Modularized Calibration with laGP

Grant Hutchings

February 2022

1 Introduction

Consider the study of a physical process for which experiments are parameterized by a set of controllable inputs $\boldsymbol{x} = (x_1, \ldots, x_{p_x})$. Suppose we have n noisy observations of this process at a number of input parameter settings $\boldsymbol{X}^F = [\boldsymbol{x}_1^F, \ldots, \boldsymbol{x}_n^F]^T$. We denote a field observation at input setting $\boldsymbol{x}^F$ as $y^F(\boldsymbol{x}^F)$ and the set of field observations $\boldsymbol{Y}^F(\boldsymbol{X}^F) = [y_1^F(\boldsymbol{x}_1^F), ..., y_n^F(\boldsymbol{x}_n^F)]^T$. Field observations are often expensive and therefore limited, so scientists frequently turn to computer models to study the process. Suppose we have a computer model for the process which depends not only on $\boldsymbol{x}$, but also on a set of calibration parameters $\boldsymbol{t} = (t_1, ..., t_{p_t})$. We will denote the response from our computer model as $y^M(\boldsymbol{x}, \boldsymbol{t})$ and the set of m computer model runs as $\boldsymbol{Y}^M(\boldsymbol{X}^M) = [y_1^F(\boldsymbol{x}_1^M), ..., y_n^F(\boldsymbol{x}_n^M)]^T$. The goal of calibration is to determine the $\boldsymbol{t}^*$ which is optimal for the computer model. Optimal can mean different things depending on if the computer model is an unbiased estimate of the field data or not. We will discuss this more in the following section.

2 Unbiased Calibration

Given a certain unknown $\boldsymbol{t}^*$, we expect our deterministic computer model to be an unbiased estimate of the field data. In other words, the field data is a noisy version of the simulator output.

$$y^F(\boldsymbol{x}^F) = y^M(\boldsymbol{x}^F, \boldsymbol{t}^*) + \epsilon;\ \epsilon \sim N(0, \sigma_{y^F}^2) \tag{1}$$

where $\sigma_{y^F}^2$ is the observation error variance. Calibration is the process of determining this $\boldsymbol{t}^*$. Computer models are often not sufficiently fast to use for calibration, as we would need to run the model at many different $(\boldsymbol{x}, \boldsymbol{t})$ pairs to determine $\boldsymbol{t}^*$. Instead we build a statistical model, or emulator, of the simulator using a Gaussian Process. This laGP emulator can be used for fast calibration.

2.1 Scalar Response

Let us first consider the case where the response from the field data and computer code are scalar as shown in Equation 1. Replacing our computer model with an laGP emulator $\eta(\boldsymbol{x}, \boldsymbol{t})$ we have

$$\begin{aligned} y^F(\boldsymbol{x}^F) &= \eta(\boldsymbol{x}^F, \boldsymbol{t}^*) + \epsilon;\ \epsilon \sim N(0, \sigma_{y^F}^2) \\ &\sim \mathcal{N}(\mu_\eta, \sigma_\eta^2 + \sigma_{y^F}^2) \end{aligned} \tag{2}$$

Before we fit $\eta(\boldsymbol{x}, \boldsymbol{t})$ to the simulator data $\boldsymbol{Y}^M$, there is some clever pre-computing we can do to facilitate fast calibration later. We follow the procedure of https://arxiv.org/pdf/2005.00386.pdf

where inputs $(\boldsymbol{x}, \boldsymbol{t})$ are scaled, or "stretched and compressed", by estimates of the global GP length-scale parameters. This means that future laGP fits using scaled inputs do not need to estimate length-scales, as they will be very near to 1. We will later show how this greatly reduces calibration time. We begin by fitting a full GP to a small subset of the available suite of computer model runs. This subset can be chosen randomly, or using some kind of space-filling algorithm. The size of this subset can be chosen based on available computational resources. Given a set of estimated length-scales $[\hat{\boldsymbol{\theta}}_x, \hat{\boldsymbol{\theta}}_t]$ we compute $[\boldsymbol{X}^M_{sc}, \boldsymbol{T}^M_{sc}] = [\boldsymbol{X}^M, \boldsymbol{T}^M]/\sqrt{[\hat{\boldsymbol{\theta}}_x, \hat{\boldsymbol{\theta}}_t]}$ and similarly for $\boldsymbol{X}^F_{sc}$.

