Parameter space exploration with Gaussian process trees

Robert B. Gramacy RBGRAMACY@AMS.UCSC.EDU
Herbert K. H. Lee HERBIE@AMS.UCSC.EDU
University of California, Santa Cruz, Applied Math & Statistics Department
Baskin Engineering: 1156 High St. Santa Cruz, CA 95064

William G. Macready WGM@EMAIL.ARC.NASA.GOV
University Affiliated Research Center
NASA Ames Research Center: Mail Stop 269-4, Moffett Field, CA 94035 USA

Abstract

Computer experiments often require dense sweeps over input parameters to obtain a qualitative understanding of their response. Such sweeps can be prohibitively expensive, and are unnecessary in regions where the response is easy predicted; well-chosen designs could allow a mapping of the response with far fewer simulation runs. Thus, there is a need for computationally inexpensive surrogate models and an accompanying method for selecting small designs. We explore a general methodology for addressing this need that uses non-stationary Gaussian processes. Binary trees partition the input space to facilitate non-stationarity and a Bayesian interpretation provides an explicit measure of predictive uncertainty that can be used to guide sampling. Our methods are illustrated on several examples, including a motivating example involving computational fluid dynamics simulation of a NASA reentry vehicle.

1. Introduction

As computing power has advanced so too has the fidelity of computer simulation. However, this fidelity often comes at great computational expense. Computational fluid dynamics simulations in which fluid flow phenomena are modeled are an excellent example — fluid flows over complex surfaces may be modeled accurately, but only at the cost of supercomputer resources.

Appearing in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004.

A simulation model defines a mapping (perhaps non-deterministic) from parameters describing the input to one or more output responses. Without an analytic representation of this mapping, the simulation must be run for many different inputs in order to build up an understanding of the simulation's possible outcomes. Computational expense and/or high dimensional input usually prohibits a naive approach to the mapping of the response surface. A computationally inexpensive approximation to the simulation (O'Hagan et al., 1999) with active learning is one possible remedy. If the approximation is a good match to the simulation, then samples may be drawn in regions of the input space where the output response is changing significantly. For models which return both predictions and associated confidences, regions can be identified where the model is unsure of the response.

We focus on Gaussian processes (GPs) as a suitable approximation for a number of reasons. GPs are conceptually straightforward, easily accommodate prior knowledge in the form of covariance functions, and return a confidence around predictions. In spite of these benefits there are three important difficulties in applying standard GPs in our setting. Firstly, inference on the GP scales poorly with the number of data points; typically requiring $\mathcal{O}(N^3)$ time, where N is the number of data points. Secondly, GP models are usually stationary as the same covariance structure is used throughout the entire input space. In the applications we have in mind, where subsonic flow is quite different than supersonic flow, this limitation is unacceptable. Thirdly, the error (standard deviation) associated with a predicted response under a GP model does not directly depend on any of the previously observed output responses. Instead, it depends only upon the previously sampled input settings $\{\mathbf{x}_i\}_{i=1}^N$ and the covariance matrix $\mathbf{C}(\mathbf{x}_i, \mathbf{x}_j)$. All of these shortcomings may be addressed by partitioning the input space into regions, and fitting separate GPs within each region. Partitioning allows for non-stationary behavior, and can ameliorate some of the computational demands (by fitting models to less data). Finally, a fully Bayesian interpretation yields uncertainty measures for predictive inference which can help direct future sampling.

This work draws upon two successful approaches. The use of trees and recursive partitioning for achieving non-stationarity has a long history (Breiman et al., 1984; Denison et al., 1998)— the Bayesian application of which is well worked out— and the use of GPs for active learning has also seen recent success (Seo et al., 2000). Alternative approaches for non-stationary modeling include mixtures of GPs (Tresp, 2001) and infinite mixtures of GPs (Rasmussen & Ghahramani, 2002).

This paper is structured as follows. We define the problem and review the necessary background in Section 2. Section 3 provides details on the use of Bayesian treed GP models including inference and prediction. Section 4 considers how the treed GP model is used to adaptively select input parameters. Section 5 presents results on real and simulated data, and provides a comparison of the proposed adaptive sampling scheme to two others used in the literature.

2. Background and related work

We model the simulation output as (Sacks et al., 1989)

$$\mathbf{Y}(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\beta} + \mathbf{W}(\mathbf{x}) \tag{1}$$

where $\mathbf{Y}$ is the (possibly multivariate) output of the computer model, $\mathbf{x}$ is a particular (multivariate) input value, $\boldsymbol{\beta}$ are linear trend coefficients, and $\mathbf{W}(\mathbf{x})$ is a zero mean random process with covariance $\mathbf{C}(\mathbf{x}, \mathbf{x}') = \sigma^2 \mathbf{K}(\mathbf{x}, \mathbf{x}')$, and $\mathbf{K}$ is a correlation matrix. Stationary Gaussian processes (Sacks et al., 1989) are a popular example of a model that fits this description.

