Moving Least Squares Regression for High-Dimensional Stochastic Simulation Metamodeling

Peter Salemi
Barry L. Nelson
Jeremy Staum

Department of Industrial Engineering and Management Sciences
Northwestern University
2145 Sheridan Road
Evanston, IL, 60208-3119, U.S.A.

November 1, 2014

Abstract

Simulation metamodeling is building a statistical model based on simulation output as an approximation to the system performance measure being estimated by the simulation model. In high-dimensional metamodeling problems, larger numbers of design points are needed to build an accurate and precise metamodel. Metamodeling techniques that are functions of all of these design points experience difficulties because of numerical instabilities and high computation times. We introduce a procedure to implement a local smoothing method called Moving Least Squares (MLS) regression in high-dimensional stochastic simulation metamodeling problems. Although MLS regression is known to work well when there are a very large number of design points, current procedures are focused on two and three-dimensional cases. Furthermore, our procedure accounts for the fact that we can make replications and control the placement of design points in stochastic simulation. We provide a bound on the expected approximation error, show that the MLS predictor is consistent under certain conditions, and test the procedure with two examples that demonstrate better results than other existing simulation metamodeling techniques.

1 Introduction

Stochastic simulation is often used to model complex systems to support decision making. For example, Yang et al. [2011] use a simulation model of a semi-conductor wafer fabrication system to estimate the expected throughput for any given scenario. Simulation runs may be time-consuming to execute, especially when many scenarios need to be investigated; for example, Tongarlak et al. [2010] describe a simulation model of a fuel injector production line that takes 8 hours to run a single replication. This burden can make simulation models impossible for use in decision making, especially when decisions need to be made quickly. However, when there is enough time between model building and decision making, the simulation can be exercised on a set of chosen scenarios, the design points, and the results can be used to construct a statistical model. This statistical model is called the simulation metamodel. Simulation metamodeling allows the experimenter to

obtain more benefits from a simulation because the simulation can be run when time is plentiful, and quick predictions can be made when decision-making time is scarce or expensive. Applications in which we need such metamodeling capability include manufacturing planning [Yang et al., 2011] and financial security pricing [Liu and Staum, 2010]. For instance, in manufacturing capacity or production planning, decision makers may want to consider trade-offs among system design and control parameters as they affect, say, throughput or cycle time. Decision-maker time may be scarce and expensive, and individual simulation experiments on complex manufacturing systems may take too much time to evaluate trade-offs interactively. In this situation a metamodel can provide simulation-level fidelity "on demand." In the security pricing context decisions may need to be made in real time in the face of changing underlying risk factors, making it impossible to execute numerically intensive simulations. Characteristic of these two (and many other) similar situations is that there is a large space of possible scenarios that could arise, with no way to know in advance which ones will be relevant, and insufficient time to execute the simulations necessary to explore them directly when needed. Even if high-performance computing could theoretically allow the simulations to be executed in near-real time, expensive computing resources are typically heavily utilized and therefore their use must be scheduled. In other words, in order for a decision-maker to run the simulation and obtain a quick answer when needed, the high-performance computing environment would have to be idle; however, these computing resources are usually scheduled for high utilization.

The higher the dimension of the metamodeling problem, where dimension is the number of variable factors in a scenario, the more decision points are typically needed to obtain an accurate and precise metamodel. In this paper, we are interested in high-dimensional metamodeling problems with a very large number of design points, such as a 75-dimensional problem with 250,000 design points.

Metamodeling techniques that are functions of all of the design points, such as weighted least squares regression and Gaussian process models, experience difficulties when there is a large number of design points because of numerical instabilities and high computation times. For example, fitting a Gaussian process model requires solving an $n \times n$ linear system, which requires $O(n^3)$ operations, where n is the number of design points. Several methods have been developed to deal with these limitations such as using pseudo-inputs which maximize the likelihood that the actual data was drawn [Snelson and Ghahramani, 2006], covariance tapering [Kaufman et al., 2008], fixed-rank kriging [Cressie and Johannesson, 2008], and treed Gaussian processes [Gramacy and Lee, 2008].

Some metamodeling techniques are based on the premise that the response surface may have a sparse representation [Shan and Wang, 2010, Lafferty and Wasserman, 2008, Vijayakumar and Schaal, 2000]. These methods often require a search to determine the important terms in the representation, and can be slow and time-consuming since the number of possible terms to consider increases exponentially as the problem dimension increases. When the variance in the replications is large, determining which factors are important becomes difficult. Many of these methods also assume a relatively small number of important factors, and can be ill-suited for problems not satisfying this assumption.

Instead of using the entire set of design points for prediction, many methods localize the prediction by only using design points near the prediction point [Vijayakumar and Schaal, 2000, Lafferty and Wasserman, 2008, Breiman et al., 1984, Altman, 1992, Watson, 1964]. The main obstacle for localization methods is choosing the window for prediction. The window determines which design points influence each prediction. As the variance in the replications increases, it becomes difficult for these methods to identify good windows around the prediction point. Some of these methods (such as Vijayakumar and Schaal [2000] and Lafferty and Wasserman [2008]) also assume a small number of relevant variables.

Moving Least Squares (MLS) regression [Lancaster and Salkauskas, 1981, Levin, 1998] is a localization method which has been studied in the fields of partial differential equations and image processing. These applications feature low-dimensional problems with a large number of design points. Much of the research has focused on different formulations and applications of MLS regression, with relatively little focus on the construction of efficient procedures to implement MLS regression. The main obstacle for any MLS regression procedure is the choice of the bandwidths of the weight function, which determine the window. Lipman et al. [2006] calculates an error bound for the MLS predictor and then searches for the bandwidth that minimizes this error bound. The weight function is assumed to be isotropic, i.e., there is only one bandwidth parameter, so a line search is used to find the optimal bandwidth. The line search can be time consuming since the error bound must be calculated during each step of the search. Furthermore, noise in the observations satisfies a known bound. Adamson and Alexa [2006] proposed a method that uses the empirical covariance matrix of the k-nearest neighbors of the prediction point to assign a weight to each of the k-nearest neighbors. Since the empirical covariance matrix is positive-definite, the eigenvectors form the axes of an ellipsoid, the lengths of which depend on the eigenvalues. The weight given to each of the k-nearest neighbors is determined by where the point lies in the ellipsoid. As pointed out in Adamson and Alexa [2006], there is no way to ensure the ellipsoid covers all of the k-nearest neighbors, and no method is proposed to choose a good value for k.

Locally Weighted Least Squares regression (LWLSR) [Ruppert and Wand, 1994] is a particular type of MLS regression, where we assume the noise in the simulation output is of a specified form (given in Section 3.2). As with MLS regression, the main obstacle for any LWLSR technique is the selection of bandwidths for the weight function. The most common approach is to minimize the approximate mean squared error (AMSE) of the LWLSR predictor with respect to the bandwidths [Ruppert et al., 1995a, Hengartner et al., 2002, Fan and Gijbels, 1995, Doksum et al., 2000]. The main difference between each of these methods is how they estimate the AMSE and the choice of plug-in estimators for the parameters on which the AMSE relies. The majority of LWLSR methods focus on the one-dimensional case or use an isotropic weight function, which does not work well when there are multiple dimensions [Wand and Jones, 1993]. Also, the proposed plug-in estimators do not exploit the characteristics of stochastic simulation, namely, access to replications and the placement of design points. Furthermore, the plug-in estimators for the variance are usually designed under the assumption of homoscedasticity [Ruppert et al., 1995a]. These methods also have no way of controlling the number of design points used for prediction, which can slow down computations and detract from the benefit obtained by localization. Other examples of LWLSR methods include using eigenvalues [Prewitt and Lohr, 2006] and estimating the bias empirically [Ruppert, 1997]. See also Cleveland et al. [1988] and Loader [1999] for a discussion of the related method called local regression.

In this paper, we introduce MLS regression into the field of stochastic simulation metamodeling and present a procedure to implement MLS regression in high dimensions. Our procedure can also be used for high-dimensional LWLSR problems, since current procedures focus on the one and two-dimensional cases. Instead of using an isotropic weight function, we use an anisotropic weight function whose bandwidths differ in each dimension. To choose the optimal bandwidths for the MLS predictor, we solve an optimization problem whose objective function is the AMSE of the LWLSR predictor. Unlike existing methods, the optimization problem used to choose the bandwidths is constrained. By putting constraints on the bandwidths, we can control the number of design points used for prediction, which allows our method to produce predictions relatively quickly even when there is a large number of design points. Furthermore, the constrained optimization problem can be solved very efficiently using a variable-pegging procedure. We also introduce new plug-in estimators for the parameters of the method, including the density of design points, the variance

of a replication, and the second derivatives at the prediction point. The plug-in estimators for the density of design points and the variance of a replication at the prediction point exploit the fact that, in the setting of stochastic simulation, we control the placement of design points and can make replications. The plug-in estimators for the second derivatives at the prediction point can be calculated in high-dimensions, unlike existing plug-in estimators (for example, the plug-in estimators in Ruppert et al. [1995b]). Finally, we provide a bound on the expected approximation error and show that the predictor is consistent under certain conditions.

Critically, we do not assume the number of relevant variables is small or that the response surface has a low-dimensional representation. Furthermore, we do not assume the simulation output has homogeneous variance throughout the design space. We want to have good predictions by having a very large number of design points and a space-filling experiment design.

In the next section, we formulate the simulation metamodeling problem and discuss the experiment designs we use in our procedure. Section 3 reviews MLS regression and LWLSR, on which we base our MLS procedure, followed by the presentation of our MLS procedure in Section 4. We then provide a bound on the expected approximation error and establish the consistency of our MLS predictor in Section 5, and discuss estimation of the parameters in Section 6. Lastly, we present results of numerical experiments using two queueing examples in Section 7.

2 Experiment Design

We are interested in predicting a response surface, for example the expected waiting time for a customer in a queue. Denote the response surface at a design point $\mathbf{x}$ by $y(\mathbf{x})$. For the queue example, $\mathbf{x}$ could include arrival rates, service rates, etc. Denote the design space, the set of all possible values of the design variables, by $\mathbb{X}$, which we assume is the unit hypercube (which may be attained by rescaling the natural design variables). Furthermore, let $\{\{\mathbb{X}_n, \mathbb{R}_n\}; n \geq 0\}$ denote a sequence of experiment designs, where $\mathbb{X}_n = (\mathbf{x}_1^n, \mathbf{x}_2^n, \ldots, \mathbf{x}_n^n)$ is the vector containing the first n generated design points, and $\mathbb{R}_n = (R_1^n, R_2^n, \ldots, R_n^n)$ is the vector containing the number of replications we allocate to each design point in $\mathbb{X}_n$. In other words, for the nth sequential design we allocate R_i^n replications to $\mathbf{x}_i^n$. The $\mathbb{X}_n$, $n \geq 1$, are not necessarily nested. We introduce this sequential setting as an asymptotic regime for analyzing our procedure later in the paper. We assume that

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{\mathbf{x}_i^n \in A\} = \int_A g(\mathbf{z}) d\mathbf{z},$$

for all rectangles $A \subseteq \mathbb{X}$, where $g(\mathbf{z})$ is the limiting density of design points at $\mathbf{z} \in \mathbb{X}$. We also assume that

$$\lim_{n\to\infty} \frac{1}{C_n} \sum_{i=1}^{n} \mathbb{I}\{\mathbf{x}_i^n \in A\} R_i^n = \int_A \tilde{g}(\mathbf{z}) d\mathbf{z},$$

for all rectangles $A \subseteq \mathbb{X}$, where $C_n = \sum_{i=1}^n R_i^n$ is the total number of replications allocated in the nth design, and $\tilde{g}(\mathbf{z})$ is the limiting density of effort spent at $\mathbf{z} \in \mathbb{X}$. In our procedure, we assume that g and $\tilde{g}$ are uniform densities on the unit hypercube $[0,1]^d$, i.e., $g(\cdot) = \tilde{g}(\cdot) = 1$ and $\mathbb{X} = [0,1]^d$.

