Multi-Level Monte Carlo Metamodeling

Imry Rosenbaum and Jeremy Staum
Department of Industrial Engineering and Management Sciences
Northwestern University

November 23, 2015

Abstract

Approximating the function that maps the input parameters of the simulation model to the expectation of the simulation output is an important and challenging problem in stochastic simulation metamodeling. Because an expectation is an integral, this function approximation problem can be seen as parametric integration—approximating the function that maps a parameter vector to the integral of an integrand that depends on the parameter vector. S. Heinrich and co-authors have proved that the multi-level Monte Carlo (MLMC) method improves the computational complexity of parametric integration, under some conditions. We prove similar results under different conditions that are more applicable to stochastic simulation metamodeling problems in operations research. We also propose a practical MLMC procedure for stochastic simulation metamodeling with user-driven error tolerance. In our simulation experiments, this procedure was up to hundreds of thousands of times faster than standard Monte Carlo.

1 Introduction

Simulation models are used extensively to analyze complex stochastic systems. The performance of a stochastic system, according to the model, is a function of the model's input parameters. For example, the expected number of jobs present at some future time in a queueing system is a function of arrival rates and service rates. In applications such as risk analysis or system design, we want to know the performance of the system for many possible inputs. That is, we want to know the response surface μ, the function that maps each input parameter vector θ in a domain Θ to system performance $\mu(\theta)$. We cannot run the simulation model for all possible values of its inputs, so we use simulation metamodeling to approximate the response surface using a limited number of simulation runs (Kleijnen and Sargent, 1997; Barton, 1998). Because the expectation of the simulation output is an integral over a probability space, stochastic simulation metamodeling can be seen as parametric integration—approximating the function that maps a parameter vector to the integral of an integrand that depends on the parameter vector. The computational complexity theory of parametric integration shows that, although parametric integration is computationally expensive, the multi-level Monte Carlo (MLMC) method can provide a better rate of convergence than standard Monte Carlo, under some conditions (Heinrich and Sindambiwe, 1999; Heinrich, 2000, 2001; Daun and Heinrich, 2013, 2014). One of the contributions of the present article is simply to point out that MLMC can be applied to stochastic simulation metamodeling and demonstrate that this can result in very large improvements to computational efficiency. We also make two further contributions in applying MLMC to stochastic simulation metamodeling.

One of these contributions is to analyze MLMC under a set of assumptions that are better suited to stochastic simulation metamodeling in operations research than those used so far in the literature on parametric integration. In operations research, the simulation output often fails to be a differentiable function of the input parameters. For example, the max function often appears in models of queueing, inventory, logistics, and financial systems. Such behavior is allowed in the Sobolev space framework of Heinrich (2000, 2001), but the results in those papers do not yield the conclusions we desire for stochastic simulation metamodeling in operations research. As we explain in Appendix B, those results require unnecessary assumptions, and relying on them would cause difficulties in metamodeling problems with more than one parameter. These results involve assumptions that higher-order moments of simulation output exist and that there is a bound on the number of random variates used in a single replication. They do not directly provide bounds on integrated mean squared error, which is a standard criterion for error in stochastic simulation metamodeling. Our analysis avoids these drawbacks. Moreover, we have avoided the difficulties of this Sobolev space framework by providing a less abstract analysis, using concepts that are more common in the operations research literature on stochastic simulation. We believe that our new results will have a practical value and our simpler derivations will have an expository value to the operations research community. Our primary assumption is Lipschitz continuity of the simulation output. This assumption has also been used to justify infinitesimal perturbation analysis estimators of the gradient of the response surface (Glasserman, 2004). Giles (2008) used Lipschitz continuity in analyzing MLMC for simulation of stochastic differential equations (SDEs). We show that this assumption is also useful in analyzing MLMC for stochastic simulation metamodeling.

Our other contribution is to provide a practical MLMC procedure for stochastic simulation metamodeling with a user-specified target precision, and to demonstrate its efficacy in computational experiments. Heinrich and Sindambiwe (1999) proposed a fixed-budget MLMC procedure for parametric integration. Our MLMC procedure has a user-specified target precision. It is based on the MLMC procedure of Giles (2008) for simulating SDEs. Applying his ideas to stochastic simulation metamodeling required further development, because the output of a simulation metamodeling procedure is a function, whereas the output of Giles's SDE simulation procedure is a scalar.

The outline of this article is as follows. Section 2 motivates the MLMC method and presents its basic ideas. Section 3 contains the mathematical framework, assumptions, and analyses of error and computational complexity. Section 4 introduces our practical MLMC procedure, and Section 5 contains simulation experiments that demonstrate how well it performs. In Section 6, we discuss some directions for future research. The issue of which metamodeling schemes should to use with MLMC is addressed in Appendix A. Appendix B contains a detailed comparison of our assumptions and results to those of previous articles on MLMC for parametric integration.

2 Multi-Level Monte Carlo Metamodeling

MLMC is not a substitute for a metamodeling scheme; it is a way of using a metamodeling scheme, by which we mean the combination of an experiment design and a function approximation method.

2.1 Motivation

The standard Monte Carlo method for stochastic simulation metamodeling is simply to run a simulation experiment and apply a function approximation method, as follows. The design of the simulation experiment involves choosing the values of the input parameter vector at which to run the simulation. These N vectors are called the design points and denoted $\theta_1, \ldots, \theta_N$. Let $\bar{Y}(\theta_i)$ denote the average simulation output over all the replications run at design point θ_i. Some function approximation method is then used to build a metamodel from the experiment's inputs $\theta_1, \ldots, \theta_N$ and outputs $\bar{Y}(\theta_1), \ldots, \bar{Y}(\theta_N)$. For example, regression and kriging are noteworthy as function approximation methods used in simulation metamodeling (Kleijnen, 2008). The hope is that the response surface μ will be well approximated by the metamodel $\hat{\mu}$ if the experiment design provides enough information, the noise in simulation output is not too large, and an appropriate function approximation method is used. When considering estimation of the value $\mu(\theta)$ of the response surface μ at a single point θ, we call θ the prediction point and $\hat{\mu}(\theta)$ the prediction.

The standard Monte Carlo method can be a computationally expensive way to get a metamodel that is accurate globally. Under our assumptions, formally stated in Sections 3.1–3.3, we bound the bias and variance in a prediction of the form

$$\hat{\mu}(\theta) = \sum_{i=1}^{N} w_i(\theta)\bar{Y}(\theta_i) \tag{1}$$

and show how increasing computational effort reduces a bound on mean squared error. The bias can be reduced by increasing the number of design points so that they fill space more densely, allowing the estimator to place more weight on simulation output generated at design points located in a smaller neighborhood $\mathcal{N}_\theta$ of the prediction point θ. The prediction's variance can be reduced by increasing the number of independent replications that are simulated at design points in $\mathcal{N}_\theta$. Now consider once more the problem of building a globally accurate metamodel, i.e., attaining sufficiently low mean squared error at every prediction point in the domain Θ. Because neighborhoods like $\mathcal{N}_\theta$ are small, there exists a large subset of $\{\mathcal{N}_\theta : \theta \in \Theta\}$ with the property that all the neighborhoods in it are disjoint from each other. Thus, to get a globally accurate metamodel, standard Monte Carlo requires many replications to be run in each of many disjoint neighborhoods. The need to have many replications (to attack variance) in each of many disjoint neighborhoods (to attack bias) is the source of the great computational expense of producing a globally accurate metamodel in stochastic simulation with standard Monte Carlo.

The essential idea of MLMC is to design a simulation experiment in which bias and variance can be attacked separately, yielding an improvement in computational efficiency. An MLMC simulation experiment involves multiple levels, and function approximation is done at each level. The finest level, which has the most design points and the fewest replications at each design point, yields a metamodel with low bias and high variance. Its bias is the bias of the MLMC metamodel. The coarser levels have fewer design points, but more replications at each design point. The coarser levels serve, like control variates, to reduce the variance of the MLMC metamodel.

2.2 The Multi-Level Monte Carlo Method

Following the general explanation of MLMC by Giles (2013), we explain MLMC for metamodeling in terms of control variates with unknown means, beginning by supposing that there are just two levels. Then MLMC uses two sets of design points: a coarse set $\mathcal{T}_0$ of N_0 design points, and a fine set $\mathcal{T}_1$ in which the number of design points is $N_1 > N_0$. Let $\hat{\mu}_\ell(\theta, M, \varpi)$ denote the prediction at θ of a metamodel built from a simulation experiment that runs M replications of the simulation using random number stream ϖ at every design point in $\mathcal{T}_\ell$. MLMC runs a small number M_1 of replications on $\mathcal{T}_1$, yielding a high-variance fine estimator $\hat{\mu}_1(\theta, M_1, \varpi^1)$. Suppose that the point sets are nested: $\mathcal{T}_0 \subset \mathcal{T}_1$. Then a high-variance coarse estimator $\hat{\mu}_0\left(\theta, M_1, \varpi^1\right)$ is computed using the simulation output from those design points that are in both $\mathcal{T}_0$ and $\mathcal{T}_1$.

Remark 1 In MLMC, it is not required that the point sets be nested so that a coarser point set is a subset of a finer point set. However, if they are nested, a computational savings results for MLMC. The number of replications simulated to compute $\hat{\mu}_1(\theta, M_1, \varpi^1)$ and $\hat{\mu}_0(\theta, M_1, \varpi^1)$ is $M_1 N_1$ if $\mathcal{T}_0 \subset \mathcal{T}_1$, and it would be $M_1(N_0 + N_1)$ if these point sets were disjoint.

Because of the correlation between these two estimators, the coarse estimator could be used as a control variate for the fine estimator, if the mean of the coarse estimator were known. Although the mean is unknown, control variates with unknown mean can decrease variance if the unknown mean is cheap to estimate (Emsermann and Simon, 2002; Pasupathy et al., 2012). In the setting of MLMC, the coarse estimator is relatively cheap to simulate because $N_0 < N_1$. Let $\mu_\ell(\theta)$ denote the expectation of the prediction at θ of a metamodel built using $\mathcal{T}_\ell$ as the set of design points. MLMC estimates the mean $\mu_0(\theta)$ by $\hat{\mu}_0(\theta, M_0, \varpi^0)$, which is a low-variance coarse estimator with a large number $M_0 > M_1$ of replications generated using a random number stream ϖ^0 that is independent of the random number stream ϖ^1 used previously. The MLMC estimator of $\mu(\theta)$ is

$$\hat{\mu}_1(\theta, M_1, \varpi^1) - \left(\hat{\mu}_0(\theta, M_1, \varpi^1) - \hat{\mu}_0(\theta, M_0, \varpi^0)\right), \tag{2}$$

where the high-variance fine estimator $\hat{\mu}_1(\theta, M_1, \varpi^1)$ is viewed as fundamental, the correlated high-variance coarse estimator $\hat{\mu}_0(\theta, M_1, \varpi^1)$ is its control variate with unknown mean, and the independent low-variance coarse estimator $\hat{\mu}_0(\theta, M_0, \varpi^0)$ is an estimator for the mean of the control variate. The bias of the MLMC estimator is the same as the bias of the fine-level estimator $\hat{\mu}_1(\theta, M_1, \varpi^1)$, because the coarse-level estimators $\hat{\mu}_0(\theta, M_1, \varpi^1)$ and $\hat{\mu}_0(\theta, M_0, \varpi^0)$ have the same expectation, $\mu_0(\theta)$. For purposes of analyzing variance, we rewrite the MLMC estimator as the sum of two independent terms, a baseline estimator and a refinement estimator:

$$\hat{\mu}_0(\theta, M_0, \varpi^0) + \left(\hat{\mu}_1(\theta, M_1, \varpi^1) - \hat{\mu}_0(\theta, M_1, \varpi^1)\right). \tag{3}$$

The baseline estimator has low variance because M_0 is large, but it may have unacceptably large bias because it uses only a coarse estimator, which has mean $\mu_0(\theta)$. The refinement estimator, which has mean $\mu_1(\theta) - \mu_0(\theta)$, is intended to reduce bias. It has low variance because of the high correlation between its two terms, a fine estimator and a coarse estimator generated using common random numbers (CRN).

Remark 2 It may seem odd to use CRN across design points in stochastic simulation metamodeling. Using CRN instead of independent sampling across design points can increase the mean squared error of metamodeling schemes that perform smoothing, such as stochastic kriging (Chen et al., 2012). It would indeed be acceptable to use independent sampling in the baseline estimator. However, it is essential to MLMC to use CRN somehow to reduce the variance of the refinement estimator, i.e., to generate correlation between the fine estimator and a control variate with unknown mean.

