Simultaneous Bayesian calibration and engineering design with an application to a vibration isolation system

Blinded version
for review

Calibration of computer models and the use of those models for design are two activities traditionally carried out separately. This paper generalizes existing Bayesian inverse analysis approaches for computer model calibration to present a methodology combining calibration and design in a unified Bayesian framework. This provides a computationally efficient means to undertake both tasks while quantifying all relevant sources of uncertainty. Specifically, compared with the traditional approach of design using parameter estimates from previously completed model calibration, this generalized framework inherently includes uncertainty from the calibration process in the design procedure. We demonstrate our approach on the design of a vibration isolation system. We also demonstrate how, when adaptive sampling of the phenomenon of interest is possible, the proposed framework may select new sampling locations using both available real observations and the computer model. This is especially useful when a misspecified model fails to reflect that the calibration parameter is functionally dependent upon the design inputs to be optimized.

Nomenclature

β Inverse correlation length for Gaussian process input
$\delta(\cdot)$ Discrepancy between model and true system
ε_c Measurement error
ε_d Discrepancy between optimal system output and $\mathbf{y}_t$
$\eta(\cdot)$ Computer model of the system of interest
ζ Damping ratio of a dynamic vibration system
$\boldsymbol{\eta}$ Vector of outputs of $\eta(\cdot)$
$\boldsymbol{\theta}_c$ True value of parameter to be calibrated
$\boldsymbol{\theta}_d$ Optimal design input
λ Marginal precision of Gaussian process
ρ Reparameterization of β
σ_c^2 Variance of ε_c
σ_d^2 Variance of ε_d
$\boldsymbol{\phi}_\delta$ Hyperparameters of Gaussian process model of $\delta(\cdot)$
$\boldsymbol{\phi}_\eta$ Hyperparameters of Gaussian process surrogate for $\eta(\cdot)$
$\mathcal{D}$ $(\boldsymbol{\eta}^T, \mathbf{y}^T)^T$ for some observations $\mathbf{y}$
$C_\eta(\cdot)$ Covariance of Gaussian process emulator of η
$C_\delta(\cdot)$ Covariance of Gaussian process model of δ
k Elastic modulus of leaf spring
$f(\cdot)$ The system of interest
g Gain of dynamic vibration system
m Mass of oscillator in dynamic vibration system
$m_\eta(\cdot)$ Mean of Gaussian process emulator of η
$m_\delta(\cdot)$ Mean of Gaussian process model of δ
T Time period of one oscillation of a dynamic vibration system
t_c Value of calibration parameter used as input in $\eta(\cdot)$
t_d Value of design variable used as input in $\eta(\cdot)$
$\mathbf{x}$ All model inputs in the operational domain of $f(\cdot)$
$\mathbf{y}_r$ Vector of observations of the system of interest
$\mathbf{y}_s$ Vector of outputs of the computer model of $f(\cdot)$
$\mathbf{y}_t$ Vector of target outcomes for the system of interest
$\mathbf{z}$ All known and/or controllable inputs of $f(\cdot)$

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 of unknown, or imperfectly known, 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 design 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. Since calibration and design are typically carried out separately, existing design techniques operate under the assumption that the model is an accurate approximation of the real system of interest. In practice, models used for design 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. We refer to this new approach as DCTO, for dual calibration to target outcomes. In addition to avoiding the idealization that the model used for design is unbiased, DCTO allows one to focus calibration efforts on regions of interest, prioritizing them over other areas of the model range. 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. Having a combined framework for calibration and design is especially of interest when those 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 [1]. Since their seminal paper, the methodology has seen numerous extensions and refinements [2–7]. Henceforth, we refer to this approach to calibration as 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) metamodel of the computer model of interest and a GP prior on the model discrepancy $\delta(\cdot)$, and using the available observations y_r of the real system to find a posterior distribution $\boldsymbol{\theta}_c, \delta(\cdot)|y_r$. 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 $\delta(\cdot)$.

Herein, we leverage the KOH framework to find a posterior distribution, not only on unknown model parameters, but also on controllable design settings. We achieve this via an approach called counterfactual Bayes. In traditional model calibration, one uses Bayes' rule to discover a posterior distribution of calibration parameters using real observations, so that the 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 of optimal design settings. The idea of counterfactual Bayes is to identify artificial observations, or target outcomes, y_t such that the resulting likelihood is highest in the optimal design region — i.e., target outcomes y_t such that their occurrence is strong evidence that the design settings are optimal. Hence, in addition to calibrating the unknown model parameters against experimental observations, one uses the KOH framework to also find a posterior distribution of design settings given the target outcomes. Given the nature of y_t, this is *de facto* a distribution of 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. And like KOH, DCTO is especially well-suited to problems that rely on black-box functions.

We may divide optimization approaches in such cases broadly into three camps [8]. Gradient-based approaches [9] 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 [10] such as evolutionary algorithms [11–13], particle swarm optimization [14, 15], and simulated annealing [16] 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 [17, 18], but these come at the cost of even further inflating the number of function evaluations required. This problem can be mitigated by relying on a surrogate model, but the resulting uncertainty quantification is accomplished by separate methods that are layered on top of the independent heuristic approach. On the other hand, our combined approach to calibration and design includes uncertainty quantification as an intrinsic aspect of the framework.

The third camp is the diverse collection of response surface methodologies (RSMs) [19] used for optimization. RSMs operate by fitting a predictive model to an existing set of model runs, to form a computationally inexpensive metamodel which is then used to explore the model output. The concept of calibration to target outcomes that is built into DCTO is an example of an RSM, using GPs for its metamodel fit. Other popular versions of RSMs include efficient global optimization (EGO) [20, 21] and stepwise uncertainty reduction (SUR) [22–28]. EGO and SUR 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 [20, 29], whereas SUR's acquisition function seeks to reduce the volume of excursion sets below the current best known solutions [24]. 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). Because they rely on sequential sampling, EGO and SUR 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. The value of exploration is in guiding future adaptive sampling to avoid settling on merely local optima. In cases where adaptive sampling is not possible, this value cannot be realized, and hence it is preferable to adopt a pure-exploitation result. As a result, although these acquisition functions constitute distributions of sampling locations, by their nature they are not interpretable as distributions of the *optimal* design settings for a given problem, and hence these distributions do not quantify uncertainty regarding the location of that optimum. By contrast, our combined approach (understood as a pure-exploitation method) quantifies remaining uncertainty regarding the location of the system optimum.

An example of an RSM more closely resembling our approach to design is described by Olalotiti-Lawal and Datta-Gupta [30]. Their approach defines a distribution which is designed to lie both on and near the Pareto front (PF) of the objective function and generates a posterior distribution which includes quantified uncertainties via Markov Chain Monte Carlo (MCMC) [31]. However, the posterior distribution in that work is designed by the authors and is not dictated by the model itself; as such, its interpretability is not entirely clear. By contrast, our approach 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 design using the KOH framework is model-driven and interpretable as uncertainty regarding the optimal values for the design inputs and the resulting system output.