With our stretched and compressed input space, we are ready to calibrate. Unbiased calibration is straightforward in that the optimal $\boldsymbol{t}^*$ is one that maximizes the likelihood associated with the normal distribution in with Equation 2. A calibration algorithm is given below.

1. For a given $\boldsymbol{t}_{sc} = \boldsymbol{t}/\sqrt{\hat{\boldsymbol{\theta}}_t}$, fit an laGP model to the simulator data with inputs $[\boldsymbol{X}^M_{sc}, \boldsymbol{T}^M_{sc}]$ and make predictions at $[\boldsymbol{X}^F_{sc}, \boldsymbol{t}_{sc}]$. As our inputs are scaled, length-scales are fixed at 1, and a small nugget is added for numerical stability.

$$\begin{aligned} \eta(\boldsymbol{X}^F, \boldsymbol{t}) = laGP(&X = [\boldsymbol{X}^M_{sc}, \boldsymbol{T}^M_{sc}], \\ &Z = \boldsymbol{Y}^M, \\ &XX = [\boldsymbol{X}^F_{sc}, \boldsymbol{t}_{sc}], \\ &d = 1, \\ &g = 1e-7) \end{aligned} \tag{3}$$

2. Compute the residuals for the predictions: $\boldsymbol{r} = \boldsymbol{Y}^F(\boldsymbol{X}^F_{sc}) - \boldsymbol{\mu}_\eta(\boldsymbol{X}^F_{sc}, \boldsymbol{t}_{sc}) \ \in \mathbb{R}^n$.

3. Compute the log-likelihood of these residuals subject to Equation 2 with $\hat{\sigma}^2_{y^F} = \frac{1}{n}\boldsymbol{r}^T\boldsymbol{r}$.

$$ll \propto -\frac{1}{2}\left(log(|\boldsymbol{\Sigma}_\eta + \hat{\sigma}^2_{y^F}\boldsymbol{I}_n|) + \boldsymbol{r}^T(\boldsymbol{\Sigma}_\eta + \hat{\sigma}^2_{y^F}\boldsymbol{I}_n)^{-1}\boldsymbol{r}\right) \tag{4}$$

   where $\boldsymbol{\Sigma}_\eta \in \mathbb{R}^{n\times n}$ is a diagonal matrix of prediction variances returned by laGP.

Steps 1-3 can be run over a grid of possible $\boldsymbol{t}$ values, or can be implemented as an objective function in a more sophisticated optimizer in order to find the $\boldsymbol{t}$ which maximizes the likelihood.

2.2 Functional Response

We follow a similar approach to Higdon '08 to extend this procedure observations and computer simulations with functional response. Suppose the simulation response is of dimension p and observed data is of dimension q. Throughout the following section we will index experiments with the index $i = 1, ..., n$, simulations with the index $j = 1, ..., m$, and orthogonal bases, or principal components with $k = 1, ..., n_{pc}$. We now have

$$\boldsymbol{y}^F(\boldsymbol{x}^F) = \boldsymbol{y}^M(\boldsymbol{x}^F) + \boldsymbol{\epsilon};\ \boldsymbol{\epsilon} \sim N(\boldsymbol{0}_n, \sigma^2_{y^F}\boldsymbol{I}_q) \tag{5}$$

where $\boldsymbol{y} \in \mathbb{R}^q$. In the above equation, and throughout the documents, $\boldsymbol{y}^M(\boldsymbol{x}^F)$ implies not only that we are evaluating the simulator at the observed input $\boldsymbol{x}^F$, but also on the observed support.