As discussed in the introduction, we require more flexibility than offered by a stationary GP. To achieve non-stationarity we turn to binary trees, using them to partition the input space, and then fit a GP to each partition. This approach bears some similarity to the models of Kim et al. (2002), who fit separate GPs in each element of a Voronoi tessellation. Our approach is better geared toward problems with a smaller number of distinct partitions, leading to a simpler overall model. Using a Voronoi tessellation allows an intricate partitioning of the space, but has the trade-off of added complexity and can produce a final model that is difficult to interpret. The complexities of this added flexibility are not warranted in our application.

2.1. Stationary Gaussian Processes

GPs are a popular kernel-based method for regression and classification. Though the method can be traced back to Kriging (Matheron, 1963), it is only recently that they have been broadly applied in machine learning. Consider a training set $D = \{\mathbf{x}_i, \mathbf{t}_i\}_{i=1}^N$ of m_X-dimensional input parameters and m_Y-dimensional simulation outputs. We indicate the collection of inputs as the $N \times m_X$ matrix $\mathbf{X}$ whose ith row is $\mathbf{x}_i^\top$. A GP (Seo et al., 2000) is a collection of random variables $\mathbf{Y}(\mathbf{x})$ indexed by $\mathbf{x}$ having a jointly Gaussian distribution for any subset of indices. It is specified by a mean $\boldsymbol{\mu}(\mathbf{x}) = E\big(\mathbf{Y}(\mathbf{x})\big)$ and correlation function $\mathbf{K}(\mathbf{x}, \mathbf{x}') = E\big([\mathbf{Y}(\mathbf{x}) - \boldsymbol{\mu}(\mathbf{x})][\mathbf{Y}(\mathbf{x}') - \boldsymbol{\mu}(\mathbf{x}')]^\top\big)$. Given a set of observations D, the resulting density over outputs at a new point $\mathbf{x}$ is easily found to be Gaussian with

$$\begin{aligned} \text{mean} \quad & \hat{y}(\mathbf{x}) = \mathbf{k}^\top(\mathbf{x})\mathbf{K}^{-1}\mathbf{t}, \;\; \text{and} \\ \text{variance} \quad & \sigma_{\hat{y}}^2(\mathbf{x}) = \sigma^2[\mathbf{K}(\mathbf{x}, \mathbf{x}) - \mathbf{k}^\top(\mathbf{x})\mathbf{K}^{-1}\mathbf{k}^\top(\mathbf{x})]. \end{aligned}$$

For simplicity we assume that the output is scalar (i.e., we are modeling each output response independently and so $m_Y = 1$) so that the image of the covariance function is a scalar. To keep this equation simple, the linear term from (1) has been omitted here. We define $\mathbf{k}^\top(\mathbf{x})$ to be the N-vector whose ith component is $\mathbf{K}(\mathbf{x}, \mathbf{x}_i)$, $\mathbf{K}$ to be the $N \times N$ matrix with i, j element $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{t}$ to be the N-vector of observations with i component t_i. It is important to note that the uncertainty, $\sigma_{\hat{y}}^2(\mathbf{x})$, associated with the prediction has no direct dependence on the observed simulation outputs $\mathbf{t}$. Typically, the covariance function depends on hyperparameters which are determined either by maximizing the likelihood of D or integrating over them.

2.2. Bayesian Treed Models

A tree model partitions the input space and infers a separate model within each partition. Partitioning is often done by making binary splits on the value of a single variable (e.g., speed > 0.8) so that partition boundaries are parallel to coordinate axes. Partitioning is recursive, so each new partition is a sub-partition of a previous one. For example, a first partition may divide the space in half by whether the first variable is above or below its midpoint. The second partition will then divide only the space below (or above) the midpoint of the first variable, so that there are now three partitions (not four). Since variables may be revisited, there is no loss of generality by using binary splits as multiple splits on the same variable will be equivalent to a non-binary split. These sorts of models are often referred to as Classification and Regression Trees (CART) (Breiman et al., 1984). CART has become popular because of its ease of use, clear interpretation, and ability to provide a good fit in many cases.

The Bayesian approach is straightforward to apply to tree models (Chipman et al., 1998; Denison et al., 1998). Key is the specification of a meaningful prior for the size of the tree. Here we follow Chipman et. al who specify the prior through a tree-generating process. Starting with a null tree (all data in a single partition), the tree $\mathcal{T}$ is probabilistically split recursively, with each partition η being split with probability $p_{\rm SPLIT}(\eta, \mathcal{T}) = a(1 + q_\eta)^{-b}$ where q_η is the depth of η in $\mathcal{T}$ and a and b are parameters chosen to give an appropriate size and spread to the distribution of trees. More details are available in Chipman et. al (1998). We expect a relatively small number of partitions, and choose these parameters accordingly. As part of the process prior, we further require that each new region have at least five data points, since the parameters of a GP cannot be effectively estimated if there are too few points in a partition.

3. Non-stationary GPs via Trees

We begin by defining the model conditional on a particular tree, and later discuss integrating over possible trees.

