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

March 19, 2019

Abstract

Abstract

Keywords:

*The authors gratefully acknowledge *please remember to list all relevant funding sources in the unblinded version*

1 Introduction

The goal of traditional Kennedy-O'Hagan style calibration (KOH, Kennedy and O'Hagan, 2001) is to find a posterior distribution on unknown parameters by calibrating a computer model using real-world observations of the modeled phenomenon. By contrast, the design methodology of calibration to target outcomes (CTO) uses the KOH framework to find a posterior distribution on optimal input settings in the model by "calibrating" a computer model using artificial observations that reflect performance and cost targets for the modeled system. The goal of the work described here is to combine 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 θ 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.

2 The model

Let f be the model of interest, and partition its inputs as (x, θ_1, θ_2) where x denotes the known and/or controllable inputs, θ_1 denotes the parameters targeted for KOH calibration, and θ_2 denotes the input settings targeted for design via CTO. If f 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 f with mean $m_0(x, t_1, t_2)$ and covariance function $C_0((x, t_1, t_2), (x', t_1', t_2'))$. From here on in this discussion, assume that a GP surrogate is used for f. Typically, there will be some systematic discrepancy between the output of f and the true system, even if the true value of θ_1 is used. We model this discrepancy, between the true system and f at the true value of θ_1, with another GP prior δ_1 having mean $m_1(x, t_2)$ and covariance function $C_1((x, t_2), (x', t_2'))$. In addition to the discrepancy between f and the true system, there may also be a systematic discrepancy between the true system and our performance/cost targets, particularly if the latter are especially optimistic. We model this discrepancy using a GP prior with mean $m_2(x)$ and covariance function $C_2(x, x')$. Notice that this discrepancy is between the true system at optimal settings for θ_2 and f using the true value of θ_1 and the optimal settings for θ_2. In addition to systematic discrepancy between f and reality, measurement error must be included in the model. Thus let $\epsilon \sim N(0, \sigma^2)$ denote the (known) i.i.d. variance of (real) observations at any given values of x, t_2. Finally, we assume that η, δ_1, δ_2, and ϵ are all mutually independent.

A collection of simulation runs is needed to train the GP code surrogate. Let $(\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}})$ 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 the real observations at $\mathbf{x_r}, \mathbf{t_{2r}}$, and let $\mathbf{y_d}$ be a set of artificial "observations" reflecting our cost and performance targets at control settings $\mathbf{x_d}$. Call $\mathbf{y_d}$ the *target outcomes*. Finally, let $\mathbf{y} = (\mathbf{y_s}, \mathbf{y_r}, \mathbf{y_d})^T$. Then it follows that $\mathbf{y} \sim \mathrm{N}(\mathbf{m}, \mathbf{C})$,

where

$$\mathbf{m} = \begin{pmatrix} m_0(\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}}) \\ m_0(\mathbf{x_r}, \mathbf{1}\theta_1, \mathbf{t_{2r}}) + m_1(\mathbf{x_1}, \mathbf{t_{2r}}) \\ m_0(\mathbf{x_d}, \mathbf{1}\theta_1, \mathbf{1}\theta_2) + m_1(\mathbf{x_d}, \mathbf{1}\theta_2) + m_2(\mathbf{x_d}) \end{pmatrix},$$

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

$$\mathbf{C_{11}} = C_0\left((\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}}), (\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}})\right)$$

$$\mathbf{C_{21}} = C_0\left((\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}}), (\mathbf{x_r}, \mathbf{1}\theta_1, \mathbf{t_{2r}})\right)$$

$$\mathbf{C_{31}} = C_0\left((\mathbf{x_s}, \mathbf{t_{1s}}, \mathbf{t_{2s}}), (\mathbf{x_d}, \mathbf{1}\theta_1, \mathbf{1}\theta_2)\right)$$