Next we show in detail how and why it can be better to use the MLMC estimator than to use the fine-level estimator alone. They have the same bias, but under the right circumstances, the MLMC estimator can achieve lower variance given the same computational budget C, measured in total number of replications simulated. For the sake of simplicity in this analysis, based on Equation (1), suppose that

$$\hat{\mu}_\ell(\theta, M, \varpi) = \sum_{i=1}^{N_\ell} w_i^\ell(\theta) \bar{Y}(\theta_i^\ell, \varpi), \tag{4}$$

where the point set $\mathcal{T}_\ell = \{\theta_1^\ell, \ldots, \theta_{N_\ell}^\ell\}$ and $\bar{Y}(\theta_i^\ell, \varpi)$ is the average simulation output over M replications run at design point θ_i^ℓ using random number stream ϖ. Then there exist constants $u_1(\theta)$, $v_0(\theta)$, and $v_1(\theta)$ such that the fine estimator's variance $\mathrm{Var}\left[\hat{\mu}_1(\theta, M, \varpi)\right] = u_1(\theta)/M$, and in MLMC, the baseline estimator's variance $\mathrm{Var}\left[\hat{\mu}_0(\theta, M_0, \varpi^0)\right] = v_0(\theta)/M_0$ and the refinement estimator's variance $\mathrm{Var}\left[\hat{\mu}_1(\theta, M_1, \varpi^1) - \hat{\mu}_0(\theta, M_1, \varpi^1)\right] = v_1(\theta)/M_1$. If we were to use the fine estimator alone, the budget would allow for $M = C/N_1$ replications per design point, yielding $\mathrm{Var}\left[\hat{\mu}_1(\theta, M, \varpi)\right] = u_1(\theta)N_1/C$. For MLMC, the computational cost is M_0N_0 for the baseline estimator and M_1N_1 for the refinement estimator (Remark 1). The MLMC variance is $v_0(\theta)/M_0 + v_1(\theta)/M_1$. It is minimized subject to the budget constraint $M_0N_0 + M_1N_1 = C$ by $M_0 = C\sqrt{v_0(\theta)/N_0}/(\sqrt{v_0(\theta)N_0} + \sqrt{v_1(\theta)N_1})$ and $M_1 = C\sqrt{v_1(\theta)/N_1}/(\sqrt{v_0(\theta)N_0} + \sqrt{v_1(\theta)N_1})$ (Giles, 2013). The resulting MLMC variance is $(\sqrt{v_0(\theta)N_0} + \sqrt{v_1(\theta)N_1})^2/C$. To simplify, suppose that $u_1(\theta) = v_0(\theta)$, i.e., the variance is the same for the coarse and fine estimators. Then $v_1(\theta) = v_0(\theta)(1 - \rho^2)$ where ρ is the correlation between $\hat{\mu}_1(\theta, M_1, \varpi^1)$ and $\hat{\mu}_0(\theta, M_1, \varpi^1)$ induced by CRN. With this simplification, the variance of the fine estimator alone is $v_0(\theta)N_1/C$, and the variance of MLMC is $(v_0(\theta)N_1/C)(\sqrt{N_0/N_1} + \sqrt{1-\rho^2})^2$. This illustrates how this two-level version of MLMC reduces variance when the coarse estimator has a low ratio N_0/N_1 of its computational cost to that of the fine estimator, yet it has a high correlation ρ with the fine estimator.

From the variance reduction factor $(\sqrt{N_0/N_1} + \sqrt{1-\rho^2})^2$, we can see why MLMC benefits from having more than just two levels. We want the correlation ρ to be large to make the coarse estimator effective as a control variate, which makes the refinement estimator's variance per replication small. This suggests making N_0 close to N_1 so that the coarse estimator and fine estimators are similar. We want N_0 to be much smaller than N_1 to make the unknown mean of the coarse estimator cheap to estimate, i.e., to make the baseline estimator's cost per replication small. This tension can be resolved by allowing for more than one level of refinement. The sequence of point sets $\mathcal{T}_0, \ldots, \mathcal{T}_L$ is such that N_0 is small, but $N_{\ell-1}$ and N_ℓ are similar for $\ell = 1, \ldots, L$.

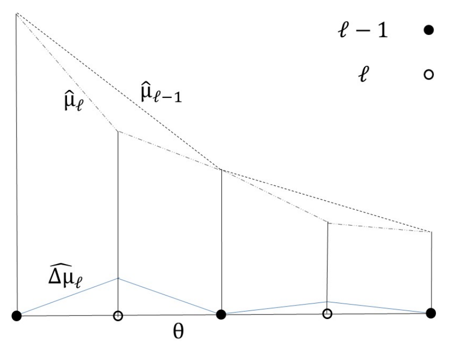


Figure 1: Estimating the refinement $\widehat{\Delta\mu}_\ell$ in linear interpolation due to using the ℓth grid (solid and open dots) instead of the $(\ell-1)$th grid (solid dots only).

The MLMC estimator of $\mu(\theta)$ is

$$\hat{\mu}_0(\theta, M_0, \varpi^0) + \sum_{\ell=1}^{L} \widehat{\Delta\mu}_\ell(\theta, M_\ell, \varpi^\ell), \tag{5}$$

where the ℓth refinement estimator

$$\widehat{\Delta\mu}_\ell(\theta, M_\ell, \varpi^\ell) = \hat{\mu}_\ell\left(\theta, M_\ell, \varpi^\ell\right) - \hat{\mu}_{\ell-1}\left(\theta, M_\ell, \varpi^\ell\right) \tag{6}$$

is the change in the metamodel's prediction caused by augmenting the coarser set $\mathcal{T}_{\ell-1}$ of design points to include all the points in the finer set $\mathcal{T}_\ell$. This is illustrated in Figure 1 for linear interpolation on a one-dimensional grid. The purpose of adding the refinement estimators to the baseline estimator $\hat{\mu}_0(\theta, M_0, \varpi^0)$ is to reduce bias. The bias of the MLMC estimator is the same as that of the finest estimator, generated on $\mathcal{T}_L$, because the mean of the MLMC estimator is $\mu_0(\theta) + \sum_{\ell=1}^{L}(\mu_\ell(\theta) - \mu_{\ell-1}(\theta)) = \mu_L(\theta)$. Equation (5) can be rewritten in terms of control variates with unknown means as

$$\hat{\mu}_L(\theta, M_L, \varpi^L) - \sum_{\ell=0}^{L-1} \left(\hat{\mu}_\ell\left(\theta, M_{\ell+1}, \varpi^{\ell+1}\right) - \hat{\mu}_\ell\left(\theta, M_\ell, \varpi^\ell\right)\right). \tag{7}$$

Here $\hat{\mu}_\ell\left(\theta, M_{\ell+1}, \varpi^{\ell+1}\right)$ serves as a control variate for $\hat{\mu}_{\ell+1}\left(\theta, M_{\ell+1}, \varpi^{\ell+1}\right)$, the unknown mean $\mu_\ell(\theta)$ is estimated by $\hat{\mu}_\ell\left(\theta, M_\ell, \varpi^\ell\right)$, which in turn gets its own control variate $\hat{\mu}_{\ell-1}(\theta, M_\ell, \varpi^\ell)$ with unknown mean, etc. In Equation 7, the finest estimator $\hat{\mu}_L(\theta, M_L, \varpi^L)$ is viewed as fundamental, and the remaining terms serve to reduce the variance of the MLMC estimator. The variance reduction is made possible by the first-order autocorrelation in the sequence $\hat{\mu}_L(\theta, M_L, \varpi^L), \hat{\mu}_{L-1}\left(\theta, M_L, \varpi^L\right) - \hat{\mu}_{L-1}\left(\theta, M_{L-1}, \varpi^{L-1}\right), \ldots, \hat{\mu}_0\left(\theta, M_1, \varpi^1\right) - \hat{\mu}_0\left(\theta, M_0, \varpi^0\right)$.

3 Theoretical Analysis

In this section, we lay out the mathematical framework in which we analyze stochastic simulation metamodeling. We also present our assumptions and the elements of an error analysis for stochastic simulation metamodeling. Finally, we provide a computational complexity theorem for MLMC.

3.1 Stochastic Simulation Metamodeling

Our goal is to approximate a function $\mu : \Theta \to \mathbb{R}$ that arises as an expectation: $\mu(\theta) = \mathbb{E}[Y(\theta)]$, where the input θ is a parameter vector that falls within a domain Θ. For example, θ may contain the prices that we set for each of the products our company sells, $Y(\theta)$ may be our profit in selling the products at these prices, and then $\mu(\theta)$ would be the expected profit. As another example, θ may contain the price and volatility of a stock and the strike price of an Asian option on the stock, $Y(\theta)$ may be the discounted payoff of an Asian option on such a stock and with such a strike price, and then $\mu(\theta)$ would be the present value of the Asian option. These examples are discussed in detail in Section 5. In stochastic simulation metamodeling, we approximate the response surface μ by a metamodel $\hat{\mu}$. The metamodel is built after running a simulation experiment in which the simulation is run at design points $\theta_1, \dots, \theta_N$. At each design point θ_i, we expend computational effort M_i. We assume that M_i is a number of independent replications, each yielding a realization of $Y(\theta)$. MLMC is also potentially applicable to steady-state simulations in which the computational effort M_i represents run length. Because the variance and bias of the simulation output behave differently as computational effort increases in steady-state simulation, we leave this topic for future research.

Our error criterion for assessing the quality of a stochastic simulation metamodeling procedure is mean integrated squared error:

$$\begin{aligned} \text{MISE} &= \mathbb{E}\left[\|\hat{\mu}-\mu\|_2^2\right] = \mathbb{E}\left[\int_\Theta \left(\hat{\mu}(\theta)-\mu(\theta)\right)^2 \, \mathrm{d}\theta\right] = \int_\Theta \mathbb{E}\left[\left(\hat{\mu}(\theta)-\mu(\theta)\right)^2\right] \, \mathrm{d}\theta \\ &= \int_\Theta \text{Var}\left[\hat{\mu}(\theta)\right] \, \mathrm{d}\theta + \int_\Theta \left(\mathbb{E}\left[\hat{\mu}(\theta)\right]-\mu(\theta)\right)^2 \, \mathrm{d}\theta = \|\text{Var}\left[\hat{\mu}\right]\|_1 + \|\text{Bias}\left[\hat{\mu}\right]\|_2^2. \end{aligned} \tag{8}$$

MISE equals integrated mean squared error because the order of integration over Θ and expectation can be interchanged by Tonelli's theorem.

To control the MISE of $\hat{\mu}$, we need to control bias and variance across the domain Θ. To do this, we rely on Assumption 1 about the domain, Assumptions 2 and 3 about the behavior of the simulation output $Y(\theta)$ as a function of the input θ, and Assumption 4 about the metamodeling scheme.

Assumption 1 The parameter domain Θ is a compact subset of $\mathbb{R}^d$ for some $d \in \mathbb{N}$.

3.2 Simulation Output

To formulate our next assumptions, we describe the simulation output Y as a random field. That is, the simulation output is a measurable function $Y : \Theta \times \Omega \to \mathbb{R}$, where $(\Omega, \mathcal{F}, \mathbb{P})$ is a probability space; Lebesgue measure is the measure on Θ. For any input $\theta \in \Theta$, the simulation output is a random variable denoted $Y(\theta)$. The response surface μ is the mean function of the random field: $\mu(\theta) = \mathbb{E}[Y(\theta)] = \int Y(\theta, \omega) \, \mathrm{d}\mathbb{P}(\omega)$.

For Monte Carlo, we want finite variance.

Assumption 2 For each $\theta \in \Theta$, the variance of the simulation output $Y(\theta)$ is finite.

Assumption 3 is that the simulation output Y is almost surely Lipschitz continuous, and the coefficient of Lipschitz continuity has finite second moment. The same assumption has also been used to analyze infinitesimal perturbation analysis (also called "pathwise") Monte Carlo estimators of the gradient of the response surface. For example, Glasserman (2004, § 7.2) uses some assumptions, on a more detailed framework in which the simulation output $Y(\theta)$ arises as a function of underlying random variates generated by the simulation, that are sufficient for Assumption 3 to hold. Using those assumptions, he derives results similar to Equations (10) and (11) below.

Assumption 3 There exists a random variable κ such that $\mathbb{E}[\kappa^2]$ is finite and

$$|Y(\theta_1, \omega) - Y(\theta_2, \omega)| \le \kappa(\omega) \|\theta_1 - \theta_2\|, \quad \text{for all } \theta_1, \theta_2 \in \Theta$$

holds almost surely.

For example, consider simulation output that is exponentially distributed with mean $\theta \in \Theta = [0, \theta_{\max}]$. This can be described as a random field Y given by $Y(\theta, \omega) = -\theta(\ln \omega)$ for $\omega \in \Omega = (0, 1)$, with the probability measure $\mathbb{P}$ being uniform on Ω. Assumption 3 holds with coefficient $\kappa(\omega) = -\ln(\omega)$, which has $\mathbb{E}[\kappa^2] = 2$.

We identify a scenario $\omega \in \Omega$ with a sequence of random numbers used in simulation. The function $Y(\cdot, \omega)$ is a realization of the random field Y and describes how the simulation output changes as a function of the input θ while the random numbers used in the simulation stay the same. In MLMC, it is essential to use common random numbers in running the simulation with different inputs (Remark 2). The same random number stream is used to run the simulation at multiple design points, and it contains independent substreams, which are used to generate independent replications of the simulation (L'Ecuyer et al. 2002). Our notation is that the random number stream $\varpi = \left\{\omega^j\right\}_{j \in \mathbb{N}}$, where the substream ω^j generates the jth replication. The average simulation output from M independent replications run with input θ and random number stream ϖ is

$$\bar{Y}\left(\theta, M, \varpi\right) = \frac{1}{M} \sum_{j=1}^{M} Y\left(\theta, \omega^j\right). \tag{9}$$

We need to consider different random number streams $\varpi^0, \varpi^1, \ldots$ because distinct levels in MLMC are simulated independently, i.e., with different random number streams. Equation (9) shows that the average simulation output forms a random field $\bar{Y} : \Theta \times \mathbb{N} \times \Omega^{\infty} \to \mathbb{R}$. That is, for any input $\theta \in \Theta$ and positive number M of replications, $\bar{Y}\left(\theta, M\right)$ is a random variable; for any random number stream $\varpi \in \Omega^{\infty}$, the function $\bar{Y}(\cdot, \cdot, \varpi)$ describes how the average simulation output would change if we changed the input and the number of replications while using the same random number stream. To make the substreams independent, the probability space built on Ω^{∞} involves product measure and the σ-field generated by the set of all cylinders, as is standard (Billingsley, 1995, §2).

Two crucial inequalities follow from Assumption 3. In Section 3.3, we use the Lipschitz continuity of the response surface,

$$|\mu(\theta + h) - \mu(\theta)| = \left|\mathbb{E}\left[Y\left(\theta + h\right)\right] - \mathbb{E}\left[Y\left(\theta\right)\right]\right| \leq \mathbb{E}\left[\left|Y\left(\theta + h\right) - Y\left(\theta\right)\right|\right] \leq \mathbb{E}\left[\kappa\right] \|h\|, \tag{10}$$

to bound the bias in metamodeling. The bound on the variance of the change in simulation output between nearby points,

$$\mathrm{Var}\left[Y\left(\theta + h\right) - Y\left(\theta\right)\right] \leq \mathbb{E}\left[\left(Y\left(\theta + h\right) - Y\left(\theta\right)\right)^2\right] = \mathbb{E}\left[\kappa^2\right] \|h\|^2, \tag{11}$$

yields a bound on the variance in metamodeling in Section 3.4.