If we have m computer model runs, stack the response vectors in columns to produce the matrix $\boldsymbol{Y}^M = [\boldsymbol{y}^M_1|...|\boldsymbol{y}^M_m] \in \mathbb{R}^{p\times m}$ and perform an SVD decomposition $\boldsymbol{Y}^M = \boldsymbol{U}\boldsymbol{D}\boldsymbol{W}^T = \boldsymbol{B}^M\boldsymbol{W}^T$ where

$\boldsymbol{B}^M = \boldsymbol{U}\boldsymbol{D}$. In practice we approximate the response using only the first n_{pc} basis vectors and associated weights.

$$\boldsymbol{Y}^M \approx \boldsymbol{B}^M[:, 1:n_{pc}]\boldsymbol{W}^T[1:n_{pc}, :] \; n_{pc} \leq min(p, q) \tag{6}$$

To cast field observations into this basis, we take $\boldsymbol{B}^F = \boldsymbol{B}^M$ if the field observations and simulations are on the same support. If not, we compute compute basis vectors $\boldsymbol{b}^F$ by interpolating the columns of $\boldsymbol{B}^M$ at the observed support. We can then compute the matrix of weights, $\boldsymbol{Z}^T$ as

$$\begin{aligned} & \boldsymbol{Y}^F = \boldsymbol{B}^F\boldsymbol{Z}^T \\ \implies \quad & (\boldsymbol{B}^F)^T\boldsymbol{\Sigma}_y\boldsymbol{Y}^F = (\boldsymbol{B}^F)^T\boldsymbol{\Sigma}_y\boldsymbol{B}^F\boldsymbol{Z}^T \\ \implies \quad & \boldsymbol{Z}^T = ((\boldsymbol{B}^F)^T\boldsymbol{\Sigma}_y\boldsymbol{Y}^F)^{-1}(\boldsymbol{B}^F)^T\boldsymbol{\Sigma}_y\boldsymbol{B}^F \end{aligned}$$

where $\boldsymbol{\Sigma}_y = \hat{\gamma}_y I_n$ and $\hat{\gamma}_y$ is the estimated observation precision (Go back to Hidgon to make sure I understand why the precision is included).

With our data properly decomposed, our emulator consists of n_{pc} independent laGP models on the basis weight vectors, the rows of $\boldsymbol{W}^T$ which can be combined through the basis vectors to reproduce the functional response. We will denote the emulator as $\eta(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{B}\boldsymbol{w}(\boldsymbol{x}, \boldsymbol{t})$ where $\boldsymbol{w}(\boldsymbol{x}, \boldsymbol{t}) \in \mathbb{R}^{n_{pc}}$ contains a basis weight for each of the k components.

Calibration is quite similar to the scalar case where we use predictions from the emulator at the observed inputs. Given the unbiased simulator assumption, we have

$$\boldsymbol{y}_i^F(\boldsymbol{x}^F) = \boldsymbol{B}^F\boldsymbol{w}_i(\boldsymbol{x}^F, \boldsymbol{t}^*) + \boldsymbol{\mathcal{E}}_i$$

where

$$\begin{aligned} & \boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{\mu}_{wi}, \boldsymbol{\Sigma}_{wi}) \\ & \boldsymbol{\mathcal{E}}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2_{y^F}\boldsymbol{I}_q) \\ \implies & \boldsymbol{B}^F\boldsymbol{z}_i \sim \boldsymbol{B}^F\mathcal{N}(\boldsymbol{\mu}_{wi}, \boldsymbol{\Sigma}_{wi}) + \mathcal{N}(\boldsymbol{0}, \sigma^2_{y^F}\boldsymbol{I}) \\ \implies & \boldsymbol{B}^F\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{B}^F\boldsymbol{\mu}_{wi}, \boldsymbol{B}^F\boldsymbol{\Sigma}_{wi}(\boldsymbol{B}^F)^T + \sigma^2_{y^F}\boldsymbol{I}) \\ \implies & \boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{\mu}_{wi}, \boldsymbol{\Sigma}_{wi} + \sigma^2_{y^F}((\boldsymbol{B}^F)^T\boldsymbol{B}^F)^{-1}) \end{aligned} \tag{7}$$

This is the likelihood we will use for calibration. Once again we begin by fitting n_{pc} independent full GP models to a subset of the computer model runs to estimate length-scale parameters for each component of the model. With a set of estimated length-scales for each of the n_{pc} models, inputs can be stretched and compressed (sc) independently for each model. Calibration proceeds similarly to the scalar response case.

