri, M. J., J. O. Berger, R. Paulo, J. Sacks, J. A. Cafeo, J. Cavendish, C.-H. Lin, and J. Tu (2007b). A framework for validation of computer models. *Technometrics 49*(2), 138–154.

Binois, M., V. Picheny, P. Taillandier, and A. Habbal (2019, feb). The Kalai-Smorodinski solution for many-objective Bayesian optimization.

Bonyadi, M. R. and Z. Michalewicz (2017, mar). Particle swarm optimization for single objective continuous space problems: A review.

Branke, J., K. Deb, K. Miettinen, and R. Slowinski (Eds.) (2008). *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Volume 5252 LNCS. Springer.

Brochu, E., V. M. Cora, and N. de Freitas (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning.

Brynjarsdóttir, J. and A. O'Hagan (2014). Learning about physical parameters: The importance of model discrepancy. *Inverse Problems 30*(11).

Chevalier, C., J. Bect, D. Ginsbourger, E. Vazquez, V. Picheny, and Y. Richet (2014). Fast Parallel Kriging-Based Stepwise Uncertainty Reduction With Application to the Identification of an Excursion Set. *Technometrics 56*(4).

Dean, A., D. Voss, and D. Draguljić (2017). *Response Surface Methodology*, pp. 565–614. Cham: Springer International Publishing.

Deb, K. and H. Gupta (2006, dec). Introducing robustness in multi-objective optimization. *Evolutionary Computation 14*(4), 463–494.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation 6*(2), 182–197.

Ehrett, C., D. A. Brown, E. Chodora, C. Kitchens, and S. Atamturktur (2019). Coupling material and mechanical design processes via computer model calibration.

Ezzat, A. A., A. Pourhabib, and Y. Ding (2018, jul). Sequential Design for Functional Calibration of Computer Models. *Technometrics 60*(3), 286–296.

Gelfand, A. E. and A. F. M. Smith (1990, jun). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association 85*(410), 398–409.

Geman, D. and B. Jedynak (1996). An active testing model for tracking roads in satellite images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*(1), 1–14.

Higdon, D., M. Kennedy, J. C. Cavendish, J. A. Cafeo, and R. D. Ryne (2004). Combining field data and computer simulations for calibration and prediction. *SIAM Journal on Scientific Computing 26*(2), 448–466.

Jones, D. R., M. Schonlau, and W. J. Welch (1998). Efficient Global Optimization of Expensive Black-Box Functions. Technical report.

Kennedy, M. C. and A. O'Hagan (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B 63*(3), 425–464.

Kim, M., T. Hiroyasu, M. Miki, and S. Watanabe (2004). SPEA2+: Improving the performance of the strength pareto evolutionary algorithm 2. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 3242*, 742–751.

Lee, K. Y. and M. A. El-Sharkawi (2007, jun). *Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems*. John Wiley and Sons.

Liu, F., M. J. Bayarri, and J. O. Berger (2009). Modularization in Bayesian analysis, with emphasis on analysis of computer models. *Bayesian Analysis 4*(1), 119–150.

Mason, K., J. Duggan, and E. Howley (2017, dec). Multi-objective dynamic economic emission dispatch using particle swarm optimisation variants. *Neurocomputing 270*, 188–197.

Miguel Hernández-Lobato, J., M. A. Gelbart, R. P. Adams, M. W. Hoffman, Z. Ghahramani, J. M. Hernández-Lobato, M. A. Gelbart, R. P. Adams, M. W. Hoffman, and Z. Ghahramani Hernández-Lobato (2016). A General Framework for Constrained Bayesian Optimization using Information-based Search. Technical report.

Mockus, J., V. Tiesis, and A. Zilinskas (1978). The application of bayesian methods for seeking the extremum. *Towards global optimization 2*(117-129), 2.

Nocedal, J. and S. J. Wright (2006). *Numerical optimization*. Springer.

Olalotiti-Lawal, F. and A. Datta-Gupta (2015, sep). A Multi-Objective Markov Chain Monte Carlo Approach for History Matching and Uncertainty Quantification. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers.

Paulo, R., G. García-Donato, and J. Palomo (2012). Calibration of computer models with multivariate output. *Computational Statistics and Data Analysis 56*, 3959–3974.

Picheny, V. (2015). Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing 25*(6), 1265–1280.

Picheny, V., M. Binois, and A. Habbal (2019, jan). A Bayesian optimization approach to find Nash equilibria. *Journal of Global Optimization 73*(1), 171–192.

Regis, R. G. and C. A. Shoemaker (2004). Local function approximation in evolutionary algorithms for the optimization of costly functions. *IEEE Transactions on Evolutionary Computation 8*(5), 490–505.

Robert, C. P. and G. Casella (2004). Monte Carlo Optimization. In *Monte Carlo Statistical Methods* (2 ed.)., Chapter 5, pp. 157–204. New York, NY: Springer New York.

Villemonteix, J., E. Vazquez, and E. Walter (2009, aug). An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization 44*(4), 509–534.

Williams, B., D. Higdon, J. Gattiker, L. Moore, M. McKay, and S. Keller-McNulty (2006). Combining experimental data and computer simulations, with an application to flyer plate experiments. *Bayesian Analysis 1*(4), 765–792.

Zhou, A., B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang (2011, mar). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation 1*(1), 32–49.