For the nth sequential design, we run the simulation at $\mathbb{X}_n = (\mathbf{x}_1^n, \mathbf{x}_2^n, \ldots, \mathbf{x}_n^n)$. At design point $\mathbf{x}_i^n$ we run R_i^n *i.i.d.* replications of the simulation, and we denote the simulation output of the jth replication by $\mathrm{Y}_j^n(\mathbf{x}_i^n)$, which we assume is an unbiased estimator of $y(\mathbf{x}_i^n)$. The estimate that we

obtain at design point $\mathbf{x}_i^n$ is the sample average

$$\bar{\mathrm{Y}}^n(\mathbf{x}_i^n; R_i^n) = \frac{1}{R_i^n} \sum_{j=1}^{R_i^n} \mathrm{Y}_j^n(\mathbf{x}_i^n).$$

We will also need an estimate of the variance $\sigma^2(\mathbf{x}_i^n)$ of a replication at $\mathbf{x}_i^n$, which we estimate by the sample variance

$$S^2(\mathbf{x}_i^n; R_i^n) = \frac{1}{R_i^n - 1} \sum_{j=1}^{R_i^n} (\mathrm{Y}_j^n(\mathbf{x}_i^n) - \bar{\mathrm{Y}}^n(\mathbf{x}_i^n; R_i^n))^2.$$

If we do not have access to replications, such as in the case of steady-state simulations, we only need an estimate for the variance of the one replication for our procedure.

For ease of notation, we drop the superscripts for the nth sequential design and let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ denote the design points in the nth sequential design, and $R_1, R_2, \ldots, R_n$ denote the replications allocated to each design point in $\mathbb{X}_n$.

3 Local Smoothing Approaches

In this section, we discuss the smoothing methodologies of MLS regression and LWLSR [Ruppert and Wand, 1994]. Both approaches require a positive weight function of the form $\mathrm{K}_{\mathbf{H}}(\mathbf{u}) = |\mathbf{H}|^{-1/2}\mathrm{K}(\mathbf{H}^{-1/2}\mathbf{u})$, where K is a compactly supported d-variate kernel such that $\int \mathrm{K}(\mathbf{u})d\mathbf{u} = 1$, and $\mathbf{H}$ is a $d \times d$ symmetric positive definite matrix depending on n. The matrix $\mathbf{H}$ is called the bandwidth matrix and its entries are called the bandwidth parameters. The bandwidth matrix determines the shape of the contours of the weight function $\mathrm{K}_{\mathbf{H}}$. The number of non-zero entries in the bandwidth matrix is the number of bandwidth parameters that must be chosen before one can apply either of the two smoothing methodologies. In high-dimensional problems, allowing the bandwidth matrix to have non-zero values off the diagonal would result in too many parameters. Therefore we will only consider diagonal bandwidth matrices in our procedure. A diagonal bandwidth matrix will cause the contours of the kernel to be parallel to the main coordinate axes, whereas a full bandwidth matrix would allow the contours of the kernel to be arbitrarily rotated. We do not dwell on this restriction because it has been shown that the improvement gained by allowing off-diagonal entries to be non-zero is not nearly as great as the benefit from allowing the diagonal entries to vary from one another [Wand and Jones, 1993]. Furthermore, the choice of kernel is not as important as the choice of bandwidth matrix, $\mathbf{H}$ [Wand and Jones, 1993]. We employ a kernel which is a function of the maximum norm, given by $||\mathbf{u}||_\infty = \max\{|u_1|, |u_2|, \ldots, |u_d|\}$ for $\mathbf{u} \in \mathbb{R}^d$. This kernel is

$$\mathrm{K}(\mathbf{u}) = \max\{1 - ||\mathbf{u}||_\infty, 0\},$$

and its support is the d-dimensional unit hypercube, which is shown in Figure 1(a) for the case $d = 2$.

The weight function that is induced from this kernel has a compact rectangular support with the bandwidth parameters lying on the diagonal of the bandwidth matrix determining the half-length of each edge of the rectangle. The diagonal bandwidth matrix $\mathbf{H} = \mathrm{diag}\{h_1^2, h_2^2, \ldots, h_d^2\}$ will yield the weight function

$$\mathrm{K}_{\mathbf{H}}(\mathbf{u}) = |\mathbf{H}|^{-\frac{1}{2}} \max\{1 - ||\mathbf{H}^{-\frac{1}{2}}\mathbf{u}||_\infty, 0\},$$

whose support is shown in Figure 1(b) for the two-dimensional case where $h_1 = 1$ and $h_2 = 0.25$, in relation to the support of the kernel in Figure 1(a).

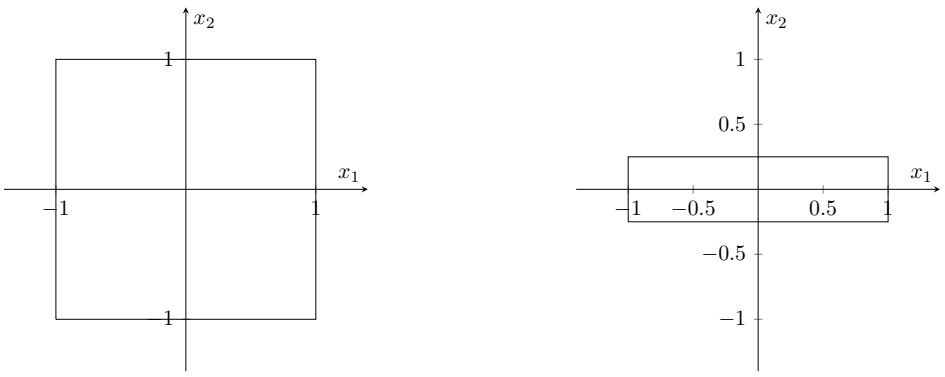

Figure 1: (a) The compact support of the kernel $K(\cdot)$ in two dimensions, which is the unit hypercube. (b) An example of the compact support of the weight function $K_{\mathbf{H}}(\cdot)$ in two dimensions, with $h_1 = 1$, and $h_2 = 0.25$.

3.1 Moving Least Squares Regression

MLS regression reinterprets the metamodeling problem as predicting $y(\mathbf{x})$ for any specific $\mathbf{x} \in \mathbb{X}$ instead of building a metamodel to approximate the entire response surface y. Each design point is assigned a weight, which is similar to weighted least squares regression except that the weight given to a design point depends on the particular prediction point, with the weight being determined by the weight function, $\mathrm{K}_{\mathbf{H}}(\cdot)$. Therefore, every time we predict the response surface at a different prediction point we solve a different weighted least squares problem. In the following, let Π_k^d denote the space of d-variate polynomials of degree k, and let $p_1, p_2, \ldots, p_m$ denote the basis functions of Π_k^d. In this paper, we take the basis functions of Π_k^d to be the standard basis which is the set of $\binom{d+k}{k}$ monomials. The polynomial, $\hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0,\mathbf{H}}$, used for approximating the response surface $y(\mathbf{x}_0)$ at the prediction point $\mathbf{x}_0$ is

$$\hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0,\mathbf{H}} = \arg\min_{p \in \Pi_k^d} \left\{ \sum_{i=1}^{n} (\bar{\mathrm{Y}}(\mathbf{x}_i; R_i) - p(\mathbf{x}_i))^2 \mathrm{K}_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}_0) \right\}. \tag{1}$$

This is the standard approach to MLS regression [Bos and Salkauskas, 1989]. The optimal solution to this problem is obtained from the weighted least squares solution

$$\hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}) = \mathbf{P}(\mathbf{x})^\top (\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1} \mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{Y},$$

where $\mathbf{P}$ is the $n \times m$ matrix whose i^{th} row is $(p_1(\mathbf{x}_i - \mathbf{x}_0), p_2(\mathbf{x}_i - \mathbf{x}_0), \cdots, p_m(\mathbf{x}_i - \mathbf{x}_0))$, and

$$\begin{aligned}
\mathbf{Y} &= (\bar{\mathrm{Y}}(\mathbf{x}_1; R_1), \bar{\mathrm{Y}}(\mathbf{x}_2; R_2), \ldots, \bar{\mathrm{Y}}(\mathbf{x}_n; R_n))^\top \\
\mathbf{W}(\mathbf{x}_0) &= \mathrm{diag}\left\{ \mathrm{K}_{\mathbf{H}}(\mathbf{x}_1 - \mathbf{x}_0), \mathrm{K}_{\mathbf{H}}(\mathbf{x}_2 - \mathbf{x}_0), \ldots, \mathrm{K}_{\mathbf{H}}(\mathbf{x}_n - \mathbf{x}_0) \right\} \\
\mathbf{P}(\mathbf{x}) &= (p_1(\mathbf{x} - \mathbf{x}_0), p_2(\mathbf{x} - \mathbf{x}_0), \ldots, p_m(\mathbf{x} - \mathbf{x}_0))^\top.
\end{aligned}$$

For each prediction point, $\mathbf{x}_0 \in \mathbb{X}$, we get a different approximating polynomial, $\hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0,\mathbf{H}}$.

The minimization in Problem (1) is done over the polynomial space Π_k^d. Since d is the dimension of the design space, the only factor that we are able to choose is k. The dimension of Π_k^d is $\binom{d+k}{k}$, so for large d we must be careful to not pick k too large. Otherwise, we must invert a $\binom{d+k}{k} \times \binom{d+k}{k}$ matrix to obtain the prediction, which is infeasible when d and k are large. We will use the space of linear polynomials, Π_1^d.

3.2 Locally Weighted Least Squares Regression

The weight function $\mathrm{K}_{\mathbf{H}}$ depends on bandwidth parameters that determine the shape and size of its contours. The main problem in MLS regression is optimizing these bandwidth parameters with respect to some criterion. LWLSR is a particular type of MLS regression, where we assume the outputs obtained from the simulation are of the form $\bar{\mathrm{Y}}(\mathbf{x}_i; R_i) = y(\mathbf{x}_i) + (\sigma(\mathbf{x}_i)/\sqrt{R_i})\epsilon_i$, where $\sigma(\mathbf{x}_i)$ is the standard deviation of a replication at $\mathbf{x}_i$ and the ϵ_i are mutually independent and identically distributed random variables with zero mean and unit variance. Using this assumption, we can obtain an expression for the AMSE of the LWLSR predictor and use this expression to choose the bandwidth parameters for the MLS predictor.

LWLSR with linear polynomials uses a first-order Taylor expansion to approximate the function value at the prediction point. The LWLSR prediction at $\mathbf{x}_0$ is $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}) \triangleq \hat{\beta}_0$, where $\hat{\beta}_0$ is from the solution to the problem

$$\min_{\beta_0, \boldsymbol{\beta}_1} \sum_{i=1}^{n} (\bar{\mathrm{Y}}(\mathbf{x}_i; R_i) - \beta_0 - \boldsymbol{\beta}_1^\top (\mathbf{x}_i - \mathbf{x}_0))^2 \mathrm{K}_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}_0),$$

which is just a reformulated version of the MLS problem in Section 3.1 when we use the space Π_1^d in MLS regression. Note that $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}) = \hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0, \mathbf{H}}(\mathbf{x}_0)$.