3.1. Hierarchical Model

A tree $\mathcal{T}$ recursively partitions the input space into into R non-overlapping regions: $\{r_\nu\}_{\nu=1}^R$. Each region r_ν contains data $D_\nu = \{\mathbf{X}_\nu, \mathbf{t}_\nu\}$, consisting of n_ν observations. Each split in the tree is based on a selected dimension $u_j \in \{1, \dots, m_X\}$ and an associated split criterion s_j, so that one of the resulting sub-partitions consist of those observations in $\{\mathbf{X}_\nu, \mathbf{t}_\nu\}$ with the u_jth parameter less than s_j, and the other contains those observations greater than or equal to s_j.

Given a tree $\mathcal{T}$, we fit a stationary GP with linear trend (1) independently within each region. The $n_\nu \times n_\nu$ covariance matrix for the process in the νth region is denoted $\mathbf{K}_\nu$ and the linear trend coefficients are $\boldsymbol{\beta}_\nu$. We denote the full set of coefficients across all regions as $\boldsymbol{\beta}^\top = [\boldsymbol{\beta}_1^\top, \dots, \boldsymbol{\beta}_R^\top]$ (and similarly for all other region-specific parameters). The hierarchical generative model we use is:[1]

$$\begin{aligned} \mathbf{t}_\nu | \boldsymbol{\beta}_\nu, \sigma_\nu^2, d_\nu, g_\nu &\sim N(\mathbf{F}_\nu \boldsymbol{\beta}_\nu, \sigma_\nu^2 \mathbf{K}_\nu), \\ \boldsymbol{\beta}_\nu | \sigma_\nu^2, \mathbf{W}, \boldsymbol{\beta}_0 &\sim N(\boldsymbol{\beta}_0, \sigma_\nu^2 \mathbf{W}) \\ \boldsymbol{\beta}_0 &\sim N(\boldsymbol{\mu}, \mathbf{B}), \\ \sigma_\nu^2 &\sim IG(\alpha_0/2, q_0/2), \\ \mathbf{W}^{-1} &\sim W((\rho \mathbf{V})^{-1}, \rho), \end{aligned}$$

with $\mathbf{F}_\nu = (\mathbf{1}, \mathbf{X}_\nu)$, and $\mathbf{W}$ is a $(m_X+1)\times(m_X+1)$ matrix. N, IG, and W are the Normal, Inverse-Gamma, and Wishart distribution, respectively. The GP correlation structure for each partition, $\mathbf{K}_\nu$, is chosen from an isotropic power family with a fixed power p_0, but unknown range d_ν and nugget g_ν parameters:

$$K_\nu(\mathbf{x}_j, \mathbf{x}_k) = \exp\left\{ -\frac{[(\mathbf{x}_j - \mathbf{x}_k)^\top (\mathbf{x}_j - \mathbf{x}_k)]^{p_0}}{d_\nu} \right\} + g_\nu \delta_{j,k}, \tag{2}$$

where $\delta_{\cdot,\cdot}$ is the Kronecker delta function. For notational convenience we continue to refer $\mathbf{K}$ as a correlation matrix, even though with the nugget term, g, in $K(\cdot,\cdot)$ of Eq. (2) it is no longer technically a correlation matrix. Hierarchical mixture-priors on d and g can express our prior belief that the global covariance structure is non-stationary. Below, we shall refer to parameters to such hierarchical priors as $\boldsymbol{\gamma}$. Finally, constants $\boldsymbol{\mu}, \mathbf{B}, \mathbf{V}, \rho, \alpha_0, q_0, p_0$ are treated as known.

3.2. Prediction

Prediction under the above GP model is straightforward (Hjort & Omre, 1994). The predicted value of $\mathbf{y}$ at $\mathbf{x}$ is normally distributed with mean and variance

$$\hat{y}(\mathbf{x}) = \mathbf{f}^\top(\mathbf{x})\boldsymbol{\beta}_\nu + \mathbf{k}_\nu^\top \mathbf{K}_\nu^{-1}(\mathbf{t}_\nu - \mathbf{F}_\nu \boldsymbol{\beta}_\nu), \tag{3}$$

$$\hat{\sigma}(\mathbf{x})^2 = \sigma_\nu^2[\kappa(\mathbf{x},\mathbf{x}) - \mathbf{q}_\nu^\top(\mathbf{x})\mathbf{C}_\nu^{-1}\mathbf{q}_\nu(\mathbf{x})], \tag{4}$$

where $\mathbf{C}_\nu^{-1} = (\mathbf{K}_\nu + \mathbf{F}_\nu \mathbf{W} \mathbf{F}_\nu^\top)^{-1}$, $\mathbf{q}_\nu(\mathbf{x}) = \mathbf{k}_\nu(\mathbf{x}) + \mathbf{F}_\nu \mathbf{W}_\nu \mathbf{f}(\mathbf{x})$, $\mathbf{f}^\top(\mathbf{x}) = (1, \mathbf{x}^\top)$, $\kappa(\mathbf{x}, \mathbf{y}) = K_\nu(\mathbf{x}, \mathbf{y}) + \mathbf{f}^\top(\mathbf{x})\mathbf{W}\mathbf{f}(\mathbf{y})$, and $\mathbf{k}_\nu(\mathbf{x})$ is a $n_\nu-$vector with $\mathbf{k}_{\nu,j}(\mathbf{x}) = K_\nu(\mathbf{x}, \mathbf{x}_j)$, for all $\mathbf{x}_j \in \mathbf{X}_\nu$.