3.3 Bias

Equation (10) can support a bound on the bias in metamodeling. The bias in metamodeling depends on the metamodeling scheme and the set of design points $\{\theta_1, \ldots, \theta_N\}$ at which the simulation is run. A key feature of the experiment design is the mesh size of the set of design points,

$$\Delta = \max_{\theta \in \Theta} \min_{i=1,\ldots,N} \|\theta_i - \theta\|. \tag{12}$$

First, let us consider a very simple metamodeling scheme and experiment design: nearest neighbor approximation on a grid. The prediction of the value of the response surface at the prediction point θ is the simulation output observed at the design point nearest to θ, i.e., $\hat{\mu}\left(\theta\right) = \bar{Y}(\theta_{i(\theta)})$, where $i(\theta) = \arg\min_{i=1,\ldots,N} \|\theta_i - \theta\|$. Figure 2 illustrates this metamodeling scheme in an example where the simulation output $Y(\theta)$ is an exponential random variable with intensity θ. The bias in this metamodeling scheme is $\mu(\theta_{i(\theta)}) - \mu(\theta)$. To apply Equation (10), we need to bound the distance from the prediction point to the nearest design point. This is determined by the experiment design. Let Θ, the domain in which the prediction point can be located, be the hypercube $[0,1]^d$. Let the experiment design be a grid of $N = k^d$ points, each of the form $(i_1/(k+1), \ldots, i_d/(k+1))$, where each $i_1, \ldots, i_d$ is an integer between 1 and k. The mesh size of the grid with N points is $\Delta(N) = \sqrt{d}/(k+1) < d^{1/2} N^{-1/d}$. From Equation (10), it follows that the bias $\mathbb{E}[\hat{\mu}(\theta)] - \mu(\theta)$ of nearest neighbor approximation with this grid design is bounded by $\mathbb{E}\left[\kappa\right] d^{1/2} N^{-1/d}$, uniformly for all $\theta \in [0,1]^d$.

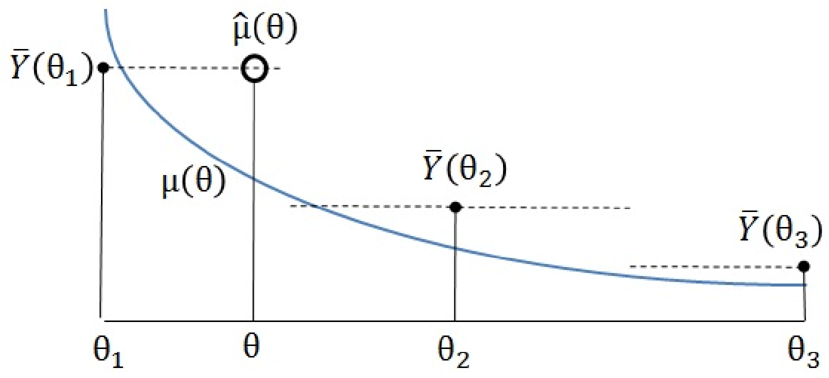


Figure 2: Nearest neighbor approximation, where the parameter θ is the intensity of an exponential random variable.

For the more general analysis of the way in which the bias in metamodeling depends on the experiment design and metamodeling scheme, an asymptotic analysis suffices to support the computational complexity analysis in Section 3.5. Consider a sequence of experiment designs indexed by ℓ, in which the number of design points $N_\ell \to \infty$. The notation for the point set in the ℓth experiment design is $\theta_1^\ell, \ldots, \theta_{N_\ell}^\ell$. The experiment design also includes the number of replications to simulate at each design point, but this is not essential to address while analyzing bias. Let $\hat{\mu}_\ell$ represent the metamodel built from the ℓth experiment. As a general matter, what we need is to show that Equation (10) supports a conclusion that $\|\mathrm{Bias}\,[\hat{\mu}_\ell]\,\|_2^2$, the integrated squared bias term in Equation (8), is $\mathcal{O}(s^{-2\alpha\ell})$ for some constants s and α.

The following assumption provides one way of doing that. It expresses the idea that the metamodel's prediction is a weighted sum that places most of the weight on design points near the prediction point, if there are many design points.

Assumption 4 For each $\theta \in \Theta$ and ℓ, Equation (4) holds, and each weight $w_i^\ell(\theta)$ on the simulation output from design point θ_i^ℓ in predicting at θ in the ℓth experiment design is a non-negative constant, determined by the experiment design and θ. Define $I^\ell(\theta; r) = \{i : \|\theta - \theta_i^\ell\| \leq r\}$, the set of design points in the ℓth experiment design that are within distance r of θ. There exist constants $s > 1$ and $\alpha > 0$ and sequences $\{p_\ell\}_{\ell\in\mathbb{N}}$ and $\{r_\ell\}_{\ell\in\mathbb{N}}$ such that p_ℓ and r_ℓ are $\mathcal{O}(s^{-\alpha\ell})$, and for all $\theta \in \Theta$,

$$\sum_{i \notin I^\ell(\theta; r_\ell)} w_i^\ell(\theta) \leq p_\ell \quad \text{and} \quad \left| 1 - \sum_{i \in I^\ell(\theta; r_\ell)} w_i^\ell(\theta) \right| \leq p_\ell.$$

That is, little weight is placed on design points that are far from the prediction point, and the total weight placed on design points that are near the prediction point nearly equals one. Appendix A.1 provides examples of metamodeling schemes that satisfy Assumption 4, including grids with piecewise linear interpolation, k-nearest-neighbor approximation, Shepard's method, or kernel smoothing. A scheme that uses experiment designs other than grids, such as low-discrepancy sequences (Asmussen and Glynn, 2007), can also satisfy Assumption 4, if the mesh size converges to zero. A function approximation method that does not fit Assumption 4 is linear regression of the simulation outputs $\bar{Y}(\theta_1^\ell), \ldots, \bar{Y}(\theta_{N_\ell}^\ell)$ on the inputs $\theta_1^\ell, \ldots, \theta_{N_\ell}^\ell$. Linear regression puts large amounts of weight on design points far from the prediction point and generates negative weights.

Proposition 1 If Assumptions 1, 3 and 4 hold, then $\|\mathrm{Bias}\,[\hat{\mu}_\ell]\,\|_2^2$ is $\mathcal{O}(s^{-2\alpha\ell})$.

Proof. Proof. For any $\theta \in \Theta$, the bias is

$$\begin{aligned}
|\mathbb{E}[\hat{\mu}_\ell(\theta)] - \mu(\theta)| &= \left| \sum_{i=1}^{N_\ell} w_i^\ell(\theta)\mu(\theta_i^\ell) - \mu(\theta) \right| \\
&= \left| \sum_{i \in I^\ell(\theta; r_\ell)} w_i^\ell(\theta)(\mu(\theta_i^\ell) - \mu(\theta)) + \sum_{i \notin I^\ell(\theta; r_\ell)} w_i^\ell(\theta)(\mu(\theta_i^\ell) - \mu(\theta)) - \left(1 - \sum_{i=1}^{N_\ell} w_i^\ell(\theta)\right)\mu(\theta) \right| \\
&\leq \sum_{i \in I^\ell(\theta; r_\ell)} w_i^\ell(\theta) \left|\mu(\theta_i^\ell) - \mu(\theta)\right| + \sum_{i \notin I^\ell(\theta; r_\ell)} w_i^\ell(\theta) \left|\mu(\theta_i^\ell) - \mu(\theta)\right| + \left|1 - \sum_{i=1}^{N_\ell} w_i^\ell(\theta)\right| |\mu(\theta)| \\
&\leq (1 + p_\ell)\mathbb{E}[\kappa]r_\ell + p_\ell \mathbb{E}[\kappa]\|\Theta\| + 2p_\ell|\mu(\theta)|
\end{aligned}$$

where $\|\Theta\|$ is the maximum distance between any two points in Θ. The derivation of all of the terms in the last line depends on the bounds of sums of weights in Assumption 4. The Lipschitz continuity of μ expressed in Equation (10) follows from Assumption 3. The Lipschitz continuity of μ is used in the derivation of the first and second terms in the last line. Continuity of μ also implies that μ is bounded on the compact domain Θ. Therefore we get a bound on bias that is uniform over all $\theta \in \Theta$:

$$|\mathbb{E}[\hat{\mu}_\ell(\theta)] - \mu(\theta)| \leq (1 + p_\ell)\mathbb{E}[\kappa]r_\ell + p_\ell \left(\mathbb{E}[\kappa]\|\Theta\| + 2\sup_{\theta \in \Theta} |\mu(\theta)|\right).$$

Both terms on the right side are $\mathcal{O}(s^{-\alpha\ell})$ because of the asymptotic behavior of the sequences $\{p_\ell\}_{\ell \in \mathbb{N}}$ and $\{r_\ell\}_{\ell \in \mathbb{N}}$ under Assumption 4. Therefore the integrated squared bias $\|\text{Bias}[\hat{\mu}_\ell]\|_2^2$ is $\mathcal{O}(s^{-2\alpha\ell})$.

□

Consider again the example of nearest-neighbor approximation on a grid in $[0,1]^d$. Let the number of points in the ℓth grid be $N_\ell = 2^{d\ell}$. Let $r_\ell = \sqrt{d}/(N_\ell^{1/d} + 1)$, the mesh size of the ℓth grid. Assumption 4 holds with $s = 2$, $\alpha = 1$, and $p_\ell = 0$. The bias is $\mathcal{O}(2^{-\ell})$ at any prediction point, and the integrated squared bias is $\mathcal{O}(2^{-2\ell})$.

3.4 Variance

Equation (11) provides the foundation for bounding the variance of simulation output (Lemma 1), the integrated variance in simulation metamodeling (Proposition 2), and the integrated variance in estimating refinements in MLMC (Proposition 4).

Lemma 1 If Assumptions 1, 2 and 3 hold, then $\sup_{\theta \in \Theta} \text{Var}[Y(\theta)] < \infty$.

Proof. Proof. For any $\theta, \theta' \in \Theta$,

$$\sqrt{\text{Var}[Y(\theta')]} \leq \sqrt{\text{Var}[Y(\theta)]} + \sqrt{\text{Var}[Y(\theta') - Y(\theta)]} \leq \sqrt{\text{Var}[Y(\theta)]} + \sqrt{\mathbb{E}[\kappa^2]}\|\Theta\|,$$

where the last inequality follows from Equation (11). □

As it was for bias, our bounds on variance in metamodeling are obtained only under some conditions on the metamodeling scheme and the sequence of point sets $\{\mathcal{T}_\ell\}_{\ell \in \mathbb{N}}$, and we rely on Assumption 4.

Proposition 2 If Assumptions 1, 2, 3 and 4 hold and M replications are simulated at each design point, then there exists a constant $\bar{v}$ such that $\|\text{Var}[\hat{\mu}_\ell(\cdot, M)]\|_1 \leq \bar{v}/M$ for all ℓ and M.

Proof. Proof. For any θ and ℓ, from Equation (4) we can see that

$$\text{Var}[\hat{\mu}_\ell(\theta, M)] \leq \left(\sum_{i=1}^{N_\ell} w_i^\ell(\theta)\right)^2 \frac{1}{M} \sup_{\theta' \in \Theta} \text{Var}[Y(\theta')].$$

The supremum is finite by Lemma 1. The squared sum of weights is bounded by $(1+2p_\ell)^2$. Because p_ℓ converges to 0, $(1+2p_\ell)^2$ is bounded over ℓ. Therefore we get

$$\|\mathrm{Var}\left[\hat{\mu}_\ell(\cdot, M)\right]\|_1 \leq \bar{v} = \|\Theta\| \sup_{\ell\in\mathbb{N}}(1+2p_\ell)^2 \sup_{\theta'\in\Theta} \mathrm{Var}[Y(\theta')].$$

□

Next we consider the variance and integrated variance of refinement estimators in MLMC. Similar to Equation (4), define

$$\hat{Y}_\ell(\theta,\omega) = \sum_{i=1}^{N_\ell} w_i^\ell(\theta) Y(\theta_i^\ell, \omega), \tag{13}$$

the metamodel prediction at θ formed from $\{Y(\theta_i,\omega) : \theta_i \in \mathcal{T}_\ell\}$.

Proposition 3 If Assumptions 2 and 4 hold, then $\mathrm{Var}\left[\hat{\mu}_0(\theta, M)\right] = \mathrm{Var}\left[\hat{Y}_0(\theta)\right]/M$ and $\mathrm{Var}\left[\widehat{\Delta\mu}_\ell(\theta, M)\right] = \mathrm{Var}\left[\hat{Y}_\ell(\theta) - \hat{Y}_{\ell-1}(\theta)\right]/M$.

Proof. Proof. From Equation (13), we get

$$\hat{\mu}_\ell(\theta, M, \varpi) = \sum_{i=1}^{N_\ell} w_i^\ell(\theta) \frac{1}{M}\sum_{j=1}^{M} Y(\theta_i^\ell, \omega^j) = \frac{1}{M}\sum_{j=1}^{M} \hat{Y}_\ell(\theta, \omega^j)$$

and hence

$$\widehat{\Delta\mu}_\ell(\theta, M, \varpi) = \frac{1}{M}\sum_{j=1}^{M}\left(\hat{Y}_\ell(\theta,\omega^j) - \hat{Y}_{\ell-1}(\theta,\omega^j)\right).$$

□

Define $v_\ell(\theta) = M_\ell \mathrm{Var}\left[\widehat{\Delta\mu}_\ell(\theta, M_\ell)\right]$, the variance per replication in estimating the refinement at the ℓth level.

Proposition 4 If Assumptions 1, 2, 3 and 4 hold, then $\|v_\ell\|_1$ is $\mathcal{O}(s^{-2\alpha\ell})$.