To analyze the MSE of the predictor so we can obtain an expression for the AMSE at the prediction point $\mathbf{x}_0 \in \mathbb{X}$, assume that we have a sequence of bandwidth matrices $\{\mathbf{H}_n : n \geq 1\}$. We need the following assumptions, taken from Ruppert and Wand [1994].

Assumption 1. *The prediction point $\mathbf{x}_0$ is in the interior of $\mathbb{X}$. At $\mathbf{x}_0$, $\sigma^2(\cdot)$ is continuous, the limiting densities of design points and simulation effort, i.e. g and $\tilde{g}$, are continuously differentiable, and all second-order derivatives of y are continuous. Also, $g(\mathbf{x}_0) > 0$, $\tilde{g}(\mathbf{x}_0) > 0$, and $0 < \sigma^2(\mathbf{x}_0) < \infty$.*

Assumption 2. *The sequence of bandwidth matrices $\{\mathbf{H}_n : n \geq 1\}$ is such that $n^{-1}|\mathbf{H}_n|$ and each entry of $\mathbf{H}_n$ tends to zero as $n \to \infty$ with $\mathbf{H}_n$ remaining symmetric and positive definite. Also, there is a fixed constant L such that the condition number of $\mathbf{H}_n$ is at most L, for all n.*

Let $\mathbf{x}_0$ be a point that satisfies Assumption 1 and let $\{\mathbf{H}_n : n \geq 1\}$ be a sequence of bandwidth matrices that satisfies Assumption 2. From Ruppert and Wand [1994], we have

$$\mathrm{E}\{\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}_n) - y(\mathbf{x}_0) | \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} = \frac{1}{2}\mu_2(\mathrm{K})\mathrm{tr}\{\mathbf{H}_n \nabla_y^2(\mathbf{x}_0)\} + o_P\{\mathrm{tr}(\mathbf{H}_n)\} \tag{2}$$

$$\mathrm{Var}\{\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}_n) | \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} = \frac{R(\mathrm{K})\sigma^2(\mathbf{x}_0)}{C_n |\mathbf{H}_n|^{1/2} \tilde{g}(\mathbf{x}_0)}\{1 + o_P(1)\}, \tag{3}$$

where o_P denotes order in probability, $\mu_2(\mathrm{K}) = \int_{\mathbb{R}} \mathbf{x}_i \mathrm{K}(\mathbf{x})\, d\mathbf{x}$, $R(\mathrm{K}) = \int_{\mathbb{R}} \mathrm{K}(\mathbf{x})^2\, d\mathbf{x}$, and $\nabla_y^2(\mathbf{x}_0)$ is the Hessian of y evaluated at $\mathbf{x}_0$. For the diagonal bandwidth matrix $\mathbf{H} = \mathrm{diag}\{h_1^2, h_2^2, \ldots, h_d^2\}$, the AMSE of the estimator $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H})$ is given by the sum of the leading order terms in (2) and (3),

$$\begin{aligned} \mathrm{AMSE} &= \frac{1}{4}\mu_2(\mathrm{K})^2 \mathrm{tr}\{\mathbf{H}\nabla_y^2(\mathbf{x}_0)\}^2 + \frac{R(\mathrm{K})\sigma^2(\mathbf{x}_0)}{C_n |\mathbf{H}|^{1/2} \tilde{g}(\mathbf{x}_0)} \\ &= \frac{1}{4}\mu_2(\mathrm{K})^2 (h_1^2 \mathcal{D}_1(\mathbf{x}_0) + \cdots + h_d^2 \mathcal{D}_d(\mathbf{x}_0))^2 + \frac{R(\mathrm{K})\sigma^2(\mathbf{x}_0)}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^d h_i}, \end{aligned} \tag{4}$$

where $\mathcal{D}_i(\mathbf{x}_0)$ denotes the second partial derivative $\partial^2 y(\mathbf{x}_0)/\partial x_i^2$. Equation (4) shows the bias-variance trade-off with respect to the bandwidth parameters. The first term in the sum represents

the bias of the estimator, while the second term represents variance. When the bandwidth parameters are small, the bias of the estimator $\hat{\beta}_0$ is small, but fewer design points are used in the prediction, making the variance of the estimator high. For large bandwidth parameters, the opposite happens.

We can use the bias-variance trade-off to choose the bandwidth parameters by minimizing the AMSE equation. In the bias term, given by the first part of Equation (4), directions corresponding to larger changes in the response surface (i.e., larger second partial derivatives) result in smaller bandwidth parameters corresponding to those directions. This regulates the bias because weight decays more rapidly in directions where there are larger changes in the response surface. In the variance term, given by the second part of Equation (4), a higher variance at the prediction point, $\sigma^2(\mathbf{x}_0)$, with all other parameters fixed, will increase the bandwidth parameters, incorporating more design points in the approximation and therefore filtering out the larger noise. The limiting density of effort spent at the prediction point, $\tilde{g}(\mathbf{x}_0)$, with all other parameters fixed, will give smaller bandwidth parameters to prediction points in regions of higher density. Intuitively, this is because in regions where we have spent the most simulation effort, we would like the prediction to be based on design points closer to the prediction point, making the bandwidth parameters smaller, and hence decreasing the bias.

4 Moving Least Squares Procedure

We provide a brief outline of the procedure, with details following in Section 4.1, Section 4.2, and Section 6.

1. Run the simulation model at design points satisfying the conditions in Section 2. Compute the sample averages across replications and estimate the variance of a replication at each of the design points.

2. For each prediction point $\mathbf{x}_0$

   (a) Estimate the second derivatives and the variance of a replication at $\mathbf{x}_0$ using the methods in Section 6.

   (b) Calculate the bandwidth parameters of the weight function by solving MP(1) in Section 4.1 using the **Bandwidth Procedure** in the appendix. *Optional* : Put an upper bound on the number of design points used for prediction, as discussed in Section 4.2.

   (c) Predict the mean response at the prediction point. The MLS prediction is given by the optimal solution of Equation (1), using the bandwidth parameters calculated in Step 2(b). *Optional* : Include interaction terms in Equation (1), as discussed in Section 4.2.

4.1 Moving Least Squares Procedure

Let $\mathbf{H}_{l,r} = \text{diag}\{(h_1^l \vee h_1^r)^2, (h_2^l \vee h_2^r)^2, \ldots, (h_d^l \vee h_d^r)^2\}$ where $x \vee y = \max\{x, y\}$. For our procedure we will use the weight function $\text{K}_{\mathbf{H}_{l,r}}(\cdot)$, given by

$$\text{K}_{\mathbf{H}_{l,r}}(\mathbf{u}) = |\mathbf{H}_{l,r}|^{-\frac{1}{2}} \max\{1 - ||\mathbf{H}_{l,r}^{-\frac{1}{2}}\mathbf{u}||_\infty, 0\},$$

with the associated prediction window defined by the region $\Omega \triangleq \{\mathbf{x} \in \mathbb{X} : |x_i - x_{0,i}| \leq h_i^l \vee h_i^r, \forall i\}$. Each bandwidth parameter h_i in Equation (4) has been replaced with two separate parameters to deal with effects for prediction points lying near the boundary. The variable h_i^l denotes the

distance from the left edge of the prediction window to the prediction point in the ith coordinate, and the variable h_i^r denotes the distance from the right edge to the prediction point. The bandwidth parameters $h_1^l \vee h_1^r, h_2^l \vee h_2^r, \ldots, h_d^l \vee h_d^r$ determine the bandwidth in the corresponding coordinate direction. For example, $h_1^l \vee h_1^r$ determines how fast the weight decays in the direction along the first basis vector of $\mathbb{R}^d$. The region Ω is the intersection of the compact support of the kernel $K_{\mathbf{H}}$ and the design space $\mathbb{X}$, so the design points that fall in the region will be the design points used for prediction, hence the name "prediction window".

Assuming we have estimates $s^2(\mathbf{x}_0)$ and $\widehat{\mathcal{D}_i(\mathbf{x}_0)}$, of $\sigma^2(\mathbf{x}_0)$ and $\mathcal{D}_i(\mathbf{x}_0)$, for $i = 1, 2, \ldots, d$, the bandwidth parameters are found by solving MP(1), whose objective function is a modification of the AMSE equation, and then transforming the optimal solution.

$$
\begin{aligned}
\text{MP(1)}: \min_{\{h_1,\ldots,h_d\}} \quad & \frac{1}{4}\mu_2(\mathrm{K})^2 \left(h_1^2 \left|\widehat{\mathcal{D}_1(\mathbf{x}_0)}\right| + \cdots + h_1^2 \left|\widehat{\mathcal{D}_d(\mathbf{x}_0)}\right| \right)^2 + \frac{R(\mathrm{K}) s^2(\mathbf{x}_0)}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^d h_i} \\
\text{s.t.} \quad & \dim(\Pi_1^d) + \delta \le n g(\mathbf{x}_0) \prod_{i=1}^d (h_i^l + h_i^r) \\
& 2h_i = h_i^l + h_i^r \quad \text{for } \ i = 1, 2, \ldots, d \\
& 0 \le h_i^l \le x_{0,i}, \quad \text{for } \ i = 1, 2, \ldots, d \\
& 0 \le h_i^r \le 1 - x_{0,i}, \quad \text{for } \ i = 1, 2, \ldots, d \\
& h_i^l + h_i^r \le f_n, \quad \text{for } \ i = 1, 2, \ldots, d,
\end{aligned}
$$

where f_n, defined in Section 5.2, depends on d, and the density and number of design points. The constraint $h_i^l + h_i^r \le f_n$ will be discussed in Section 5.2, as will estimation of $\sigma^2(\mathbf{x}_0)$ and the second partial derivatives.

The bandwidths in Equation (4) represent the half-widths of the prediction window, when the prediction window is symmetric about the prediction point. In an effort to keep the same interpretation for the bandwidths in the objective function of MP(1), where the prediction window may not be symmetric about the prediction point, we have the constraint $2h_i = h_i^l + h_i^r$.

The motivation for the second constraint is the following. To ensure that the number of design points used for prediction is at least the dimension of Π_1^d and to protect against having linearly dependent columns in the matrix $\mathbf{P}$ of the solution to Problem 1, we set a lower bound, $\dim(\Pi_1^d)+\delta$, on the number of design points that lie within the prediction window. We use $\delta = 5d$. An approximation to the number of design points that lie within the prediction window is $n g(\mathbf{x}_0) \prod_{i=1}^d (h_1^l + h_1^r)$. This can be interpreted as the density of design points at the prediction point $n g(\mathbf{x}_0)$ times the volume of the prediction window which gives us the total number of design points included in the prediction window. The limiting density of design points that makes $n g(\mathbf{x}_0) \prod_{i=1}^d (h_1^l + h_1^r)$ the best approximation is the uniform density, which is the density we use in this procedure. The constraints $0 \le h_i^l \le x_{0,i}$ and $0 \le h_i^r \le 1 - x_{0,i}$ ensure that the bandwidth parameters are confined to the unit hypercube, so that $\prod_{i=1}^d (h_1^l + h_1^r)$ is the volume of the prediction window.

The second derivatives in the AMSE equations have been replaced by the absolute values of the second derivatives to ensure that the bandwidth parameters behave well when some second derivatives are positive and some are negative. To see the motivation for this change, consider the case where the response surface has both positive and negative second partial derivatives. By setting the bandwidth parameters in the proper proportion to each other, the AMSE equation would appear to kill the approximate bias. We could then reduce the variance by increasing the size of the prediction window. However this increase in window size reduces the validity of the bias approximation, so for a fixed value of n, Equation (4) may cease to be a good approximation to

the MSE when a large window is used. Thus, we take a conservative approach to the window size and use an upper bound on the AMSE.