3.3. Estimating the model parameters

The data $D_\nu = \{\mathbf{X}, \mathbf{t}\}_\nu$ are used to estimate the parameters $\boldsymbol{\theta}_\nu \equiv \{\boldsymbol{\beta}_\nu, \sigma_\nu^2, d_\nu, g_\nu\}$, for $\nu = 1, \dots, R$. Parameters to the hierarchical priors ($\boldsymbol{\theta}_0 = \{\mathbf{W}, \beta_0, \boldsymbol{\gamma}\}$) depend only on $\{\boldsymbol{\theta}_\nu\}_{\nu=1}^R$. Conditional on the tree $\mathcal{T}$, we write the full set of parameters as $\boldsymbol{\theta} = \boldsymbol{\theta}_\mathbf{0} \cup \bigcup_{\nu=1}^R \boldsymbol{\theta}_\nu$. Samples from the posterior distribution of $\boldsymbol{\theta}$ are gathered using Markov chain Monte Carlo (MCMC) (Gelman et al., 1995).

[1]We omit the dependence on $\mathcal{T}$.

$\boldsymbol{\beta}_\nu, \sigma^2_\nu, \boldsymbol{\beta}_0, \mathbf{W}$ are updated with Gibbs steps. The other parameters require Metropolis Hastings (MH) steps. It is advantageous for mixing to analytically integrate out $\boldsymbol{\beta}$ and σ^2 to get a marginal posterior when updating d_ν and g_ν.

3.4. Tree Structure

Integrating out dependence on the tree structure $\mathcal{T}$ is accomplished by reversible-jump MCMC (RJ-MCMC) (Richardson & Green, 1997). We implement the tree operations *grow, prune, change*, and *swap* similar to those in Chipman et al. (1998). Tree proposals can change the size of the parameter space ($\boldsymbol{\theta}$). To keep things simple, proposals for new parameters— via an increase in the number of partitions R— are drawn from their priors, thus eliminating the Jacobian term usually present in RJ-MCMC. New splits are chosen uniformly from the set of marginalized input locations $\mathbf{X}$.

Swap and *change* tree operations are straightforward because the number of partitions (and thus parameters) stays the same. In a *change* operation we propose moving an existing split-point $\{u, s\}$, to either the next greater or lesser value of s (s_+ or s_-) along the uth dimension of (marginalized) locations from $\mathbf{X}$. This is accomplished by sampling s' uniformly from the set $\{u_\nu, s_\nu\}_{\nu=1}^{\lceil R/2 \rceil} \times \{+,-\}$. Parameters $\boldsymbol{\theta}_r$ in regions below the split-point $\{u, s'\}$ are held fixed. Uniform proposals and priors on split-points cause the MH acceptance ratio for *change* to reduce to a simple likelihood ratio.

The *swap* operation is similar, however we slightly augment the one described in Chipman et al. (1998). Swaps proposed on parent-child internal nodes which split on the same variable are always rejected because a child region below both parents becomes empty after the operation. However, if instead a *rotate* operation from Binary Search Trees (BSTs) is performed, the the proposal will almost always accept. Rotations are a way of adjusting the configuration (and thus height) of a BST without violating the BST property. Rotations encourage better mixing of the Markov chain by providing a more dynamic set of candidate nodes for pruning, thereby helping it escape local minima. Since the partitions at the leaves remain unchanged, the likelihood ratio of a proposed rotate is always 1. The only "active" part of the MH acceptance ratio is the prior on tree $\mathcal{T}$, preferring trees of minimal depth.

Grow and *prune* operations are more complex because they add or remove partitions, and thus cause a change in the dimension of the parameter space. The first step for either operation is to select a leaf node (for *grow*), or the parent of a pair of leaf nodes (for *prune*). We choose the node uniformly from the set of legal candidates. When a new region r is added, new parameters $\{d, g\}_r$ must be proposed, and when a region is taken away the parameters must be absorbed by the parent region, or discarded. When evaluating the MH acceptance ratio for either operation we marginalize over the $\{\boldsymbol{\beta}, \sigma^2\}_r$ parameters. One of the newly grown children is selected (uniformly) to receive the d and g parameters of its parent. To ensure that the resulting Markov chain is ergodic and reversible, the other new sibling draws its d and g parameters from their priors. Symmetrically, *prune* operations randomly select parameters d and g for the consolidated node from one of the children being absorbed. If the *grow* or *prune* operation is accepted, σ_r^2 can next be drawn from its marginal posterior (with $\boldsymbol{\beta}_r$ integrated out) after which draws for $\boldsymbol{\beta}_r$ and the other parameters for the rth region can then proceed as usual.