The combined approach to calibration and design presented here thus contrasts with traditional approaches by providing for quantification of remaining uncertainty in the joint posterior distribution of the calibration and design inputs using an integrated framework incorporating both sets of inputs. The rest of the paper is organized as follows. Section 2 describes the difficulties involved in extending KOH into a framework that incorporates both design and calibration, illustrating this by considering the failings of a naïve method for combining the two procedures, followed by a description of the proposed DCTO framework for extending KOH. Section 3 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. We showcase the application of DCTO with sequential sampling using a synthetic example, comparing its results to that of a more traditional approach of design following calibration. In Section 4 we apply DCTO to a dynamic vibration system, using a set of experimental observations simultaneously to calibrate a finite element model and to select gain factor settings to achieve the optimal vibration isolation outcome, while demonstrating DCTO's thorough quantification of the relevant uncertainties. Section 5 concludes with discussion of the results and thoughts about future directions.*

*Note that KOH is usually conceived of as a means of calibrating a computer model with respect to a set of experimental observations. However, the KOH framework, and by extension DCTO, are applicable more generally whenever one has access to both low-fidelity and high-fidelity sources of information and seeks to calibrate the former with respect to the latter. This includes the case in which both the high-fidelity and low-fidelity sources of information are computer models (e.g. with different levels of computational expense). For ease of exposition, we follow the common convention, and present DCTO in terms of calibrating a computer model using experimental observations. Nonetheless, in some cases (such as in our discussion of sequential sampling in Section 3), the methods discussed may apply more naturally in the context of employing two computer models of varying fidelity.

2 Dual calibration to target outcomes

2.1 Separate calibration and design

The version of KOH considered here is that which finds a posterior distribution of a parameter of interest for calibration, $\boldsymbol{\theta}$, using a GP emulator with hyperparameters $\boldsymbol{\phi}_\eta$. Similarly, one may also use a GP prior with hyperparameters $\boldsymbol{\phi}_\delta$ to model discrepancy between the computer model $\eta(\cdot)$ and the true function $f(\cdot)$ 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 $\boldsymbol{\phi}_\eta = [\beta,\ \lambda]$. In our adaptation, $\boldsymbol{\theta} = (\boldsymbol{\theta}_c, \boldsymbol{\theta}_d)$ is partitioned into parameters $\boldsymbol{\theta}_c$ to be calibrated and inputs $\boldsymbol{\theta}_d$ to be optimized for design purposes. 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

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

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 $\eta()$. When calibrating to target outcomes as in DCTO, by contrast, $\mathbf{y}$ is a set of target outcomes 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 design leveraging a simulation model that also requires traditional calibration, then, one might consider combining 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 design). However, this approach will not work for two reasons. Firstly, though the matter of whether a given input is considered to be a calibration parameter is dependent upon the nature of the research problem under investigation, typically the inputs to be calibrated are not design settings under researcher control. Secondly, for successful calibration one must train one's model on observations of reality rather than on unobserved target outcomes.

Hence, model calibration and system design 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 for model-assisted design. 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 design.

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 in the operational domain, within which the calibrated model's predictions are reliable. We can write $\eta_c(\mathbf{z})$ as $\eta_c(\mathbf{x}, \boldsymbol{\theta}_d)$. Then one can perform design 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 design. A crucial difference between calibration and design is that for the design 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 that is a point mass at 0.

A problem with the above-described approach of performing calibration prior to a separate design optimization is that relying on static calibration estimates $\widehat{\boldsymbol{\theta}}_c$ ignores uncertainty remaining after calibration with respect to the true value of $\boldsymbol{\theta}_c$ (if one takes a Bayesian view, one may prefer to consider calibration as finding optimal settings for inducing a model to approximate reality, rather than as finding true values of some parameter; for ease exposition, we will continue to use the latter phrasing). In order to produce results that take into account all sources of uncertainty, it is necessary to integrate calibration and design, so that the uncertainty remaining from calibration is propagated through the design process. This can be accomplished either asynchronously (so that the posterior distribution of $\widehat{\boldsymbol{\theta}}_c$ is sampled while undertaking design) or, for lower computational overhead, synchronously (so that a single MCMC run is used to perform both calibration and design). In either case, it will be useful to produce an integrated model for the combined tasks of calibration and design which describes the use of both procedures, and which makes clear the relationship between them. This integrated model will also serve to demonstrate the unified framework underlying the combined approach.

2.2 Integrated model

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, and $\mathbf{x}$ denotes the remaining 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. We 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 possible outputs of a model that lacks ε_d. Including ε_d ensures that there is nonzero probability of an observation falling in the vicinity of the targets. 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 (depending on the choice of targets) 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 the settings of the 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 we wish to observe at $\mathbf{x_t}$. Finally, let $\mathbf{y} = (\mathbf{y_s}^T, \mathbf{y_r}^T, \mathbf{y_t}^T)^T$, and $\mathbf{1}$ a vector of ones. Then it follows that $\mathbf{y} \sim N(\mathbf{m}, \mathbf{C})$, where

$$\mathbf{m} = \begin{pmatrix} m_s(\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}) \\ m_s(\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}) + m_\delta(\mathbf{x_r}, \mathbf{t_{dr}}) \\ m_s(\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T) + m_\delta(\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T) \end{pmatrix}, \tag{2}$$

$$\mathbf{C} = \begin{pmatrix} \mathbf{C_{11}} & \mathbf{C_{12}} & \mathbf{C_{13}} \\ \mathbf{C_{21}} & \mathbf{C_{22}} & \mathbf{C_{23}} \\ \mathbf{C_{31}} & \mathbf{C_{32}} & \mathbf{C_{33}} \end{pmatrix}, \tag{3}$$

$$\mathbf{C_{11}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}})\right) \tag{4}$$

$$\mathbf{C_{21}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}})\right) \tag{5}$$

$$\mathbf{C_{31}} = C_\eta\left((\mathbf{x_s}, \mathbf{t_{cs}}, \mathbf{t_{ds}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right) \tag{6}$$

$$\mathbf{C_{12}} = \mathbf{C_{21}}^T \tag{7}$$

$$\mathbf{C_{22}} = C_\eta\left((\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}), (\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}})\right) + C_\delta\left((\mathbf{x_r}, \mathbf{t_{dr}}), (\mathbf{x_r}, \mathbf{t_{dr}})\right) + \sigma_c^2\mathbf{I} \tag{8}$$

$$\mathbf{C_{32}} = C_\eta\left((\mathbf{x_r}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{t_{dr}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + C_\delta\left((\mathbf{x_r}, \mathbf{t_{dr}}), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T)\right) \tag{9}$$

$$\mathbf{C_{13}} = \mathbf{C_{31}}^T \tag{10}$$

$$\mathbf{C_{23}} = \mathbf{C_{32}}^T \tag{11}$$

$$\mathbf{C_{33}} = C_\eta\left((\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_c^T, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + C_\delta\left((\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T), (\mathbf{x_t}, \mathbf{1}\boldsymbol{\theta}_d^T)\right) + \sigma_c^2\mathbf{I} + \sigma_d^2\mathbf{I} \tag{12}$$

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 an extension of the KOH framework to include design using target outcomes.

A primary benefit of DCTO is that the design process includes quantification of all sources of uncertainty. Perform-

ing calibration and then subsequently undertaking design using static estimates for $\widehat{\boldsymbol{\theta}}_c$ and $\widehat{\delta}$ does not properly account for the uncertainty surrounding the estimates. Another benefit of the combined approach appears in cases in which the model is misspecified in failing to account for functional dependence of $\boldsymbol{\theta}_c$ on $\boldsymbol{\theta}_d$. 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 3.

For DCTO, we employ modularity in the manner of [32]. A modular analysis intentionally falls short of being a full Bayesian analysis, either for computational benefits, or to quarantine "suspect" aspects of the model, so that the posterior distributions of parameters of interest are robust to model misspecification. 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 $\boldsymbol{\theta}_c$. To modularize DCTO, we estimate the emulator hyperparameters via maximum likelihood, and we refrain 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 $\mathbf{y} = (\mathbf{y}_s^T, \mathbf{y}_r^T, \mathbf{y}_t^T)^T$, we 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 $\mathbf{y}_t$.