Proof. Proof. The desired conclusion holds if we can find a bound on $v_\ell(\theta)$ that is uniform over all $\theta \in \Theta$ and is $\mathcal{O}(s^{-2\alpha\ell})$. By Proposition 3, for any $\ell \geq 1$ and $\theta \in \Theta$, $v_\ell(\theta) = \mathrm{Var}\left[\hat{Y}_\ell(\theta) - \hat{Y}_{\ell-1}(\theta)\right]$. We then obtain

$$v_\ell(\theta) = \mathrm{Var}\left[\hat{Y}_\ell(\theta) - Y(\theta) + Y(\theta) - \hat{Y}_{\ell-1}(\theta)\right] \leq \left(\sqrt{\mathrm{Var}\left[\hat{Y}_\ell(\theta) - Y(\theta)\right]} + \sqrt{\mathrm{Var}\left[\hat{Y}_{\ell-1}(\theta) - Y(\theta)\right]}\right)^2.$$

It suffices to show that there is a bound on $\mathrm{Var}\left[\hat{Y}_\ell(\theta) - Y(\theta)\right]$ that is uniform over θ and is $\mathcal{O}(s^{-2\alpha\ell})$. We have

$$\hat{Y}_\ell(\theta) - Y(\theta) = \sum_{i\in I^\ell(\theta;r_\ell)} w_i^\ell(\theta)(Y(\theta_i^\ell) - Y(\theta)) + \sum_{i\notin I^\ell(\theta;r_\ell)} w_i^\ell(\theta)(Y(\theta_i^\ell) - Y(\theta)) - \left(1 - \sum_{i=1}^{N_\ell} w_i^\ell(\theta)\right) Y(\theta),$$

so the variance

$$\mathrm{Var}\left[\hat{Y}_\ell(\theta) - Y(\theta)\right] \leq 2\mathrm{Var}\left[\sum_{i\in I^\ell(\theta;r_\ell)} w_i^\ell(\theta)(Y(\theta_i^\ell) - Y(\theta))\right] + 2\mathrm{Var}\left[\sum_{i\notin I^\ell(\theta;r_\ell)} w_i^\ell(\theta)(Y(\theta_i^\ell) - Y(\theta))\right]$$
$$+ 2\mathrm{Var}\left[\left(1 - \sum_{i=1}^{N_\ell} w_i^\ell(\theta)\right) Y(\theta)\right].$$

Thus, it suffices to exhibit bounds on the variances of each of these three terms that are uniform in θ and are $\mathcal{O}(s^{-2\alpha\ell})$. Let us call these three variances

1. the variance due to nearby design points,
2. the variance due to distant design points, and
3. the variance due to weights' failure to sum to one.

Because of Assumption 3, Equation (11) holds. Assumption 4 provides bounds on the sums of weights. Using these facts, we get bounds on each of these three variances that are uniform in θ:

1. The variance due to nearby design points is bounded by $(1+p_\ell)\mathbb{E}[\kappa^2]r_\ell^2$. This is where it is essential that a common random number stream ϖ^ℓ is used to simulate at all design points in the ℓth level. Because of that, we can use Lipschitz continuity to bound $|Y(\theta_i^\ell,\omega^j)-Y(\theta,\omega^j)|\leq\kappa(\omega^j)|\theta_i^\ell-\theta|$; Equation (11) expresses a corresponding bound on variance.
2. The variance due to distant design points is bounded by $p_\ell^2\mathbb{E}[\kappa^2]\|\Theta\|^2$, where $\|\Theta\|$ is the maximum distance between any two points in Θ.
3. The variance due to weights' failure to sum to one is bounded by $4p_\ell^2\sup_{\theta\in\Theta}\mathrm{Var}[Y(\theta)]$. The supremum is finite by Lemma 1.

The bounds are all $\mathcal{O}(s^{-2\alpha\ell})$ due to the asymptotic behavior of the sequences $\{p_\ell\}_{\ell\in\mathbb{N}}$ and $\{r_\ell\}_{\ell\in\mathbb{N}}$ under Assumption 4. $\square$

3.5 Computational Complexity

In our setting, MLMC is asymptotically more computationally efficient than standard Monte Carlo in reducing MISE, defined in Equation (8). Theorem 1 gives bounds for the computational complexity of standard Monte Carlo and MLMC under the following conditions:

1. The integrated squared bias in metamodeling with the ℓth experiment design $\|\mathrm{Bias}\,[\hat{\mu}_\ell]\,\|_2^2\leq(bs^{-\alpha\ell})^2$.
2. The integrated variance in metamodeling $\|\mathrm{Var}\,[\hat{\mu}_\ell(\cdot,M)]\,\|_1\leq\bar{v}M^{-1}$ for all ℓ and M.
3. The integrated variance per replication in estimating the refinement at the ℓth level $\|v_\ell\|_1\leq\sigma^2s^{-2\alpha\ell}$.
4. The number N_ℓ of points in the set $\mathcal{T}_\ell\leq cs^{\gamma\ell}$.

Under such conditions, similar but more general theorems about MLMC hold, for other problems such as simulation of SDEs as well as parametric integration (Giles, 2013, 2015).

The specifics of stochastic simulation metamodeling enter in two ways. One is in Propositions 1-4, which showed that Conditions 1-3 follow from Assumptions 1-4. The other would be in showing that a specific metamodeling scheme, in which the experiment design satisfies Condition 4 with given s and γ, satisfies Assumption 4 for a certain value of α. For example, this was done at the end of Section 3.3 for nearest-neighbor approximation on a grid. Theorem 1 is of general use in proving the benefit of MLMC even when not all of Assumptions 1-4 hold. For example, Theorem 1 may be applied to a metamodeling scheme that does not satisfy Assumption 4 if it can be shown directly that Conditions 1-4 hold.

In the following theorem, we see that computational complexity does not depend on s at all, or on the rates α and γ separately, but only on the ratio $\phi=\gamma/2\alpha$. As we will see in the next section, the factor s^γ for growth in the number of points matters in practice for our MLMC procedure. However, the ratio ϕ is all that matters for computational complexity in an asymptotic sense.

Theorem 1 Under Conditions 1–4, the total number of replications that must be simulated to attain a MISE less than ϵ^2 is $\mathcal{O}(\epsilon^{-2(1+\phi)})$ for standard Monte Carlo, whereas for MLMC it is

- $\mathcal{O}(\epsilon^{-2})$ if $\phi < 1$,
- $\mathcal{O}((\epsilon^{-1}(\log \epsilon^{-1}))^2)$ if $\phi = 1$, or
- $\mathcal{O}(\epsilon^{-2\phi})$ if $\phi > 1$.

Proof. Proof. For standard Monte Carlo with point set $\mathcal{T}_L$ and M replications per design point, the integrated squared bias $\|\text{Bias}\,[\hat{\mu}_L]\,\|_2^2 \leq b^2 s^{-2\alpha L}$, and the integrated variance $\|\text{Var}\,[\hat{\mu}_L(\cdot, M)]\,\|_1 \leq \bar{v}M^{-1}$. Take $M = 2\bar{v}\epsilon^{-2}$ and $L = (\log_s(\sqrt{2}b\epsilon^{-1}))/\alpha$. This guarantees MISE $\leq \epsilon^2$. We have $N_L \leq cs^{\gamma L}$, which is $\mathcal{O}(\epsilon^{-\gamma/\alpha})$. The computational cost MN_L is $\mathcal{O}(\epsilon^{-(2+\gamma/\alpha)})$.

For MLMC, the proof is the same as the proof of Theorem 1 of Cliffe et al. (2011) or Giles (2013). There is a difference in the setting: the random output of our simulation metamodeling procedure is a function and we are concerned with its MISE, whereas Giles deals with the MSE of random scalar output. This difference is immaterial once we have formulated conditions on integrated squared bias and integrated variance that fit into the framework of conditions that Giles imposes on bias and variance. The other difference is that we have set Giles's rate β for the decay of refinement estimators' variance equal to 2α, because of the effect of our Assumption 3 through Equation (11) and Proposition 4.

□

Theorem 1 can be interpreted as saying that stochastic simulation metamodeling may be hard, if $\phi > 1$, or secretly easy, if $\phi < 1$. "Easy" means that the computational cost to attain MISE of ϵ^2 is $\mathcal{O}(\epsilon^{-2})$, which is the usual computational complexity of Monte Carlo estimation of the expectation of a random variable. "Hard" means that the computational cost to attain MISE of ϵ^2 is not $\mathcal{O}(\epsilon^{-2})$, but grows more rapidly as $\epsilon \to 0$. Although stochastic simulation metamodeling is always hard if you use standard Monte Carlo, it may be easy if you use MLMC. It is easy with MLMC if $\phi < 1$, or roughly speaking, if doubling the number of design points causes the squared bias and refinement variance to shrink by a factor of more than 2. If $\phi > 1$, then stochastic simulation metamodeling is hard even with MLMC, but the computational complexity is reduced by a factor of ϵ^{-2} compared to standard Monte Carlo.

Remark 3 It is not obvious that it is meaningful to compare $\mathcal{O}(\epsilon^{-2(1+\phi)})$ and $\mathcal{O}(\epsilon^{-2\phi})$ in this way; Theorem 1 merely describes the asymptotic behavior of upper bounds on computational cost. There exist stochastic simulation metamodeling problems for which the bounds are not tight: for example, if the simulation output $Y(\theta)$ is linear in the parameter θ, then piecewise linear interpolation on a grid has no bias. Information-based complexity theory can clarify such situations by showing that the bounds are tight for the worst-case random field Y among a relevant set of random fields. Such a result is given by Heinrich and Sindambiwe (1999).

For the purpose of illustrating how the dimension d of the domain Θ, the function approximation method, and the sequence of point sets $\{\mathcal{T}_\ell\}_{\ell \in \mathbb{N}}$ combine to tell us what α and γ (and therefore $\phi = \gamma/2\alpha$) should be, let us reconsider a previous example. The example is nearest-neighbor approximation on a grid in $\Theta = [0,1]^d$, with the ℓth grid $\mathcal{T}_\ell$ having $N_\ell = 2^{d\ell}$ points and mesh size $\sqrt{d}/(N_\ell^{1/d} + 1)$. This gives us $s = 2$ and $\gamma = d$. We showed before how Assumption 4 holds and implies that $\alpha = 1$. Therefore $\phi = d/2$. Higher dimension d can make stochastic simulation metamodeling harder. In the one-dimensional case, the computational complexity is $\mathcal{O}(\epsilon^{-3})$ for standard Monte Carlo and $\mathcal{O}(\epsilon^{-2})$ for MLMC. If $d > 2$, then the computational complexity is $\mathcal{O}(\epsilon^{-(d+2)})$ for standard Monte Carlo and $\mathcal{O}(\epsilon^{-d})$ for MLMC.

4 A Practical Procedure

Theorem 1 describes the computational complexity of attaining MISE less than a target ϵ^2 using MLMC without specifying how to attain it. Giles (2008) and Cliffe et al. (2011) analyze the optimal number L of levels and number M_ℓ of replications to use at each level, but these quantities depend on unknowns such as the bias and refinement variance at each level. Here we present a practical MLMC procedure for attaining the target MISE of ϵ^2. We closely follow the procedure of Giles (2008), modifying it because it deals with Monte

Carlo approximation of a scalar, whereas we are dealing with Monte Carlo approximation of a function, the response surface μ.

The user of the MLMC procedure must provide the target MISE ϵ^2, an initial number M^0 of replications to use at each level, a sequence of point sets $\{\mathcal{T}_\ell\}_{\ell\in\mathbb{N}}$ whose size $N_\ell = |\mathcal{T}_\ell|$ increases by a factor of roughly s^γ at each level, and a rate α such that integrated squared bias and refinement variance at the ℓth level are $\mathcal{O}(s^{-2\alpha\ell})$, as in the conditions established in Section 3.5. The user does not need to know exactly the rate α at which the bias and refinement variance decrease as the level increases. Suppose that α is chosen to be unnecessarily small, so that it is true that $\|\text{Bias}[\hat{\mu}_\ell]\|_2^2$ and $\|v_\ell\|_1$ are $\mathcal{O}(s^{-2\alpha\ell})$, but these statements are weak. Then the procedure will tend to proceed to a higher level than necessary, due to its stopping criterion.

The stopping criterion uses the factor $s^{2\alpha}$ to guess whether the integrated squared bias of the MLMC estimator with ℓ levels is small enough. The integrated squared bias of the MLMC estimator with ℓ levels is the same as that of $\hat{\mu}_\ell$, the metamodel built on the point set $\mathcal{T}_\ell$ alone. Condition 1 in Section 3.5 establishes a bound on this of $b^2 s^{-2\alpha\ell}$. If this bound were tight, then we would have

$$\left\|\mathbb{E}\left[\widehat{\Delta\mu}_\ell\right]\right\|_2^2 = \|\mathbb{E}\left[\hat{\mu}_\ell - \hat{\mu}_{\ell-1}\right]\|_2^2 \geq b^2 s^{-2\alpha\ell}(s^{2\alpha}-1) = (s^{2\alpha}-1)\,\|\text{Bias}\left[\hat{\mu}_\ell\right]\|_2^2 .$$

The goal of $(s^{2\alpha}-1)\,\|\text{Bias}\left[\hat{\mu}_\ell\right]\|_2^2 \leq \epsilon^2/2$ suggests a stopping criterion of

$$\left\|\widehat{\Delta\mu}_\ell\right\|_2^2 \leq (s^{2\alpha}-1)\frac{\epsilon^2}{2},$$

which is conservative, considering that $\mathbb{E}\left[\left\|\widehat{\Delta\mu}_\ell\right\|_2^2\right] \geq \left\|\mathbb{E}\left[\widehat{\Delta\mu}_\ell\right]\right\|_2^2$ promotes delayed stopping. Nonetheless, to avoid stopping too early due to unluckily low values of $\left\|\widehat{\Delta\mu}_\ell\right\|_2^2$, we also include a second estimator of $\left\|\mathbb{E}\left[\widehat{\Delta\mu}_\ell\right]\right\|_2^2$ in the test. The other estimator is $s^{-2\alpha}\left\|\widehat{\Delta\mu}_{\ell-1}\right\|_2^2$, based on extrapolating from the previous level. We stop if both estimators are below the threshold. That is, the stopping criterion is

$$\max\left\{\left\|\widehat{\Delta\mu}_\ell\right\|_2^2, s^{-2\alpha}\left\|\widehat{\Delta\mu}_{\ell-1}\right\|_2^2\right\} \leq (s^{2\alpha}-1)\frac{\epsilon^2}{2}. \tag{14}$$

In practice, the integrals cannot be observed, so we approximate them by averages over a set $\mathcal{P}$ of prediction points. Specifically, we sampled those prediction points independently and uniformly over Θ.

