51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference
 18th
12 - 15 April 2010, Orlando, Florida

AIAA 2010-2851

2010 AIAA SDM Student Symposium Parameter Estimation via Gaussian Processes and Maximum Likelihood Estimation

Nicholas West*

Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA, 94305

Laura Swiler†

Sandia National Laboratories, Albuquerque, NM, 87185, USA

Computer models usually have a variety of parameters that can (and need to) be tuned so that the model better reflects reality. This problem is called calibration and is an inverse problem. We assume that we have a set of observed responses to given inputs in a physical system and a computer model that depends on parameters that models the physical system being studied. It is often the case that many more simulations can be run than experiments conducted, so we typically have many more simulation results (at various parameter values) than experimental results (at the "true" parameter value). In this paper, we use Maximum Likelihood Estimation (MLE) to calibrate model parameters. We assume that the response data is vector-valued, e.g. a response is given as a function of time. We approximate the underlying models with Gaussian Processes (GPs) and fit the parameters of the GPs with MLE. Specifically, we propose a decomposition approach to identify the basis vectors that allows for efficient calculation of the parameters. Experimental data is then used to calibrate the model parameters. This approach is demonstrated on one test problem.

I. Introduction

Computer models usually have a variety of *parameters* that can (and need to) be tuned so that the model better reflects reality. This problem is called *calibration* and is an inverse problem. We assume that we have a set of observed responses to given inputs in a physical system and a computer model that depends on parameters and models the physical system being studied. It is often the case that many more simulations can be run than experiments conducted, so we typically have many more simulation results (at various parameter values) than experimental results (at the "true" parameter value). There is a large literature surrounding the problem of model calibration, ranging from interval analysis, to least squares fitting, to fully probabilistic Bayesian methods.

A typical calibration approach is least squares analysis. In a classical least squares analysis, the model parameters are assumed unknown but fixed. They are estimated by identifying parameter values which minimize an objective function called the "error sum of squares" which is the sum of the squared residual terms measuring the difference between the model prediction and the experimental results. There is a vast literature on linear regression [Draper and Smith[1]] and nonlinear regression methods [Seber and Wild[2]] In addition, methods have been developed for generating confidence intervals around optimal parameter

*Doctoral Candidate, Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA, 94305, USA

†Principal Member of Technical Staff, Optimization and Uncertainty Estimation Dept., P.O. Box 5800, Sandia National Laboratories, MS 1318

estimates. These methods usually attempt to incorporate uncertainty in the experimental data. It is common to use linear approximations for nonlinear models in the calculation of confidence regions. In general, the computation of confidence intervals on parameters estimated by least squares methods is computationally expensive and can be inaccurate [Vugrin et al[3]]. Interval analysis is a less common approach used in calibration. In interval analysis, one attempts to match intervals spanned by the simulation data with intervals spanned by the experimental data by determining intervals on some of the input uncertainties which lead to the condition of experimental result intervals approximately matching those of simulation results [Romero 2007[4]].

Note that regression methods usually incorporate some type of error associated with the observations but generally do not incorporate other types of uncertainty such as model form uncertainty, model bias or inadequacy, and uncertain model parameters. [Xiong et al.[5]] provides a nice overview of more recent approaches for calibration (they refer to calibration as a form of model updating). Specifically, in Bayesian calibration, the parameters are assumed unknown but fixed over the course of the experiment. The lack of knowledge of the parameters is reflected in prior distributions which are assigned, and these prior distributions are updated based on observed data through the Bayesian framework [Kennedy and O'Hagan[6] and Higdon et al[7]]. In contrast to Bayesian methods which determine a distribution for each of the input parameters, Maximum Likelihood Estimation methods (MLE) attempt to find a particular set of parameter values which result in a maximal value of a likelihood function, usually by using an optimization method. In this paper, we use a MLE approach to determining the parameters governing the Gaussian process which is part of the Bayesian calibration framework.

II. Model Formulation

To frame the calibration problem we suppose that we have a mathematical model of the process under study (generally a computational model) that is tune-able with a set of parameters, θ. Additionally, it is possible to conduct physical experiments for relatively few inputs; by their nature, these experiments are conducted at a "true" value of the parameters. We therefore have a set of simulation model results $\{Y_i^m = f(X_i^m, t_i^m)\}$ for $i = 1, \cdots, n$ and a set of experimental results $\{Y_i^o = f(X_i^o)\}$[a], where $f(X_i^o) \approx f(X_i^o, \theta^*) + \varepsilon_i$.

