Adaptive exploration of computer experiment parameter spaces

Robert B. Gramacy, Herbert K. H. Lee
and William G. Macready
{rbgramacy,herbie}@ams.ucsc.edu
wgm@email.arc.nasa.gov

Many complex phenomena are difficult to investigate directly through controlled experiments. Instead, computer simulation is becoming a commonplace alternative to providing insight into such phenomena. The drive towards higher fidelity simulation continues to tax the fastest of computers, even in highly distributed computing environments. 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. In this article, we discuss the problem of fitting a response surface for a computer model when we also have the ability to design the experiment adaptively, updating the experiment as we learn about the model– a task to which we feel the Bayesian approach is particularly well-suited. Much of what is presented here follows our work in Gramacy et al. (2004).

Without an analytic representation of the mapping between inputs and outputs, simulations must be run for many different input configurations in order to build up an understanding of its possible outcomes. Computational expense of the simulation and/or high dimensional inputs often prohibit the naive approach of running the experiment over a dense grid of possible inputs. However, computationally inexpensive surrogate models can often provide accurate approximations to the simulation, especially in regions of the input space where the response is easily predicted.

For example, consider a model for the computational fluid dynamics of flight conditions for a proposed reusable NASA launch vehicle called the Langley Glide-Back Booster. The simulations involve the integration of the inviscid Euler equations over a mesh of 1.4 million cells. 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. There are three input parameters (side slip angle, Mach number, angle of attack). Six outputs are monitored (lift, drag, pitch, side-force, yaw, roll). The left panel of Figure 1 shows lift as a function of speed and angle of attack. Of note is the large ridge at Mach 1, where the flight abruptly transitions from subsonic to supersonic. While most of the output space is rather smooth, the ridge is clearly not. Thus there is interest in being able to automatically explore this surface, learning about the ridge and spending relatively more effort there than in the smooth regions.

The above experiment is an example of a situation where surrogate models combined with active learning techniques could direct future sampling, dramatically reducing the size of the final experimental design, saving thousands of hours of computing time. Sampling can be focused on input configurations where the surrogate model is least sure of its predicted response, either because the output response is changing significantly or because there are relatively few nearby data points already examined.

The traditional surrogate model used to approximate outputs to computer experiments is the Gaussian process (GP). GPs are conceptually straightforward, easily accommodate prior knowledge in the form of covariance functions, and return a confidence around predictions. In spite of its simplicity, there are three important disadvantages to standard GPs in our setting. Firstly, inference on the GP scales poorly with the number of data points, typically requiring computing time that grows with the cube of the sample size. Secondly, GP models are usually stationary in that 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.

All of these shortcomings may be addressed by partitioning the input space into regions, and fitting separate GPs within each region. Partitioning allows for modeling of non-stationary behavior, and can ameliorate some of the computational demands by fitting models to less data. Finally, a fully Bayesian

approach yields uncertainty measures for predictive inference which can help direct future sampling.

Bayesian Treed GP Models

A tree model partitions the input space and infers a separate model within each region. Partitioning is accomplished by making (recursive) binary splits on the value of a single variable (e.g., speed > 0.8) so that partition boundaries are parallel to coordinate axes. These sorts of models are often referred to as Classification and Regression Trees (CART). 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, provided that one can specify a meaningful prior for the size of the tree. We follow Chipman et al. (1998) 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 $p_{\text{SPLIT}}(\eta, \mathcal{T}) = a(1+q_\eta)^{-b}$ 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. We expect a relatively small number of partitions, and choose a and b accordingly.

Extending the work of Chipman et. al (2002), we fit a stationary GP with linear trend independently within each of R regions, $\{r_\nu\}_{\nu=1}^R$, depicted by the tree, $\mathcal{T}$. The GP correlation structure for each partition is chosen either from the isotropic power family, or separable power family of unknown (random) parameterization. In both cases, the correlation function takes the form $K_\nu(\mathbf{x}_j, \mathbf{x}_k) = K_\nu^*(\mathbf{x}_j, \mathbf{x}_k) + g_\nu \delta_{j,k}$ where $\delta_{\cdot,\cdot}$ is the Kronecker delta function, and K_ν^* is a *true* correlation representative from a parametric family. Priors which encode our belief that the global covariance structure is non-stationary are chosen for parameters to K_ν^* and g_ν.