1. For a given $\boldsymbol{t}$, fit n_{pc} independent laGP models to the computer model data with scaled inputs $[\boldsymbol{X}^M_{sc}, \boldsymbol{T}^M_{sc}]$ and predict at $[\boldsymbol{X}^F_{sc}, \boldsymbol{t}_{sc}]$. As our inputs are stretched and compressed, length-scales are fixed at 1, and a small nugget is used. laGP returns the mean and variance of the sampling distribution

$$\begin{aligned} \boldsymbol{\mu}_{wk}, \boldsymbol{\sigma}^2_{wk} = laGP(& X = [\boldsymbol{X}^M_{sc}, \boldsymbol{T}^M_{sc}]_k, \\ & Z = \boldsymbol{W}^T[k, :], \\ & XX = [\boldsymbol{X}^F_{sc}, \boldsymbol{t}_{sc}]_k, \\ & d = 1, \\ & g = 1e-7); \; k = 1, ..., n_{pc} \end{aligned} \tag{8}$$

Subscripts on input matrices are to indicate that inputs are scaled differently for each k. Store prediction means and variances in matrices $\boldsymbol{M}_w = [\boldsymbol{\mu}_{w1}|...|\boldsymbol{\mu}_{wn_{pc}}]^T \in \mathbb{R}^{n_{pc} \times n}$, $\boldsymbol{S}_w = [\boldsymbol{\sigma}^2_{w1}|...|\boldsymbol{\sigma}^2_{wn_{pc}}]^T \in \mathbb{R}^{n_{pc} \times n}$

2. Compute the residual matrices:

$$\boldsymbol{R}_w = \boldsymbol{Z}^T - \boldsymbol{M}_w$$
$$\boldsymbol{R}_y = \boldsymbol{Y}^F - \boldsymbol{B}^F \boldsymbol{M}_w$$

3. Estimate error variance: Store $\boldsymbol{R}_y$ in a vector $\boldsymbol{r}_y$ and estimate

$$\hat{\sigma}^2_y = \frac{1}{n * p} \boldsymbol{r}_y^T \boldsymbol{r}_y$$

4. Compute the log likelihood for each experiment from the normal distribution in Equation 7.

$$ll \propto \sum_{i=1}^{n} -\frac{1}{2}(log(|\boldsymbol{\Sigma}_i|) + \boldsymbol{R}_w[:, i](\boldsymbol{\Sigma}_i)^{-1} \boldsymbol{R}_w[:, i])) \tag{9}$$

where $\boldsymbol{\Sigma}_i = \hat{\sigma}^2_{y^F} (\boldsymbol{B}^F)^T \boldsymbol{B}^F + \boldsymbol{\Sigma}_{wi}$ and $\boldsymbol{\Sigma}_{wi} = \texttt{diag}(\boldsymbol{S}_w[:, i])$.

2.3 Open Questions

2.3.1 Prediction Variance

Equation 7 describes the sampling distribution for the observed weights.

$$\boldsymbol{z}_i \sim N(\boldsymbol{\mu}_{wi}), \hat{\sigma}^2_y((\boldsymbol{B}^F)^T \boldsymbol{B})^{-1} + \boldsymbol{\Sigma}_{wi})$$

But, we could also sample from

$$\boldsymbol{Y}_i \sim N(\boldsymbol{B}^F \boldsymbol{\mu}_{wi}, \boldsymbol{B}^F \boldsymbol{\Sigma}_{wi} (\boldsymbol{B}^F)^T + \hat{\sigma}^2_y \boldsymbol{I})$$