4. Adaptive Sampling

Having described the predictive algorithm used to model $P(\mathbf{y}|\mathbf{x})$, we now consider how to choose new sampling locations based on this distribution. Two criteria have been previously proposed. The simplest choice is to maximize the information gained about model parameters $\{\boldsymbol{\theta}, \mathcal{T}\}$ by selecting the $\tilde{\mathbf{x}}$ which has the greatest standard deviation in predicted output (Mackay, 1992). Given its simplicity this is the method we explore here. An alternative measure is to select $\tilde{\mathbf{x}}$ minimizing the resulting expected squared error over the input space (Cohn, 1994). A comparison between these two methods using standard GPs appears in Seo et al. (2000). In the results described below we use the difference between the 95% and 5% quantiles of the predicted output value as the measure of uncertainty— an MCMC scaled-approximate standard deviation. A comparison of our methods to the algorithms described by Seo et al. is included in Section 5.3.

To further improve our adaptive sampling we shall exploit Latin hypercube (LH) designs (Box et al., 1978). LHs systematically choose points that are spread out, taking on values throughout the region, but in different combinations across dimensions, thereby obtaining nearly full coverage with fewer points than a full gridding. To create a LH (McKay et al., 1979) with n samples in a m_X-dimensional space, one starts with an n^{m_X} grid over the search space. For each row in the first dimension of the grid, a row in each other dimension is chosen randomly without replacement, so that exactly one sample point appears in each row for each dimension. Within these chosen grid cells, the actual sample point is typically chosen randomly. In one dimension a LH design is equivalent to a complete grid, but as the number of dimensions grows, the number of points in the LH design stays constant, and the computational savings grow exponentially with m_X.

5. Results and Discussion

In this section we demonstrate an adaptive sampling scheme based on the Bayesian treed GP model of Section 3. Given N previous samples and their responses we use the model and its predictive quantiles to select a new location at which to request a response. For all experiments herein this is accomplished as follows: 15,000 MCMC rounds are performed, in which the parameters $\boldsymbol{\theta}|\mathcal{T}$ are updated. Every fourth round we also update the tree structure ($\mathcal{T}$) by drawing probabilistically from the discrete distribution $\{2/5, 1/5, 1/5, 1/5\}$, and attempting a {change, grow, prune, swap} operation accordingly. The first 5,000 of the 15,000 rounds are treated as burn-in, after which predictions are made using the parameters sampled during the remaining 10,000 rounds. Suppose that there are currently N locations $\mathbf{x}_i$ for which we have a response t_i.[2] An initial LH sample of size N_0 is used to get things started. At the beginning of the MCMC rounds we lay down a LH sample of N' new locations on which to predict. Quantiles (95th and 5th) are computed at each of the N' predictive locations. Based on their difference, one is selected. Every third adaptive sample is chosen probabilistically, treating the quantiles as a discrete distribution, while the rest are chosen by taking the maximum. Probabilistic samples are taken for robustness, as the maximum is only the optimal choice when the model is specified completely correctly. Finally, a response is elicited at the chosen input location, and the pair is then added into the data. The process is repeated, the model refit, and another adaptive sample is chosen from a new set of N' LH samples.

5.1. Synthetic Data

1-d Sinusoidal dataset:

Our first example is a simulated dataset on the input space $[0, 60]$. The true response is (Higdon, 2002):

$$t(x) = \left\{ \sin\left(\frac{\pi x}{5}\right) + \frac{1}{5}\cos\left(\frac{4\pi x}{5}\right) \right\} \theta(x - 35.75) \quad (5)$$

where θ is the step function defined by $\theta(x) = 1$ if $x > 0$ and $\theta(x) = 0$ otherwise. Zero mean Gaussian noise with sd $= 0.1$ is added to the response. This dataset typifies the type of non-stationary response surface that our model was designed to exploit.

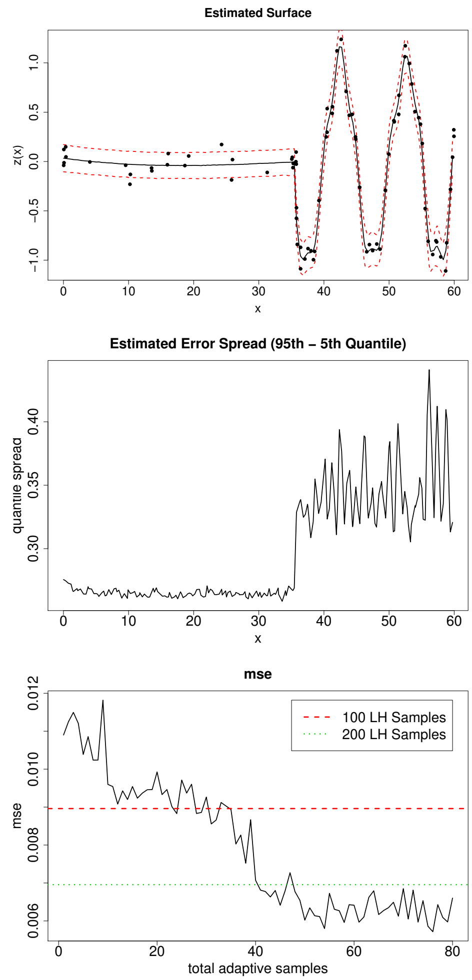

Figure 1. Top : surface estimate (mean and quantiles) with the initial and adaptively sampled data shown. *Middle*: MCMC quantile-based error over the domain. *Bottom:* MSE decreases over time.