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

$$\mathbf{C_{22}} = C_0\left((\mathbf{x_r}, \mathbf{1}\theta_1, \mathbf{t_{2r}}), (\mathbf{x_r}, \mathbf{1}\theta_1, \mathbf{t_{2r}})\right) + C_1\left((\mathbf{x_r}, \mathbf{t_{2r}}), (\mathbf{x_r}, \mathbf{t_{2r}})\right) + \sigma_2\mathbf{I}$$

$$\mathbf{C_{32}} = C_0\left((\mathbf{x_r}, \mathbf{1}\theta_1, \mathbf{t_{2r}}), (\mathbf{x_d}, \mathbf{1}\theta_1, \mathbf{1}\theta_2)\right) + C_1\left((\mathbf{x_r}, \mathbf{t_{2r}}), (\mathbf{x_d}, \mathbf{1}\theta_2)\right)$$

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

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

$$\mathbf{C_{33}} = C_0\left((\mathbf{x_s}, \mathbf{1}\theta_1, \mathbf{1}\theta_2), (\mathbf{x_s}, \mathbf{1}\theta_1, \mathbf{1}\theta_2)\right) + C_1\left((\mathbf{x_s}, \mathbf{1}\theta_2), (\mathbf{x_s}, \mathbf{1}\theta_2)\right) + C_2(\mathbf{x_d}, \mathbf{x_d})$$

Note that when $\mathbf{y_d}$ and $\mathbf{x_d}$ 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 generalizes the KOH framework.

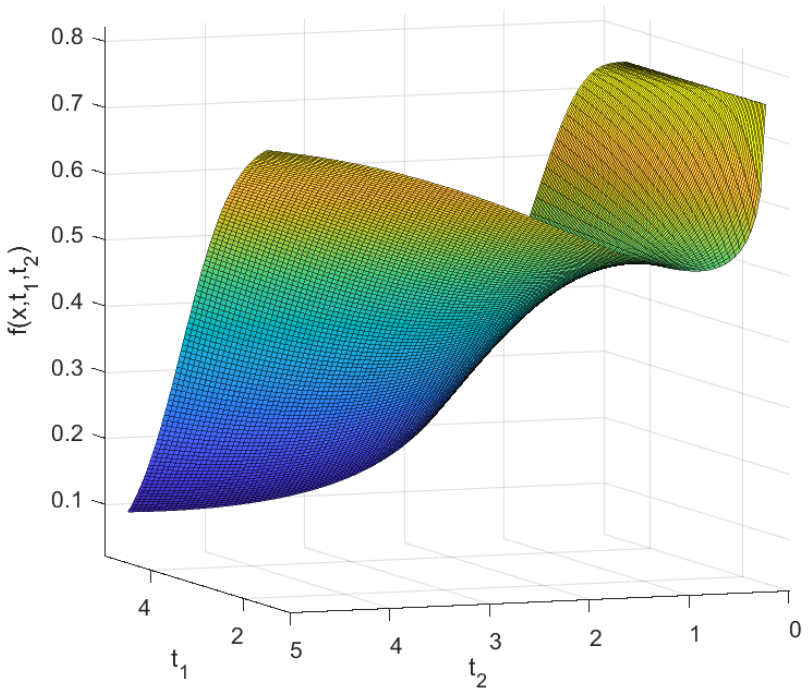

Figure 1: Example computer model output over the support of the calibration parameter t_1 and the design parameter t_2.

3 Example with simulated data

Consider the function of three inputs $f(x, t_1, t_2) = x/(t_2^{t_1-1} \exp(-0.75t_2) + 1)$. Figure 1 shows the output of this function for $x = 1$ over the range $(t_1, t_2) \in [1.5, 4.5] \times [0, 5]$. We arbitrarily set $\theta_1 = 2$ to be the "true" value of t_1. For any value of x and t_1, the optimal (minimizing) value of t_2 is $(4/3)(t_1 - 1)$, so we have $\theta_2 = 4/3$. Figure 2 shows the locations of the true and optimal values (respectively) of θ_1 and θ_2. There it is clear that the true value of θ_1 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 f 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.