Most literature on the *Design and Analysis of Computer Experiments* (Santner et al., 2003; Sacks et al., 1989) deliberately omits the nugget parameter (g) on grounds that computer experiments are deterministic. However, there are many reasons why one may wish to study a computer experiment with a model that includes an explicit noise component. In particular, the experiment may, in fact, be nondeterministic. Our collaborators tell us that their CFD solvers are often started with random initial conditions, involve forced random restarts when diagnostics indicate that convergence is poor, and that input configurations arbitrarily close to one another often fail to achieve the same estimated convergence. Thus a conventional GP model without a small-distance noise process (e.g. a nugget) can be a mismatch to such inherently non-smooth data.

The data $\{\mathbf{X}, \mathbf{t}\}_\nu$ in region r_ν are used to estimate the parameters $\boldsymbol{\theta}_\nu$ of the model active in the region. Parameters to the hierarchical priors depend only on $\{\boldsymbol{\theta}_\nu\}_{\nu=1}^R$. Samples from the posterior distribution are gathered using Markov chain Monte Carlo (MCMC). Integrating out dependence on the tree structure $\mathcal{T}$ is accomplished by reversible-jump MCMC. We implement the tree operations *grow, prune, change*, and *swap* similar to those in Chipman et al. (1998).

Adaptive Sampling

In the world of Machine learning, adaptive sampling would fall under the blanket of a research focus called *active learning*. Active learning techniques are currently being applied successfully in areas such as computational drug design/discovery by aiding in the search for compounds that are active against a biological target. However, we are not aware of any other active learning algorithms that use non-stationary modeling to help select small designs.

In the statistics community, the traditional approach to sequential data solicitation goes under the general heading of *(Sequential) Design of Experiments* (Santner et al., 2003). Depending on whether the goal of the experiment is inference or prediction (as described by a choice of utility), different algorithms for obtaining optimal designs can be derived. For example, one can choose the Kullback-Leibler distance between the posterior and prior distributions (with parameters $\boldsymbol{\theta}$) as a utility. For Gaussian process models with correlation function $\mathbf{K}$, this is equivalent to maximizing $\det(\boldsymbol{K})$. Subsequently chosen input configurations are called $D-$optimal designs. Choosing quadratic loss leads to what are called $A-$optimal designs. An excellent review of

Bayesian approaches to the design of experiments is contained in Chaloner & Verdinelli (1995) .

A hybrid approach to designing experiments employs active learning techniques. The idea is to consider a set of candidate input configurations and choose a rule for deciding the order in which they should be added to the design. For example, consider an approach which maximizes the information gained about model parameters by selecting the location $\tilde{\mathbf{x}}$ which has the greatest standard deviation in predicted output. This approach has been called ALM for Active Learning–Mackay, and has been shown to approximate maximum expected information designs. Given its simplicity this is the method we explored first. MCMC posterior predictive samples provide a convenient estimate of location-specific variance; namely the width of predictive quantiles.

An alternative algorithm is to select $\tilde{\mathbf{x}}$ minimizing the resulting expected squared error averaged over the input space, called ALC for Active Learning–Cohn. Conditioning on $\mathcal{T}$, the reduction in variance at a point $\mathbf{y}$ given that the location $\mathbf{x}$ is added into the data has a simple closed form. Averaging over $\mathbf{y}$ gives an estimate of the reduction in predictive variance obtained by adding $\mathbf{x}$ into the design—easily computed using MCMC methods. A comparison between ALC and ALM using standard GPs appears in (Seo et al., 2000).

Given these two hybrid approaches to sequential design, constructing a list of input configurations to send to available computing agents is simply a matter of sorting candidate locations ranked via either ALM or ALC. That way, the most informative locations are first in line for simulation when agents become available. Candidates could come from a pre-defined grid, a random sub-sample, a Latin Hypercube (LH) sample, an optimal design (e.g. a sequentially D-optimal design), or some combination (e.g. LH subsample of a D-optimal design).