The **Bandwidth Procedure** in the appendix solves MP(1) and then transforms the optimal solution to get the bandwidth parameters. Denote the output of the **Bandwidth Procedure** by $\mathbf{h}^* = \{h_1^{l\,*}, h_1^{r*}, h_2^{l\,*}, h_2^{r*}, \ldots, h_d^{l\,*}, h_d^{r*}\}$ and let $\mathbf{H}_{l,r}^* = \text{diag}\{(h_1^{l\,*} \vee h_1^{r*})^2, (h_2^{l\,*} \vee h_2^{r*})^2, \ldots, (h_d^{l\,*} \vee h_d^{r*})^2\}$. The weight function used for prediction is given by

$$\text{K}_{\mathbf{H}_{l,r}^*}(\mathbf{u}) = |\mathbf{H}_{l,r}^*|^{-\frac{1}{2}} \max\{1 - ||{\mathbf{H}_{l,r}^*}^{-\frac{1}{2}}\mathbf{u}||_\infty, 0\},$$

4.2 Modifications to the MLS Procedure

The first constraint in MP(1) controls how many design points (approximately) fall within the prediction window by regulating the size of the prediction window. For high-dimensional problems when there is a very large number of design points, we may want to limit the amount of design points that we use for prediction for the computing time to be acceptable. We can limit the number of design points that fall within the prediction window by placing an upper bound on the first constraint in MP(1). We denote the upper bound by Mass_{UB}, and we use $\text{Mass}_{\text{UB}} = 2000$ in this paper (of course, this could be much higher depending on computing power). In this case, the bandwidth parameters are found by solving MP(1) with the added constraint $ng(\mathbf{x}_0)\prod_{i=1}^d (h_i^l + h_i^r) \leq \text{Mass}_{\text{UB}}$, which can also be solved using the **Bandwidth Procedure** in the appendix. We denote the new optimization problem (MP(1) with the added constraint $ng(\mathbf{x}_0)\prod_{i=1}^d (h_i^l + h_i^r) \leq \text{Mass}_{\text{UB}}$) by MP(2).

The AMSE expression is the result of using LWLSR for prediction, which results in second partial derivatives in the bias term. The second partial derivatives arise because we use a linear approximation, and therefore cannot account for higher-order derivatives. The bias term in Equation (4) is only an approximation to the bias at the prediction point, and will underestimate the amount of bias since the approximation does not consider the higher-order partial derivatives, and assumes that the prediction window is symmetric about the prediction point. Although the bias of the LWLSR predictor for a prediction point near the center of the design space is of the same order as for a prediction point lying near the boundary, namely, $o_P\{\text{tr}(H)\}$, we would still like to try to reduce the bias. In an effort to further reduce the bias, we use a stepwise regression method to determine if there are necessary second-order terms that should be included in the model. Note that for higher-order terms to be added, δ may have to be increased to ensure non-singularity of the matrix $\mathbf{P}$. Denote the prediction window of $\text{K}_{\mathbf{H}}^*$ by Ω^*, and let $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_{|\Omega^*|}^*$ denote the $|\Omega^*|$ design points that fall into the prediction window. The stepwise procedure is as follows:

0. Initialize the $|\Omega^*| \times (d+1)$ regression matrix $\mathbf{X}$, with ith row $[1, x_{1,1}^*, x_{1,2}^*, \ldots, x_{1,d}^*]$, and let $\mathbf{Y}$ denote the vector of observations at the design points in the prediction window. Also, let $\mathbf{R}$ denote the regression matrix consisting of all possible second-order terms.

1. Normalize and center the columns of $\mathbf{R}$.

2. Calculate the vector of correlations $\mathbf{c} = \mathbf{R}^\top \left(\mathbf{Y} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}\right)$. Choose the ith term corresponding to, say, $x_j x_k$, such that $c_i = \max\{\mathbf{c}\}$.

3. Add a column to $\mathbf{X}$ corresponding to $x_j x_k$, and remove the corresponding column from $\mathbf{R}$. If $c_i \leq \rho$ orthe maximum number of iterations is reached, stop. Otherwise, go to 2.

This stepwise procedure starts with a linear approximation, and adds second-order terms to the approximating polynomial in a greedy manner by choosing the next term that is most correlated with the residuals of the current approximating polynomial. The procedure stops when either the correlations become too weak (are less than ρ), or the maximum number of iterations is reached.

5 Error Analysis

In this section, we give a bound on the expected approximation error, as well as show that the estimator is consistent as the amount of simulation effort increases to infinity.

5.1 Approximation Error

We can bound the expected approximation error using the second-order partial derivatives, and the variance of the simulation output and number of replications at each of the design points. Let $C^2(\mathbb{X})$ be the space of twice-continuously differentiable functions on $\mathbb{X}$. Furthermore, for the vectors $\mathbf{x} = (x_1, x_2, \ldots, x_d)^\top$ and $\mathbf{v} = (v_1, v_2, \ldots, v_d)^\top$, let $\partial^{|\mathbf{v}|}/\partial\mathbf{x}^{\mathbf{v}} = \partial^{v_1+v_2+\cdots+v_d}/\partial x_1^{v_1}\partial x_2^{v_2}\cdots\partial x_d^{v_d}$ and $\mathbf{x}^{\mathbf{v}} = x_1^{v_1}x_2^{v_2}\cdots x_d^{v_d}$.

Theorem 5.1. *Let $y \in C^2(\mathbb{X})$. If $\bar{Y}(\mathbf{x}_i; R_i) = y(\mathbf{x}_i) + (\sigma(\mathbf{x}_i)/\sqrt{R_i})\epsilon_i$, for $i = 1, 2, \ldots, n$, where the ϵ_i are mutually independent and identically distributed random variables with zero mean and unit variance, then*

$$\mathrm{E}\left[(\hat{y}^{\mathrm{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}_0) - y(\mathbf{x}_0))^2\right] \leq \left(\frac{1}{2}\sum_{|\mathbf{v}|=2}\sum_{i=1}^{n} C_i^{\mathbf{v}}|\mathbf{x}_i - \mathbf{x}_0|^{\mathbf{v}}|\Xi_i|\right)^2 + \sum_{i=1}^{n}\frac{\sigma_i^2(\mathbf{x}_i)}{R_i}\Xi_i^2,$$

where

$$C_i^{\mathbf{v}} = \sup_{0\leq\eta\leq 1}\left|\frac{\partial^{|\mathbf{v}|}y(\eta(\mathbf{x}_i - \mathbf{x}_0) + \mathbf{x}_0)}{\partial\mathbf{x}^{\mathbf{v}}}\right|,$$

$\Xi_i = \frac{\det(\mathbf{P}^\top\mathbf{W}(\mathbf{x}_0)\mathbf{P}^i)}{\det(\mathbf{P}^\top\mathbf{W}(\mathbf{x}_0)\mathbf{P})}$, *and $\mathbf{P}^i$ is the matrix $\mathbf{P}$ with the first column replaced with the ith standard basis vector.*

5.2 Consistency Results

We now discuss the consistency of the estimator $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}^*_{l,r})$, with the bandwidths obtained by minimizing MP(1) in Section 4.1. For the purpose of analysis, consider a sequential design indexed by n. We analyze two cases of the experiment design: $C_n/n \to \infty$ and $C_n = O(n)$, where C_n is the total number of simulation replications allocated in the nth design. In the first case, the number of replications allocated to each design point becomes infinite, whereas in the second case, the number of replications per design point is bounded by a constant. We deal with consistency in each of these two cases separately.

As mentioned in Section 4.1, f_n depends on the dimension d, and the density and number of design points. Furthermore, f_n converges to zero as $n \to \infty$ to ensure that as the simulation effort increases, the bandwidths of the prediction window will shrink to zero. Consider the two-dimensional case where the x_1 coordinate has a second partial derivative of zero and the x_2 coordinate has a second partial derivative that is greater than zero. This will cause the prediction window to take the shape of a telephone pole, with the long edge in the x_1 coordinate. As the simulation effort increases, the volume of the prediction window will shrink to zero even though h_1 remains equal to one (reaching the boundary of the unit hypercube). However, higher-order derivatives in the x_1 coordinate may be greater than zero, leading to bias in the prediction that is not detected by the AMSE Equation (4). Therefore, we do actually want to shrink h_1, which is the purpose of f_n and the constraint $h_1^l + h_1^r \leq f_n$. Without these constraints, the bandwidth parameters may not shrink to zero and the estimator may not be consistent. Each case of the experiment design will require a different definition of f_n, given in the respective definition.

All three proofs follow the same format. We first show that **Assumption 2** holds, making all of the conditions of Theorem 2.1 of Ruppert and Wand [1994] satisfied. Since the conditions of Theorem 2.1 of Ruppert and Wand [1994] are met, Equations (2) and (3) are the conditional bias and conditional variance of $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}^*_{l,r})$, respectively. Then we show that Equations (2) and (3) converge to zero in probability, which proves the claim of consistency. In the following, the bandwidths, variance, and second partial derivative estimates are functions of n to explicitly show the dependence on n. For brevity, let $\mathcal{D}_i(n)$ denote $\widehat{\mathcal{D}_i(\mathbf{x}_0)}$ for the nth experiment design, and let $s^2(n)$ denote $s^2(\mathbf{x}_0)$ for the nth sequential design.

For the following three theorems, we will make use of this condition:

Condition 1. *The prediction point $\mathbf{x}_0 \in \mathbb{X}$ satisfies Assumption 1, $P(\limsup_{n\to\infty} \mathcal{D}_i(n) < \infty) = 1$ for $i = 1, 2, \ldots, d$, and $P(\limsup_{n\to\infty} s^2(n) < \infty) = 1$.*

In the case $C_n/n \to \infty$, the only restrictions we need on the second partial derivative or variance estimates is boundedness, since the constraint $\dim(\Pi_1^d) + \delta \leq n g(\mathbf{x}_0) \prod_{i=1}^d h_i(n)$ ensures that the volume of the prediction window converges to zero slowly enough. However, we need the sequence f_n to converge to zero quickly enough to meet the regularity conditions given in Assumption 2.

Theorem 5.2. *Assume that $C_n/n \to \infty$ and Condition 1 is satisfied. If the bandwidths are chosen according to MP(1), with $f_n = (M/g(\mathbf{x}_0))^{1/d}(1/n)^{1/d}$, where $M > dim(\Pi_1^d) + \delta$, then $\hat{y}^{LOC}(\mathbf{x}_0; \mathbf{H}^*_{l,r}) \xrightarrow{p} y(\mathbf{x}_0)$ as $n \to \infty$.*

In the case $C_n = O(n)$, the second derivative estimates can get arbitrarily large, as long as they do not stay large. Similarly, the variance estimates can get arbitrarily small, as long as they do not stay small. These conditions ensure that the volume of the prediction window converges to zero at the correct rate, resulting in consistency of the estimator. We need f_n to converge to zero slower than in the case of Theorem 5.2 to ensure that the volume of the prediction window does not converge to zero too quickly.

Theorem 5.3. *Assume that $P(\liminf_{n\to\infty} s^2(n) > 0) = 1$, $C_n = O(n)$, and Condition 1 is satisfied. If the bandwidths are chosen according to MP(1), with $f_n = (M/g(\mathbf{x}_0))^{1/d}(1/n)^{1/(d+1)}$, where $M > dim(\Pi_1^d) + \delta$, then $\hat{y}^{LOC}(\mathbf{x}_0; \mathbf{H}^*_{l,r}) \xrightarrow{p} y(\mathbf{x}_0)$ as $n \to \infty$.*

In the case of MP(2), placing an upper bound pushes the volume of the prediction window to zero faster, and limits the number of design points that are included in the prediction window. For the estimator to be consistent in this case, we need to allocate more and more replications to each design point, so that the simulation effort included in the prediction window goes to infinity. This increase in replications per design point is given by the condition $C_n/n \to \infty$ as $n \to \infty$.