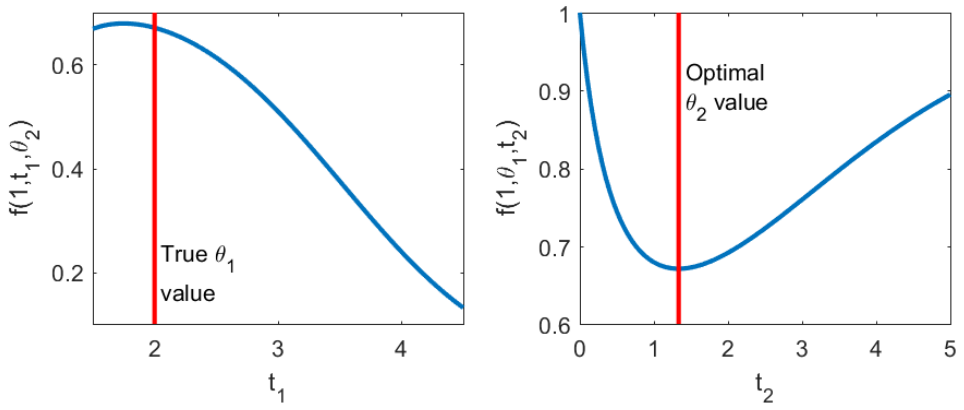

Figure 2: The lefthand plot shows the computer model output at $x = 1$ and optimal θ_2 for each value of the calibration parameter t_1 The righthand plot show the model output at $x = 1, t_1 = \theta_1$ for each value of the design parameter t_2.

3.1 Results

We used DCTO on four versions of the problem. First, we assumed that f is free from discrepancy – i.e. that $f(x, \theta_1, t_2)$ is an unbiased estimator of the "true" system $g(x, t_2)$. The other three versions each assume that f suffers from some form of discrepancy. Let g_1, g_2, g_3 denote the "true" systems in these three cases. We set

$$g_1(x, t_2) = f(x, \theta_1, t_2)\,(1 - c(x - .5)(x - 1)/x))$$

$$g_2(x, t_2) = f(x, \theta_1, t_2) - c(x - .5)(x - 1)\left(t_2 - \frac{4}{3}\right)^2 + d$$

$$g_3(x, t_2) = f(x, \theta_1, t_2) + cxt_2 + d$$

Where c, d are constants which determine how severe the discrepancy is in each case. The function g_1 has a multiplicative discrepancy dependent only on x. This discrepancy does not affect the optimal value of t_2. The discrepancy of g_2 is additive, and is dependent

upon both x and θ_1. Though this discrepancy can affect the optimal value of t_2, in the case that $\theta_1 = 2$ (which is what we assume to be the truth) it does not. Thus under g_1 and g_2, it remains the case that the optimal value of t_2 is $\theta_2 = 4/3$. By contrast, g_3 has an additive discrepancy which does affect the optimal setting for t_2. For g_3, optimal t_2 is dependent upon both the true value of θ_1 and upon the value of c. For example, for $\theta_1 = 2$ and $c = 0.055$, the optimal t_2 is $\theta_2 \approx 1$. Figure 3 shows the discrepancies for two different versions (corresponding to different settings of (c, d)) of each g_i.

We applied DCTO to each of seven cases: the non-discrepancy case, and the two different versions of each g_i shown in Figure 3. We found that in these cases, no appreciable 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 50 "observations" of g_i on a latin hypercube design over the supports of x and θ_2, setting θ_1 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 $t_1, t_2, \boldsymbol{\rho}_{\delta_1}, \lambda_{\delta_1}, \rho_{\delta_2}, \lambda_{\delta_2}$, where the latter four are the hyperparameters of the product power exponential covariance functions C_1 and C_2. Figure 4 shows the results for g_0, the case of no discrepancy. Strong Bayesian learning has occurred, in that the posterior distributions of θ_1, θ_2 are peaked around their true and optimal values, respectively. The skewness apparent in the posterior distribution of θ_2 occurs in all of the results gathered here, and is likely due to the shape of the objective function f, which increases sharply for $t_1 < 2$ and

