Sequential Design for Solving Inverse Problem for Expensive Deterministic Computer Simulators

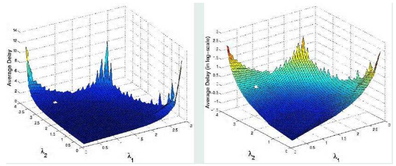
P. Ranjan[1], D. Bingham[2], and G. Michailidis[3]

[1]Acadia University, [2]Simon Fraser University and
[3]University of Michigan, Ann Arbor

1 Introduction

Computer simulation is frequently used as a cost-effective means to study complex deterministic processes. While a computer code (simulator) can often be viewed as an inexpensive way to gain insight into a system, it can still be computationally costly. Consequently, it is desirable to perform only a limited number of simulation trials. A computer experiment frequently involves the modelling of complex systems using a deterministic computer code.

In this article, we develop sequential design methodology for estimating the contour (also called a level set or iso-surface) of a non-degenerate complex computer code. In the network queuing application which motivated this work, the problem is estimating a contour of a globally or locally monotone response surface, where the contour identifies the boundary which distinguishes "good" from "bad" performance.



2 Methodology and Algorithm

The proposed approach consists of two main components. Firstly, a stochastic process model is used as a surrogate for the computer model which helps provide an estimate of the contour, and also uncertainty, given the current set of computer trials. Secondly, a new criterion is proposed which is used to identify computer trials aimed specifically at improving the contour estimate.

2.1 Model

Without loss of generality, assume that the design space (or input space) is the unit hypercube, $\chi = [0,1]^d$. The outputs for the simulation trials are held in the n-dimensional vector $y = y(X) = (y_1, y_2, \ldots, y_n)'$. The output of the simulator, $y(x_i)$, is modelled as:

$$y(x_i) = \mu + z(x_i); \qquad i = 1, ..., n, \tag{1}$$

where, μ is the overall mean, $z(x_i)$ is a spatial process with $E(z(x_i)) = 0$, $Var(z(x_i)) = \sigma_z^2$, $cov(z(x_i), z(x_j)) = \sigma_z^2 R_{ij}$ and,

$$R_{ij} = \text{corr}(z(x_i), z(x_j)) = \prod_{k=1}^{d} \exp\left\{-\theta_k (x_{ik} - x_{jk})^{p_k}\right\} \qquad \forall i, j. \tag{2}$$

More generally, $y(X)$ has multivariate normal distribution, $y(X) \sim N_n(\mathbf{1_n}\mu, \Sigma)$, where $\Sigma = \sigma_z^2 R$ and $R = [R_{ij}]$. The model can now be used to estimate responses at any non-sampled point x^*. The best linear unbiased predictor (BLUP) for $y(x^*)$ is (see Henderson 1975 for details)

$$\hat{y}(x^*) = \hat{\mu} + r' R^{-1}(y - \mathbf{1_n}\hat{\mu}), \tag{3}$$

with mean squared error

$$s^2(x^*) = \sigma_z^2 \left(1 - r' R^{-1} r + \frac{(1 - \mathbf{1_n}' R^{-1} r)^2}{\mathbf{1_n}' R^{-1} \mathbf{1_n}}\right), \tag{4}$$

where $r = (r_1(x^*), ..., r_n(x^*))'$, and $r_i(x^*) = \text{corr}(z(x^*), z(x_i))$. In practice the parameters are replaced with estimates obtained from MLE.

2.2 Improvement Function

To efficiently use the available resources, one would like to maximize the information provided by new computer trials in the neighborhood of the contour. In similar spirit to Jones et al. (1998), we propose an *improvement function* to help identify optimal computer trials. Define the improvement function as,

$$I(x) = \epsilon^2(x) - min\left\{(y(x) - a)^2, \epsilon^2(x)\right\}, \tag{5}$$

where $\epsilon(x) = \alpha s(x)$ represent the radius of the band around the estimated contour, $S(a)$, at process value $y = a$. The new trial is the maximizer of the expected improvement criterion $E[I(x)]$ given by