At this point one might jump to a least-squares formulation, however the computational model may be prohibitively expensive for this approach to be feasible. We therefore seek an inexpensive surrogate for the true model. Gaussian Processes (GPs) [see Cressie[8] and Rasmussen and Williams[9]] have proved useful and effective in this context [Higdon et al 2005,[10] Kennedy and O'Hagan,[6] McFarland[11]]. We employ a GP as a surrogate for the model to capture the response to the inputs and parameters. A GP is a stochastic process, which for any finite number of points $\{x_i\}$, $i = 1, \cdots, n$ is jointly Gaussian, ie: $(X(x_1), X(x_2), \cdots, X(x_n))$ has a multivariate Gaussian distribution. It is completely characterized by its mean function $\mu(x)$ and covariance function $c(x, x')$. For this paper we will assume that the model has been de-trended (see McFarland[11] for a model with a trend term) so that $\mu(x) = 0$; we further assume that the process is stationary so that $c(x, x') = c(x - x')$, i.e. the correlation / covariance is a function of the proximity of the inputs. We use a Gaussian covariance function

$$c(r) = \sigma^2 e^{-\sum \frac{r_i^2}{\lambda_i}}$$

where σ^2 is the variance and λ_i is the correlation length of the i^{th} input variable.

In many settings it is convenient to think of each experiment or model run as returning a vector of values (either a set of variables such as pressure, temperature and velocity or the responses at different times). If one assumes that the process (now in an extended dimension) is still Gaussian, progress can be made in approximating it efficiently. This will be discussed in a following section.

Following [Kennedy and O'Hagan,[6] McFarland[11] and Higdon et al 2005[10]] we assume that the modeled results and the observed results are samples from a GP with (unknown) parameters $\{\sigma_i^2\}$ and $\{\lambda_j^i\}$. We

[a]Note that the notation here, $f(X)$, is used only to suggest that the experimental results take some functional relationship of the inputs similar to, but not the same as, the computational model.

assume that

$$Y_i^M \sim \eta(X_i^m, t_i^m) + e_i$$

where η is a GP and e_i is a noise term (a nugget) used to account for discrepancies in the GP and the underlying function. This nugget term also helps to stabilize the numerical computations. The experimental observations are modeled by

$$Y_i^o \sim \eta(X_i^o, \theta^*) + \delta(X_i^o) + \varepsilon_i$$

where θ^* represents the optimal parameter selection (when fit); $\delta(\cdot)$ is a discrepancy term and is modeled as a GP (assumed to be mean zero and stationary); and ε_i is an observational noise term. In many settings δ is taken to be zero.

III. Modeling Vector-Valued Gaussian Processes

Suppose that X is a Gaussian vector; it is well known that solving the eigenvalue problem for its covariance matrix gives a discrete expansion of the random vector in terms of basis vectors and uncorrelated random variables. We are motivated by this expansion and suggest that we model $X : \mathbb{R}^d \times \Omega \to \mathbb{R}^p$ as an expansion on a basis, weighted by uncorrelated GPs. Following [Higdon et al 2005[10]], let $\{v_i\}$ be a set of basis vectors that span $\mathbb{R}^p$ and let $\{w_i(x)\}$ be a set of uncorrelated (and thus independent) GPs, we use the approximation

$$X(x) \approx \sum_{i=1}^{p} v_i w_i(x) = VW(x).$$

The types of vector-valued Gaussian processes that we would like to use this model on have the form that the index variable is really no different than any of the other variables - it should have its own correlation length. It is unclear what a correlation length between two different index values (eg pressure and velocity) would mean. We explore the limitations of such a description below.