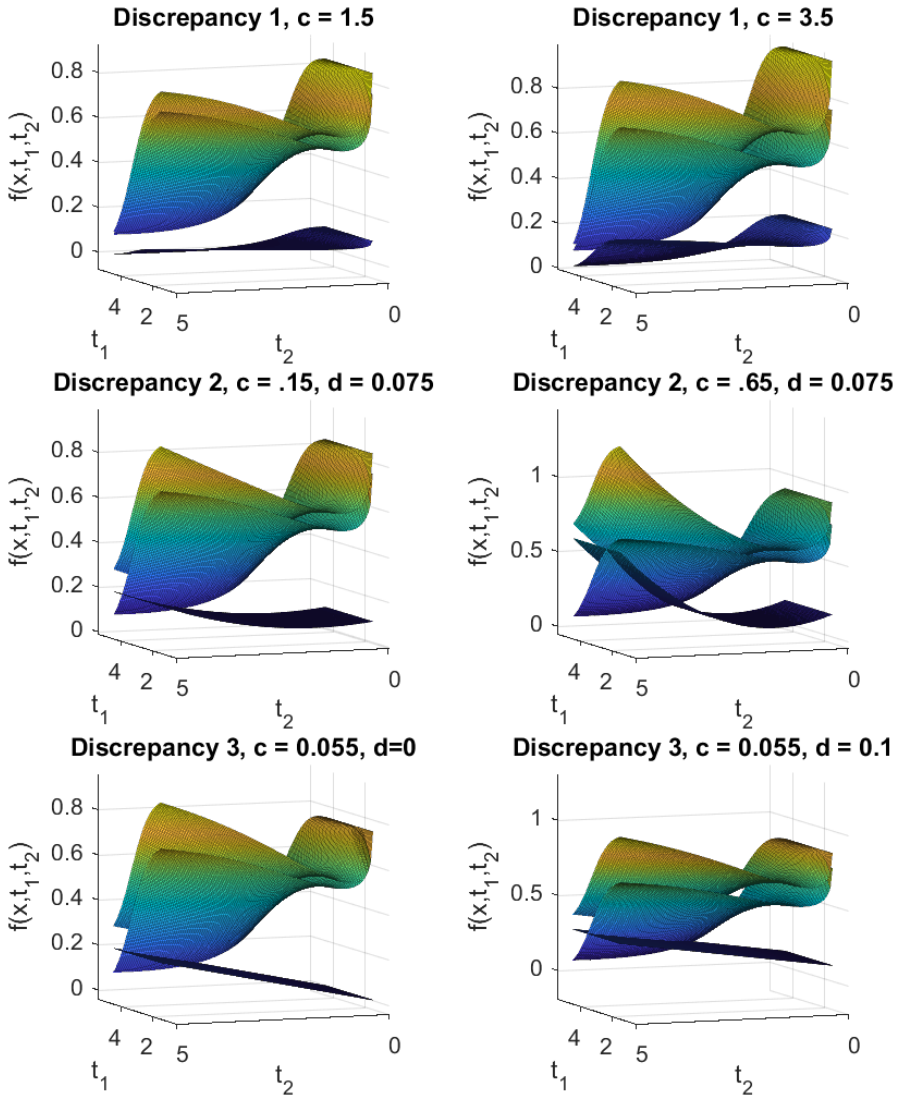

Figure 3: The i^{th} row shows g_i (the objective function with discrepancy), f (the computer model), and the discrepancy $g_i - f$. In each row, a less aggressive version of the discrepancy appears on the left, and a more aggressive on the right. In each plot, the topmost surface is g_i, the middle surface is f, and the bottom surface is the discrepancy $g_i - f$.