Which for some reason increases our predictive variability. I believe I understand now why the predictive variances are different, but I'm not sure what the solution is. Consider the following

$$Bz \sim B\mathcal{N}(\mu_w, \Sigma_w) + \mathcal{N}(0, \sigma^2 I) \tag{1}$$
$$z \sim \mathcal{N}(\mu_w, \Sigma_w) + (B^T B)^{-1} B^T \mathcal{N}(0, \sigma^2 I) \tag{2}$$
$$Bz \sim B\mathcal{N}(\mu_w, \Sigma_w) + B(B^T B)^{-1} B^T \mathcal{N}(0, \sigma^2 I) \tag{3}$$

(1) and (3) are equivalent iff $B(B^T B)^{-1} B^T = I$ which we might expect to be true as B came from SVD and is the product of an orthogonal matrix and a diagonal matrix. It turns out that $B(B^T B)^{-1} B^T = I$ if we use all the columns of B, but not if we use only the first nPC columns. For this reason we are getting different predictive variances.

2.3.2 Error Variance

For calibration we could compute σ^2_{yi} for each experiment instead. We could use these individually for the likelihood, but average them at the end for the final error variance estimate. My thought is that this might improve calibration by reducing the chance that a good error estimate for one experiment is balanced out by a bad one for another experiment. However we are assuming that σ^2 is constant for all experiments. I tried this on the ball drop example (for only 1 random data set) and I got a worse estimate of $\boldsymbol{t}^*$ so maybe its not a good idea.

3 Biased Calibration

3.1 Functional Response

We now consider the situation where the simulator is known to be a biased estimate of the observed data. Our statistical model is similar to Equation 7 although we must add another term to account for this bias.

$$\boldsymbol{y}^F(\boldsymbol{x}^F) = \eta(\boldsymbol{x}^F, \boldsymbol{t}^*) + \delta(\boldsymbol{x}^F) + \boldsymbol{\epsilon};\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2_{y^F}\boldsymbol{I}_q) \tag{10}$$

where $\delta(\boldsymbol{x})$ accounts for the bias between the emulator and the observed data. We decompose our computer simulations as before and the full emulator is of the form $\eta(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{B}\boldsymbol{w}(\boldsymbol{x}, \boldsymbol{t})$. We will similarly model the bias using a basis representation computed via SVD $\delta(\boldsymbol{x}) = \boldsymbol{D}\boldsymbol{v}(\boldsymbol{x})$, with a Gaussian Process on $\boldsymbol{v}$. A full GP as oppose to laGP is usually feasible for the bias as n is small to moderate. We will soon discuss how $\boldsymbol{D}$ and $\boldsymbol{v}$ are calculated.

We again calibrate using a maximum-likelihood approach where the likelihood is derived from equation 10.

$$\begin{aligned}
& \boldsymbol{y}_i^F(\boldsymbol{x}_i^F) = \eta(\boldsymbol{x}_i^F, \boldsymbol{t}^*) + \delta(\boldsymbol{x}_i^F) + \boldsymbol{\epsilon}_i \\
\implies & \boldsymbol{y}_i^F(\boldsymbol{x}_i^F) = \boldsymbol{B}^F\hat{\boldsymbol{w}}_i + \boldsymbol{D}\hat{\boldsymbol{v}}_i + \boldsymbol{\epsilon}_i \\
& \boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{\mu}_{wi}, \boldsymbol{\Sigma}_{wi}) \\
& \boldsymbol{v}_i \sim \mathcal{N}(\boldsymbol{\mu}_{vi}, \boldsymbol{\Sigma}_{vi}) \\
& \boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2_{y^F}\boldsymbol{I}_q) \\
\implies & \boldsymbol{y}_i^F(\boldsymbol{x}_i^F) \sim \boldsymbol{B}^F\mathcal{N}(\boldsymbol{\mu}_{wi}, \boldsymbol{\Sigma}_{wi}) + \boldsymbol{D}\mathcal{N}(\boldsymbol{\mu}_{vi}, \boldsymbol{\Sigma}_{vi}) + \mathcal{N}(\boldsymbol{0}, \sigma^2_{y^F}\boldsymbol{I}_q) \\
\implies & \boldsymbol{y}_i^F(\boldsymbol{x}_i^F) - \boldsymbol{B}^F\boldsymbol{\mu}_{wi} \sim \mathcal{N}(\boldsymbol{D}\boldsymbol{\mu}_{vi}, \boldsymbol{B}^F\boldsymbol{\Sigma}_{wi}(\boldsymbol{B}^T)^F + \boldsymbol{D}\boldsymbol{\Sigma}_{vi}\boldsymbol{D}^T + \sigma^2_{y^F}\boldsymbol{I}_q)
\end{aligned} \tag{11}$$