3 Dependence of θ_c on θ_d

3.1 Background

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 [33–35]. 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 [33, 34, 36].

In a case where 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. When calibration and design are undertaken simultaneously, as in DCTO, 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 design interest, so that the calibration takes on values that are most applicable in that region.

When observations may be made adaptively, other RSM approaches such as EGO [20, 21] or SUR [22–28] may be more efficient than the KOH framework for estimating optimal design settings, though the KOH framework offers more interpretable and model-driven uncertainty quantification. Further, 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. 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. The process of performing DCTO with adaptive sampling is described in Algorithm 1. When adaptively evaluating the objective function, the locations of the input settings $\mathbf{x}_i$ which are not being optimized for design can be selected 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. The result of applying this algorithm 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 $\theta_c(\theta_d)$.

Algorithm 1: DCTO with adaptive sampling

1 Set $\mathbf{y} = [\mathbf{y}_r^T \mathbf{y}_t^T]^T$ where $\mathbf{y}_t$ are the target outcomes and $\mathbf{y}_r = [\,]$ is an empty array.
2 Begin MCMC burn-in. Set $i = 1$. Let m be the budget of function evaluations. While $i \leq m$:
  2.1 Complete n iterations of MCMC burn-in (where e.g. $n = 100$).
  2.2 Draw $\widehat{\theta}_d$ from the available size $n \cdot i$ sample of $t_d|\mathbf{y}$.
  2.3 Evaluate $f(\mathbf{x}_i, \widehat{\theta}_d)$.
  2.4 Set $\mathbf{y}_r = [\mathbf{y}_r^T f(\mathbf{x}_i, \widehat{\theta}_d)]^T$.
3 Continue burn-in until convergence.
4 Draw a sample of desired size from the posterior distributions of θ_c, θ_d.

3.2 Simulated example

To demonstrate the use of DCTO with adaptive sampling in a case of functional dependence of the calibration parameter on design settings, we apply the method to a simulated system described by the function of three inputs

$$f_0(x, t_c, t_d) = x/(t_d^{t_c - 1} \exp(-0.75 t_d) + 1). \qquad (13)$$

Figure 1 shows the output of this function for $x = 1$ over the range $(t_c, t_d) \in [1.5, 4.5] \times [0, 5]$. For any value of x and t_c, the optimal (minimizing) value of t_d is $(4/3)(t_c - 1)$. Suppose

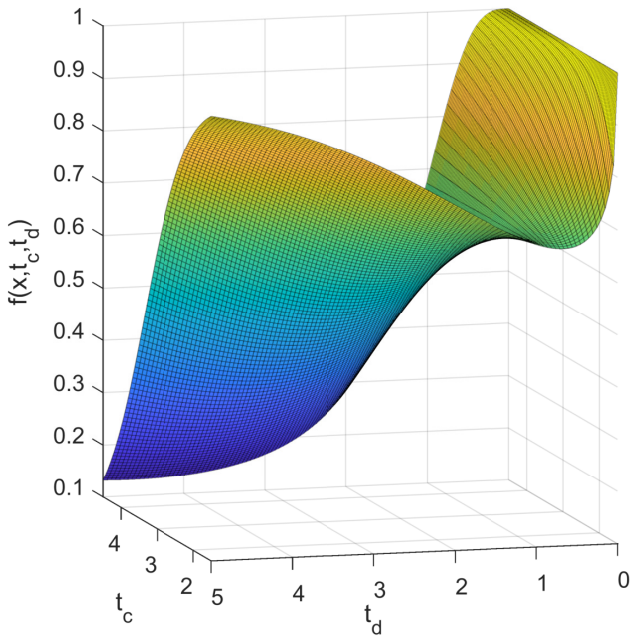

Fig. 1: Example computer model output over the support of the calibration parameter t_c and the design parameter t_d.

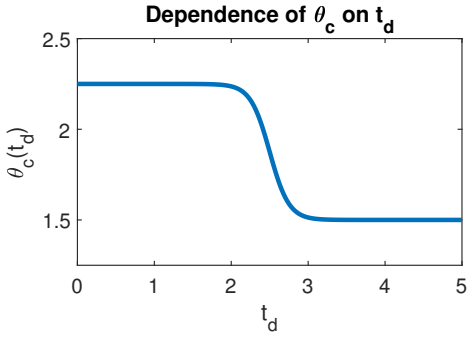

Fig. 2: True value of the calibration parameter θ_c for each value in the domain of t_d

that the calibration parameter's "true" value is functionally dependent on the design input, with the relationship:

$$\theta_c(t_d) = 2.25 - .75\frac{\exp\left(40\left(\frac{t_d-1.5}{.75}-.5\right)\right)}{1+\exp\left(40\left(\frac{t_d-1.5}{.75}-.5\right)\right)} \tag{14}$$

which would be unknown in a real application. Figure 2 shows this relationship. Figure 3 shows the locations of the true value of θ_c and the optimal value of θ_d with respect to the function $f(x,\theta_c(t_d),t_d)$ of the design input t_d (where that optimum is independent of the value of x). In Figure 3 it is clear that the true value of θ_c is far from optimal, in the sense that if this value *were* within our control (which, being a calibration parameter, it is not), we would prefer to place it at the upper end of its support, at 4.5. Thus η showcases the

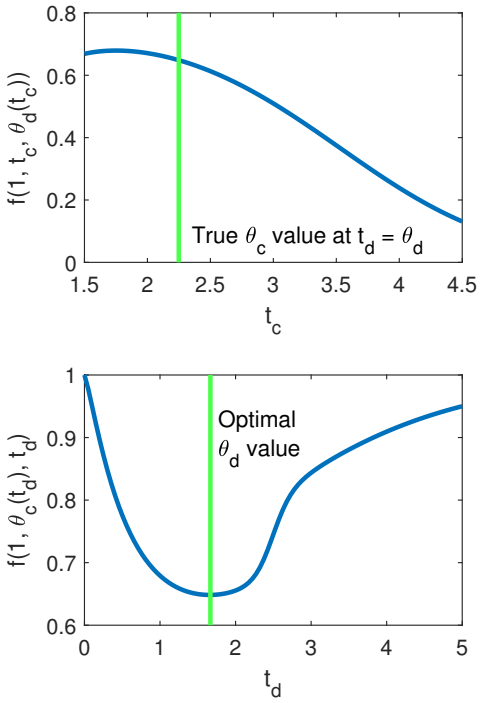

Fig. 3: The top plot shows the computer model output at $x = 1$ and optimal design setting for each value of the calibration parameter t_c. The bottom plot show the model output at $x = 1, t_c = \theta_c(t_d)$ for each value of the design parameter t_d.

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.

We apply DCTO to four versions of the problem. First, we assume that η is free from discrepancy; i.e. that $\eta(x,\theta_c,t_d)$ is an unbiased estimator of the "true" system $f_0(x,t_d)$, described by Equation 13 above. 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

$$f_1(x,t_d) = \eta(x,\theta_c,t_d)\,(1-a(x-.5)(x-1)/x)) \tag{15}$$

$$f_2(x,t_d) = \eta(x,\theta_c,t_d) - a(x-.5)(x-1)\left(t_d-\frac{4}{3}\right)^2 + b \tag{16}$$

$$f_3(x,t_d) = \eta(x,\theta_c,t_d) + axt_d + b \tag{17}$$

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 and a. This discrepancy does not affect the optimal value of t_d. The discrepancies of f_2 and f_3 are both additive. Figure 4 shows the discrepancies for two different versions (corresponding to different settings of (a,b)) of each f_i.