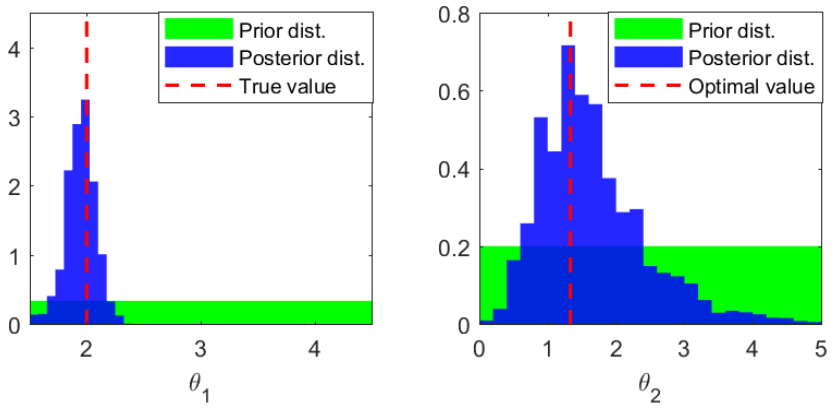

Figure 4: Prior and posterior distributions of the calibration parameter θ_1 and design parameter θ_2, along with their true/optimal values, for DCTO carried out when there is no discrepancy between the true system and the computer model.

increases much more gently for $t_1 > 2$.

Figure 5 shows the results for g_1 at two settings of c, and Figure 6 shows the results for g_2 at two settings of (c, d). Somewhat counterintuitively, even stronger Bayesian learning occurs with respect to θ_1 in the case of g_1 than in the case of g_0, in each of the two settings of c. By contrast, and less surprisingly, the posterior distributions for θ_1 are somewhat wider in the case of g_2, for each of the two settings of (c, d). Nonetheless, the posterior distributions for g_2, as for g_0 and g_1, still peak at the true value of θ_1 and at the optimal value of θ_2.

Matters change in the case of g_3. Figure 7 (upper left) shows that the true value of θ_1 is well into the tail of the posterior distribution. Surprisingly, increasing d from 0 to 0.1 and keeping $c = 0.055$, the results are significantly better, even though the discrepancy in this case is larger. In the lower left of Figure 7, we see that the posterior distribution again peaks sharply on the true value of θ_1. In all versions of the discrepancy function explored here, the posterior distribution is roughly similar with respect to θ_2; wider than

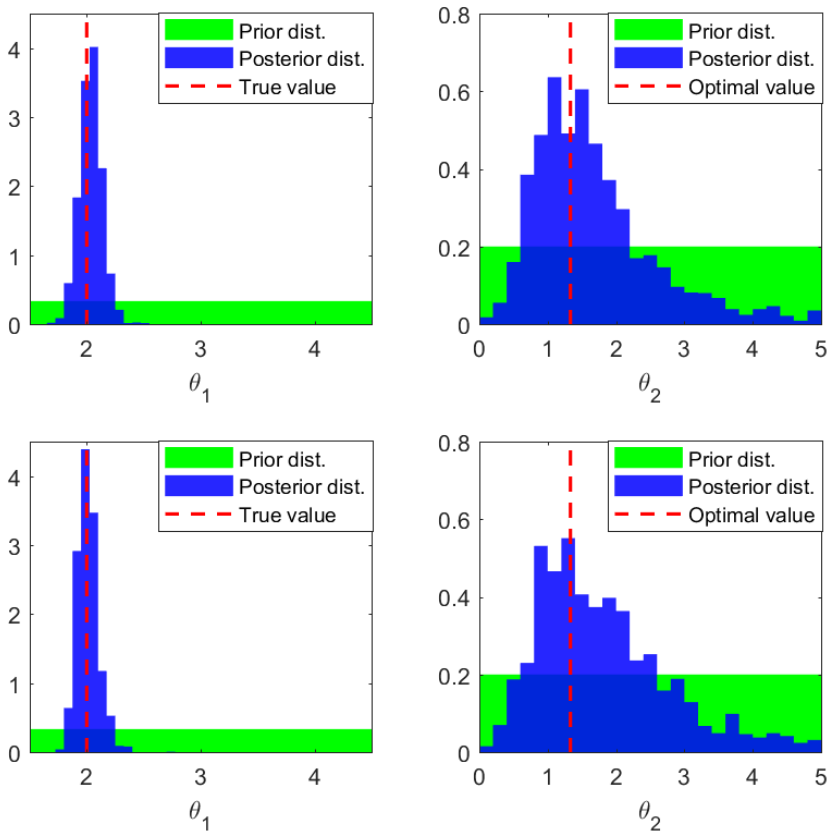

Figure 5: Prior and posterior distributions of the calibration parameter θ_1 and design parameter θ_2, along with their true/optimal values, for DCTO in the case of true systems g_1. The top row corresponds to a smaller discrepancy, with $c = 1.5$; the bottom row coresponds to a larger discrepancy, with $c = 3.5$.