For the original process $X(x)$ (which is a vector valued Gaussian process), under the interpretation that the vector notation is convenient short hand, the covariance function between $X(x_i)$ and $X(x_j)$ is described as

$$\mathrm{E}[X(x_i)X(x_j)^T] = \{C_{k\ell}(x_i - x_j)\}, \;\; k, \ell = 1, \cdots, d$$

where

$$C_{k\ell}(x_i, x_j) = \mathrm{E}[X(x_i, t_k)X(x_j, t_\ell)] = \sigma^2 e^{-\sum_q (x_{iq} - x_{jq})_q^2 / \lambda_q} e^{-(t_k - t_l)^2/\lambda_t} = \sigma_{k\ell}^2 C(x_i, x_j)$$

and $C(x_i, x_j)$ is the standard Gaussian covariance function. Now, consider the approximation given above for the process; call this $\hat{X}$ to avoid confusion.

$$\mathrm{E}[\hat{X}(x_i)\hat{X}^T(x_i)] = \mathrm{E}[VW(x_i)W^T(x_j)V^T] = V\mathrm{E}[W(x_i)W^T(x_j)] = \{\hat{C}_{k\ell}(x_i - x_j)\}, \;\; k, \ell = 1, \cdots, d$$

where

$$\hat{C}_{k\ell}(x_i, x_j) = \sum_q v_{kq} v_{q\ell} C_q(x_i, x_j), \;\; C_q(x_i, x_j) = \sigma_q^2 e^{-\sum_r (x_{ir} - x_{jr})^2 / \lambda_r^q}.$$

Note that, if we take the basis vectors V to be fixed, then this model has a reduced number of parameters: $d + n_x$. This is in contrast to the $d^2 + n_x + 1$ parameters in the original model. The main difference, however, is that this model does not explicitly know about the variability in the last coordinate, being used as the index in the vector. Selection of the basis vectors will be discussed below.

IV. Maximum Likelihood Estimation

The technique of Maximum Likelihood Estimation (MLE) is often considered the gold-standard for statistical parameter estimation. There are two approaches to likelihood maximization for this problem: [Kennedy and O'Hagan[6]] argues that, in general, there will be many more samples from the computational

model than there are observations, and thus fitting the parameters in η to only the computational output looses very little. On the other hand [McFarland[11]] and [Higdon et al 2005[10]] approach the problem (at least theoretically) in a combined approach where all estimation is done at the same step for all terms. In [McFarland[11]] estimation is of the GP parameters with the bias term included; in [Higdon et al 2005[10]] the estimation of all parameters is done in one step of the MCMC algorithm which treats variance and correlation lengths slightly differently. Our approach follows [Kennedy and O'Hagan[6]] where our modifications have the most improvement in efficiency; however, the techniques may also be applied to the problem of maximizing the likelihood over all parameters.

The likelihood of observing the modeled output is conditionally (on the parameters) Gaussian. Write $Y = ({Y_1^m}^T, \cdots, {Y_n^m}^T)^T$ as the concatenation of all of the observed variables and define $\hat{\Sigma}$ as the matrix with n^2 $d \times d$ blocks, the i^{th}, j^{th} block given by

$$C_{ij} = \mathrm{E}\left[Y(X_i^m, t_i^m) Y(X_j^m, t_j^m)^T\right],$$

the covariance between modeled responses i and j. When we have expanded η on a set of basis vectors $\{v_i\}$ with GP weights, this becomes

$$C_{ij} = V\mathrm{E}\left[\mathrm{diag}(w_1(X_i^m, t_i^m) w_1(X_j^m, t_j^m), \cdots, w_d(X_i^m, t_i^m) w_d(X_j^m, t_j^m))\right] V^T.$$

Define $\Sigma = \hat{\Sigma} + \sigma^2 \mathrm{I}$ as the covariance with the lack-of-fit / nugget parameter (σ^2 is the variance of each component of e). The likelihood of the data is

$$L = \frac{1}{(2\pi)^{\frac{nd}{2}}} |\Sigma|^{-\frac{1}{2}} e^{-\frac{1}{2} Y^T \Sigma^{-1} Y}.$$

It is more practical to minimize the negative-log-likelihood ($-\log L$)

$$NLL = \frac{nd}{2} \log 2\pi + \frac{1}{2} \log |\Sigma| + \frac{1}{2} Y^T \Sigma^{-1} Y.$$

For an arbitrary matrix Σ, the most efficient method of evaluating the NLL is by factorizing Σ, and as Σ is symmetric and positive definite the Cholesky decomposition is the choice algorithm, yielding $\Sigma = R^T R$ where R is an upper triangular matrix. This allows for efficient solution of the implied system solves in $Y^T \Sigma^{-1} Y$ and the determinant computation. The Cholesky decomposition is an order n^3 algorithm and thus requires $O(nd)^3$ work. When implementing the evaluation of this function numerically, the determinant of Σ can be computed as the product of the diagonals of R; these can be sufficiently close to zero that underflow can occur leading to an unbounded objective function. To remedy this, we take the sum of the logs of the diagonals (as we only need the log of the determinant).