$$\begin{aligned} E[I(x)] \;=\; & \left[\alpha^2 s^2(x) - (\hat{y}(x) - a)^2\right]\left[\Phi\left(\frac{a - \hat{y}(x)}{s(x)} + \alpha\right) - \Phi\left(\frac{a - \hat{y}(x)}{s(x)} - \alpha\right)\right] \\ & + 2(\hat{y}(x) - a)s^2(x)\left[\phi\left(\frac{a - \hat{y}(x)}{s(x)} + \alpha\right) - \phi\left(\frac{a - \hat{y}(x)}{s(x)} - \alpha\right)\right] \\ & - \int_{a-\alpha s(x)}^{a+\alpha s(x)} (y - \hat{y}(x))^2 \phi\left(\frac{y - \hat{y}(x)}{s(x)}\right) dy. \end{aligned} \tag{6}$$

The expected improvement based criteria are specifically very efficient as it enforce a balance between global and local search for the contour of interest. While acting simultaneously, the first and last term encourage sampling near the contour, but in regions where we have little sampled information. The second term allows the sampling mechanism to occasionally jump away from the estimated contour to more sparsely sampled regions of the input space where the contour may plausibly exist. It is the latter scenario which justifies the optimization of $E[I(x)]$.

2.3 Contour Extraction

An attractive feature of the GASP model is that an implicit function can be easily constructed thereby allowing for easy extraction of the contour. For the GASP model outlined in Section 2.1, the d-dimensional implicit function which defines the contour estimate $S(a)$ is

$$a = \hat{\mu} + \hat{r}' \hat{R}^{-1}(y - \mathbf{1_n}\hat{\mu}). \tag{7}$$

The implicit function gives a compact and convenient representation of the contour.

2.4 Sequential Approach

1. Perform an initial experiment design (e.g., Latin Hypercube design, maximin design) of sample size $n = n_0$.
2. Fit the response surface in (3) to the available data.
3. Identify the design point which maximizes $E[I(x)]$ (shown in 6) and perform a computer trial at this design point.
4. Update the simulator data (i.e., $X = [X'\ x_{max}]'$, $Y = [Y'\ y(x_{max})]'$) and set $n = n + 1$.
5. Iterate between Step 2 and Step 4 until the experiment budget has been exhausted or some stopping criterion has been achieved.
6. Extract the desired contour $\hat{S} = \{x : \hat{y}(x) = a\}$ from this final surface.

2.5 Convergence of $E[I(x)]$

Theorem-1: *Under the correlation structure defined in equation(2) and the expected improvement function defined in equation(6),*

$$\lim_{n\to\infty} \sup_{x\epsilon\chi} E[I(x)] = 0.$$

More precisely, $\sup_{x\epsilon\chi} E[I(x)] = O\left(\frac{1}{log(n)}\right)$ *and therefore converges to zero in limit.*

3 Examples

For each of the examples, several choices of n_0 are considered and the performance is observed. In Section 3.1-3.3, we present the results that illustrate the sequential strategy. To provide a basis of comparison, we compared the proposed methodology to the simpler approach of laying out all design points according to a single Latin hypercube design (McKay et al. 1979) and extracting a contour from the estimated response surface.

3.1 Goldprice Function

Let $x_1, x_2 \in [0,1]$, and the output of the computer model $f(x_1, x_2)$ be generated from be the Goldprice function (Andre, Siarry and Dognon 2000). Let the contour of interest be $S(1.5 * 10^5)$. The contour at this height is not a contiguous curve, but the implicit function method we have used can easily extract the corresponding iso-surface.

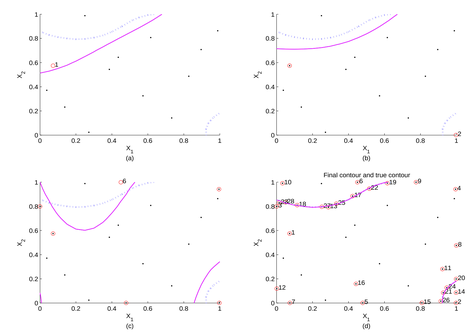


Illustration of sequential approach for contour estimation ($n = 40$ and $n_0 = 12$). The dotted curve refers to the true contour, the solid curve to the estimated contour and the dotted points refer to the initial design points. The red circles denote the location of new trials obtained by optimizing the expected improvement function.