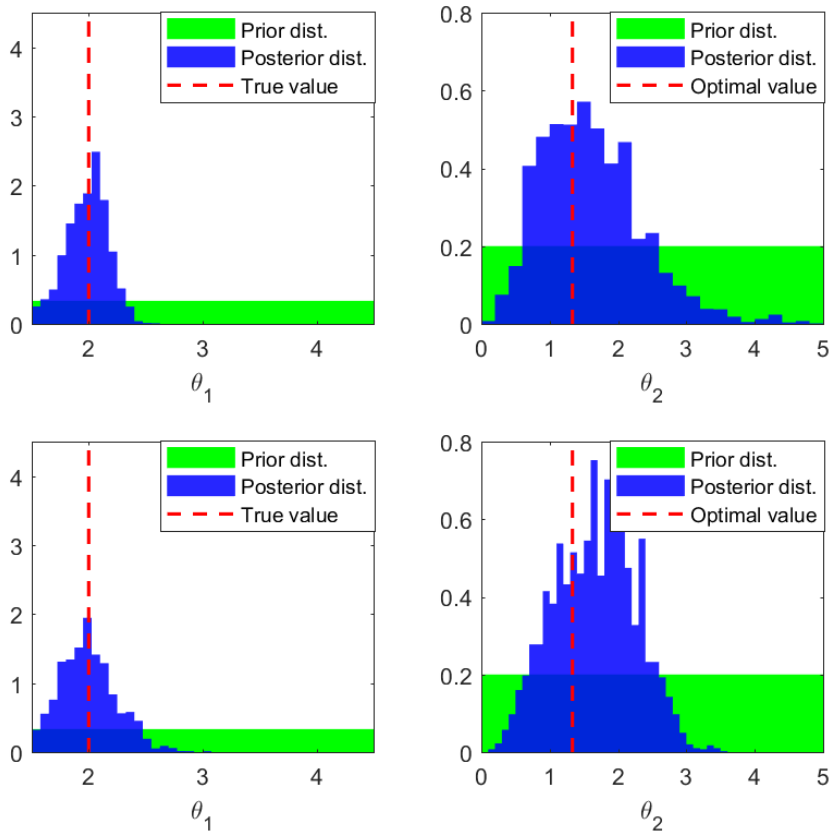

Figure 6: Prior and posterior distributions of the calibration parameter θ_1 and design parameter θ_2, along with their true/optimal values, for DCTO in the case of true systems g_2. The top row corresponds to a smaller discrepancy, with $(c, d) = (.15, .075)$; the bottom row coresponds to a larger discrepancy, with $(c, d) = (.65, .075)$.