Calibration is as follows

1. For a given $\boldsymbol{t}$, do step 1. from the calibration algorithm in Section 2.2 to get the matrices $\boldsymbol{M}_w$, $\boldsymbol{S}_w$

2. Compute bias matrix
$$\boldsymbol{R}_y = \boldsymbol{Y}^F - \boldsymbol{B}^F\boldsymbol{M}_w$$

3. Compute basis for bias
$$\texttt{SVD}(\boldsymbol{R}_y) = \boldsymbol{U}_b\boldsymbol{S}_b\boldsymbol{V}^T = \boldsymbol{D}\boldsymbol{V}^T$$
keeping only the first n^*_{pc} columns of $\boldsymbol{D}$ and rows of $\boldsymbol{V}^T$ where n^*_{pc} is chosen so that 95% of the variability in $\boldsymbol{R}_y$ is accounted for.

4. Fit a separable GP to each component of the bias, estimating all parameters and predict at the same input locations.
$$\begin{aligned}
\boldsymbol{\mu}_{vk}, \boldsymbol{\sigma}^2_{vk} = GP(& X = [\boldsymbol{X}^F, \boldsymbol{t}], \\
& Z = \boldsymbol{V}^T[k, :], \\
& XX = [\boldsymbol{X}^F, \boldsymbol{t}]);\ k = 1, ..., n^*_{pc}
\end{aligned} \tag{12}$$

5. Store predictive means and variances as before in matrices $\boldsymbol{M}_v = [\boldsymbol{\mu}_{v1}|...|\boldsymbol{\mu}_{vn^*_{pc}}]^T$, $\boldsymbol{S}_v = [\boldsymbol{\sigma}^2_{v1}|...|\boldsymbol{\sigma}^2_{vn^*_{pc}}]^T$

6. Compute residuals

$$\boldsymbol{R}_\epsilon = \boldsymbol{R}_y - \boldsymbol{D}\boldsymbol{M}_v$$

and vectorize $\boldsymbol{r}_\epsilon = \texttt{Vec}(\boldsymbol{R}_\epsilon)$ to estimate

$$\hat{\sigma}^2_{y^F} = \frac{1}{n * p}\boldsymbol{r}_\epsilon^T \boldsymbol{r}_\epsilon$$

7. Compute log-likelihood

$$ll \propto \sum_{i=1}^{n} -\frac{1}{2}\Big(log(|\boldsymbol{\Sigma}_i|) + \boldsymbol{R}_\epsilon[:,i]^T \boldsymbol{\Sigma}_i^{-1} \boldsymbol{R}_\epsilon[:,i]\Big)$$

where $\boldsymbol{\Sigma}_i = \boldsymbol{B}^F \boldsymbol{\Sigma}_{wi} (\boldsymbol{B}^T)^F + \boldsymbol{D}\boldsymbol{\Sigma}_{vi}\boldsymbol{D}^T + \hat{\sigma}^2_{y^F}\boldsymbol{I}_q$ and $\boldsymbol{\Sigma}_{vi} = \texttt{diag}(\boldsymbol{S}_v[:,i])$