Theorem 5.4. *Assume that Condition 1 is satisfied. If the bandwidths are chosen according to MP(2) with $f_n = (M/g(\mathbf{x}_0))^{1/d}(1/n)^{1/d}$, where $M > dim(\Pi_1^d) + \delta$, then $\hat{y}^{LOC}(\mathbf{x}_0; \mathbf{H}^*_{l,r}) \xrightarrow{p} y(\mathbf{x}_0)$ as $n \to \infty$ if and only if $C_n/n \to \infty$ as $n \to \infty$.*

6 Parameter Estimation

As mentioned in Section 4, estimation of $\sigma^2(\mathbf{x}_0)$ and $\mathcal{D}_i(\mathbf{x}_0)$ is required to solve MP(1). As is often done in LWLSR, we use plug-in estimates $s^2(\mathbf{x}_0)$ and $\widehat{\mathcal{D}_i(\mathbf{x}_0)}$ (see, for example, Ruppert et al. [1995b]) for $\sigma^2(\mathbf{x}_0)$ and $\mathcal{D}_i(\mathbf{x}_0)$, respectively. In existing LWLSR techniques, estimation of the density $g(\mathbf{x}_0)$ of design points around the prediction point is also required. However, in our

procedure, we control the placement of design points and can use the densities discussed in Section 2 as plug-in estimates.

The computationally expensive part of parameter estimation is finding nearest neighbors. A possible solution is to use ϵ-approximate nearest neighbors that involves preprocessing the data using a balanced-box decomposition tree, but we will not discuss this here and refer the reader to Arya et al. [1998].

6.1 Variance Estimation

Having access to replications from the simulation makes it easy for us to get an estimate of the variance of a replication at each design point. However, we need an estimate of the variance of a replication at the prediction point as it pertains to determining the size of the prediction window. We use the variance estimates at neighbors of $\mathbf{x}_0$ to estimate $\sigma^2(\mathbf{x}_0)$ and we denote the estimate by

$$s^2(\mathbf{x}_0) \triangleq \frac{1}{k} \sum_{\mathbf{x}_i \in \mathrm{I}_k(\mathbf{x}_0)} S^2(\mathbf{x}_i; R_i),$$

where $\mathrm{I}_k(\mathbf{x}_0)$ is the set of the k nearest design points to $\mathbf{x}_0$. From our experiments, we have found that the choice of k is not critical, as long as we use enough neighbors to reduce the noise of the variance estimates at the design points. We have found that $k = \min\{5d, n\}$ is a sufficient number of neighbors to reduce the noise.

6.2 Second Derivative Estimation

To estimate the second partial derivatives, we fit a third-order polynomial in a neighborhood of the prediction point and use the coefficients of the second-order terms as estimates of the second partial derivatives. Ruppert and Wand [1994] suggest using an r-order polynomial to estimate partial derivatives of order m, where $r - m$ is an odd integer. In this paper, we use $r = m + 1$. A third-order polynomial with all interaction terms has $\binom{d}{3} + 1$ terms, which makes the regression problem too expensive in high dimensions. Thus, we do not include any interaction terms in the third-order polynomial and solve

$$\min_{\beta_0, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3} \sum_{\mathbf{x}_i \in \mathrm{I}_{k^*}(\mathbf{x}_0)} \left(\bar{\mathrm{Y}}(\mathbf{x}_i; R_i) - \beta_0 - \sum_{j=1}^{3} \boldsymbol{\beta}_j^\top (\mathbf{x}_i - \mathbf{x}_0)^j \right)^2, \tag{5}$$

where $(\mathbf{x}_i - \mathbf{x}_0)^m \triangleq [(x_{i,1} - x_{0,1})^m, (x_{i,2} - x_{0,2})^m, \ldots, (x_{i,d} - x_{0,d})^m]^\top$. We use $2\hat{\boldsymbol{\beta}}_2$, where $\hat{\boldsymbol{\beta}}_2$ is from the solution of Problem (5), as our estimate of the second partial derivatives. To find k^*, the optimal number of neighbors to be used in the estimation of the second partial derivatives, we use the **Nearest-Neighbor Procedure** in the appendix. This procedure is a variation of the procedure used in Ruppert et al. [1995b], and searches for the optimal number of neighbors to fit the cubic polynomial by maximizing the goodness-of-fit criterion $R^2(k)$, which denotes the R^2 statistic using the k nearest neighbors, over k.

7 Numerical Experiments

Our goal is to investigate how the differentiability of the response surface, number of design points, variance of the simulation output, and dimension affect the procedure. We use two queueing

simulations, a multi-product M/G/1 queue and a multi-product Jackson network, whose simulation response surfaces are the expected number of products in the queue and expected cycle time of a product, respectively. The response surface for the multi-product M/G/1 queue is differentiable everywhere, while the response surface for the multi-product Jackson network is non-differentiable in some places.

The n design points we use in each experiment are the first n points from the Sobol Sequence [Sobol, 1967]. We fix the number of replications at each design point to 64. For each replication, the simulation run-length is chosen to obtain constant relative standard deviation over the design space using a heavy-traffic approximation to the asymptotic variance presented in Whitt [1989]. The relative standard deviation we use here is $\left(\sigma(\mathbf{x}_i)/\sqrt{N_i}\right)/|y(\mathbf{x}_i)|$, so, for example, a relative standard deviation of 0.25 means $\sigma(\mathbf{x}_i)/|y(\mathbf{x}_i)| = 2 = 0.25\sqrt{64}$. Using designs generated by the Sobol sequence and fixing the number of replications at each design point satisfies our assumption of a uniform limiting density of design points and simulation effort. In our experiments, we use an upper bound of 2000 in the MLS procedure, i.e., $\text{Mass}_{\text{UB}} = 2000$.

The prediction points $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{150}$ are 150 points uniformly sampled from the unit hypercube, $[0,1]^d$, rescaled to fit inside the hypercube $[0.1, 0.9]^d$. We use 150 prediction points in our experiments only for the sake of estimating the quality of the predictions from the metamodel; we do not envision using the metamodel 150 times in reality. We repeat the experiment 50 times to get 50 predictions at each prediction point. We evaluate the predictions using Root Empirical Relative Mean Squared Error

$$\text{RERMSE} = \sqrt{\frac{1}{7500}\sum_{j=1}^{50}\sum_{i=1}^{150}\left(\frac{\hat{y}_j(\mathbf{p}_i)}{y(\mathbf{p}_i)} - 1\right)^2},$$

where $\hat{y}_j(\mathbf{p}_i)$ is the estimated value of $y(\mathbf{p}_i)$ on the jth experiment at the ith prediction point.

Alternative methods that we compare against our method are the MLS regression method of Lipman et al. [2006] using the data-independent error bound and assuming we know the magnitude of the error in the simulation output (which we refer to as the "vanilla MLS" method), Classification and Regression Trees (CART) [Breiman et al., 1984] implemented using the **rpart** package in R, RODEO [Lafferty and Wasserman, 2008], stochastic kriging [Ankenman et al., 2010] using the Gaussian correlation function (implemented using the **mlegp** package in R), and weighted least squares regression (WLS). Although global metamodeling methods, such as stochastic kriging and WLS, are known not to perform well when the number of design points is large, we include them to show when these methods start breaking down and how our MLS method overcomes these difficulties.

7.1 Multi-Product M/G/1 Queue

In the multi-product M/G/1 queue, $d-1$ types of products arrive to a queue according to a Poisson Process. Let the service rate of product i be μ_i. The vector of design variables is $\mathbf{x} = (x_1, x_2, \ldots, x_{d-1}, \rho)$, where ρ is the traffic intensity and the x_i determine the arrival rates for the $d-1$ types of products. For $\mathbf{x} = (x_1, x_2, \ldots, x_{d-1}, \rho)$ the arrival rate for product i is $\lambda_i = cx_i$ where $c = \rho/\sum_{i=1}^{d-1}(x_i/\mu_i)$ and $\mu_i \in [1,5]$. The response surface that we estimate with the simulation is the steady-state expected waiting time in the queue. The closed-form solution for the steady-state expected waiting time used for evaluating the predictions is

$$y(\mathbf{x}) = \frac{\rho\sum_{i=1}^{d-1}\frac{cx_i}{\mu_i^2}}{(1-\rho)\sum_{i=1}^{d-1}\frac{cx_i}{\mu_i}}.$$

Table 1: Relative difference for the multi-product M/G/1 queue example.

d	n	RSD	relative difference					
			MLS	vanilla MLS	CART	RODEO	SK	WLS
5	500	0.05	-58%	-14%	-42%	>0%	-49%	>0%
		0.1	-62%	-20%	-45%	>0%	-53%	-12%
		0.25	-67%	-26%	-53%	>0%	-63%	-62%
25	5000	0.05	-48%	>0%	>0%	>0%	∅	>0%
		0.1	-52%	>0%	>0%	>0%	∅	-10%
		0.25	-59%	>0%	>0%	>0%	∅	-62%
75	150000	0.05	-40%	>0%	>0%	>0%	∅	∅
		0.1	-42%	>0%	>0%	>0%	∅	∅
		0.25	-53%	>0%	>0%	>0%	∅	∅

The design space is $[5, 10]^{d-1} \times [0.8, 0.95]$, which after rescaling is the d-dimensional unit hypercube.

7.2 Multi-Product Jackson Network

In the multi-product Jackson Network, $d-1$ products arrive to the first station of a system of 3 single-server stations according to a Poisson Process. The service rate at station j is μ_j, which is independent of the product type. The vector of design variables is $\mathbf{x} = (x_1, x_2, \ldots, x_{d-1}, \rho)$, where ρ is the traffic intensity and the x_i determine the arrival rates for the $d-1$ types of products to the first station. For $\mathbf{x} = (x_1, x_2, \ldots, x_{d-1}, \rho)$ the arrival rate for product i is $\lambda_i = cx_i$ where $c = \max_j \rho / \sum_{i=1}^{d-1}(x_i \delta_{ij}/\mu_j)$ and $\mu_i \in [1, 5]$. We denote the number of visits to station j by product i by δ_{ij}. The response surface that we estimate with the simulation is the expected cycle time of product 1, which has the closed-form solution

$$y(\mathbf{x}) = \sum_{j=1}^{3} \frac{\delta_{1j}}{\mu_j - \sum_{k=1}^{d-1} cx_k \delta_{kj}}.$$

The design space is $[5, 10]^{d-1} \times [0.8, 0.95]$, which after rescaling is the d-dimensional unit hypercube.

7.3 Experiment Results

Tables 1–2 display the relative difference of RERMSE and relative standard deviation using our MLS method, the vanilla MLS method, CART, RODEO, stochastic kriging using the Gaussian correlation function, and WLS. A table entry of ∅ means that the corresponding R package used to fit the model ran out of memory. Table 1 gives the results for the multi-product M/G/1 queue example, and Table 2 gives the results for the multi-prodcut Jackson network example. These values are calculated by subtracting the relative standard deviation used to choose the run length in the experiment from the RERMSE and standardizing by dividing the difference with the relative standard deviation. For example, if we used a relative standard deviation of 0.25, and obtained an RERMSE of 0.1 for that experiment, the value in the table would be $100\% \times (0.1 - 0.25)/0.25 = -60\%$. Thus, as can be seen directly from the definition, a relative difference of -100% is the best possible.