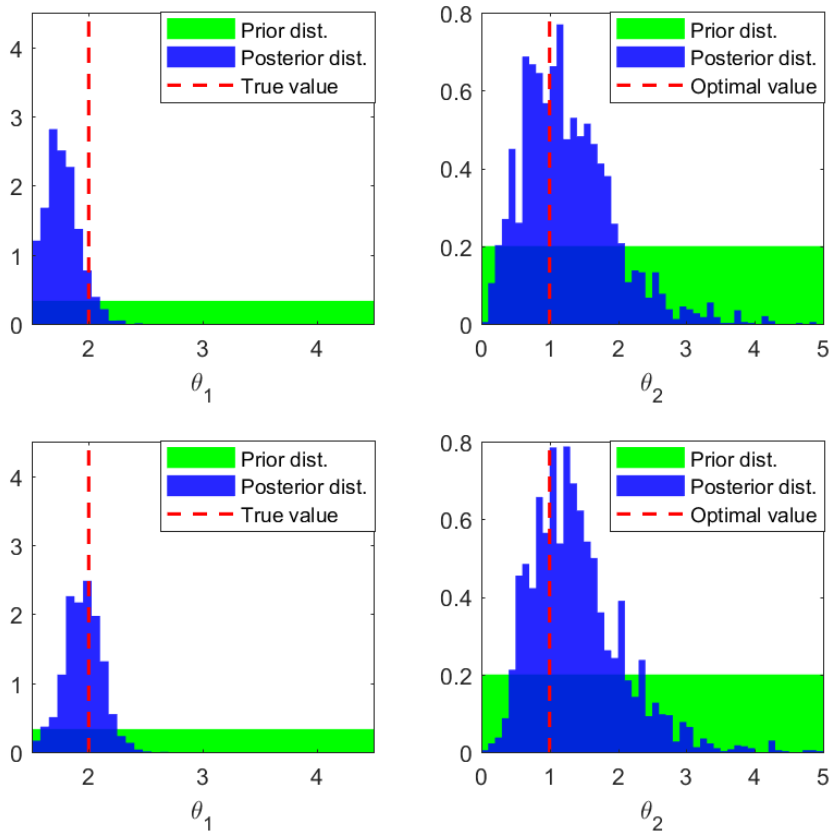

Figure 7: Prior and posterior distributions of the calibration parameter θ_1 and design parameter θ_2, along with their true/optimal values, for DCTO in the case of true systems g_2. The top row corresponds to a smaller discrepancy, with $(c, d) = (.055, 0)$; the bottom row corresponds to a larger discrepancy, with $(c, d) = (.055, .1)$.

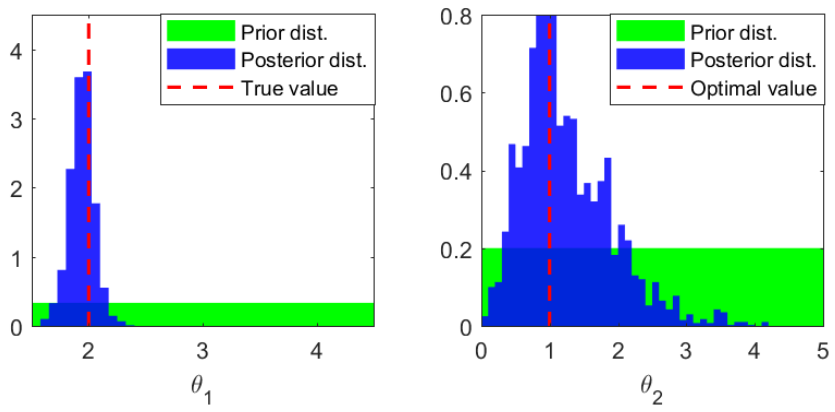

Figure 8: Prior and posterior distributions of the calibration parameter θ_1 and design parameter θ_2, along with their true/optimal values, for DCTO in the case of true systems g_2 with $(c, d) = (.055, 0)$ and an informative prior for δ_1.

the posteriors for θ_1, but peaking near the optimal θ_2 even when, as in the top row of Figure 7, this is not true of θ_1.

In order to achieve better results in the case of g_3 with $d = 0$, we attempted to place a more informative prior on the discrepancy function δ_1. We pursued two different strategies to do this. Firstly, we integrated the discrepancy $g_3 - f$ over the supports of x and t_2, finding the average value of the discrepancy. We then re-ran DCTO using $m_1(x, t_2) = \int (g_3(z, w) - f(z, w)) dz dw$ as the (constant) mean of the GP prior on δ_1. This corresponds to a case in which one knows on average how far one's model tends to be from the true system. The results, however, were not appreciably different from the original results using a mean of 0 for the GP prior on δ_1. Secondly, we again ran DCTO using $m_1(x, t_2) = g_3(x, t_2) - f(x, t_2)$, so that the GP prior on δ_1 was the true discrepancy. This corresponds to a case in which one has (e.g. through extensive experimentation) a more thorough understanding of the form of the model's discrepancy with the true system. This second strategy proved fruitful, producing the results in Figure 8.

References

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.