3.2 Motivating Example

The design region for this application is not rectangular, and we have finitely many grid points to choose the trials from, thus we used a minimax design (Johnson, Moore and Ylvisaker 1990) rather than a Latin hypercube design as our starting design. We consider estimating the contour at delay of $a = 0.75$ (ie., $S(0.75)$).

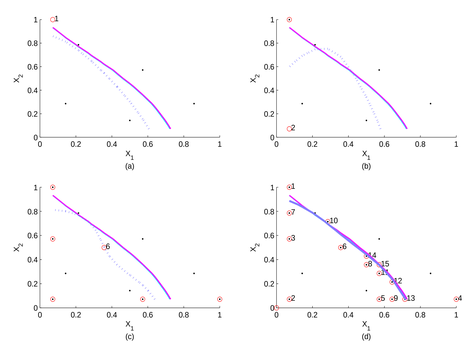


Illustration of sequential approach for contour estimation ($n = 20$ and $n_0 = 5$) for the queuing example. The solid curve refers to the true contour, the dotted curve to the estimated contour and the dotted points refer to the initial design points. The red circles denote the location of new trials obtained by optimizing the expected improvement function.

3.3 Goodness of Fit Measures

One can envision many ways to measure the closeness between the true (C_t) and the estimated contours ($C_{n_0,k}$). We present three such measures. Denote the discrete sample from $C_{n_0,k}$ as $\tilde{C}_{n_0,k} = \{x_1^k, \ldots, x_m^k\}$, where $x_i^k = (x_i^{1k}, \ldots, x_i^{dk})$ and similarly, for the true contour $\tilde{C}_t = \{x_1^t, \ldots, x_m^t\}$.

The first discrepancy measure calculates the *lack in correlation* between the estimated and true contours, and is denoted by

$$M_1 = \sum_{l=1}^{d} (1 - corr(x^{lk}, x^{lt})). \tag{8}$$

Here, x^{lk} and x^{lt} are the l-th coordinate of $C_{n_0,k}$ and C_t respectively. The second discrepancy measure is the average L_2 Euclidean distance between $C_{n_0,k}$ and C_t

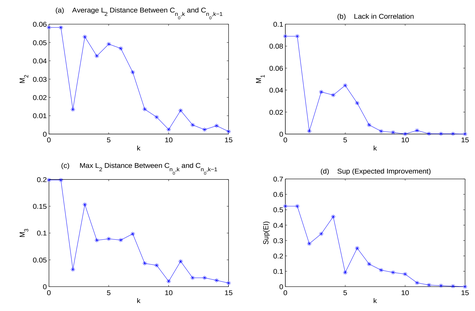
$$M_2 = \frac{1}{|\tilde{C}_{n_0,k}|} \sum_{x \in \tilde{C}_{n_0,k}} d(x, \tilde{C}_t), \tag{9}$$

where $d(x, \tilde{C}_t) = \min\{||x - y||_2 : y \in \tilde{C}_t\}$. Lastly, the third discrepancy measure is the maximum L_2 distance between $C_{n_0,k}$ and C_t

$$M_3 = \max\{d(x, \tilde{C}_t) : x \in \tilde{C}_{n_0,k}\}. \tag{10}$$

This discrepancy measure aims to measure maximum separation between two contours, and guards against the worst case scenario.

Since the true function is unknown in practice, by replacing C_t with $C_{n_0,k-1}$ in equations (8), (9) and (10), we get a similar set of discrepancy measures which can be used for diagnostic purposes.



4 Concluding Remarks

In this article, we have developed a sequential methodology for estimating a contour from a complex computer simulator that often requires fewer simulator runs compared to "all points in one shot" designs. Future work includes simultaneous investigation of multiple contours from complex computer models.

References

Jones, D., Schonlau, M., and Welch, W. (1998), "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, 13, 455-492.

Ranjan, P., Bingham, D. and Michailidis, G. (2008), "Sequential Experiment Design for Contour Estimation from Complex Computer Codes", *Technometrics*, 50(4), 527-541.