From Tables 1–2, it is clear that our procedure is successful in filtering out the noise obtained from using noisy observations at the design points. Our MLS procedure produced better predictions, in terms of relative difference, in each case except for the 25-dimensional M/G/1 queue example

Table 2: Relative difference for the multi-product Jackson network example.

d	n	RSD	relative difference					
			MLS	vanilla MLS	CART	RODEO	SK	WLS
5	500	0.05	-52%	-10%	-37%	>0%	-43%	>0%
		0.1	-58%	-17%	-41%	>0%	-51%	-6%
		0.25	-63%	-23%	-51%	>0%	-60%	-52%
25	5000	0.05	-45%	>0%	>0%	>0%	∅	>0%
		0.1	-50%	>0%	>0%	>0%	∅	-7%
		0.25	-57%	>0%	>0%	>0%	∅	-55%
75	150000	0.05	-36%	>0%	>0%	>0%	∅	∅
		0.1	-38%	>0%	>0%	>0%	∅	∅
		0.25	-44%	>0%	>0%	>0%	∅	∅

when the relative standard deviation was set at 0.25. However, as will be seen in Table 6, when we increased the number of design points, our MLS procedure produced better predictions than WLS. Our MLS procedure works even when the number of design points is large, whereas techniques such as stochastic kriging and WLS fail to produce results when the number of design points is larger than 5,000 and 50,000, respectively (represented by ∅ in the tables).

The local metamodeling methods include our MLS method, the vanilla MLS method, and RODEO. From Tables 5–7, we can see that our MLS method scales well in high dimensions when both RERMSE and runtime are considered. The vanilla MLS method suffered from bad predictions since the method uses an isotropic weight function and the data-independent error bounds are not close (tight) to the actual errors, which leads to incorrectly chosen bandwidths. The method also suffered from long runtimes (which can be seen in Tables 3–4) because of the computations required at each step of the line search. RODEO suffered from bad predictions in all cases, possibly because the assumption of sparsity is not met, and RODEO is designed for problems when the number of relevant variables is sparse. Furthermore, RODEO assumes a homogeneous variance throughout the design space, and this assumption is not met for the M/G/1 queue since the variance is heterogeneous throughout the design space.

The global metamodeling methods include CART, stochastic kriging, and WLS. In 25 and 75 dimensions, the **rpart** package reaches the maximum tree depth, which results in poor prediction of the response surface. Stochastic kriging cannot be used when the number of design points is large, since the variance-covariance matrix is $n \times n$, and the inversion of the variance-covariance matrix is $O(n^3)$; this inversion causes **mlegp** to run out of memory. From Tables 1–2, as well as Tables 5–7, we can see that a significant improvement over WLS can be obtained when we localize the prediction using MLS. Although a large number of design points is needed for the bandwidths to remain local in higher dimensions, our method still produces results that are superior to WLS because MLS assigns different weight to each design point depending on the particular prediction point. Therefore, even though design points that fall in the prediction window may be 'far' away, they can still be assigned a very small weight.

7.3.1 Comparison of Runtimes

For a comparison of runtimes, Tables 3–4 give an overview of the average runtimes of the MLS procedure and the alternative methods that we use for comparison. Table 3 gives the average runtime during setup; for CART, this includes building the regression tree; for stochastic kriging, this includes estimation of the parameters and inverting the covariance matrix; for WLS, this

Table 3: Average runtime (across all experiments with same dimension and number of design points) during setup, for the multi-product M/G/1 queue example.

d	n	runtime					
		MLS	vanilla MLS	CART	RODEO	SK	WLS
5	500	∅	∅	46.3 sec	∅	14.3 min	0.26 min
	10000	∅	∅	1.5 min	∅	∅	1.2 min
	50000	∅	∅	2.1 min	∅	∅	1.7 min
25	5000	∅	∅	1.7 min	∅	∅	0.9 min
	50000	∅	∅	3.4 min	∅	∅	2.3 min
	100000	∅	∅	4.1 min	∅	∅	5.5 min
75	150000	∅	∅	4.9 min	∅	∅	∅
	200000	∅	∅	5.23 min	∅	∅	∅
	250000	∅	∅	5.98 min	∅	∅	∅

Table 4: Average runtime (across all experiments with same dimension and number of design points) for one prediction, for the multi-product M/G/1 queue example.

d	n	runtime					
		MLS	vanilla MLS	CART	RODEO	SK	WLS
5	500	1.03 sec	>2 min	0.11 sec	>2 min	0.2 sec	0.03 sec
	10000	1.47 sec	>2 min	0.21 sec	>2 min	∅	0.03 sec
	50000	2.63 sec	>2 min	0.48 sec	>2 min	∅	0.03 sec
25	5000	3.93 sec	>2 min	0.69 sec	>2 min	∅	0.09 sec
	50000	4.5 sec	>2 min	0.78 sec	>2 min	∅	0.09 sec
	100000	8.9 sec	>2 min	1.32 sec	>2 min	∅	0.09 sec
75	150000	10.3 sec	>2 min	2.03 sec	>2 min	∅	∅
	200000	12.4 sec	>2 min	2.32 sec	>2 min	∅	∅
	250000	15.9 sec	>2 min	2.68 sec	>2 min	∅	∅

includes estimating the regression coefficients. There is no setup for our MLS procedure, the vanilla MLS procedure, and RODEO, so the corresponding table entries have an entry of ∅. Table 4 gives the average runtime for one prediction, given that the metamodels for CART, stochastic kriging, and WLS have already been built. The majority of time in the MLS procedure was spent on estimation of the second partial derivatives and sorting the data matrix in high dimensions.

7.3.2 Procedure using Actual Second Derivative Values

Although the procedure can handle many more design points than the number used to calculate the values in Tables 1–2, there was not much observed decrease in the RERMSE when more design points were used. One possible explanation is that the estimated second partial derivatives tended to be larger than the true second partial derivatives. These larger estimates make our procedure choose smaller prediction windows than is actually optimal, hence limiting the smoothing capability of the procedure and resulting in limited improvement in RERMSE as the number of design points increases. Tables 5–7 show the results of experiments when both the estimated and actual second derivative values were used in our MLS procedure, along with the other methods we use for comparison. We use the name "MLS-deriv" to refer to our MLS procedure when the actual second derivative values are used. From Tables 5–7, we can see that there is significant improvement

Table 5: Relative difference for the 5 dimensional M/G/1 queue example.

RSD	n	relative difference						
		MLS	MLS-deriv	vanilla MLS	CART	RODEO	SK	WLS
0.05	500	-58%	-53%	-14%	-42%	>0%	-49%	>0%
	10000	-64%	-84%	-28%	-63%	>0%	∅	>0%
	50000	-72%	-91%	-37%	-75%	>0%	∅	>0%
0.1	500	-62%	-59%	-20%	-45%	>0%	-53%	-12%
	10000	-70%	-86%	-33%	-80%	>0%	∅	-12%
	50000	-74%	-92%	-46%	-85%	>0%	∅	-14%
0.25	500	-67%	-64%	-26%	-53%	>0%	-63%	-62%
	10000	-71%	-89%	-39%	-84%	>0%	∅	-63%
	50000	-78%	-94%	-52%	-86%	>0%	∅	-65%

Table 6: Relative difference for the 25 dimensional M/G/1 queue example.

RSD	n	relative difference						
		MLS	MLS-deriv	vanilla MLS	CART	RODEO	SK	WLS
0.05	5000	-48%	-60%	>0%	>0%	>0%	∅	>0%
	50000	-51%	-82%	>0%	>0%	>0%	∅	>0%
	100000	-56%	-89%	>0%	>0%	>0%	∅	>0%
0.1	5000	-52%	-66%	>0%	>0%	>0%	∅	-10%
	50000	-57%	-85%	>0%	>0%	>0%	∅	-11%
	100000	-61%	-91%	>0%	>0%	>0%	∅	-14%
0.25	5000	-59%	-73%	>0%	>0%	>0%	∅	-62%
	50000	-67%	-86%	>0%	>0%	>0%	∅	-64%
	100000	-72%	-92%	>0%	>0%	>0%	∅	-63%

in the prediction ability of our MLS procedure when the actual second derivative values are used in the procedure. As mentioned before, a table entry of ∅ means that the corresponding R package used to fit the model ran out of memory. We can see that the vanilla MLS, CART, RODEO, stochastic kriging, and WLS all encounter problems when they are implemented with a large number of design points. Although the prediction ability is significantly improved when we use the actual second derivative values, our MLS procedure (with estimated second derivatives) resulted in better predictions over every other method to which we compared, except for CART in the 5-dimensional M/G/1 queue example, and WLS in the 25-dimensional M/G/1 queue example when the relative standard deviation is set at 0.25.

8 Conclusion and Future Research

In this paper, we introduced a procedure to implement a local smoothing method called MLS regression in high-dimensional stochastic simulation metamodeling problems. Our procedure accounts for the fact that we can make replications and control the placement of design points in stochastic simulation. Furthermore, we provided a bound on the expected approximation error and showed that the MLS predictor is consistent under certain conditions. Lastly, we tested the procedure on two examples that demonstrated better results than other existing simulation metamodeling techniques. Since the performance of our procedure was improved significantly when we used the

Table 7: Relative difference for the 75 dimensional M/G/1 queue example.

RSD	n	relative difference						
		MLS	MLS-deriv	vanilla MLS	CART	RODEO	SK	WLS
0.05	150000	-40%	-53%	>0%	>0%	>0%	∅	∅
	200000	-44%	-62%	>0%	>0%	>0%	∅	∅
	250000	-53%	-71%	>0%	>0%	>0%	∅	∅
0.1	150000	-42%	-58%	>0%	>0%	>0%	∅	∅
	200000	-47%	-72%	>0%	>0%	>0%	∅	∅
	250000	-52%	-76%	>0%	>0%	>0%	∅	∅
0.25	150000	-53%	-63%	>0%	>0%	>0%	∅	∅
	200000	-59%	-76%	>0%	>0%	>0%	∅	∅
	250000	-65%	-80%	>0%	>0%	>0%	∅	∅

true values of the second partial derivatives, obtaining better second partial derivative estimates is a subject of future research.

Acknowledgements

This paper is based upon work supported by the National Science Foundation under Grant No. CMMI-0900354. Portions of this paper were published in Salemi et al. [2012].