We apply DCTO with and without adaptive sampling to each of seven cases, without using an emulator: the non-discrepancy case, and the two different versions of each f_i

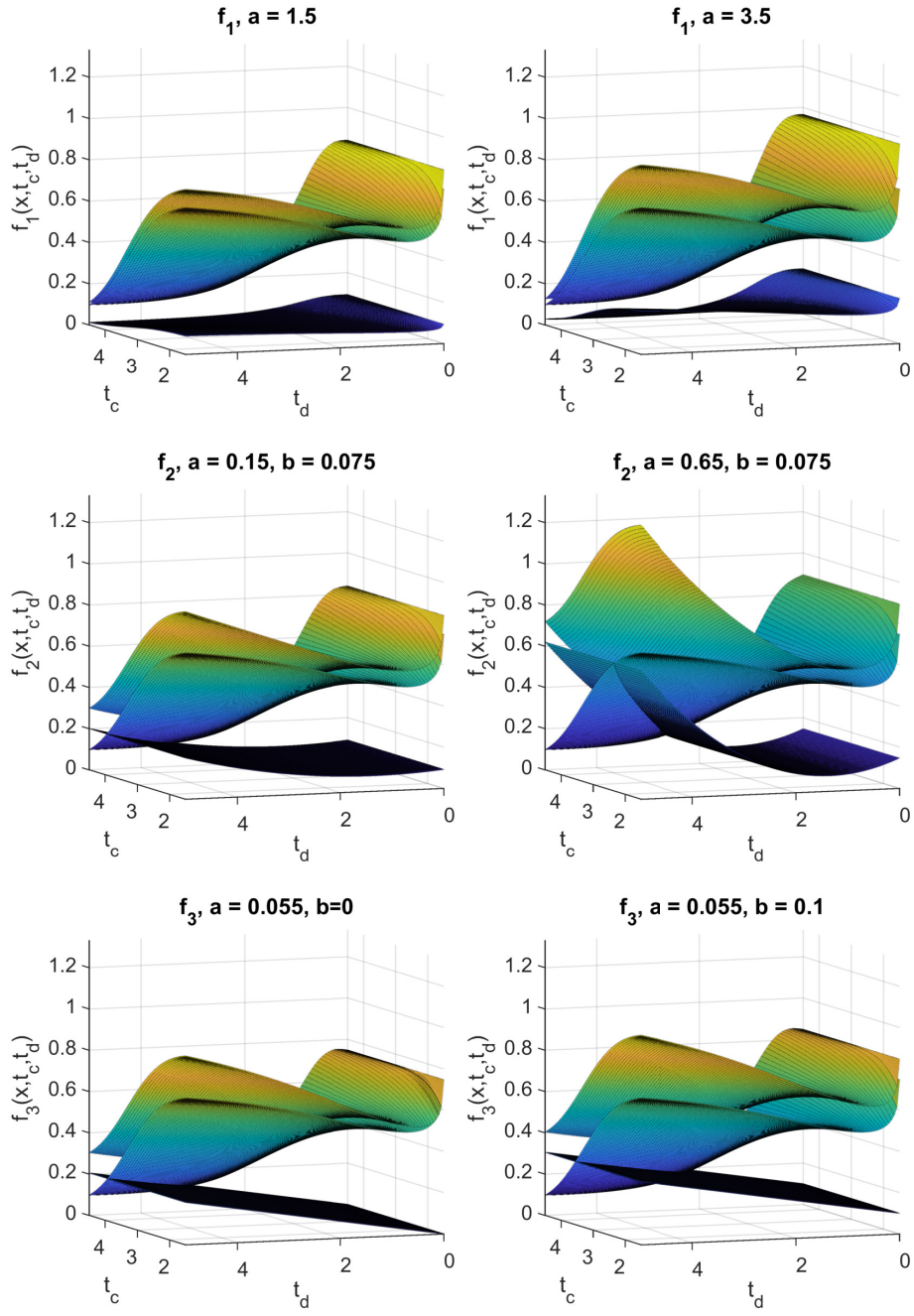

Fig. 4: The i^{th} row shows f_i (the objective function with discrepancy), η (the computer model, identical in each panel), and the discrepancy $f_i - \eta$, all at $x = 0.75$. In each row, a less aggressive version of the discrepancy appears on the left, and a more aggressive on the right. In each plot, the topmost surface is f_i, the middle surface is η, and the bottom surface is the discrepancy $f_i - \eta$.

shown in Figure 4. In each case, we gather 20 "observations" of f_i on a latin hypercube design over the supports of x and t_d, setting θ_c equal to its "true" value of $\theta_c(t_d)$. After standardizing the response to have mean 0 and standard deviation 1, we add i.i.d. N(0,0.05) noise to the response. An example of the resulting "observations" from non-adaptive DCTO, with noise, appears in Figure 5. We carry out DCTO using Metropolis-Hastings-within-Gibbs MCMC, drawing 8000 realizations each (discarding the first 4000 as burn-in) of $t_c, t_d, \boldsymbol{\rho}_\delta, \lambda_\delta, \sigma_d^2$, where $\boldsymbol{\phi}_\delta = (\boldsymbol{\rho}_\delta^T, \lambda_\delta)^T$. For the adaptive sampling application of DCTO, we begin the MCMC with 0 observations of f_i, making a new observation after every 100 steps of MCMC until we reached the total budget of 20. An example of the resulting difference between the adaptive sampling approach and relying on a space filling design, with regard to the sampling distribution of our observations

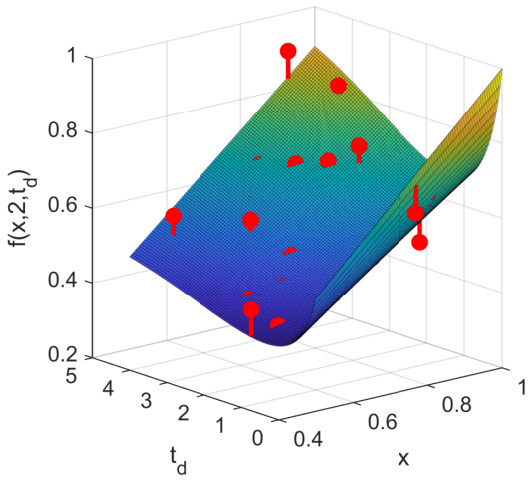

Fig. 5: Noisy observations of the system, and the true system mean, for $f = f_0$ (no discrepancy).

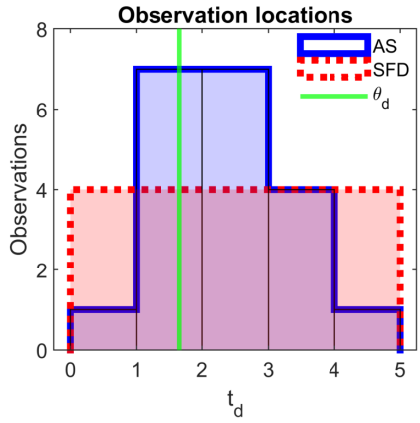

Fig. 6: Design input values for observations made under the adaptive sampling approach (AS) and under a space-filling design (SFD), along with the optimal value θ_d visible as the system global minimum in Figure 3.

of the objective function, appears in Figure 6. There, one can see that the adaptive sampling approach manages to expend its budget on observations that are near to the region of design interest. This explains the superior performance of adaptive sampling (discussed below) in both design and in calibration (since the value of the calibration parameter is dependent upon that of the design input). This ameliorative effect would likely be even greater in a higher-dimensional case, in which a space-filling design would (due to the curse of dimensionality) tend to generate observations even farther from the region of design interest.