Figure 1 shows the model after $N_0 = 20$ initial LH samples and 80 adaptive samples (with $N' = 200$) were drawn iteratively (as outlined above). The top graphs show a snap-shot of the predictive mean and predictive 5% and 95% quantiles after the 80th adaptive sample, illustrating that of 80 adaptive samples (dots) only about 25% are in the flat region. The quantile-based error differences (shown in middle graph) are indeed lower in this region. These errors also point to high model uncertainty around the split-point between the two regions, and consequently this part of the space has the highest concentration of adaptive samples. The bottom graph illustrates how the model adapts over time, showing that the mean standard error (MSE) decreases steadily as samples are added,

[2] We first translate and re-scale the data so that it lies on the unit cube in $\mathbb{R}^{m_X}$.

despite the fact that very few are added in the flat region. Moreover, it shows that adaptive sampling yields significantly lower MSE than using the same Bayesian treed GP model with 100 and even 200 LH samples. This corresponds to a factor of two decrease in the total number of samples compared to LH sampling. As might be expected, there were on average two partitions constructed in each round with quite small variance.

Detailed analysis of the learned models clearly illustrates another difference between sampling adaptively and using a LH design. The predictive surface based entirely on LH samples does not achieve the same accuracy (or resolution) as the surface which is based on the same number adaptive samples as it fails to discover the secondary structure contributed by the cosine term in (5). We also note that the stationary model (which does not partition the input space) fits the sinusoidal region about as well as the non-adaptive, non-stationary model, however the fit of the stationary model in the flat region is poor because it assumes a homogeneous correlation structure.

Our implementation in `C/C++` using the `ATLAS` library running on an Intel Xeon at 2 GHz required 10 seconds for sample 21 (first adaptive) and 65 seconds for sample 100 (last).

2-d Exponential dataset:

Next we show results for a two-dimensional input space in $[-2, 6] \times [-2, 6]$. The true response is given by

$$y(\mathbf{x}) = x_1 \exp(-x_1^2 - x_2^2) \tag{6}$$

As before, a small amount of Gaussian noise (with sd $= 0.001$) is added. There are $N_0 = 20$ initial LH samples and an additional 60 adaptively selected samples (from $N' = 2N$). Note that besides its dimensionality, a key difference between this data set and the last is that it is not defined using step functions; this smooth function does not have any artificial breaks between regions.

Figure 2 shows results which are similar to those for the 1-d sinusoidal data set. After the 60th adaptive sample, the *top-left* graph shows the fitted response generated by the sample shown at the *top-right*, most of which were, by design, drawn in $[-2, 2] \times [-2, 2]$ as the corresponding quantile based errors were largest here (not shown). The *bottom* graph compares adaptive sampling with LH sampling by MSE. The two horizontal lines in this graph represent the MSE of Bayesian treed GP models fit on 80 and 160 LH samples. Notice that the adaptive sampler beats out the LH design using half as many samples. Drawing the first and last adaptive samples took 10 and 45 seconds, respectively.

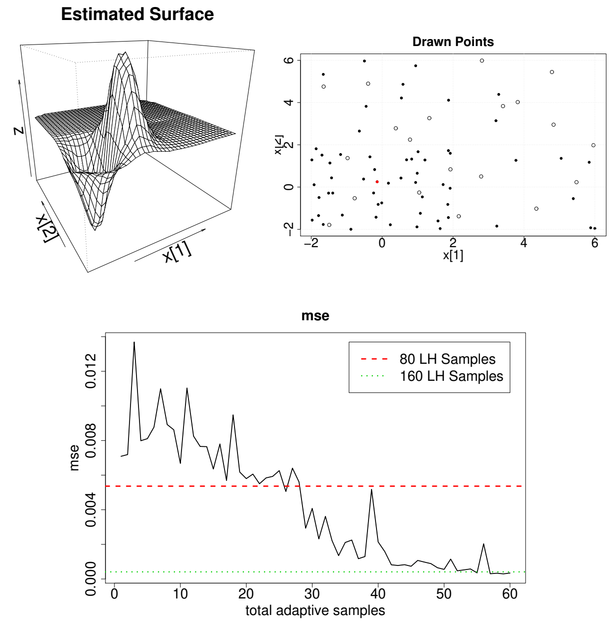

Figure 2. Top Left: surface estimate (mean and quantiles) with the initial and adaptively sampled data shown. *Top Right*: sampled points (initial-circles, and adaptive-dots) *Bottom* : MSE decreases over time.