A Appendix

Lemma A.1. *Consider optimization problems MP*(1) *and MP*(2)*. Denote the optimal solution to MP*(1) *by* $h_{i,1}$*, for* $i = 1, 2, \ldots, d$*, and the optimal solution to MP*(2) *by* $h_{i,2}$*, for* $i = 1, 2, \ldots, d$*. Then,* $h_{i,1} \leq h_{i,2}$*, for* $i = 1, 2, \ldots, d$*.*

Proof. Let $\mathcal{D}_i = \widehat{\mathcal{D}_i(\mathbf{x}_0)}$, for $i = 1, 2, \ldots, d$, and $s^2 = s^2(\mathbf{x}_0)$. The objective function of MP(1) is strictly convex and the constraints are affine, so any feasible solution that satisfies the Karush-Kuhn-Tucker (KKT) conditions is a unique global optimum. Without loss of generality, assume that $\mathcal{D}_1 \leq \mathcal{D}_2 \leq \cdots \leq \mathcal{D}_d$. The optimal solution to MP(1) is of the form, $h_{1,1} = f_n/2, h_{2,1} = f_n/2, \ldots, h_{d',1} = f_n/2, h_{d'+1,1} < f_n/2, \ldots, h_{d,1} < f_n/2$, for some $0 \leq d' \leq d$. This is because if there exists i, j with $i - j > 1$ such that $h_{i,1} = f_n/2$ and $h_{j,1} < f_n/2$, then the objective function can be decreased by swapping the values of the bandwidths, which is a feasible solution. The corresponding KKT conditions for MP(1) are

$$(h_{i,1})^2 \mathcal{D}_i \mu_2(\mathrm{K})^2 \left(\sum_{i=1}^{d} (h_{i,1})^2 \mathcal{D}_i \right) - \frac{R(\mathrm{K}) s^2}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^{d} h_{i,1}} \leq 0,$$

for $i = 1, 2, \ldots, d'$, and

$$(h_{i,1})^2 \mathcal{D}_i \mu_2(\mathrm{K})^2 \left(\sum_{i=1}^{d} (h_{i,1})^2 \mathcal{D}_i \right) - \frac{R(\mathrm{K}) s^2}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^{d} h_{i,1}} = 0$$

for $i = d' + 1, d' + 2, \ldots, d$. Using the last $d - d'$ equations from the KKT conditions, we can see that the free variables are of the form $h_{i,1} = k_1(1/\sqrt{\mathcal{D}_i})$ for some constant k_1. Thus, the optimal solution to MP(1) can be written in the form $h_{i,1} = \min\{f_n/2, k_1(1/\sqrt{\mathcal{D}_1})\}$ for $i = 1, 2, \ldots, d$.

Similarly, the objective function of MP(2) is strictly convex and the constraints are either affine or quasi-convex, so any feasible solution that satisfies the KKT conditions is a unique global optimum. Using the same arguments as in the case of MP(2), it can be shown that the optimal solution to MP(2) is of the form $h_{i,2} = \min\{f_n/2, k_2(1/\sqrt{\mathcal{D}_1})\}$, for $i = 1, 2, \ldots, d$, for some constant k_2. However, the KKT condition for a variable that does not hit its upper bound, $f_n/2$, is

$$(h_{i,2})^2 \mathcal{D}_i \mu_2(\mathrm{K})^2 \left(\sum_{i=1}^{d} (h_{i,2})^2 \mathcal{D}_i \right) - \frac{R(\mathrm{K}) s^2 + \lambda n g(x_0) C_n \tilde{g}(x_0) 2^d (\prod_{i=1}^{d} h_{i,2})^2}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^{d} h_{i,2}} = 0,$$

where $\lambda > 0$. Since $\lambda n g(x_0) C_n \tilde{g}(x_0) 2^d (\prod_{i=1}^{d} h_{i,2})^2 > 0$, we have that $k_1 \leq k_2$. □

Bandwidth Procedure
Input: $\delta, B = \mathrm{Mass_{UB}}$ or ∞. Output: $h_1^{l\,*}, h_1^{r*}, h_2^{l\,*}, h_2^{r*}, \ldots, h_d^{l\,*}, h_d^{r*}$
Perform a line search over the interval $[\dim(\Pi_1^d) + \delta, B]$, using the Golden Search Method [Bazaraa et al., 2006]. For each $i \in [\dim(\Pi_1^d) + \delta, B]$, the value $q(i)$ used in the line search is the optimal value of the optimization problem

$$\begin{aligned}
\min_{\{h_1,\ldots,h_d\}} \quad & \frac{1}{4}\mu_2(\mathrm{K})^2 \left(\sum_{i=1}^{d} h_i^2 \left| \widehat{\mathcal{D}_i(\mathbf{x}_0)} \right| \right)^2 + \frac{R(\mathrm{K}) s^2(\mathbf{x}_0)}{C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^{d} h_i} \\
\text{s.t.} \quad & n g(\mathbf{x}_0) \prod_{i=1}^{d} (h_i^l + h_i^r) = Q \\
& 2h_i = h_i^l + h_i^r \quad \text{for } i = 1, 2, \ldots, d \\
& 0 \leq h_i^l \leq x_{0,i}, \quad \text{for } i = 1, 2, \ldots, d \\
& 0 \leq h_i^r \leq 1 - x_{0,i}, \quad \text{for } i = 1, 2, \ldots, d \\
& h_i^l + h_i^r \leq f_n, \quad \text{for } i = 1, 2, \ldots, d.
\end{aligned}$$

This optimization problem can be solved using the **Inner Procedure** below, with $\Phi = Q$. This procedure is based on a variation of the variable pegging procedure presented in Bitran and Hax [1981]. Denote the optimal solution to the line search by i^* and let the corresponding optimal solution to the associated optimization problem be denoted by $h_1^{l\,*}, h_1^{r*}, h_2^{l\,*}, h_2^{r*}, \ldots, h_d^{l\,*}, h_d^{r*}$. This solution is optimal for MP(1) or MP(2).

Inner Procedure
Input: Φ. Output: $h_1^{l\,*}, h_1^{r*}, h_2^{l\,*}, h_2^{r*}, \ldots, h_d^{l\,*}, h_d^{r*}$

0. Initialize $\mathbb{J}^1 = \{1, \ldots, d\}$, $\mathbf{P}^1 = \ln\left(\frac{\Phi}{n g(\mathbf{x}_0) 2^d}\right)$, and Iteration $\beta = 1$.

1. For all $j \in \mathbb{J}^\beta$, set $h_j^\beta = \frac{1}{|\mathbb{J}^\beta|}\mathbf{P}^\beta - \frac{1}{2}\ln\left(\left|\widehat{\mathcal{D}_j(\mathbf{x}_0)}\right|\right) + \frac{1}{2|\mathbb{J}^\beta|}\sum_{k\in\mathbb{J}^\beta} \ln\left(\left|\widehat{\mathcal{D}_k(\mathbf{x}_0)}\right|\right)$. If $h_j^\beta \leq \ln(\min\{1/2, f_n/2\})$ for all $j \in \mathbb{J}^\beta$, set $h_j^* = h_j^\beta$ for all $j \in \mathbb{J}^\beta$, and go to 3. Otherwise go to 2.

2. Let $\mathbb{J}_+^\beta = \{j \in \mathbb{J}^\beta : h_j^\beta \geq \ln(\min\{1/2, f_n/2\})\}$. Define $h_j^* \triangleq \ln(\min\{1/2, f_n/2\}), \forall j \in \mathbb{J}_+^\beta$ and let $\mathbb{J}^{\beta+1} = \mathbb{J}^\beta \setminus \mathbb{J}_+^\beta$, $\mathbf{P}^{\beta+1} = \mathbf{P}^\beta - |\mathbb{J}_+^\beta| \ln(\min\{1/2, f_n/2\})$. If $\mathbb{J}^{\beta+1} = \varnothing$ go to 3. Else, $\beta \leftarrow \beta + 1$ and go to 1.

3. For all $i = 1, \ldots, d$: Set $h_i^* \leftarrow e^{h_i^*}$. If $h_i^* \leq \min\{x_{0,i}, 1 - x_{0,i}, f_n/2\}$, set $h_i^{l*} = h_i^{r*} = h_i^*$. Else, if $x_{0,i} \leq 1 - x_{0,i}$ set $h_i^{l*} = \min\{x_{0,i}, f_n/2\}$ and $h_i^{r*} = 2h_i^* - \min\{x_{0,i}, f_n/2\}$. Else, set $h_i^{r*} = \min\{1 - x_{0,i}, f_n/2\}$ and $h_i^{l*} = 2h_i^* - \min\{1 - x_{0,i}, f_n/2\}$.

Nearest-Neighbor Procedure
Search over the grid $\Delta = [7d, 8d, \ldots, \min\{20d, \lfloor n/d \rfloor\}]$. For each $k \in \Delta$, the value that is used in the line search is $R^2(k) \triangleq 1 - \frac{\mathbf{Y}_k^\top (\mathbf{I}_{k\times k} - \mathbf{X}_k(\mathbf{X}_k^\top \mathbf{X}_k)^{-1}\mathbf{X}_k^\top)\mathbf{Y}_k}{\mathbf{Y}_k^\top (\mathbf{I}_{k\times k} - \frac{1}{k}\mathbf{J}_{k\times k})\mathbf{Y}_k}$, where $\mathbf{Y}_k$ and $\mathbf{X}_k$ is the vector of observations and the regression matrix of the k nearest neighbors to the prediction point, respectively, and $\mathbf{I}_{k\times k}$ is the $k \times k$ identity matrix, and $\mathbf{J}_{k\times k}$ is the $k \times k$ matrix of ones. Choose the k that maximizes $R^2(k)$.

Proof of Theorem 5.1

Proof. Recall from Section 3.1 that

$$\hat{y}^{\text{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}_0) = \mathbf{P}(\mathbf{x}_0)^\top (\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{Y},$$

where the ith entry of $\mathbf{Y}$ is $y(\mathbf{x}_i) + e_i$, and e_i is the error associated with the simulation output at the ith design point. Thus, we can write $\mathbf{Y}$ as $\mathbf{y} + \mathbf{e}$, where $\mathbf{y} = (y(\mathbf{x}_1), y(\mathbf{x}_2), \ldots, y(\mathbf{x}_n))^\top$, and $\mathbf{e} = (e_1, e_2, \ldots, e_n)^\top$. Furthermore, since $\mathbf{P}(\mathbf{x}_0) = (1, 0, \ldots, 0)^\top$ we have $\hat{y}^{\text{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}_0) = c_1$, where $\mathbf{c}$ satisfies the normal equations

$$\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P}\mathbf{c} = \mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)(\mathbf{y} + \mathbf{e}).$$

Since $y \in C^2(\mathbb{X})$, we can use the second-order Taylor expansion for y to express $\mathbf{y}$ as

$$\mathbf{y} = y(\mathbf{x}_0)[\mathbf{P}]_1 + \sum_{i=1}^{d} \frac{\partial y(\mathbf{x}_0)}{\partial x_i}[\mathbf{P}]_{i+1} + \frac{1}{2}\sum_{|\mathbf{v}|=2} \mathbf{Q}_\mathbf{v}\mathbf{E}_\mathbf{v},$$

where $[\mathbf{P}]_k$ is the kth column of $\mathbf{P}$, $\mathbf{Q}_\mathbf{v} = \text{diag}\left\{\frac{\partial^{|\mathbf{v}|} y(\eta_1(\mathbf{x}_1 - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^\mathbf{v}}, \ldots, \frac{\partial^{|\mathbf{v}|} y(\eta_n(\mathbf{x}_n - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^\mathbf{v}}\right\}$, η_i is a scalar with $0 \le \eta_i \le 1$ for $i = 1, 2, \ldots, n$, and $\mathbf{E}_\mathbf{v}$ is an $n \times 1$ vector with ith entry $(\mathbf{x}_i - \mathbf{x}_0)^\mathbf{v}$. Substituting this representation into the normal equations and solving for $\mathbf{c}$, we get

$$\mathbf{c} = (\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\left(y(\mathbf{x}_0)[\mathbf{P}]_1 + \sum_{i=1}^{d} \frac{\partial y(\mathbf{x}_0)}{\partial x_i}[\mathbf{P}]_{i+1} + \frac{1}{2}\sum_{|\mathbf{v}|=2} \mathbf{Q}_\mathbf{v}\mathbf{E}_\mathbf{v} + \mathbf{e}\right).$$

Therefore,

$$\begin{aligned} c_1 &= y(\mathbf{x}_0) + \frac{1}{2}\sum_{|\mathbf{v}|=2}\left((\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{Q}_\mathbf{v}\mathbf{E}_\mathbf{v}\right)_1 + \left((\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{e}\right)_1 \\ &= y(\mathbf{x}_0) + \frac{1}{2}\sum_{|\mathbf{v}|=2}\sum_{i=1}^{n} \frac{\partial^{|\mathbf{v}|} y(\eta_i(\mathbf{x}_i - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^\mathbf{v}}(\mathbf{x}_i - \mathbf{x}_0)^\mathbf{v}\left((\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}[\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)]_i\right)_1 \\ &\qquad + \sum_{i=1}^{n} e_i\left((\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}[\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)]_i\right)_1. \end{aligned}$$