The stopping criterion provides some insight about choosing the spacing of the levels, i.e., choosing the factor s^γ by which the number of design points increases. As we saw in Section 3.5, $\alpha = \gamma/2\phi$, where ϕ is a constant determined by the metamodeling scheme, the structure of the point sets, and the dimension of the problem. We do not want to choose s^γ to be very large, or else doing the last level necessary to get MISE below ϵ^2 could be much more work than needed to be done. From the stopping criterion, we see that we also do not want to choose s^γ to be too close to 1, or else $s^{2\alpha}$ will be too close to 1, making the stopping criterion difficult to satisfy. In other words, if we believe that bias shrinks very slowly as we advance to the next level, then we believe that the remaining bias can be much larger than the refinement we observe at this level. In such a situation, checking the size of the refinement at one level would not work well.

There is also a question about how to estimate the variance of the level-0 metamodel or higher-level refinements. The method we used involves building a metamodel $\hat{Y}_\ell(\cdot,\omega)$ for each replication ω, as discussed in Section 3.4. For prediction point θ, $\text{Var}\left[\hat{\mu}_0(\theta, M^0)\right]$ or $\text{Var}\left[\widehat{\Delta\mu}_\ell(\theta, M^0)\right]$ are respectively estimated by the sample variances

$$\frac{1}{M^0-1}\left(\hat{Y}_0(\theta,\omega_j^0) - \hat{\mu}_0(\theta, M^0, \varpi^0)\right)^2 \quad \text{or} \quad \frac{1}{M^0-1}\left(\hat{Y}_\ell(\theta,\omega_j^\ell) - \hat{Y}_{\ell-1}(\theta,\omega_j^\ell) - \widehat{\Delta\mu}_\ell(\theta, M^0, \varpi^\ell)\right)^2, \tag{15}$$

based on Proposition 3. If the number of replications used for variance estimation is too large and the metamodeling scheme too computationally costly for this method to be desirable, then one may apply sectioning (Asmussen and Glynn, 2007, § 3.5) so that fewer metamodels need to be built. Other metamodeling schemes, such as stochastic kriging, provide their own variance estimates for a prediction $\hat{\mu}_\ell(\theta)$ at the same time as they make that prediction. In this case, those variance estimates could be used instead of Equation (15).

The procedure is as follows:

1. **Initialization** Set $\ell = 0$.

2. **Initial Simulation for Variance Estimation** For every design point in $\mathcal{T}_\ell$, and also in $\mathcal{T}_{\ell-1}$ if $\ell \geq 1$, simulate M^0 replications using common random numbers across design points. For every prediction point θ in $\mathcal{P}$, estimate the variance of $\hat{\mu}_0(\theta, M^0)$ if $\ell = 0$ or $\widehat{\Delta\mu}_\ell(\theta, M^0)$ if $\ell \geq 1$. One method to do this is in Equation (15). Denote the average of the variance estimates over all prediction points in $\mathcal{P}$ as V_ℓ.

3. **Further Simulation** Calculate the desired number of replications

$$M_\ell = \left\lceil 2\epsilon^{-2} \sqrt{\frac{V_\ell}{N_\ell}} \sum_{\ell'=0}^{\ell} \sqrt{V_{\ell'} N_{\ell'}} \right\rceil. \tag{16}$$

For every design point in the ℓth level, simulate more replications as needed.

4. **Assess Size of Refinement** If $\ell \geq 1$, for every prediction point θ in $\mathcal{P}$, calculate $\left(\widehat{\Delta\mu}_\ell\left(\theta, M_\ell, \varpi^\ell\right)\right)^2$. Denote the average over all prediction points in $\mathcal{P}$ as d_ℓ^2.

5. **Stopping Criterion** If $\ell < 2$ or $\max\{d_\ell^2, s^{-2\alpha} d_{\ell-1}^2\} > \left(s^{2\alpha} - 1\right)\epsilon^2/2$, set $\ell = \ell + 1$ and go to Step 2. Otherwise, set $L = \ell$.

6. **Final Simulation** For $\ell = 0, \ldots, L$, recalculate the desired number of replications

$$M_\ell = \left\lceil 2\epsilon^{-2} \sqrt{\frac{V_\ell}{N_\ell}} \sum_{\ell'=0}^{L} \sqrt{V_{\ell'} N_{\ell'}} \right\rceil. \tag{17}$$

For every design point in the ℓth level, simulate more replications as needed.

7. **Final Metamodel** The MLMC prediction at any prediction point θ is $\hat{\mu}_0(\theta, M_0, \varpi^0) + \sum_{\ell=1}^{L} \widehat{\Delta\mu}_\ell(\theta, M_\ell, \varpi^\ell)$.

Giles (2008) suggested Equation (17) for choosing the number of replications at each level. The goal is to make the variance (for us, the integrated variance) less than $\epsilon^2/2$ while allocating replications optimally across levels. The equation is related to the optimal allocation of replications in stratified sampling when the cost of a replication varies across strata (Asmussen and Glynn, 2007, § 5.7); the case $L = 1$ was analyzed in Section 2.2. Because Equation (17) cannot be evaluated before L is chosen, we use Equation (16) when simulating at the ℓth level. Equation (17) is expected to be more than Equation (16), so we may simulate more replications at earlier levels $\ell < L$ at the end, after choosing L.

5 Computational Experiments

This section demonstrates numerically the benefit of using our MLMC procedure by testing its performance on three examples, which are described in Section 5.1. We found that our procedure was eight to hundreds of thousands of times faster than a standard Monte Carlo procedure, depending primarily on target precision. MLMC yielded large benefits for every combination of the three examples and two function approximation methods we tested. In the examples, the metamodeling scheme combines a grid for the experiment design with multivariate linear interpolation or kernel smoothing as the function approximation method. In kernel smoothing, the kernel depends on the grid's mesh size. The result is that, in both of the function approximation methods, the prediction $\hat{\mu}(\theta)$ puts nonzero weight only on vertices of the grid cell that contains θ. The weight on the vertex θ_i is proportional to $\exp(-\|\theta - \theta_i\|)$, and the weights in the prediction are normalized to sum to 1.

The user-specified parameters of our MLMC procedure in Section 4 are the MISE target ϵ^2 and the initial number of replications M^0. In our experiments, we set $M^0 = 30$ and varied ϵ^2 to test the performance of the procedure over a wide range of target precision.

We performed multiple macro-replications of the MLMC simulation experiments, recording computational cost and estimating MISE in each of the macro-replications. We estimated MISE by selecting 100 prediction points and computing MSE at each of them, comparing $\hat{\mu}(\theta)$ at a prediction point θ to the true value $\mu(\theta)$, which is known in these examples. The MISE attained by our MLMC procedure was less than the target ϵ^2 in all experiments that we ran. Sometimes it was less than half the target. The computational cost of an MLMC experiment is $\sum_{\ell=0}^{L}(1+2^\ell)^d M_\ell$, where L is the level at which the MLMC procedure stops, and the number of replications M_ℓ is given in Equation (17). We compare the computational cost of our method to the cost of a standard Monte Carlo method, which uses a single level. In designing a standard Monte Carlo experiment, it is not obvious how to choose the number of design points. We compare to a standard Monte Carlo experiment in which the grid of design points is the same as the finest grid used by our MLMC procedure when it stops at level L. This aims for an integrated squared bias less than $\epsilon^2/2$ for the standard Monte Carlo estimator, same as for the MLMC estimator. The standard Monte Carlo experiment uses a grid of $(1+2^L)^d$ points, each having M_L^* replications, for a computational cost of $(1+2^L)^d M_L^*$. Here M_L^* is chosen so that the integrated variance of the standard Monte Carlo estimator is approximately $\epsilon^2/2$. This is similar to the criterion used to choose the number of replications in our procedure. The number of replications M_L^* is chosen based on an auxiliary experiment in which 10,000 replications were run for all the relevant grids, so as to estimate the intergrated variance of predictions based on each of those grids. That is, we did not actually run a standard Monte Carlo experiment with computational cost $(1+2^L)^d M_L^*$, which can be prohibitively high. Instead, we simply calculated $(1+2^L)^d M_L^*$ using the stopping level L of the MLMC procedure and the number of replications M_L^* chosen according to the auxiliary experiment, which was used only for this purpose. The average computational cost of the standard Monte Carlo and MLMC experiments is reported in Section 5.2.

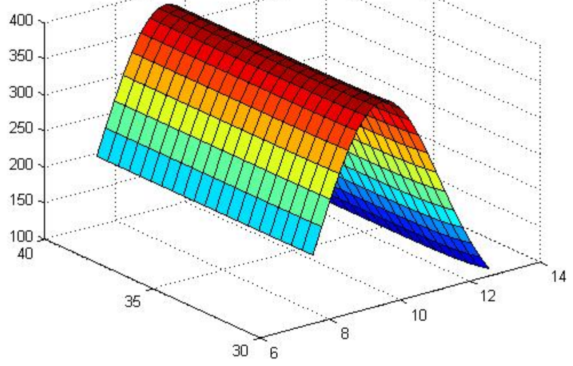
5.1 Examples

5.1.1 Inventory System

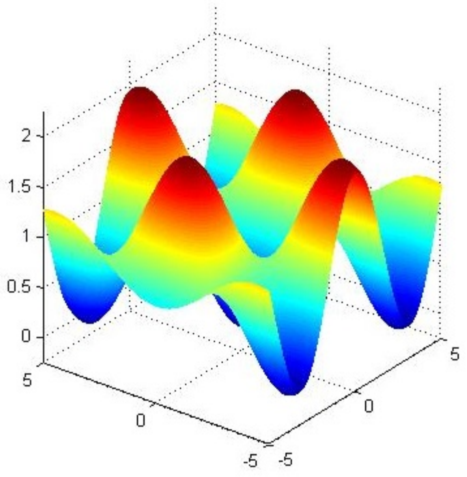
This example involves a model of expected profit in an inventory system given by Aydin and Porteus (2008). The input parameter vector contains the prices θ_1 and θ_2 that a firm sets for two products, and it varies over the domain $\Theta = [6, 14] \times [30, 40]$. The prices make the expected demand for product i proportional to $q_i = \exp(\alpha_i - \theta_i)/(1 + \exp(\alpha_1 - \theta_1) + \exp(\alpha_1 - \theta_1))$ in a logit model with coefficients $\alpha_1 = 10$ and $\alpha_2 = 25$. The stochastic demand for product i is $D_i = U_i q_i$ where U_1 and U_2 are independent random variables uniformly distributed between 100 and 500. The costs of the products are $c_1 = 6$ and $c_2 = 20$. The inventory chosen for product i is $I_i = (400(1 - c_i/\theta_i) + 100)q_i$. The profit in the inventory system is $Y(\theta) = \theta_1 \min\{D_1, I_1\} - c_1 I_1 + \theta_2 \min\{D_2, I_2\} - c_2 I_2$. Figure 3a shows the response surface μ in this example.

5.1.2 Asian Option

In this example we estimate the price of a geometric Asian call option. The price is the expectation of the discounted payoff of the option, which is $e^{-rt_n} \max\{0, \prod_{i=0}^{n} \sqrt[n+1]{S_{t_i}} - K\}$ where $t_1, \ldots, t_n$ are times at which an asset price is monitored, S_{t_i} is the price of the asset at t_i, r is an interest rate, and K is the strike price of the option. It is assumed that the asset price process S is a geometric Brownian motion with volatility σ and drift r. The input parameter vector is three-dimensional and includes the volatility σ, the strike price K, and the initial asset price S_0. Its domain $\Theta = [0.1, 0.3] \times [90, 110] \times [80, 120]$. The other parameters of



(a) Inventory system.



(b) Griewank function.

Figure 3: Response surfaces in two of our examples.

the model were fixed at $n = 4$, $t_i = i/4$, and $r = 0.01$. With these parameters, the response surface μ ranges from about 0.02 to 27 over Θ.

5.1.3 Griewank Function

The response surfaces in the Asian option and inventory system examples might be considered easy to approximate because of their mild curvature. To test the performance of MLMC in approximating a response surface whose second derivative changes sign frequently, we take the response surface μ to be the Griewank function. Figure 3b shows the Griewank function, which is given by $\mu(\theta) = 1+(\theta_1^2+\theta_2^2)/4000-\cos(\theta_1)\cos(\theta_2/\sqrt{2})$. We created a stochastic simulation example by making the simulation output Y equal to the Griewank function μ plus a mean-zero Gaussian random field with a nontrivial covariance structure. This Gaussian random field $Y - \mu = (1 + |\theta_1|)Z_1 + \exp(-\theta_2^2)Z_2$, where Z_1 and Z_2 are independent standard normal random variables.

5.2 Results

Figure 4 compares the computational cost of our MLMC procedure to the standard Monte Carlo procedure for different values of ϵ, the square root of MISE. The figure displays the log of the average computational cost of each experiment, repeated over 50-500 macro-replications. From the macro-replications, we constructed 95% normal confidence intervals for expected computational cost. These confidence intervals (which are not displayed, so as to avoid clutter in the figure) would lead to error bars of height 0.3 or less in Figure 4. The ratio of the average computational cost of the standard Monte Carlo procedure to the average computational cost of our MLMC procedure is reported in Table 1. Depending on the example and the function approximation method, MLMC is 8-200 times faster than standard Monte Carlo for the largest values of ϵ and 1,500-300,000 times faster for the smallest values of ϵ.