For optimization purposes, we drop the constant and use the function

$$NLL = \log |\Sigma| + Y^T \Sigma^{-1} Y$$

for which we can analytically derive derivatives with respect to the GP parameters. Recall that for the fitting process (of the GP to the model), Σ is a function of the correlation lengths, variances and nugget variance which need to be optimized over. Let t be an arbitrary parameter, then the derivative of NLL with respect to t is given by

$$\frac{\partial NLL}{\partial t} = \mathrm{trace}(\Sigma^{-1} \frac{\partial \Sigma}{\partial t}) + Y^T \Sigma^{-1} \frac{\partial \Sigma}{\partial t} \Sigma^{-1} Y$$

where the notation $\partial\Sigma/\partial t$ is the component-wise differentiation of the matrix.

V. Basis Vector Selection

The major improvement to this model comes from the selection of the basis vectors used to represent the GPs and, after judicious selection, use of certain properties of these vectors. [Higdon et al 2005[10]] uses the

principal components of the empirical covariance matrix for their basis without much comment. However some insight can be gained from this choice. If we write

$$Y^m = [Y_1^m, Y_2^m, \cdots Y_n^m],$$

(the matrix whose columns are the model results) then empirical covariance matrix is given by

$$\Sigma_e = \frac{1}{n-1} YY^T = USU^T$$

where U is the matrix of singular vectors and S is a diagonal matrix with the singular values. Another way to view this decomposition is as a Karhunen-Loeve expansion, and we know that the singular values correspond to the amount of variance each component contributes. This suggests a more rigorous / understandable process to determine the number of basis vectors needed to approximate a GP: we select the first d' vectors, so that the first d' singular values contribute at least $(1-\delta)\times 100\%$ of the variance. Define the corresponding set of singular vectors to be $\hat{U}$, the first d' columns of U. As this incorporates a spatial average, this only captures the most large scale features in the variance function and may cause large optimal correlation lengths.

One could now use this set of vectors in the model as it has been described, however, there are further refinements possible. With a particular set of vectors we are assuming that

$$Y_i^m = Y^m(X_i^m, t_i^m) \stackrel{D}{=} \sum_{i=1}^{d'} u_i w_i(X_i^m, t_i^m) = \hat{U} W_i^m$$

which is Gaussian. We now take specific linear combinations of the components of Y_i^m so that

$$u_k^T Y_i^m \stackrel{D}{=} u_k^T \sum_{j=1}^{d'} u_j w_j(X_i^m, t_i^m) \stackrel{D}{=} w_k(X_i^m, t_i^m).$$

due to the orthogonality of the singular vectors. Define

$$\hat{Y}_i^m = \hat{U}^T Y_i^m$$

as the new "model response". We note that the covariance between two of these new observations is simply the diagonal matrix:

$$\mathrm{E}[\hat{Y}_i^m \hat{Y}_j^{mT}] = \mathrm{diag}(\mathrm{E}[w_1(X_i^m, t_i^m) w_1(X_j^m, t_j^m)], \cdots, \mathrm{E}[w_{d'}(X_i^m, t_i^m) w_{d'}(X_j^m, t_j^m)]).$$

Now the new data vector $\hat{Y}^m = (Y_1^{mT}, \cdots, Y_n^{mT})^T$ has a banded (and very sparse) covariance matrix. By permuting the data vector so that all of the data corresponding to process w_1 are first, w_2 second, and so forth, the covariance matrix becomes block diagonal with each block being the covariance matrix for one GP.