Experimental Results

Bayesian adaptive sampling (BAS) proceeds in trials. Suppose N samples and their responses have been gathered in previous trials (or from a small initial grid, before the first trial). In the current trial the model is estimated for data $\{\mathbf{X}_i, t_i\}_{i=1}^N$. In accordance with the ALM algorithm, MCMC predictive quantiles are gathered, and sorted. Since our current experiments are based on pre-calculated pairs of input configurations and responses delivered by NASA, candidates (for now) must be chosen via random-subsample from the available data. We developed an artificial clustered simulation environment with a fixed number of agents in order to simulate the parallel and asynchronous evaluation of input configurations. After refreshing the sorted list of candidates, BAS gathers finished and running input configurations and adds them into the design. Predictive mean estimates are used as surrogate responses for unfinished (running) configurations until the true response is available. New trials start with fresh candidates.

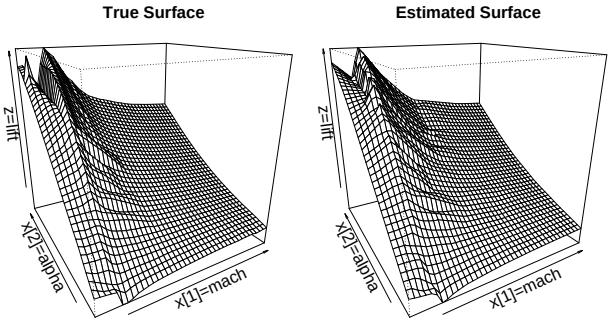

Figure 1: CFD projections. *Left:* true surface based on ∼3000 data points. *Right:* Fitted surface based on 750 adaptive samples over 100 trials.

The left side of Figure 1 shows one of the six outputs (lift) plotted as a function of speed (Mach) and angle of attack (alpha) based on the full design of more than 3000 input configurations. The third input, side slip angle (beta), is fixed at zero. A fitted surfaced based upon 750 total samples is shown on the right side of Figure 1. Configurations gathered using BAS (for beta=0) are shown in the Figure 2. Also shown in Figure 2 is a representative sample of the partitions obtained by integrating over the tree ($\mathcal{T}$). BAS has the desired behavior in that it fits different models around and on either side of the Mach 1 regions, and focuses most of the adaptive sampling around Mach 1. Further partitioning and sampling occurs for large angle of attack (alpha) near Mach 1

as indeed the response is changing most rapidly in this region.

Visually, there is little difference between the true surface (left) and the estimated surface (right) shown in Figure 1. 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. For a more detailed analysis of these results, experiments on other data, and comparisons with other approaches, the interested reader is referred to a paper we presented at ICML 2004 (Gramacy et al., 2004). Our future work includes running a live experiment on the NASA supercomputers.

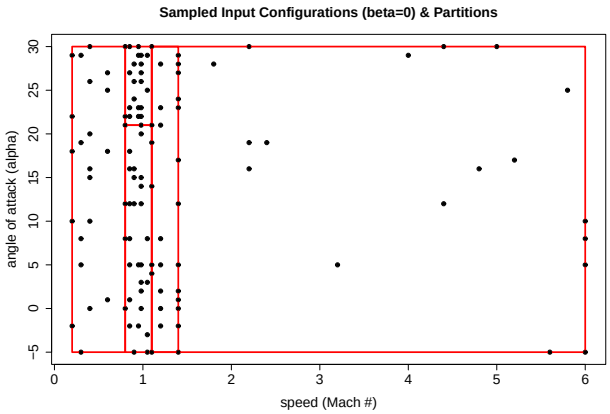

Figure 2: Adaptively sampled input locations (for a slice of side-slip-angle (beta = 0)).

In conclusion, creating a surrogate model for computer experiments is a problem that will continue to be of interest, as additional computing resources are put toward more accurate simulations rather than faster results. The Bayesian approach allows a natural mechanism for creating a sequential design based on the current estimated uncertainty.

References

Chaloner, K., & Verdinelli, I. (1995). Bayesian experimental design, a review. *Statistical Science*, *10 No. 3*, 273–1304.

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

Chipman, H. A., George, E. I., & McCulloch, R. E. (2002). Bayesian treed models. *Machine Learning*, *48*, 303–324.

Gramacy, R. B., Lee, H. K. H., & Macready, W. (2004). Parameter space exploration with Gaussian process trees. *Proceedings of the International Conference on Machine L earning* (pp. 353–360). Omnipress & ACM Digital Library.

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

Santner, T. J., Williams, B. J., & Notz, W. I. (2003). *The design and analysis of computer experiments*. New York, NY: Springer-Verlag.

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.