We can compare the slopes of lines in Figure 4 to the asymptotic computational complexity described in Theorem 1. For the inventory system example, the slopes of the lines on the log-log plot are about -4 for standard Monte Carlo and a bit steeper than -2 for MLMC, which is consistent with Theorem 1 with $\phi = d/2 = 1$. The slope for MLMC is similar in the Griewank function example, which also has $d = 2$. However, the plots for standard Monte Carlo in the Griewank function example and for both methods in the Asian option example are not linear: the slope changes over the range of ϵ that we investigated. One possible cause of this phenomenon is variability in the stopping level L of the MLMC procedure, which also affects

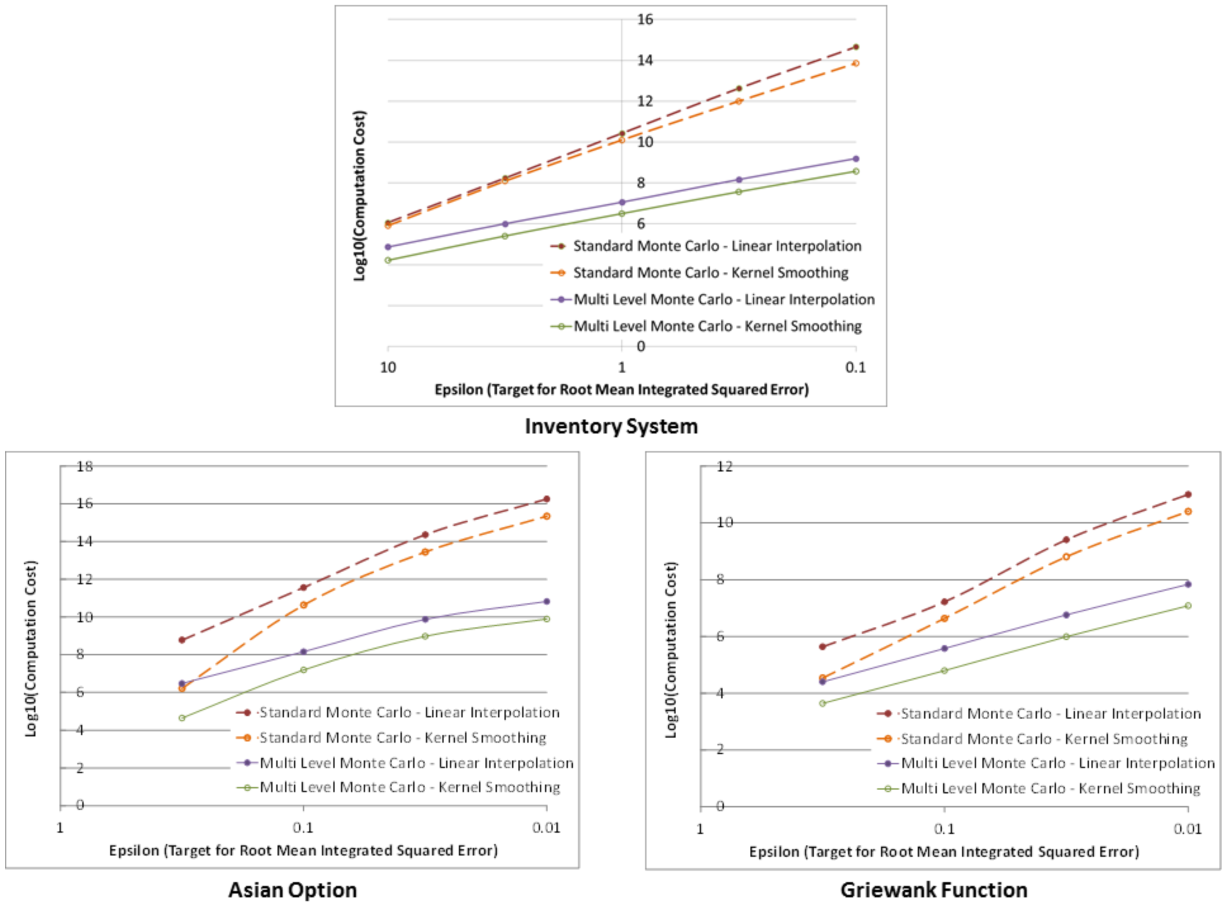


Figure 4: Computational cost of simulation experiments.

	Inventory System		Asian Option		Griewank Function	
$\log_{10} \epsilon$	Kernel Smoothing	Linear Interpolation	Kernel Smoothing	Linear Interpolation	Kernel Smoothing	Linear Interpolation
1.0	4.9×10^1	1.5×10^1				
0.5	4.8×10^2	1.7×10^2				
0.0	4.0×10^3	2.4×10^3				
-0.5	2.7×10^4	2.9×10^4	3.7×10^1	2.0×10^2	7.9×10^0	1.7×10^1
-1.0	1.9×10^5	2.9×10^5	2.8×10^3	2.5×10^3	7.0×10^1	4.4×10^1
-1.5			3.0×10^4	3.1×10^4	6.6×10^2	4.4×10^3
-2.0			2.9×10^5	2.7×10^5	2.1×10^3	1.5×10^4

Table 1: Reduction in computational cost achieved by using multi-level Monte Carlo.

the computational cost of the standard Monte Carlo procedure through the choice of grid. Another possible explanation for the difference between the theoretical rates in Theorem 1 and slopes seen in Figure 4 is that the theorem describes asymptotic growth rates of upper bounds on computational cost. Some values of ϵ in our computational experiments may be too small to see asymptotic behavior, and some function approximation methods may perform exceptionally well for some grids and some response surfaces. For example, the performance of kernel smoothing is surprisingly good for the largest value of ϵ for standard Monte Carlo in the Griewank function example and for both standard Monte Carlo and MLMC in the Asian option example.

6 Future Research

We believe that MLMC is a very valuable tool for stochastic simulation metamodeling and worthy of further development for this application. Here we sketch a few directions for future research in this area.

We provided a simple MLMC procedure for stochastic simulation metamodeling, building on the MLMC procedure of Giles (2008) for SDE simulation. Recent advances in the design of MLMC procedures for SDE simulation and other applications might be applied to build better MLMC procedures for stochastic simulation metamodeling. Haji-Ali et al. (2014) explored optimization of the sequence of experiment designs, the stopping criterion for number of levels, and the number of replications. Hoel et al. (2012) developed an adaptive experiment design which, in our setting, might correspond to using a different experiment design for each realization of the simulation output, $Y(\cdot, \omega)$. Mes et al. (2011) also developed an adaptive experiment design featuring multiple levels of coarseness; although their problem was discrete optimization via simulation, their work may be relevant to MLMC in stochastic simulation metamodeling.

We showed how MLMC can improve the rate of convergence in stochastic simulation metamodeling when error is measured based on the $\|\cdot\|_2$ norm and under conditions including Lipschitz continuity of the simulation output. Similar results could be derived using different norms and different conditions on the simulation output or also response surface. We previously derived similar results under the condition of Hölder continuity (Rosenbaum and Staum, 2013), but for operations research applications, it would be much more valuable to allow for some discontinuity in the simulation output. We are presently working on this topic. The $\|\cdot\|_\infty$ norm and higher-order smoothness are discussed in the Appendices.

McLeish (2011) provided a general method for eliminating the bias from a Monte Carlo estimator whose bias can be made to converge to zero by increasing computational effort. In the language of MLMC, his method involves randomizing the number of replications M_ℓ for each level ℓ in such a way that the number of levels L is random, $M_1, \ldots, M_L$ are non-increasing, $M_\ell = 0$ for $\ell > L$, and $\mathbb{E}[M_\ell] > 0$ for all ℓ. This method can improve the computational complexity of simulating SDEs with time-discretization bias (Rhee and Glynn, 2012, 2013); recently it was also used to approximate expectations taken under the equilibrium distribution of a positive Harris recurrent Markov chain (Glynn and Rhee, 2014). The same idea should be applicable to stochastic simulation metamodeling.

Finally, consider simulation metamodeling when a simulation run at a design point θ provides a biased estimator of $\mu(\theta)$, such as simulation of SDEs with time-discretization bias or steady-state simulation with initial-condition bias. Suppose that the bias-elimination technique described in the previous paragraph is applied to the simulation runs. Then the simulation metamodeling problem is returned to the framework considered in this article, in which the simulation runs provide unbiased estimators. Alternatively, for some such simulation metamodeling problems with biased simulation runs, one might instead try to control two kinds of bias simultaneously: the bias from function approximation, reduced by using many design points, and the bias inherent in simulation runs, reduced by increasing the computational effort per simulation run. Future research may address these questions of experiment design and the effectiveness of such simulation metamodeling procedures.

A Applicability of MLMC

With which metamodeling schemes should MLMC be applied? MLMC improves the computational complexity of simulation metamodeling if our assumptions are satisfied. Among them, Assumption 4 is an assumption about the metamodeling scheme: the function approximation method and sequence of experiment designs. To satisfy Assumption 4, it is necessary that the function approximation method yield a prediction in the form of Equation (1), and that the sequence of experiment designs have a mesh size, defined in Equation (12), that converges to zero at an appropriate rate. In Appendix A.1, we describe some schemes that satisfy Assumption 4. Appendix A.2 contains further remarks on function approximation methods that might be used with MLMC.

First, we make a general remark about experiment design. For simplicity, we have used grids in our examples and computational experiments. However, a grid does not fill high-dimensional space efficiently. In simulation metamodeling, it seems better to use other space-filling experiment designs, such as low-discrepancy point sets (Asmussen and Glynn, 2007) or sparse grids (Barthelmann et al., 2000).

A.1 Some Specific Metamodeling Schemes that Satisfy Assumption 4

Heinrich (2000) mentioned piecewise multivariate linear interpolation as a function approximation method that is suitable for MLMC. Consider a d-dimensional grid for which the vector h specifies the width of a grid cell in each of the coordinates; the mesh size is $\|h\|$. Let $\mathcal{N}(\theta)$ be the set of design points neighboring the prediction point θ. If θ is in the interior of a grid cell, then $\mathcal{N}(\theta)$ is the set of the 2^d vertices of this grid cell. Piecewise linear interpolation on a grid can be done with tensor-product piecewise linear B-splines (Hastie et al., 2009), yielding a prediction in the form of Equation (1) with weight

$$w_i(\theta) = \prod_{j=1}^{d} \frac{|\theta_j - \theta_{ij}|}{h_j} \text{ if } \theta_i \in \mathcal{N}(\theta), \quad w_i(\theta) = 0 \text{ if } \theta_i \notin \mathcal{N}(\theta).$$

Let h^ℓ specify the cell widths in the ℓth experiment design. If $\|h^\ell\|$ is $\mathcal{O}(s^{-\alpha\ell})$, then Assumption 4 holds with $r_\ell = \|h^\ell\|$ and $p_\ell = 0$.

Next we consider function approximation methods that work with scattered data, i.e., the experiment design need not have a special structure such as a grid. Assume that Δ^ℓ, the mesh size of the ℓth experiment design, is $\mathcal{O}(s^{-\alpha\ell})$.

One simple method is k-nearest-neighbor approximation (Hastie et al., 2009), which puts weight $1/k$ on each of the k design points nearest the prediction point and weight 0 on all other design points in Equation (1). Assumption 4 holds with $r_\ell = 2k\Delta^\ell$ and $p_\ell = 0$.

As an alternative, let the weight $w_i(\theta)$ on design point θ_i depend on the distance $\|\theta - \theta_i\|$ to the prediction point θ. A non-increasing function ψ is used to set the influence $\psi(t)$ given to a design point at distance t from the prediction point. The weight for design point θ_i in Equation (1) is

$$w_i(\theta) = \frac{\psi(\|\theta - \theta_i\|)}{\sum_{i'=1}^{N} \psi(\|\theta - \theta_{i'}\|)}.$$

If ψ is strictly positive, then the prediction is a weighted average involving all of the design points; if $\psi(t) = 0$ for t exceeding a threshold $\bar{t}$, then only nearby design points enter into the weighted average. Shepard's method (Buhmann, 2003) uses such weights with a function ψ such that $\psi(0) = \infty$ and $\psi(t)$ is finite for $t > 0$. The purpose of this is to ensure $\hat{\mu}(\theta_i) = \bar{Y}(\theta_i)$, i.e., Shepard's method is an interpolation method. The same formula for weights is used in kernel smoothing (Hastie et al., 2009). If $\psi(0)$ is finite, this causes smoothing, meaning that $\hat{\mu}(\theta_i)$ can depend on simulation output at design points other than θ_i. Let ψ^ℓ be the weighting function used with the ℓth experiment design. We consider two different cases in which Assumption 4 holds for Shepard's method and kernel smoothing.

1. Suppose that there is a threshold $\bar{t}^\ell > \Delta^\ell$ such that $\psi^\ell(t) = 0$ for all $t > \bar{t}^\ell$, and $\bar{t}^\ell$ is $\mathcal{O}(s^{-\alpha\ell})$. Then Assumption 4 holds with $r_\ell = \bar{t}^\ell$ and $p_\ell = 0$.