The effects in computational savings are great when computing values of the negative-log-likelihood. By using a truncated set of vectors to describe the random field we have reduced the work required to construct the covariance matrix, however its dimension remains the same. However, when we work with the truncated variables, there are large improvements. First, the size of the resulting covariance matrix is now $nd' \times nd'$ which is at most the size of the previous matrix. However, the greatest savings comes from the fact that it is block diagonal, and each block is a positive definite covariance matrix. We need only factor each block to obtain the Cholesky decomposition of the covariance matrix. This amounts to d' $O(n^3)$ operations rather than the one $O((nd')^3)$ cost of factoring the larger matrix.

In terms of the optimization, there is no major savings, other than the reduction in time to evaluate the objective function. If, however, the nugget term is removed, then the blocks are independent (they only depend on their own parameters) and the likelihood function can be viewed as a product of likelihoods for

each GP. Rather than solving one large optimization problem, we can now solve d' smaller optimization problems. We are still investigating under what situations the nugget term can be removed and still have a numerically stable algorithm.

The discrepancy process $\delta(\cdot)$ is also modeled as a linear combination of "basis" functions, however, their selection is more context specific [Higdon et al 2005[10]]. Let $\{b_i\}$, $i = 1, \cdots, n_d$ be a set of linearly independent vectors. The discrepancy term is modeled by

$$\delta(x) = \sum_{i=1}^{n_d} b_i v_i(x) = BV(x)$$

where $\{v_i\}$, $i = 1, \cdots, n_d$ is a set of independent GPs. It is convenient to think of $b_{ij} = b_i(t_j)$ where t_j is the index variable, and $b_i(\cdot)$ is a discrepancy function. A common choice of $b_i(t)$ is a Gaussian bump centered about a point t_i, and thus the discrepancy term can be viewed as a weighting of Gaussian errors about points t_i. The number of discrepancy terms, n_d, is selected so that there is sufficient resolution to capture discrepancies between the model and reality.

VI. Model Parameter Selection

In order to select the model parameters we must incorporate the data from the experiments, conducted, in theory, at the "true" value. Recall that our model takes each output as a linear combination of basis vectors so that

$$Y_i^m(X_i^m, t_i^m) = \sum_{j=1}^{d'} u_j w_j(X_i^m, t_i^m) = \hat{U} W_i^m$$

where $W_i^m = (w_1(X_i^m, t_i^m), w_2(X_i^m, t_i^m), \cdots, w_{d'}(X_i^m, t_i^m))^T$ so that the entire data vector Y^m takes the form

$$\begin{pmatrix} V^{(1)} & & & \\ & V^{(2)} & & \\ & & \ddots & \\ & & & V^{(n)} \end{pmatrix} \begin{pmatrix} W_1^m \\ W_2^m \\ \vdots \\ W_n^m \end{pmatrix} = VW^m$$

where $V^{(i)}$ are identical copies of $\hat{U}$. Note that the columns of V are orthogonal so that $V^T V = \mathrm{I}$. Similarly, one observation vector (from the experiment) can be written as

$$Y_i^o(X_i^o, \theta^*) = \sum_{j=1}^{d'} u_j w_j(X_i^o, \theta^*) + \sum_{j=1}^{d_b} b_j v_j(X_i^o) = \hat{U} W_i^o(\theta^*) + BV_i^o$$

where $W_i^o(\theta) = (w_1(X_i^o, \theta), w_2(X_i^o, \theta), \cdots, w_{d'}(X_i^o, \theta))^T$ and $V_i^o = (v_1(X_i^o), v_2(X_i^o), \cdots, v_{d_b}(X_i^o))^T$. In extreme shorthand we can write

$$\begin{pmatrix} Y^m \\ Y^0 \end{pmatrix} = \begin{pmatrix} V & 0 & 0 \\ 0 & \bar{V} & B \end{pmatrix} \begin{pmatrix} W^m \\ W^o(\theta) \\ V^0 \end{pmatrix} \Longrightarrow \begin{pmatrix} \hat{Y}^m \\ \hat{Y}^o \end{pmatrix} = \begin{pmatrix} \mathrm{I} & 0 & 0 \\ 0 & \mathrm{I} & \bar{V}^T B \end{pmatrix} \begin{pmatrix} W^m \\ W^o(\theta) \\ V^0 \end{pmatrix}.$$

Simple computation gives the covariance matrix of the above as

$$\Sigma(\theta) = \begin{pmatrix} \mathrm{E} W^m W^{mT} & \mathrm{E} W^m W^{oT}(\theta) \\ \mathrm{E} W^o(\theta) W^{mT} & \mathrm{E} W^o(\theta) W^{oT}(\theta) + \bar{V}^T B \mathrm{E} V^o V^{oT} D^T \bar{V} \end{pmatrix}.$$

Note that due to the nature of a Gaussian covariance function, the two-two block is actually independent of θ.