In both versions of DCTO, we modularize 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 $(\mathbf{y}_s^T, \mathbf{y}_r^T, \mathbf{y}_t^T)^T$. Convergence was verified visually and by the Gelman-Rubin statistic (≈ 1.01; [37]).

Table 1: 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 $i = c, d$. For each f_i, a and b control the size of the discrepancy as specified in Equations (15,16,17).

	$\widehat{\theta}_c$ RMSE		$\widehat{\theta}_d$ RMSE	
Objective	AS	SFD	AS	SFD
f_0 (no discrepancy)	0.188	0.433	0.163	0.479
$f_1,\ a = 1.5$	0.233	0.32	0.243	0.414
$f_1,\ a = 3.5$	0.188	0.247	0.213	0.393
$f_2, a = .15, b = .075$	0.221	0.263	0.187	0.348
$f_2, a = .65, b = .075$	0.228	0.16	0.183	0.206
$f_3, a = .055, b = 0$	0.452	0.506	0.182	0.329
$f_3, a = .055, b = .1$	0.448	0.468	0.167	0.292

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 7, 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 1.

The results show superior performance for the adaptive sampling DCTO over DCTO using a space-filling design of experiments for the true phenomenon (or high-fidelity model, in a case of calibrating a low-fidelity model to use for design purposes). The adaptive DCTO posterior means have lower RMSEs in all cases for θ_d, and in all cases except one for θ_c. Though both models suffer from the misspecification of treating as constant a calibration parameter that is more properly understood as functionally dependent upon other model inputs (particularly, e.g., in the case of f_3), the adaptive form of DCTO is consistently more robust to these difficulties. 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.

4 Case study application: vibration isolation design

The application of DCTO methodology is demonstrated on a vibration isolation design problem. Vibration isolation relies on the balance of inertia, damping, and stiffness properties where, in active vibration isolation, an additional ac-

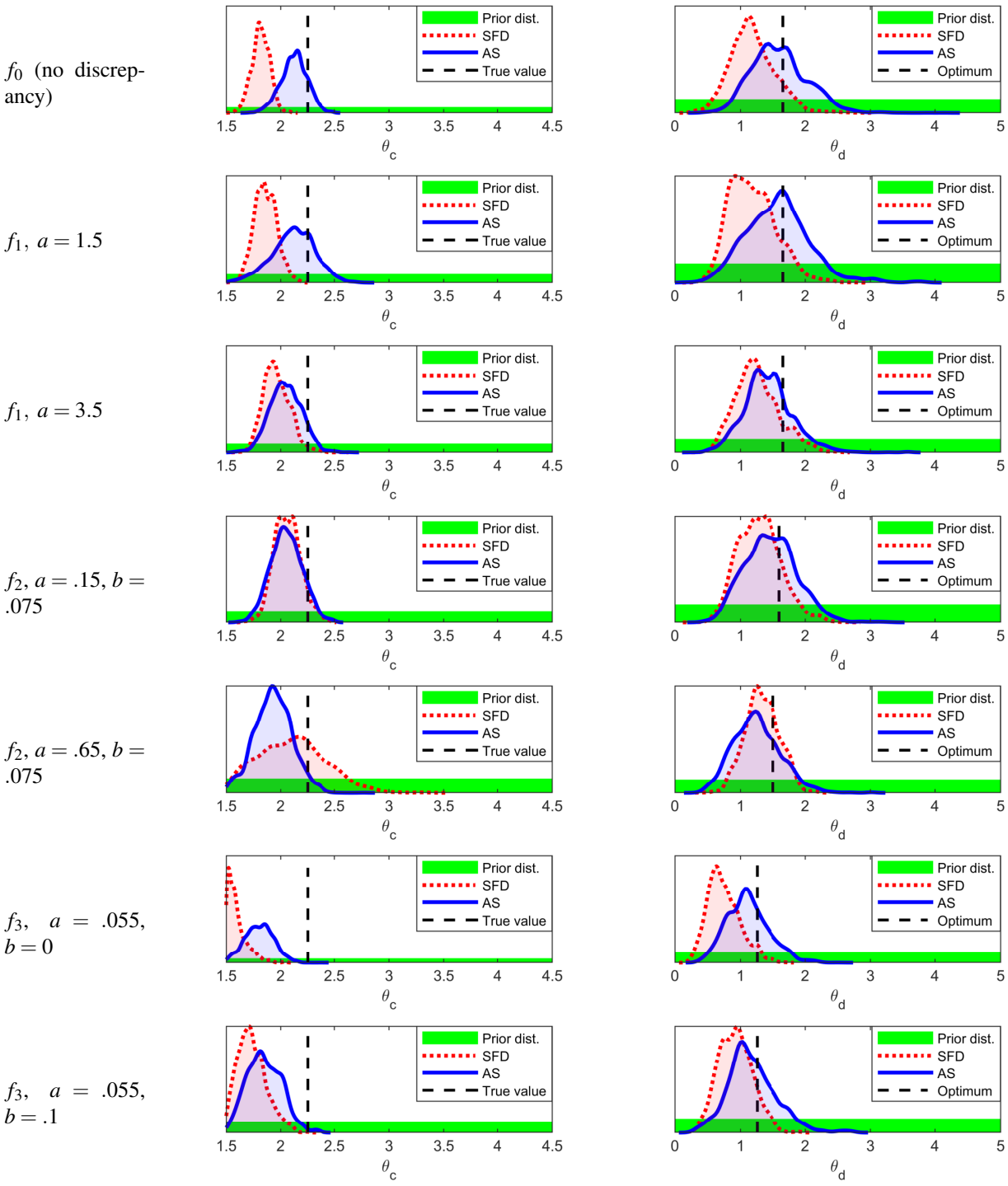

Fig. 7: Prior and posterior distributions of the calibration parameter θ_c and design parameter θ_d, along with their true/optimal values, for DCTO with adaptive sampling (AS) and with predetermined space-filling design (SFD) in each of the cases studied. For each f_i, a and b control the size of the discrepancy as specified in Equations (15,16,17).

tive gain factor enhances the system's damping behavior. To achieve the optimal vibration isolation outcome, the design engineer typically specifies the resonance and isolation frequencies and then balances mass, damping, stiffness, and the gain factor.

4.1 Case study problem

The experimental dynamical system studied herein is a one-mass oscillator subjected to passive and active vibration isolation. The system consists of a rigid rectangle frame, a rigid mass held by four identical orthogonally placed leaf springs mounted to the frame, and a voice coil actuator

(VCA) for passive and active damping.

In Figure 8a and as a simplified model, a rigid mass m oscillates in the z-direction due to a base point excitation $w(t)$. A damper with the damping coefficient b and a spring element with a stiffness constant k connect the mass to the base point. The damper and spring provide the system's internal passive damping force, active damping force, and stiffness force $F_b = b[\dot{z}(t) - \dot{w}(t)]$, $F_a = -g\dot{z}(t)$, $F_k = k[z(t) - w(t)]$ with F_a derived from a simple velocity feedback control with the gain factor g.

The inhomogeneous differential equation of the one-mass oscillator's motion in Figure 8a can be written as

$$\ddot{z}(t) + \left[2D_p\omega_0 + \frac{g}{m}\right]\dot{z}(t) + \omega_0^2 z(t) = 2D_p\omega_0\dot{w}(t) + \omega_0^2 w(t) = \omega_0^2 r(t) \tag{18}$$

using the abbreviation $2D_p\omega_0 = \frac{b}{m}$, and $\omega_0^2 = \frac{k}{m}$ including the damping ratio D_p from passive damping, with $0 < D_p < 1$, and the angular eigenfrequency ω_0. The term $\omega_0^2 r(t)$ in (18) is the excitation function, which, in this case, is the linear combination of the damper and spring base point excitation $2D_p\omega_0\dot{w}(t) + \omega_o^2 w(t)$.