2. Finally, consider a case in which positive weight is given to distant design points. Suppose that there is a function ψ such that $\lim_{t\to\infty}\psi(t)=0$ and a bandwidth h^ℓ such that for all t, $\psi^\ell(t)=\psi(t/h^\ell)$. Suppose the mesh size $\Delta^\ell \le c_\Delta s^{-\ell}$ and the number N_ℓ of points in the ℓth experiment design satisfies $N_\ell \le c_N s^{d\ell}$ for some constants c_Δ and c_N; this can be achieved by a sequence of grids. Suppose the bandwidth $h^\ell = c_h s^{-\beta\ell}$ where $c_h > 0$ and $\beta \in (0,1)$ are constants. Set $r_\ell = c_\Delta s^{-\alpha\ell}$ for $\alpha \in (0,\beta)$. This makes $r_\ell \ge \Delta^\ell$. In the ℓth experiment design, for any prediction point θ, there is at least one design point within distance Δ_ℓ, i.e., in $I^\ell(\theta;\Delta^\ell)$. This design point θ_i is in $I^\ell(\theta;r_\ell)$ and it receives un-normalized weight $\psi^\ell(\|\theta-\theta_i\|) \ge \psi(\Delta^\ell/h^\ell) \ge \psi(c_\Delta/c_h)$. Any design point $\theta_{i'}$ that is not in $I^\ell(\theta;r_\ell)$ receives un-normalized weight $\psi^\ell(\|\theta-\theta_{i'}\|) \le \psi(r^\ell/h^\ell) \le \psi((c_\Delta/c_h)s^{(\beta-\alpha)\ell})$. The total amount of normalized weight given to distant design points is

$$\begin{aligned}\sum_{i'\notin I^\ell(\theta;r_\ell)} w^\ell_{i'}(\theta) &= \frac{\sum_{i'\notin I^\ell(\theta;r_\ell)}\psi^\ell(\|\theta-\theta_{i'}\|)}{\sum_{i'\notin I^\ell(\theta;r_\ell)}\psi^\ell(\|\theta-\theta_{i'}\|)+\sum_{i\in I^\ell(\theta;r_\ell)}\psi^\ell(\|\theta-\theta_i\|)}\\ &\le \frac{\sum_{i'\notin I^\ell(\theta;r_\ell)}\psi^\ell(\|\theta-\theta_{i'}\|)}{\sum_{i\in I^\ell(\theta;r_\ell)}\psi^\ell(\|\theta-\theta_i\|)} \le \frac{N_\ell\psi((c_\Delta/c_h)s^{(\beta-\alpha)\ell})}{\psi(c_\Delta/c_h)} = p_\ell,\end{aligned}$$

where this serves to define p_ℓ. Suppose that $\psi(t)$ is $\mathcal{O}(t^{-(\alpha+d)/(\beta-\alpha)})$. Then $\psi((c_\Delta/c_h)s^{(\beta-\alpha)\ell})$ is $\mathcal{O}(s^{-(\alpha+d)\ell})$. Together with N_ℓ being $\mathcal{O}(s^{d\ell})$, this implies that p_ℓ is $\mathcal{O}(s^{-\alpha\ell})$. Thus, Assumption 4 holds.

A.2 Function Approximation Methods to Use with MLMC

Before addressing specific methods, we discuss how an approach slightly different from that in this article could be helpful in verifying a method's suitability for MLMC. Theorem 1 is founded on Equation (8), where we decomposed MISE into integrated variance plus integrated squared bias. This is similar to the standard decomposition of MSE into variance plus bias, which Giles (2008) used. A slightly different perspective was taken by Heinrich, who focused on function approximation methods that are used when the function being approximated can be observed without noise. Imagine that we could actually observe $\mu(\theta_i)$ at each of the design points. Applying the function approximation method described in Equation (1) to these observations, we would get $\tilde\mu(\theta)=\sum_{i=1}^N w_i(\theta)\mu(\theta_i)$ as the zero-noise prediction of the value of the response surface at the prediction point θ. Then MISE can be bounded as

$$\mathbb{E}[\|\hat\mu(\theta)-\mu(\theta)\|_2^2] \le \|\tilde\mu(\theta)-\mu(\theta)\|_2^2 + \mathbb{E}[\|\hat\mu(\theta)-\tilde\mu(\theta)\|_2^2], \tag{18}$$

the sum of the deterministic integrated squared error of approximating the response surface in this scheme, plus the MISE of the noisy metamodel $\hat\mu(\theta)$ considered as an estimator of the zero-noise metamodel $\tilde\mu(\theta)$. In the case of linear prediction as in Equation (1), equality holds and Equation (18) coincides with Equation (8) because $\mathbb{E}[\hat\mu(\theta)]=\tilde\mu(\theta)$. Nonetheless, adopting the perspective of Equation (18) may be advantageous in checking whether a metamodeling scheme satisfies the conditions for MLMC to improve computational complexity. Instead of trying to analyze the bias of the metamodel given noisy data, one might consult the literature for error bounds on the deterministic function approximation method. The way to deal with the last term in Equation (18) is discussed in Appendix B.4.

Consider function approximation methods that fit the form of Equation (1), rewritten as

$$\hat\mu(\theta)=\sum_{i=1}^N w_i(\theta)\bar Y(\theta_i)=\sum_{i=1}^N\sum_{j=1}^p Q_{ij}\varphi_j(\theta)\bar Y(\theta_i)=\sum_{j=1}^p\hat\beta_j\varphi_j(\theta), \tag{19}$$

where $\varphi_1,\dots,\varphi_p$ are basis functions and Q is a matrix such that the weight on the ith design point $w_i(\theta)=\sum_{j=1}^p Q_{ij}\varphi_j(\theta)$ and the coefficient on the jth basis function $\hat\beta_j=\sum_{i=1}^N Q_{ij}\bar Y(\theta_i)$. This form applies to regression, kriging, splines (Hastie et al., 2009), and radial basis functions (Buhmann, 2003), as well as their regularized or smoothing versions. These methods have much in common. Indeed, kriging can be seen as a form of regression, and in some circumstances, the same function approximation can be reached via splines, radial basis functions, or kriging (Myers, 1992).

The phrase "basis functions" indicates that $\{\varphi_n\}_{n\in\mathbb{N}}$ forms a basis for a space F of functions; the metamodel $\hat{\mu}$ must be in F. For MLMC to be applicable, there should be a sequence of functions f_n in F such that the error $f_n - \mu$ in approximating the response surface μ converges to zero in the desired sense. (In the setting of this article, it would be that the integrated squared error $\|f_n - \mu\|_2^2 = \int_\Theta (f_n(\theta) - \mu(\theta))^2 \, \mathrm{d}\theta$ goes to zero.) For example, quadratic regression is inappropriate as a method to use with MLMC, because we cannot assume that the response surface can be well approximated by a quadratic function of θ. Sobolev spaces seem to be a good choice for F. They are connected with the spline and radial basis function methods for function approximation (Fasshauer, 2012). They also accommodate the use of higher-order function approximation methods to attain better rates of convergence when the function being approximated is smoother.

A limitation of Equation (19) is that it has weights that do not depend on simulation output. In practice, the weights often involve bandwidths, smoothing parameters, or other parameters that are selected using the simulation output. Addressing that issue might complicate the analysis, but we suspect it that it need not be an obstacle to applying MLMC with these methods even when using a good parameter-selection technique.

Next, consider moving least squares regression (Levin, 1998; Salemi et al., 2012), also called "local regression" (Loader, 1999; Hastie et al., 2009). In this method, Equation (19) applies, except that the coefficients $\hat{\beta}_j(\theta)$ are allowed to depend on the prediction point θ. This allows moving least squares regression to approximate a larger space of functions than the corresponding form of ordinary regression. For example, continuously differentiable functions are locally linear, and Levin (1998) has a theorem providing the rate of convergence for moving least squares linear regression to approximate a continuously differentiable function, with error measured by the $\|\cdot\|_\infty$ norm. We conjecture that the same rate would obtain for moving least squares linear regression to approximate a Lipschitz-continuous function, with error measured by the $\|\cdot\|_2$ norm. For the reason, see Appendix B.3, where we sketch the connection between Lipschitz continuity and Heinrich's Sobolev space framework.

Finally, neural networks can approximate a large space of functions. The details depend on the architecture of the neural network, and the architecture may need to increase in size to allow approximation error to converge to zero. For example, neural networks can approximate any continuous function on a compact, convex subset of $\mathbb{R}^d$, but the required number of nodes in the network may go to infinity as the approximation error goes to zero. See Kainen et al. (2013) for an overview of results on function approximation by neural networks and associated rates of convergence. Training a neural network can be much more difficult than computing good coefficients $\hat{\beta}_j$ in Equation (19). Care should be taken to ensure that the function approximation is actually improved by expanding the point set. One might even consider, in Equation (6), using the neural network for $\hat{\mu}_{\ell-1}$ as a starting point for building the neural network for $\hat{\mu}_\ell$.

B Relation to Previous Research

MLMC for parametric integration was analyzed originally by Heinrich and Sindambiwe (1999) and Heinrich (2000, 2001), and recently by Daun and Heinrich (2013, 2014), using different assumptions on the simulation output $Y : \Theta \times \Omega \to \mathbb{R}$. Among the assumptions of Heinrich and Sindambiwe (1999) and Daun and Heinrich (2013, 2014) is that Y is continuously differentiable on $\Theta \times \Omega$. Similarly, Y must have continuous partial derivatives with respect to the parameters in θ for the analysis of Daun and Heinrich (2014) to yield useful results. As discussed in Section 1, it is common in stochastic simulation metamodeling in operations research for Y to have points of non-differentiability with respect to the parameter θ. This is allowed under the smoothness assumptions of Heinrich (2000, 2001). However, his assumptions about Θ and Ω, and the integrability properties of Y on $\Theta \times \Omega$, create obstacles to the direct application of his results for our purposes. We avoid these difficulties by assuming instead that Θ is compact, as do Heinrich and Sindambiwe (1999) and Daun and Heinrich (2013, 2014). To summarize the position of our work relative to the previous literature:

- Our work has a mixture of some assumptions of Heinrich (2000, 2001), allowing for Y not to be differentiable everywhere with respect to the parameters in θ, and the assumption of Heinrich and Sindambiwe (1999) and Daun and Heinrich (2013, 2014) that the domain Θ is compact.

- Our work is more general in that it drops the assumption that Ω is finite-dimensional, which was made in all of the articles mentioned here.
- Our work is less general in neglecting the benefits of higher-order smoothness, e.g., existence of second derivatives. Whereas we consider only Lipschitz continuity, the previous literature also shows how higher-order smoothness can improve the computational complexity of MLMC.

In this appendix, we discuss the applicability of the assumptions of Heinrich (2001) in detail, using our notation rather than his, and explain how his results relate to ours. Lastly, we present a fuller exposition of Heinrich's approach to error analysis based on the theory of random fields, and state how it can be used in our setting.

B.1 The Probability Space

Heinrich assumed that the integration in parametric integration is integration over a subset $\Omega \subset \mathbb{R}^{d_2}$ with respect to a uniform measure, and approximated integrals of the form $\int_\Omega Y(\theta, \omega)\, \mathrm{d}\omega$ by Monte Carlo sampling of random variates uniformly distributed on Ω. However, as Heinrich (2001) remarked, it is not necessary to sample uniformly on Ω; importance sampling could be used instead. We believe that it remains within the spirit of Heinrich's work to observe that, likewise, it is not necessary for the integral to be with respect to a uniform measure in the first place, nor is it necessary for the domain of integration to be a subset of $\mathbb{R}^{d_2}$ for some d_2. Indeed, d_2 does not appear in Heinrich's computational complexity theorems. Consequently, we have made no assumption about the probability space $(\Omega, \mathbb{P})$ in which we have a stochastic model and approximate expectations by Monte Carlo. We merely assume that this can be done in the usual way, with zero bias and finite variance. In particular, we wish to make it clear that our analysis of MLMC applies to simulation models in which the number of random variates generated per replication is unbounded (instead of being bounded by d_2).

B.2 The Parameter Domain

Heinrich assumed that the parameter domain $\Theta \in \mathbb{R}^d$, over which the response surface $\mu = \mathbb{E}[Y]$ is to be approximated, is a bounded open set. Making Θ an open set allows for Y and μ to be unbounded, even if they are continuous. As an example in operations research, we might think of queueing system behavior on an open set of parameter values for which the system is stable, e.g., in an M/M/1 queue with arrival rate a and service rate b, taking $\Theta = \{(a, b) : 1 < a < b < 2\}$. With an open domain like this, simulation metamodeling necessarily involves extrapolation. We suppose that in simulation metamodeling in operations research, it is not common to aim for high precision in the predictions from a metamodel outside the range of parameter values explored in the experiment design, and where the response surface may be unbounded. Instead, our Assumption 1 is that Θ is a compact set in $\mathbb{R}^d$. Similarly, Heinrich and Sindambiwe (1999) and Daun and Heinrich (2014) assumed that $\Theta = [0, 1]^d$.

Equation (10) shows that the response surface μ is Lipschitz continuous. Therefore it is bounded on Θ. Likewise, almost surely, $Y(\cdot, \omega)$ is Lipschitz continuous and therefore bounded on Θ. Similarly, if $\hat{\mu}$ is a weighted sum of simulation output, as in our Assumption 4, then $\hat{\mu}$ too is almost surely Lipschitz continuous and therefore bounded on Θ.

Heinrich further assumed that Θ has Lipschitz boundary. This is a standard regularity property to assume for bounded open subsets of $\mathbb{R}^d$ in the theory of Sobolev spaces (Adams, 1975, pp. 66-67), which Heinrich used. The assumption of Lipschitz boundary is not necessary in our simpler setting.

B.3 Smoothness and Integrability

Heinrich assumed that for some degree of smoothness $r \in \mathbb{N}$ and exponent of integrability $q \in [1, \infty]$ satisfying the Sobolev embedding condition $r/d > 1/q$, the function Y is in the space $W_q^{r,0}(\Theta \times \Omega)$, a slightly non-standard Sobolev space to be defined below. See Adams (1975) for the mathematics of Sobolev spaces.