In evaluating the likelihood of the data, conditional on some parameter selection, the two computationally demanding tasks are computing $y^T\Sigma^{-1}y$ and $\det(\Sigma)$. We make some comments here on the efficient computation of these quantities. First, observe that the one-one block of Σ is the same (unpermuted) covariance matrix of the model considered above; therefore we know that by symmetric permutation it can be transformed into a block diagonal matrix, call this transformation Π and let Q be an arbitrary (for the moment) permutation matrix, then

$$\begin{pmatrix} \Pi & \\ & Q \end{pmatrix} \Sigma \begin{pmatrix} \Pi^T & \\ & Q^T \end{pmatrix} = \begin{pmatrix} D & C \\ C^T & F+G \end{pmatrix}$$

where D is a block-diagonal matrix. By selecting Q in a similar manner to Π only for the observed projections we have that $\hat{C}$ is block -diagonal as well (although not square) and that F is also block-diagonal; G remains dense. Note that G is zero in the case of no discrepancy term.

To solve a system of the form

$$\begin{pmatrix} D & C \\ C^T & H \end{pmatrix} \begin{pmatrix} s \\ r \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}$$

we employ the Schur compliment. This gives us two systems:

$$Ds + Cr = a \quad \text{or} \quad s = D^{-1}(a - Cr) \quad \text{and} \quad C^T s + Hr = b \Rightarrow (H - C^T D^{-1} C)r = b - C^T D^{-1} a.$$

Such a solution relies on the assumption that D is easily invertible; in this case D is the same block-diagonal matrix we've been discussing and is indeed easily "inverted." In fact, the dense matrix $H = F + G$ is an $n_o d' \times n_o d'$ matrix, which is often very small. Additionally the determinant of Σ can be similarly expressed in terms of these matrices:

$$\det(\Sigma) = \det(D)\det(H - C^T D^{-1} D).$$

Once these matrices have been factored, computation of their determinants is straight forward.

VII. Confidence Interval Generation

In addition to estimating the true parameters, θ^*, it is often useful to produce a confidence interval for these parameters. We employ the *bootstrap* method [Efron and Tibshirani[12]] to produce a confidence interval. The idea of the bootstrap is the following: we use the observed data, y_o to estimate the parameters $\hat{\theta}$; conditional on $\theta = \hat{\theta}$ we use our Gaussian Process model to sample the observed data at the same inputs, call this y_d^i; from this sampled data, we now re-estimate the parameters $\hat{\theta}_i$. This produces a set of plausible parameter values; from this set we can estimate empirical confidence intervals.

Specifically, in the Gaussian Process model, we see that the simulation data and the experimental data are jointly Gaussian:

$$\begin{pmatrix} y_{sim} \\ y_{obs} \end{pmatrix} \sim N\left(0, \begin{pmatrix} A(\lambda_i, \sigma_i^2) & B(\lambda_i, \sigma_i^2, \hat{\theta}) \\ B^T(\lambda_i, \sigma_i^2, \hat{\theta}) & C(\lambda_i, \sigma_i^2) \end{pmatrix}\right)$$

where λ_i and σ_i^2 have been fit to the simulation data and $\hat{\theta}$ has been fit to the experimental data. Observe that

$$y_{obs}|y_{sim} \sim N\left(B^T A^{-1} y_{sim}, C - B^T A^{-1} B\right)$$

which can be efficiently sampled. We sample this distribution M times, collecting M iid samples, y_{obs}^i, of the experimental data. For each sampled set of data, y_{obs}^i, we repeat the parameter estimation problem, described in the previous section, producing M parameter estimates $\hat{\theta}_i$. In the event that θ is a vector-valued quantity, this set can be used to compute correlations between the parameters, in addition to simple confidence intervals for the individual parameters.