Figure 8b depicts the laboratory set-up used in this study, in which a rigid frame with mass m_f serves as a base point structure. The frame is fixed by a gliding support assumed to have no friction perpendicular to the z-direction. The frame is constrained by a damper with the damping coefficient b_f and springs with a total stiffness k_f in the z-direction, and in the same plane.

In the laboratory application, the frame suspends from a rigid mount via elastic straps vertical to the z-direction, allowing the frame to move freely in the z-direction as shown in Figure **??**c. The idealized damping b_f and k_f that constrain this movement are relatively small, compared to the b and k of the mass. The frame moves in a translational z-direction because of a time-dependent translational excitation displacement $w(t)$ in the z-direction. As shown in Figure 8c, the frame retains two supports that fix a leaf spring at its ends at A and C, with the effective bending length l on sides A-B and B-C, with the rigid mass m in the center position at B. The leaf spring is the practical realization of the spring elements in Figure 8a and b. Its stiffness $k^* = 12EI/l^3$ is a function of the bending stiffness EI, where E is the Young's modulus of the leaf spring made from carbon fiber reinforced polymer (CFRP), I is the geometrical moment of inertia, and l is the length of the leaf spring. Two leaf springs are mounted in parallel with length l on each side of A-B and B-C (see Figures **??**c and 8c). With four leaf springs, the total stiffness becomes $k = 4k^*$. The two supports at A and C in Figure 8c are adjustable along l to tune the leaf spring's bending deflection and therefore its effective stiffness k.

A VCA realizes an electromotive force F_{VCA} as the passive damping and the active force F_b and F_a (Figure 8c). The force sensor $S_{F_{\mathrm{VCA}}}$ at B in Figure 8c measures the sum of forces F_b and F_a acting on the moving mass m. The acceleration sensors $S_{a;z}$ and $S_{a;w}$ measure directly the accelerations of mass and frame, $\ddot{z}$ and $\ddot{w}$. The accelerations are transformed into velocities $\dot{w}$ and $\dot{z}$ by numerical integration in the Simulink-dSpace environment. The masses of $S_{a;z}$, $S_{F_{\mathrm{VCA}}}$ and parts of the leaf spring are included in mass m (Table 2).

Figure 8c also shows a modal hammer with a force sensor S_F to excite the frame. The hammer creates the impulse force

$$\hat{F}(t_0) = \int_{-\infty}^{\infty} F(t)\cdot\delta(t-t_0)dt, \tag{19}$$

including the Dirac-impulse function $\delta(t-t_0)$ that leads to the vibrational response of the frame

$$w(t) = \frac{\hat{F}(t_0)}{m_f\omega_{D,f}}\cdot e^{-D_f\omega_{0,f}t}\sin\omega_{D,f}t, \tag{20}$$

in the time domain, with damping ratio D_f, angular eigenfrequency $\omega_{0,f}$ and damped angular eigenfrequency $\omega_{0,f}$ of the frame's movement in z-direction. (20) is only valid for low damping $0 < D_f < 1$. This leads to the total vibration response $z(t) = r_0\{1 - e^{-D_{\omega_0}t}[\cos\omega_D t - D\frac{\omega_0}{\omega_D}\sin\omega_D t]\}$.

The particular solution r_0 is part of the general excitation function $\omega_o^2 r(t)$ in (18), which takes the form of an excitation step function $r(t) = r_o\sigma(t-t_0)$ when multiplied with the unit step function $\sigma(t-t_0)$ as the integral of the Dirac-impulse function $\delta(t-t_0)$ in (19). From the relation $2D_p\omega_0\dot{w}(t) + \omega_0^2 w(t) = \omega_0^2 r(t)$ in (18), it follows that $r_0 = \frac{1}{\omega_0}2D_p\dot{w}_0 + w_0$ with the velocity $\dot{w}_0$ and displacement w_0 at $t = t_0$ that are derived from (20).

In this demonstration, the design problem is formulated with the gain factor g being the design parameter θ_d. The elastic modulus E of the four leaf spring is assumed to be poorly known and is assigned as the calibration parameter θ_c. The mass m of the system that needs to be vibration isolated is treated as the control parameter x, and the damping ratio is the design objective y.

4.2 Experimental observations

For the dual model calibration, 12 operational conditions for the test rig are designed for varying values of the mass m and gain factor g (shown in Table 3). To excite the test rig, an impulse force is applied in the translational z-direction via a modal hammer. The time history response of the hammer excitation is shown in the left panel of Figure **??**. The right panel shows the acceleration response $\ddot{z}(t)$ of the mass, as measured by the acceleration sensor $S_{a,z}$. Since the rigid frame is constrained by a spring of small stiffness in the z-direction, the resulting relatively low resonance frequency of the frame ($\approx$1.5 Hz) does not significantly affect the mass vibration with its higher eigenfrequency (>20 Hz) when vibrating in the z-direction. The low frequency content is filtered out in the measurement chain. The hammer impact is repeated 5 times, and the impact force and the system response measurements are averaged.

One significant character of an oscillatory system is its damping (i.e. how rapidly a vibration system will decay after

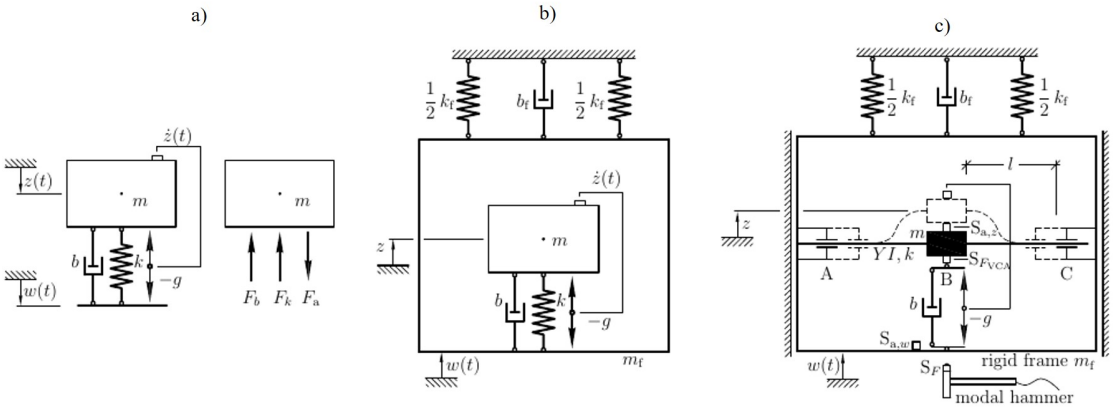

Fig. 8: Schematic diagram of the test rig for the dynamic vibration system: (a) simplified schematic representation of the one-mass oscillator, (b) one-mass oscillator with an additional frame as the base point, (c) schematic representation of the real test setup.