5.2. 3-d CFD data

The third dataset is the motivating example for this work – the output from computational fluid dynamics simulations of a proposed reusable NASA launch vehicle called the Langley-Glide-Back Booster. The simulations involved the integration of the inviscid Euler equations over a mesh of 1.4 million cells (0.8 million cells were used for the supersonic cases).

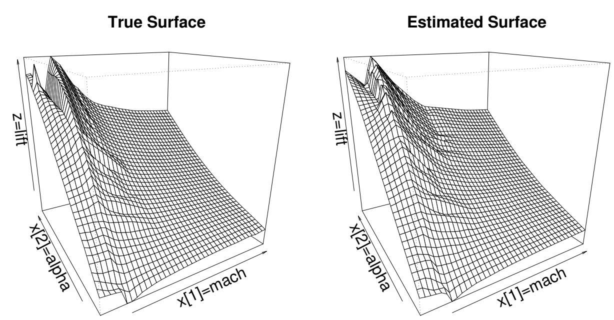

Figure 3. CFD projections. *Left:* true surface based on ∼3000 data points. *Right:* Fitted surface based on 750 adaptive samples, with the Bayesian treed GP model.

Each run of the Euler solver for a given set of parameters takes on the order of 5-20 hours on a high end workstation. Three input parameters were varied over (side slip angle, Mach number, and angle of attack) and for each setting of the input parameters six outputs were monitored. Using an interface to launch many jobs on many machines a total of around 3000 input configurations were tested. A more detailed description of this system and its results can be found in (Rogers et al., 2003).

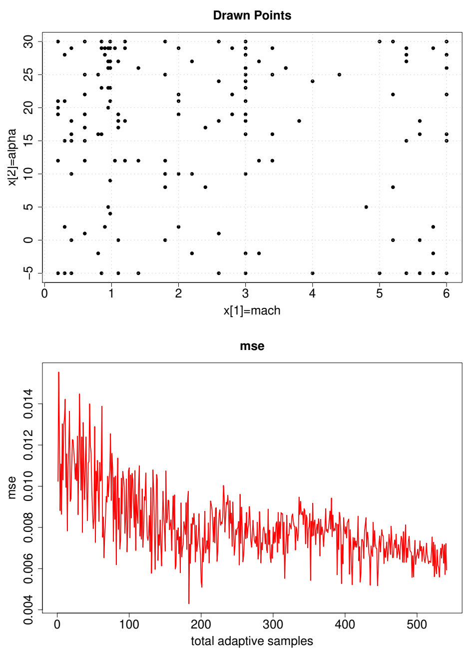

Figure 4. Top : adaptively sampled input locations. *Bottom*: MSE decreases over time.

The *left* side of Figure 3 shows one of the six outputs (lift) plotted as a function of speed (Mach) and angle of attack (alpha). The third input (side slip angle) is fixed at zero. Much of the space has a linear response, however it is highly non-linear near Mach one. We want to automatically sample points more frequently in this region. A fitted surfaced based upon 750 total samples, $N_0 = 200$ randomly selected initial subset for "peppering" and 550 chosen adaptively, is shown on the *right* side of Figure 3. Adaptively sampled configurations ($N' = N$ random subsample) are shown in the *top* panel of Figure 4.

Our Bayesian treed GP model has the desired behavior and focuses most of the adaptive sampling on the Mach 1 region. Visually, there is little difference between the true surface *(left)* and the estimated surface *(right)* shown in Figure 3. However, using a Bayesian treed GP model with adaptive sampling requires fewer than 1/4 as many samples compared to a simple gridding, saving thousands of hours of computing time. The first and last adaptive sample took 30 seconds and 10 minutes, respectively, which is fast relative to one execution of the Euler solver.

5.3. A comparator

As mentioned briefly in Section 4 our adaptive sampling scheme is motivated by an approach described by Seo et. al.. In their paper they outline two algorithms based on the predictive variances (4): ALM and ALC. ALM chooses the next adaptive sample by finding the input location $\tilde{\mathbf{x}} \in \tilde{\mathbf{X}}$ which maximizes (4). ALC chooses $\tilde{\mathbf{x}}$ to maximize the expected reduction in predictive variance at set of locations $\tilde{\mathbf{X}}$. Theoretically $\tilde{\mathbf{X}}$ could be an open subset of $\mathbb{R}^d$, although in practice it is taken to be a finite lattice or regular grid, chosen *a priori*. Section 5 describes a sampling algorithm similar to ALM, however we note two key differences. The first is the aforementioned randomization of every third adaptive sample for model robustness. The second is that Seo et. al. assume knowledge of the correct covariance structure (and parameters) at the start of the experiment — a significant assumption that we do *not* make. Furthermore they do not update their model in light of new responses.