VIII. Example Problem and Results

We consider the model calibration problem of estimating the modulus of elasticity in a cantilever beam (similar to [Swiler[13]] and [Romero 2008[14]]) We assume that the deflection in a beam is given by:

$$d = \frac{F\ell}{\epsilon I}$$

where F is a point force applied at one end of the beam, ℓ is the length of the beam, I is the beam's moment of inertia and ϵ is the modulus of elasticity, the parameter we are trying to estimate. To pose this problem as a vector-valued data set, we specify given forces $(F_1, \cdots, F_d) = F^T$, and take

$$\vec{d}(\ell, I, \epsilon) = F \cdot \frac{\ell}{\epsilon I}.$$

We generate 10 different simulation sets (of 400 input/parameter pairs), by using Latin Hypercube sampling of both the input space and the parameter space; for each simulation set, we use the same experimental data, generated with a known value of $\epsilon = 0.95$. For each set of simulated data, we apply both the ML algorithm presented above, as well as the Bayesian Markov Chain Monte Carlo algorithm in.[10] Table 1 shows the results from the 10 different cases. We see that the Maximum Likelihood estimates are closer to the true

Sample	1	2	3	4	5	6
ML	0.9429	0.9496	0.9498	0.9630	0.9381	0.9502
MCMC - Median	0.9655	0.9609	0.9706	0.9627	0.9660	0.9648
MCMC - Mean	0.9668	0.9602	0.9705	0.9628	0.9664	0.9658
Boot-CI	(0.9423, 0.9434)	(0.9491, 0.9499)	(0.9493, 0.9503)	(0.9627, 0.9632)	(0.9379, 0.9383)	(0.9498, 0.9505)
MCMC-CI	(0.9316, 1.0136)	(0.9295, 0.9975)	(0.9365, 1.0095)	(0.9291, 1.0003)	(0.9370, 0.9987)	(0.9260, 1.0141)
Sample	7	8	9	10	Mean	Variance
ML	0.9626	0.9518	0.9418	0.9628	0.9513	8.2×10^{-5}
MCMC - Median	0.9675	0.9716	0.9675	0.9710	0.9668	1.3×10^{-5}
MCMC - Mean	0.9686	0.9768	0.9674	0.9713	0.9677	2.1×10^{-5}
Boot-CI	(0.9622, 0.9630)	(0.9516, 0.9520)	(0.9412, 0.9423),	(0.9622, 0.9634)	-	-
MCMC-CI	(0.9336, 1.0088)	(0.9374, 1.0673)	(0.9301, 1.0053)	(0.9382, 1.0102)	-	-

Table 1. Parameter Estimation Results. ML: Maximum Likelihood Estimate; MCMC: Markov Chain Monte Carlo Median; Boot-CI: Parametric Bootstrap Confidence Interval; MCMC-CI: Markov Chain Monte Carlo Confidence Interval

value, and are not systematically biased (to be larger than the true value); the variance (with respect to the points used to build the underlying GP) is large (compared to the MCMC results). The MCMC method appears to produce a biased estimator (at least for this example) when either the mean or the median of the sample distribution is used; this bias is about 1% in both cases. However, the variance is smaller. This would suggest that the two methods implicitly choose to balance variance and bias in different ways.

The 95 % confidence intervals produced by the two methods are interesting to look at. The bootstrap results are based on 100 different samples (and don't change as the number of samples are increased). These confidence intervals are extremely tight, however only two of them contains the true parameter which is concerning[b] . The confidence intervals produced from the MCMC are based on 2000 samples generated by

[b]This may be due to the lack of an observation noise term in the ML code used to generate these results. It is expected that adding such a term will produce more variable samples, and thus larger confidence intervals.

the progression of the chain. These intervals are quite large, and in all cases contain the true parameter, but are skewed in the direction of the bias as well.