The exponent q governs the norm which is used in quantifying error. For $1 \leq q < \infty$, the norm of the error is

$$\|\hat{\mu} - \mu\|_q = \left(\int_\Theta |\hat{\mu}(\theta) - \mu(\theta)|^q \, \mathrm{d}\theta \right)^{1/q}.$$

For $q = \infty$, $\|\hat{\mu} - \mu\|_\infty$ is the greatest lower bound of the set of constants K such that $|\hat{\mu}(\theta) - \mu(\theta)| \leq K$ for almost every $\theta \in \Theta$. Supposing that $\hat{\mu}$ and μ are bounded on Θ (see Section B.2), $\|\hat{\mu} - \mu\|_\infty$ is simply the maximum value attained by $|\hat{\mu}(\theta) - \mu(\theta)|$ for any $\theta \in \Theta$. Thus, $q = \infty$ is of interest as the worst error of the metamodel. Also of interest is $q = 2$, because $\|\hat{\mu} - \mu\|_2^2$ is the integrated squared error which goes into MISE, a standard way to measure quality in stochastic simulation metamodeling. Separate computational complexity theorems cover the cases $1 \leq q < \infty$ and $q = \infty$, and the computational complexity is worse when one wishes to bound $\|\hat{\mu} - \mu\|_\infty$ (Heinrich, 2001). We have focused on the case $q = 2$.

The order of smoothness r relates to how many times the function Y can be weakly differentiated with respect to parameters in the vector $\theta \in \Theta$. Weak differentiation is the concept of differentiation used in Sobolev spaces. For example, the function f given by $f(\theta) = \max\{0, \theta\}$ is not differentiable on $[-1, 1]$, but it is weakly differentiable on $[-1, 1]$. The functions given by $1\{\theta > 0\}$ and $1\{\theta \geq 0\}$ are both weak derivatives of f on $[-1, 1]$. The weak derivative of a function can be thought of as being unique as an equivalence class of all weak derivatives, because the weak derivatives of a single function must be equal almost everywhere and they must equal the ordinary derivative wherever it exists. The weak derivative of f is discontinuous at 0, so f is not twice weakly differentiable. With this example in mind, we take $r = 1$ to be most appropriate for stochastic simulation metamodeling in operations research, although higher-order smoothness can occur.

To define the Sobolev space $W_q^{r,0}(\Theta \times \Omega)$, we must consider weak partial derivatives. Let α represent a multi-index, a vector of length $d + d_2$ in Heinrich's setting, such that α_i is the number of times to differentiate (weakly) with respect to the ith coordinate. Let $|\alpha|_\theta = \sum_{i=1}^d \alpha_i$ be the total number of times to differentiate with respect to the parameters in θ and $|\alpha|_\omega = \sum_{i=d+1}^{d_2} \alpha_i$ be the total number of times to differentiate with respect to the coordinates of ω; also let $|\alpha| = |\alpha|_\theta + |\alpha|_\omega$. Let $D^\alpha Y$ represent Y's weak partial derivative of order α; if the partial derivative of order α in the ordinary sense were to exist, it would be written

$$\frac{\partial^{|\alpha|} Y}{\partial \theta_1^{\alpha_1} \cdots \partial \theta_d^{\alpha_d} \partial \omega_1^{\alpha_{d+1}} \cdots \partial \omega_{d_2}^{\alpha_{d+d_2}}}.$$

The assumption that the function Y is in the space $W_q^{r,0}(\Theta \times \Omega)$ means that Y satisfies

$$\int_{\Theta \times \Omega} |D^\alpha Y|^q \, \mathrm{d}\omega \, \mathrm{d}\theta < \infty$$

for all α such that $0 \leq |\alpha|_\theta \leq r$ and $|\alpha|_\omega = 0$. To paraphrase Heinrich (2001), this means roughly that with respect to θ, $Y(\theta, \omega)$ behaves like a function in the Sobolev space $W_q^r(\Theta)$, and with respect to ω, it behaves like a function in the Sobolev space $W_q^0(\Omega) = L_q(\Omega)$. This gives us a second reason to focus on the case $q = 2$: we have no desire to impose an assumption that the simulation output has moments of order higher than 2.

However, there is an obstacle to applying Heinrich's results to our desired case of $r = 1$ and $q = 2$: the Sobolev embedding condition $r/d > 1/q$ fails to hold for dimension $d > 1$. This turns out to be irrelevant in our setting. The Sobolev embedding condition is used to ensure that functions in a Sobolev space W_q^r on a d-dimensional domain are continuous. Heinrich (2008) addresses the case in which the Sobolev embedding condition does not hold and the elements of W_q^r are equivalence classes of functions that may be discontinuous. In our setting (as shown in Appendix B.2), it follows from our Assumptions 1 and 3 that $Y(\cdot, \omega)$ is almost surely a bounded, Lipschitz-continuous function on Θ. It belongs to $W_\infty^1(\Theta)$ because it and its weak first derivatives are bounded, so have finite $\|\cdot\|_\infty$ norm. In a sense, we have $q = 2$ in the error analysis of Theorem 1 of Heinrich (2001) because we are measuring error with the $\|\cdot\|_2$ norm, yet at the same time we have $q = \infty$ for purposes of verifying the Sobolev embedding condition.

B.4 Error Analysis

In Appendix A.2, we mentioned that Heinrich has a different approach to error analysis. Let $\hat{\mu}_L^{\mathrm{MLMC}}$ represent the metamodel built by MLMC, defined in Equation (5). Similar to Equation (18), Heinrich's approach to error analysis would bound MISE as

$$\mathbb{E}\|\hat{\mu}_L^{\mathrm{MLMC}} - \mu\|_2^2 \leq \|\tilde{\mu} - \mu\|_2^2 + \mathbb{E}\left\|\hat{\mu}_L^{\mathrm{MLMC}} - \tilde{\mu}\right\|_2^2$$

where $\tilde{\mu}$ is the function approximation produced in the absence of noise. The approach proposed by Heinrich (2001) and followed up by Daun and Heinrich (2013, 2014) is more general than what is stated here. What Heinrich refers to as an "X-valued random variable" where X is a Banach space, we call a "random field." We focus on L_2 as the Banach space because of our interest in the $\|\cdot\|_2$ norm. A general approach to bounding $\mathbb{E}\|\hat{\mu}_L^{\mathrm{MLMC}} - \tilde{\mu}\|_2^2$ appears in Section 4 of Heinrich (2001), and this is carried out for a specific metamodeling scheme in Daun and Heinrich (2013): see particularly Lemma 1, Proposition 1, and Theorem 1.

The main idea is to consider the function $\hat{Y}(\cdot,\omega)$ defined in Equation (13), produced by applying the function approximation method to each realization $Y(\cdot,\omega)$ of the random field. This is the crux of the parametric integration viewpoint on stochastic simulation metamodeling. It is also analogous to the idea of Giles (2008) that a scheme for approximate solution of an SDE can be applied to "a given Brownian path" on a coarse grid and a fine grid. Even when $\hat{Y}(\cdot,\omega)$ need not be used to compute a metamodel in stochastic simulation, still we believe that it remains a fruitful concept for the design and analysis of stochastic simulation metamodeling procedures. For example, we used it in our procedure's variance estimators in Equation (15).

Acknowledgments

This article is based upon work supported by the National Science Foundation under Grant No. CMMI-0900354. Portions of this article were published in Rosenbaum and Staum (2013). The authors are very grateful to Fred Hickernell and Barry Nelson for comments on an earlier version of this article, to Stefan Heinrich for discussions, and to the associate editor and referees for comments and corrections that led to improvements in the article.

References

Adams, Robert A. 1975. *Sobolev Spaces*. Academic Press.

Asmussen, Søren, Peter W. Glynn. 2007. *Stochastic Simulation: Algorithms and Analysis*. Springer-Verlag.

Aydin, Goker, Evan L. Porteus. 2008. Joint inventory and pricing decisions for an assortment. *Operations Research* **56**(5) 1247–1255.

Barthelmann, Volker, Erich Novak, Klaus Ritter. 2000. High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics* **12** 273–288.

Barton, Russell R. 1998. Simulation metamodels. D. J. Medeiros, E. F. Watson, J. S. Carson, M. S. Manivannan, eds., *Proceedings of the 1998 Winter Simulation Conference*. IEEE Press, 167–176.

Billingsley, Patrick. 1995. *Probability and Measure*. 3rd ed. John Wiley & Sons.

Buhmann, Martin D. 2003. *Radial Basis Functions*. Cambridge University Press.

Chen, Xi, Bruce E. Ankenman, Barry L. Nelson. 2012. The effects of common random numbers on stochastic kriging metamodels. *ACM Transactions on Modeling and Computer Simulation* **22**(2) 7.

Cliffe, K., M. Giles, R. Scheichl, A. Teckentrup. 2011. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science* **14**(1) 3–15.

Daun, Thomas, Stefan Heinrich. 2013. Complexity of Banach space valued and parametric integration. Josef Dick, Frances Y. Kuo, Gareth W. Peters, Ian H. Sloan, eds., *Monte Carlo and Quasi-Monte Carlo Methods 2012*, *Springer Proceedings in Mathematics and Statistics*, vol. 65. Springer-Verlag, 297–316.

Daun, Thomas, Stefan Heinrich. 2014. Complexity of parametric integration in various smoothness classes. *Journal of Complexity* **30**(6) 750–766.

Emsermann, M., B. Simon. 2002. Improving simulation efficiency with quasi control variates. *Stochastic Models* **18**(3) 425–448.

Fasshauer, Gregory E. 2012. Green's functions: taking another look at kernel approximation, radial basis functions, and splines. Marian Neamtu, Larry Schumaker, eds., *Approximation Theory XIII: San Antonio 2010*. Springer-Verlag, 37–63.

Giles, Michael B. 2008. Multi-level Monte Carlo path simulation. *Operations Research* **56**(3) 607–617.

Giles, Michael B. 2013. Multilevel Monte Carlo methods. Josef Dick, Frances Y. Kuo, Gareth W. Peters, Ian H. Sloan, eds., *Monte Carlo and Quasi-Monte Carlo Methods 2012*, *Springer Proceedings in Mathematics and Statistics*, vol. 65. Springer-Verlag, 83–103.

Giles, Michael B. 2015. Multi-level Monte Carlo methods. *Acta Numerica* **24** 259–328.

Glasserman, Paul. 2004. *Monte Carlo Methods in Financial Engineering*. Springer-Verlag.

Glynn, Peter W., Chang-han Rhee. 2014. Exact estimation for Markov chain equilibrium expectations. *Journal of Applied Probability* **51A** 377–389.

Haji-Ali, Abdul-Lateef, Fabio Nobile, Erik von Schwerin, Raúl Tempone. 2014. Optimization of mesh hierarchies in multilevel Monte Carlo samplers.

Hastie, T., R. Tibshirani, J. H. Friedman. 2009. *The Elements of Statistical Learning*. 2nd ed. Springer-Verlag.

Heinrich, Stefan. 2000. The multilevel method of dependent tests. N. Balakrishnan, V. B. Melas, S. Ermakov, eds., *Advances in Stochastic Simulation Methods*. Birkhäuser, 47–62.

Heinrich, Stefan. 2001. Multilevel Monte Carlo methods. Svetozar Margenov, Jerzy Waśniewski, Plamen Yalamov, eds., *Large Scale Scientific Computing*, *Lecture Notes in Computer Science*, vol. 2179. Springer-Verlag, 58–67.

Heinrich, Stefan. 2008. Randomized approximation of Sobolev embeddings. Alexander Keller, Stefan Heinrich, Harald Niederreiter, eds., *Monte Carlo and Quasi-Monte Carlo Methods 2006*. Springer-Verlag, 445–459.

Heinrich, Stefan, Eugène Sindambiwe. 1999. Monte Carlo complexity of parametric integration. *Journal of Complexity* **15**(3) 317 – 341.

Hoel, Håkon, Erik von Schwerin, Anders Szepessy, Raúl Tempone. 2012. Adaptive multilevel Monte Carlo simulation. Björn Engquist, Olof Runborg, Yen-Hsi R. Tsai, eds., *Numerical Analysis of Multiscale Computations*. Springer-Verlag, 217–234.

Kainen, Paul C., Věra Kůrková, Marcello Sanguineti. 2013. Approximating multivariable functions by feedforward neural nets. Monica Bianchini, Marco Maggini, Lakhmi C. Jain, eds., *Handbook on Neural Information Processing*. Springer-Verlag, 143–181.

Kleijnen, Jack P. C. 2008. *Design and Analysis of Simulation Experiments*. Springer-Verlag.

Kleijnen, Jack P.C., Robert G. Sargent. 1997. A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research* **120** 14–29.

L'Ecuyer, Pierre, Richard Simard, E. Jack Chen, W. David Kelton. 2002. An object-oriented random-number package with many long streams and substreams. *Operations Research* **50**(6) 1073–1075.

Levin, David. 1998. The approximation power of moving least squares. *Mathematics of Computation* **67**(224) 1517–1531.

Loader, Clive. 1999. *Local Regression and Likelihood*. Springer-Verlag.

McLeish, Don. 2011. A general method for debiasing a Monte Carlo estimator. *Monte Carlo Methods and Applications* **17**(4) 301–315.

Mes, Martijn R. K., Warren B. Powell, Peter I. Frazier. 2011. Hierarchical knowledge gradient for sequential sampling. *Journal of Machine Learning Research* **12** 2931–2974.

Myers, Donald E. 1992. Kriging, cokriging, radial basis functions and the role of positive definiteness. *Computers & Mathematics with Applications* **24** 139–148.

Pasupathy, Raghu, Bruce W. Schmeiser, Michael R. Taaffe, Jin Wang. 2012. Control-variate estimation using estimated control means. *IIE Transactions* **44**(5) 381–385.

Rhee, Chang-han, Peter W. Glynn. 2012. A new approach to unbiased estimation for SDE's. C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, A. M. Uhrmacher, eds., *Proceedings of the 2012 Winter Simulation Conference*. IEEE Press, 201–207.

Rhee, Chang-han, Peter W. Glynn. 2013. Unbiased estimation with square root convergence for SDE models.

Rosenbaum, Imry, Jeremy Staum. 2013. Multi level Monte Carlo metamodeling. R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, M. E. Kuhl, eds., *Proceedings of the 2013 Winter Simulation Conference*. IEEE Press, 509–520.

Salemi, Peter, Barry L. Nelson, Jeremy Staum. 2012. Moving least squares regression for high dimensional simulation metamodeling. C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, A. M. Uhrmacher, eds., *Proceedings of the 2012 Winter Simulation Conference*. IEEE Press, 292–303.