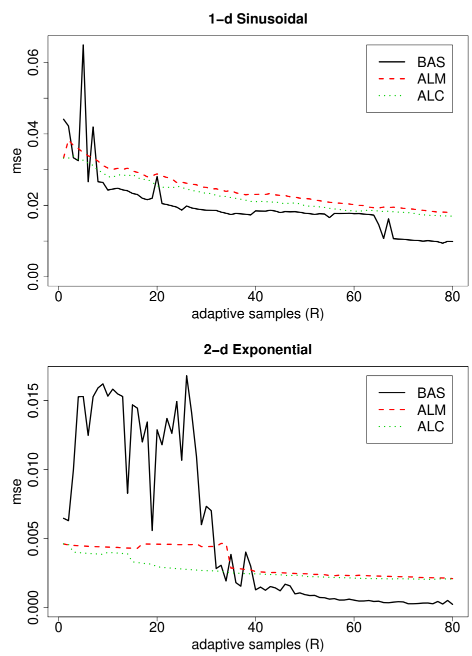

Figure 5. Top: Bayesian adaptive sampler (BAS) v. ALM and ALC in MSE on the sin-data; *Bottom:* exp-data.

Figure 5 demonstrates the advantage of adaptive sampling using Gaussian Process trees by comparing the MSE of samples chosen from a regular grid to those of ALM and ALC. ALM and ALC have the luxury of not having to learn the correct model and so fare better for the first adaptive samples. However, since our model can partition the space and adapt its parameters to account for the responses of sampled data, it eventually achieves lower error, with fewer total samples.

6. Conclusion

By building a non-stationary surrogate model we can sample adaptively and thus select smaller designs with a corresponding savings in overall computing time and a reduction in predictive variance. We have seen that savings grows as the dimension of the problem grows, allowing us to tackle problems that may not be feasible using a traditional parameter sweep, or LH sampling.

One of the next key steps is to be able systematically request more than one new sample at a time, in order to take full advantage of parallel computing resources. In a multiple-processor environment, simulator runs will finish at different times, and the main controller needs to be prepared with the next sampling point so that processors are not sitting idle. Additionally, other criteria for choosing the adaptive samples will be explored in the hopes of developing even more efficient sampling.

Acknowledgments

This work was partially supported by research subaward 08008-002-011-000 from the Universities Space Research Association and NASA. We would like to thank the area chair and three anonymous referees for their helpful suggestions.

References

Box, G. E. P., Hunter, W. G., & Hunter, J. S. (1978). *Statistics for experimenters*. New York: Wiley.

Breiman, L., Friedman, J. H., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.

Chipman, H., George, E., & McCulloch, R. (1998). Bayesian CART model search (with discussion). *Journal of the American Statistical Association*, *93*, 935–960.

Cohn, D. (1994). Neural network exploration using optimal experimental design. *Advances in Neural Information Processing Systems* (pp. 679–686). Morgan Kaufmann Publishers.

Denison, D., Mallick, B., & Smith, A. (1998). A Bayesian CART algorithm. *Biometrika*, *85*, 363–377.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (1995). *Bayesian data analysis*. London: Chapman and Hall.

Higdon, D. (2002). Space and space-time modeling using process convolutions. *Quantitative Methods for Current Environmental Issues* (pp. 37–56). London: Springer-Verlag.

Hjort, N. L., & Omre, H. (1994). Topics in spatial statistics. *Scandinavian Journal of Statistics*, *21*, 289–357.

Kim, H.-M., Mallick, B. K., & Holmes, C. C. (2002). *Analyzing non-stationary spatial data using piecewise Gaussian processes* (Technical Report). Texas A&M University – Corpus Christi.

Mackay, D. (1992). Information-based objective functions for active data selection. *Neural Computation*, *4*, 589–603.

Matheron, G. (1963). Principles of geostatistics. *Economic Geology*, *58*, 1246–1266.

McKay, M. D., Conover, W. J., & Beckman, R. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, *21*, 239–245.

O'Hagan, A., Kennedy, M. C., & Oakley, J. E. (1999). Uncertainty analysis and other inference tools for complex computer codes. *Bayesian Statistics 6* (pp. 503–524). Oxford University Press.

Rasmussen, C. E., & Ghahramani, Z. (2002). Infinite mixtures of Gaussian process experts. *Advances in Neural Information Processing Systems*. MIT Press.

Richardson, S., & Green, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society, Series B, Methodological*, *59*, 731–758.

Rogers, S. E., Aftosmis, M. J., Pandya, S. A., N. M. Chaderjian, E. T. T., & Ahmad, J. U. (2003). Automated cfd parameter studies on distributed parallel computers. *16th AIAA Computational Fluid Dynamics Conference*. AIAA Paper 2003-4229.

Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, *4*, 409–435.

Seo, S., Wallat, M., Graepel, T., & Obermayer, K. (2000). Gaussian process regression: Active data selection and test point rejection. *Proceedings of the International Joint Conference on Neural Networks IJCNN 2000* (pp. 241–246). IEEE.

Tresp, V. (2001). Mixtures of gaussian processes. *Advances in Neural Information Processing Systems 13* (pp. 654–660). MIT Press.