Finally, we present some timing results. Comparison of these two methods in terms of timings is a little difficult. For the Maximum Likelihood Estimation technique, the estimation of parameters and the generation of confidence intervals are separate, allowing one to time each part separately. For the MCMC method, the estimate is produced along with the confidence interval data. See Table 2 for timing results. It turned out that in this problem only one basis vector was needed to capture a significant percentage of the variance, and thus non of the efficiencies described were realized; however, as both the ML algorithm and the MCMC algorithm require the evaluation of the likelihood function, they could both benefit from the efficiencies described above. Another feature, and perhaps advantage, of the bootstrap method is that the samples can be generated in parallel, where as the parallelization of MCMC can be difficult.

	Mean	Max	Min
ML Estimate	263	352	220
MCMC Estimate	588	659	408
ML CI	49	51	47
MCMC CI	0.294	0.329	0.204

Table 2. Timing Results. Times are given in seconds. For confidence intervals, times reported are seconds per sample.

IX. Conclusion

We have presented an algorithm for parameter estimation from vector-valued data based on approximations of vector-valued Gaussian Processes by basis vectors weighted by independent Gaussian Processes. The use of the eigenvectors of the simulation data's covariance matrix for this basis enables us to efficiently evaluate the likelihood function of a given model. These efficiencies can be applied to any algorithm requiring evaluation of the likelihood function. Maximum Likelihood estimates have been compared to Markov Chain Monte Carlo estimates and were found to be less biased and more efficient to compute. However, the generation of confidence intervals using the ML model and parametric bootstrapping is orders of magnitude more expensive than when using MCMC, however, these intervals are quite small.

X. Acknowledgments

This work was funded in part by the Department of Energy's National Nuclear Security Administration's Predictive Science Academic Alliance Program (PSAAP).

References

[1]Draper, N. R. and Smith, H., *Applied Regression Analysis*, John Wiley, Hoboken, N. J., 1998.

[2]Seber, G. A. F. and Wild, C. J., *Nonlinear Regression*, John Wiley, Hoboken, N.J., 2003.

[3]Vugrin, K., Swiler, L. P., Roberts, R. M., Stuckey-Mack, N., and Sullivan, S., "Confidence Region Estimation Techniques for Nonlinear Regression: Three Case Studies," *Water Resources Research*, Vol. 43, 2007.

[4]Romero, V., "Validated Model? Not So Fast. The Need for Model Conditioning as an Essential Addendum to Model Validation," *AIAA-2007-1953, 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, Hawaii, April 2007, pp. 23–26.

[5]Xiong, Y., Chen, W., Tsui, K.-L., and Apley, D. W., "A better understanding of model updating strategies in validating engineering models," *Comput. Methods Appl. Mech. Engrg.*, Vol. 198, 2009, pp. 1327–1337.

[6]Kennedy, M. and O'Hagan, A., "Bayesian calibration of computer models (with discussion)," *J. R. Stat. Soc. Ser. B Stat. Methodol.*, Vol. 68, 2001, pp. 425–464.

[7]Higdon, D., Kennedy, M., Cavendish, J., Cafeo, J., and Ryne, R., "Combining field observations and simulations for calibration and prediction," *SIAM J. Sci.Comput.*, Vol. 26, 2004, pp. 448–466.

[8]Cressie, N. A. C., *Statistics for Spatial Data*, Wiley, 1993.

[9]Rasmussen, C. E. and Williams, C. K. I., *Gaussian Processes for Machine Learning*, The MIT Press, 2006.

[10]Higdon, D., Gattiker, J., Williams, B., and Rightly, M., "Computer Model Calibration Using High Dimensional Output," Tech. Rep. LA-UR-05-6410, Los Alamos National Laboratory, 2005.

[11]McFarland, J. M., *Uncertainty Analysis for Computer Simulations Through Validation and Calibration*, Ph.D. thesis, Vanderbilt University, 2008.

[12]Efron, B. and Tibshirani, R. J., *An Introduction to the Bootstrap*, Chapman and Hall, 1993.

[13]Swiler, L. P., Adams, B. M., and Eldred, M. S., "Model Calibration under Uncertainty: Matching Distribution Information," AIAA Paper AIAA-2008-5944, 2008.

[14]Romero, V. J., "Type X and Y Errors and Data and Model Conditioning for Systematic Uncertainty in Model Calibration, Validation and Extrapolation," Tech. Rep. 2008-01-1368, Sandia National Laboratories, 2008.