Table 2: Geometrical, mass, and material values of each component in the vibration isolation test rig

Category	Property	Variable	Value	Unit
Rigid frame structure	sum mass	m_f	6.2073	kg
Vibrating rigid mass	sum mass, **min**	m	0.7853	kg
	20x add. weights, small	m_{ws}	0.0760	kg
	24x add. weights, large	m_{wl}	0.2880	kg
	sum mass, **max**	m	1.1493	kg
Geometry	leaf spring length, **min**	l	0.04	m
	leaf spring length, **max**	l	0.08	m
	leaf spring cross section, width	d	0.04	m
	leaf spring cross section, height	h	0.11	m
Material	Elastic modulus	E	$6.2 \cdot 10^9$	N/m^2
	stiffness CFRP, **min**	k	25,788.1	N/m
	stiffness CFRP, **max**	k	206,305.0	N/m
VCA	passive damping coefficient, **min**	b	16	Ns/m
	passive damping coefficient, **max**	b	130	Ns/m
	passive damping ratio, **min**	D_p	0.0481	-
	passive damping ratio, **max**	D_p	0.628	-
	active gain factor, **min**	g	0	Ns/m
	active gain factor, **max**	g	95	Ns/m

the initial excitation). The damping ratio is a dimensionless measure that describes the damping level, and is calculated as $D = \left(1 + (2\pi/\delta)^2\right)^{-1/2}$, where $\delta = \frac{1}{n}\ln(\ddot{z}(t)/\ddot{z}(t+nT))$, $\ddot{z}(t)$ is the 1st peak value of mass acceleration, $\ddot{z}(t+nT)$ is the $n+1^{\text{th}}$ peak value of mass acceleration, and n is the number of peak intervals. δ is the logarithmic decrement, which is used to compute the damping ratio D_p. By following these two equations, the system responses under various numerical simulations are summarized in Table 3.

4.3 Numerical investigation

To fully explore the domain of the control parameter in this dual model calibration problem, a finite element model of the one-mass oscillator is built in ANSYS v. 2018 (Figure 10). The frame and oscillatory mass are represented by a linear solid element type C3D8R in ABAQUS. Both the frame and the mass are assigned very high stiffness values to reflect rigid body behavior. The rigid frame is constrained in the $z-$direction of vibration by a spring of a small stiffness value, and laterally, by assumed gliding support (see Figure 8c). A passive damping force, an active damping force (that

*"Overall damping ratio" refers to the combination of active and passive damping.

Table 3: A variety of experiment tests and 5-times averaged results

Case	Control Parameter x	Design Parameter θ_d	5-times Avg. System Response y
Variable unit	Mass (kg)	Gain factor (Ns/m)	Overall damping ratio* (-)
1	1.1493	0	0.0523
2	0.9653	0	0.0481
3	0.7853	0	0.0549
4	1.1493	8	0.0798
5	0.9653	8	0.0864
6	0.7853	8	0.0871
7	1.1493	41	0.308
8	0.9653	41	0.264
9	0.7853	41	0.259
10	1.1493	95	0.542
11	0.9653	95	0.527
12	0.7853	95	0.628

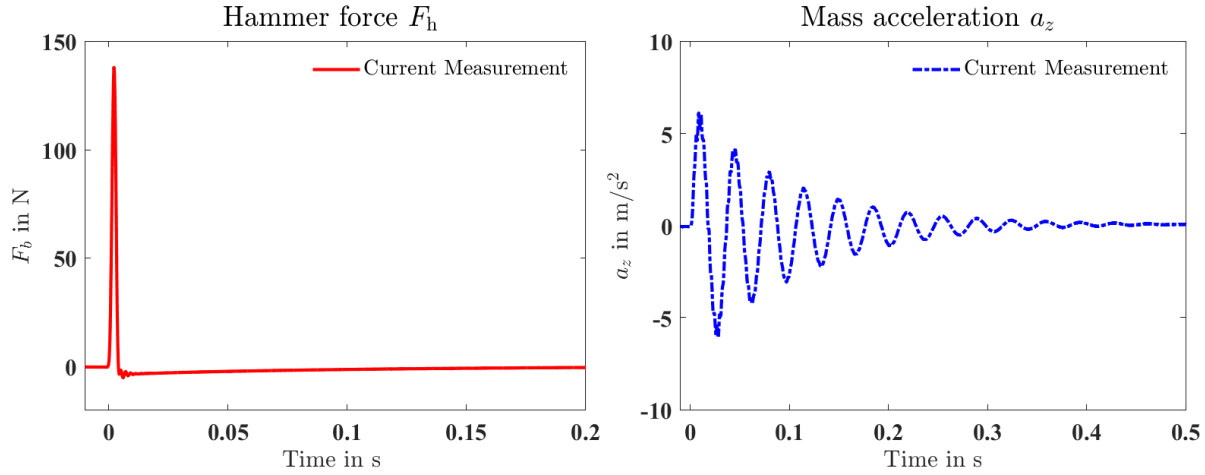

Fig. 9: Schematic diagram of the applied impulse force in the time domain (left) and five-times averaged acceleration response of the rigid mass in the time domain (right).

result from the gain factor g) and elastic forces (from the leaf springs) apply on the mass oscillator. Dashpot elements are used for the damper and gain to model velocity-dependent forces. The damper represented by DASHPOT2 element introduces a damping force as a function of the relative velocity between the rigid frame and the mass oscillator, the active damping force due to gain is modelled as a function of the absolute velocity of the mass oscillator through DASHPOT1 element, and the spring is represented by SPRING1 element in ABAQUS. A Latin Hypercube sampling is completed with 98 runs for parameters values partially shown in Table 4 for which the damping ratio of the system is calculated.

4.4 Application of DCTO to vibration isolation design

Since our goal is to minimize the damping ratio, we set our target outcomes y_t to be 0 across a range of oscillator masses. Specifically, we set a grid of size 8 over the range of oscillator masses present in the simulation and experimental data, with target outcome 0 for each point in that grid. We define our prior GP surrogate for the FE model using a mean function found via degree-2 polynomial regression on the available FE runs. For the hyperparameters of the surrogate's covariance function, we estimate them as MLEs using the quasi-Newton BFGS method [38]. We perform 10,000 iterations of MCMC using this surrogate and set of target observations, of which the first half are discarded as burn-in. The convergence of the resulting MCMC chains is assessed both visually and using the Gelman-Rubin statistic (≈ 1.01 and 1.001 for calibration and design respectively), [37]).

The total wall time required for the MCMC to complete DCTO in this case was 94 seconds (on a laptop with an Intel Core i7-9750H CPU and 16GB of RAM). The posterior distributions of the calibration and design inputs are are shown in Figure 11. Strong Bayesian learning has occurred, particularly for the design input. The posterior distribution of the elastic modulus for the system assigns high likelihood

Table 4: A partial parameterized input and corresponding numerical results

Case	Control Parameter x	Calibration Parameter θ_c	Design Parameter θ_d	System Response y
Variable unit	Mass (kg)	Elastic Modulus (N/m^2)	Gain factor (Ns/m)	Overall damping ratio (-)
1	0.9625	54037300000	11.5	0.0979
2	0.8175	58698200000	46.5	0.217
3	0.9525	72098300000	76.5	0.268
4	0.7275	70350500000	80.5	0.3496
5	1.0125	71515700000	73.5	0.2483
6	1.0875	64233000000	10.5	0.0815
...	...	...	...	...
93	1.0575	72389600000	54.5	0.1855
94	0.8775	68311300000	91.5	0.3621
95	0.7675	56950400000	19.5	0.1326
96	0.7525	71807000000	83.5	0.3489
97	0.8125	68893900000	27.5	0.1392
98	1.0525	48793800000	34.5	0.1679

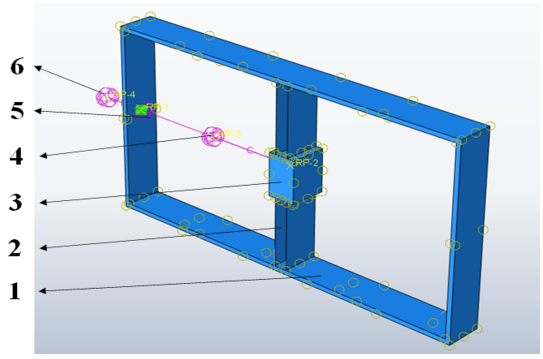

Fig. 10: The dynamic vibration system: (1) the rigid frame, (2) the leaf springs, (3) the mass oscillator, (4) the damper, (5) the active force, and (6) the spring.

to the expected value of 6.2e10, with a posterior mean of 6.188e10. For comparison with our design results, we also apply the NSGA-II algorithm [12], a gradient-free genetic algorithm, to the trained GP model surrogate. We use 100 generations and a population size 50, taking a total of 48 seconds of computation (wall time). Whereas our method performs both calibration and design, NSGA-II cannot be used for calibration, and so we apply it to a model calibrated with a point estimate (the posterior mean) of elastic modulus from our method's results. The results of NSGA-II agree with our own, in finding the optimal gain setting to be 0.

We also use the surrogate model to estimate also the posterior predictive distribution of the system after DCTO. Figure 12 shows the resulting posterior distributions of model output at various levels of oscillator mass, along with the distributions of both experimental and simulator system output. For comparison, the figure also includes the output of the surrogate model using the posterior mean of elastic modulus

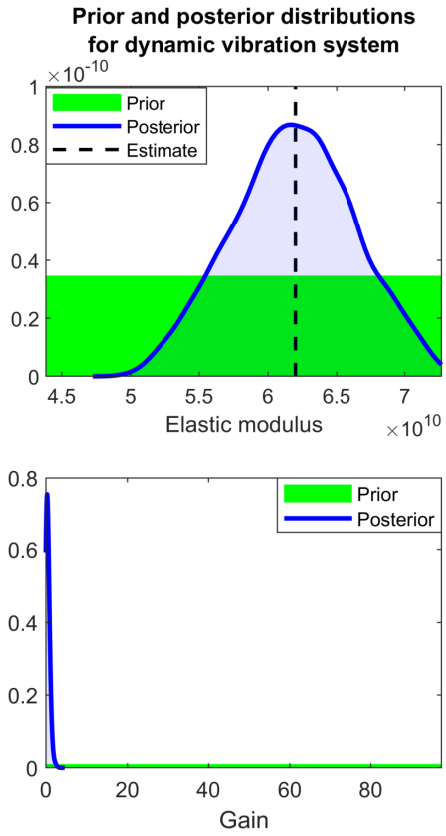

Fig. 11: The posterior distributions of the calibration and design inputs, respectively, along with their (uniform) priors. The very narrow posterior distribution of gain is concentrated at the minimum of its support.

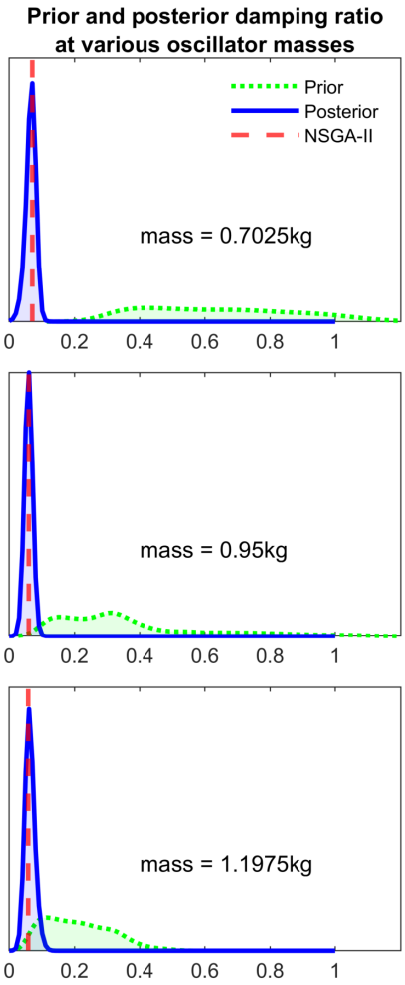

Fig. 12: The posterior distributions of the model output at three different levels of the operational domain, along with their prior distributions. The posteriors constitute a notable performance increase over the priors.

along with the NSGA-II estimate of optimal gain. Note that the predicted model outputs fall at the bottom of the ranges of observed model outputs across the domain of oscillator masses, implying a successful design outcome for the system has been achieved.

5 Conclusion

DCTO provides a method for generalizing the KOH framework for model calibration to include design. The result 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 provides a computationally efficient method of propagating the uncertainties remaining from KOH calibration through the design 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. MCMC methods struggle to converge in the context of high dimensionality; future work on this subject will include the application to DCTO of ongoing research in the area of high-dimensional MCMC [39]. Future work will also include pairing adaptive sampling DCTO with other methodologies for selecting new sampling locations, such as EGO and SUR.

References

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

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

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

[4] Bayarri, M. J., Berger, J. O., and Cafeo, J., 2007a. "Computer model validation with functional output". *The Annals of Statistics,* **35**, pp. 1874–1906.

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

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

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

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

[9] Nocedal, J., and Wright, S. J., 2006. *Numerical optimization*. Springer.

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

[11] Branke, J., Deb, K., Miettinen, K., and Slowinski, R., eds., 2008. *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Vol. 5252 LNCS. Springer.

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

[13] Kim, M., Hiroyasu, T., Miki, M., and Watanabe, S., 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**, pp. 742–751.

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

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

[16] Robert, C. P., and Casella, G., 2004. "Monte Carlo Optimization". In *Monte Carlo Statistical Methods*, 2 ed. Springer New York, New York, NY, ch. 5, pp. 157–204.

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

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

[19] Dean, A., Voss, D., and Draguljić, D., 2017. *Response Surface Methodology*. Springer International Publishing, Cham, pp. 565–614.

[20] Jones, D. R., Schonlau, M., and Welch, W. J., 1998. Efficient Global Optimization of Expensive Black-Box Functions. Tech. rep.

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

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

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

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

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

[26] Miguel Hernández-Lobato, J., Gelbart, M. A., Adams, R. P., Hoffman, M. W., Ghahramani, Z., Hernández-Lobato, J. M., Gelbart, M. A., Adams, R. P., Hoffman, M. W., and Ghahramani Hernández-Lobato, Z., 2016. A General Framework for Constrained Bayesian Optimization using Information-based Search. Tech. rep.

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

[28] Binois, M., Picheny, V., Taillandier, P., and Habbal, A., 2019. "The Kalai-Smorodinski solution for many-objective Bayesian optimization".

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

[30] Olalotiti-Lawal, F., and Datta-Gupta, A., 2018. "A multiobjective markov chain monte carlo approach for history matching and uncertainty quantification". *Journal of Petroleum Science and Engineering,* **166**, pp. 759–777.

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

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

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

[34] Atamturktur, S., Hegenderfer, J., Williams, B., Egeberg, M., Lebensohn, R. A., and Unal, C., 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), pp. 641–654.

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

[36] Brown, D. A., and Atamturktur, S., 2018. "Nonparametric functional calibration of computer models". *Statistica Sinica,* **28**(2), pp. 721–742.

[37] Gelman, A., and Rubin, D. B., 1992. "Inference from Iterative Simulation Using Multiple Sequences". *Statistical Science,* **7**(4), pp. 457–472.

[38] Fletcher, R., 2013. *Practical methods of optimization*. John Wiley & Sons.

[39] Saibaba, A. K., Bardsley, J., Brown, D. A., and Alexanderian, A., 2019. "Efficient marginalization-based MCMC methods for hierarchical Bayesian inverse problems". *SIAM/ASA Journal on Uncertainty Quantification,* **7**(3), pp. 1105–1131.