Using Cramer's rule, we have $\left((\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})^{-1}[\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)]_i\right)_1 = \frac{\det(\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P}^i))}{\det(\mathbf{P}^\top \mathbf{W}(\mathbf{x}_0)\mathbf{P})} \triangleq \Xi_i$ (see, for example, Lipman et al. [2006]), where $\mathbf{P}^i$ is the matrix $\mathbf{P}$ with the first column replaced with the ith standard basis vector. Therefore, we have

$$\hat{y}^{\text{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}_0) - y(\mathbf{x}_0) = \frac{1}{2}\sum_{|\mathbf{v}|=2}\sum_{i=1}^{n} \frac{\partial^{|\mathbf{v}|} y(\eta_i(\mathbf{x}_i - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^\mathbf{v}}(\mathbf{x}_i - \mathbf{x}_0)^\mathbf{v}\Xi_i + \sum_{i=1}^{n} e_i\Xi_i.$$

Using the Cauchy-Schwarz inequality, we have the bound

$$(\hat{y}^{\text{MLS}}_{\mathbf{x}_0,\mathbf{H}}(\mathbf{x}_0) - y(\mathbf{x}_0))^2 \leq \left(\frac{1}{2} \sum_{|\mathbf{v}|=2} \sum_{i=1}^{n} \frac{\partial^{|\mathbf{v}|} y(\eta_i(\mathbf{x}_i - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^{\mathbf{v}}} (\mathbf{x}_i - \mathbf{x}_0)^{\mathbf{v}} \Xi_i \right)^2 + \sum_{i=1}^{n} e_i^2 \Xi_i^2$$
$$\leq \left(\frac{1}{2} \sum_{|\mathbf{v}|=2} \sum_{i=1}^{n} C_i^{\mathbf{v}} |\mathbf{x}_i - \mathbf{x}_0|^{\mathbf{v}} |\Xi_i| \right)^2 + \sum_{i=1}^{n} e_i^2 \Xi_i^2,$$

where $C_i^{\mathbf{v}} = \sup_{0 \leq \eta \leq 1} \left| \frac{\partial^{|\mathbf{v}|} y(\eta(\mathbf{x}_i - \mathbf{x}_0) + \mathbf{x}_0)}{\partial \mathbf{x}^{\mathbf{v}}} \right|$. The result follows since $\text{E}[e_i^2] = \sigma_i^2 / R_i$. □

Proof of Theorem 5.2

Proof. Condition 1 ensures that MP(1) will have an optimal solution for large enough n, almost surely. Let $h_1^*(n), h_2^*(n), \ldots, h_d^*(n)$ denote the optimal solution to MP(1). Since $f_n \to 0$ as $n \to \infty$, $h_i^*(n) \to 0$ as $n \to \infty$ for $i = 1, 2, \ldots, d$, we have $n^{-1}|\mathbf{H}_{l,r}^*|$ and each entry of the bandwidth matrix tends to zero as $n \to \infty$. Let $L_{max}(n)$ and $L_{min}(n)$ denote the maximum and minimum eigenvalue of the bandwidth matrix for the nth design. Since the bandwidth matrix is diagonal, the eigenvalues are just the bandwidth parameters. The optimal solution satisfies $\dim(\Pi_1^d) + \delta \leq n g(\mathbf{x}_0) \prod_{i=1}^d h_i^*(n)$, so $\frac{\dim(\Pi_1^d)+\delta}{n g(\mathbf{x}_0) f_n^{d-1}} \leq L_{min}(n)$. Thus,

$$\frac{L_{max}(n)}{L_{min}(n)} \leq \frac{n f_n^d g(\mathbf{x}_0)}{2(\dim(\Pi_1^d) + \delta)} = \frac{M}{2(\dim(\Pi_1^d) + \delta)}.$$

Therefore, all of the conditions in Assumption 2 are satisfied, so by Theorem 2.1 of Ruppert and Wand [1994], Equations (2) and (3) are the conditional bias and variance of $\hat{y}^{\text{LOC}}(\mathbf{x}_0; \mathbf{H}_{l,r}^*)$, respectively. From the constraint $\dim(\Pi_1^d) + \delta \leq n g(\mathbf{x}_0) \prod_{i=1}^d h_i(n)$, the solution $h_i^*(n)$, for $i = 1, 2, \ldots, d$, satisfies $\dim(\Pi_1^d) + \delta \leq n g(\mathbf{x}_0) \prod_{i=1}^d h_i^*(n)$. Thus,

$$C_n \tilde{g}(\mathbf{x}_0) \frac{\dim(\Pi_1^d) + \delta}{n g(\mathbf{x}_0)} \leq C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^{d} h_i^*(n).$$

Since $C_n / n \to \infty$ as $n \to \infty$, $C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^d h_i^*(n) \to \infty$ as $n \to \infty$. From the conditions $C_n \tilde{g}(\mathbf{x}_0) \prod_{i=1}^d h_1^*(n) \to \infty$ and $h_i^*(n) \to 0$, $\forall i$, as $n \to \infty$,

$$\text{MSE}\{\hat{y}^{\text{LOC}}(\mathbf{x}_0; \mathbf{H}_{l,r}^*) | \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \xrightarrow{p} 0 \quad \text{as} \quad n \to \infty.$$

Therefore, the estimator $\hat{y}^{\text{LOC}}(\mathbf{x}_0; \mathbf{H}_{l,r}^*)$ is consistent. □

Proof of Theorem 5.3

Proof. Condition 1 ensures that MP(1) will have an optimal solution for large enough n, almost surely. Let $h_1^*(n), h_2^*(n), \ldots, h_d^*(n)$ denote the optimal solution to MP(1). Since $f_n \to 0$ as $n \to \infty$, $h_i^*(n) \to 0$ as $n \to \infty$ for $i = 1, 2, \ldots, d$, we have $n^{-1}|\mathbf{H}_{l,r}^*|$ and each entry of the bandwidth matrix tends to zero as $n \to \infty$. Let $L_{max}(n)$ and $L_{min}(n)$ denote the maximum and minimum eigenvalue of the bandwidth matrix for the nth design. Since the bandwidth matrix is diagonal, the eigenvalues are just the bandwidth parameters. We must show that there exists a constant L such that $\frac{L_{max}(n)}{L_{min}(n)} \leq L$ for $n \geq 1$. Since $P(\limsup_{n \to \infty} \mathcal{D}_i(n) < \infty) = 1$ for $i = 1, 2, \ldots, d$, for almost

every sample path there exists an $n_{1,i} < \infty$ and $\Delta_i < \infty$ such that $\mathcal{D}_i(n) < \Delta_i$ for $n > n_{1,i}$ and $i = 1, 2, \ldots, d$. Similarly, since $P(\liminf_{n\to\infty} s^2(n) > 0) = 1$, for almost every sample path there exists an $n_2 < \infty$ and $m > 0$ such that $s^2(n) > m \; \forall n > n_2$. Let $\Delta = \max\{\Delta_1, \ldots, \Delta_d\}$. Denote the optimal solution to MP(1) with the constraint $\dim(\Pi_1^d) + \delta \leq ng(\mathbf{x}_0) \prod_{i=1}^d (h_i^l(n) + h_i^r(n))$ removed by $\hat{h}_1^*(n), \hat{h}_2^*(n), \ldots, \hat{h}_d^*(n)$. Assume without loss of generality that $d' = d'(n)$ variables hit their upper bound, and let $\mathbb{H}^+ = \{1 \leq i \leq d | \hat{h}_i^*(n) = f_n/2\}$ and $\mathbb{H}^- = \{1 \leq i \leq d | \hat{h}_i^*(n) < f_n/2\}$. The Karush-Kuhn-Tucker (KKT) conditions for the variables $\hat{h}_i^*(n)$, for $i \in \mathbb{H}^-$, are

$$(\hat{h}_i^*(n))^2 \mathcal{D}_i(n) \mu_2(\mathrm{K})^2 \left(\sum_{i \in \mathbb{H}^+} \left(\frac{f_n}{2} \right)^2 \mathcal{D}_i(n) + \sum_{i \in \mathbb{H}^-} \hat{h}_i^*(n)^2 \mathcal{D}_i(n) \right) - \frac{R(\mathrm{K}) s^2(n)}{C_n \tilde{g}(\mathbf{x}_0) \left(\frac{f_n}{2} \right)^{d'} \prod\limits_{i \in \mathbb{H}^-} \hat{h}_i^*(n)} = 0.$$

Rearranging these equations, we get the implicit solution

$$\hat{h}_i^*(n) = \left(\frac{\hat{h}_i^*(n)^{d'-d} \mathcal{D}_i(n)^{\frac{d'-d-2}{2}} R(\mathrm{K}) s^2(n) \sqrt{\prod\limits_{i \in \mathbb{H}^-} \mathcal{D}_i(n)}}{C_n \tilde{g}(\mathbf{x}_0) \left(\frac{f_n}{2} \right)^{d'} \mu_2(\mathrm{K})^2 \left[(d-d') \mathcal{D}_i(n) \hat{h}_i^*(n)^2 + \left(\frac{f_n}{2} \right)^2 \sum\limits_{i \in \mathbb{H}^-} \mathcal{D}_k(n) \right]} \right)^{\frac{1}{2}}, \tag{6}$$

for $i \in \mathbb{H}^-$. From the condition $C_n = O(n)$, there exists an $n_3 < \infty$ and $\Theta < \infty$ such that $C_n \leq \Theta n$ $\forall n > n_3$. Let $n_0 = \max\{n_{1,1}, n_{2,1} \ldots, n_{1,d}, n_2, n_3\}$. Using (6) and the fact that $\hat{h}_i^*(n) \leq f_n/2$ for $i = 1, 2, \ldots, d$, we can get a lower bound on $\hat{h}_i^*(n)$,

$$\hat{h}_i^*(n) \geq \tilde{h}_i^*(n) \triangleq \left(\frac{R(\mathrm{K}) m \sqrt{\prod\limits_{i \in \mathbb{H}^-} \mathcal{D}_i(n)}}{\Theta n \tilde{g}(\mathbf{x}_0) \left(\frac{f_n}{2} \right)^{d'+2} d M^{\frac{d+4}{2}} \mu_2(\mathrm{K})^2} \right)^{\frac{1}{d-d'+2}},$$

for $i \in \mathbb{H}^-$ and $n > n_0$. From the KKT conditions, we can get a lower bound on the second partial derivatives associated with the bandwidths that do not hit their upper bounds. Indeed, for $n > n_0$,

$$\mathcal{D}_i(n) \geq \frac{R(\mathrm{K}) m}{\Theta n \tilde{g}(\mathbf{x}_0) \mu_2(\mathrm{K})^2 d M \left(\frac{f_n}{2} \right)^{d+2}}.$$

Substituting this lower bound for $\mathcal{D}_i(n)$ into $\tilde{h}_i^*(n)$, we can get a lower bound on $\tilde{h}_i^*(n)$, which we denote by $\bar{h}_i^*(n)$. For $n > n_0$,

$$\hat{h}_i^*(n) \geq \tilde{h}_i^*(n) \geq \bar{h}_i^*(n) \propto n^{-\frac{d-d'+2}{2(d-d'+2)}} f_n^{-\frac{2(d'+2)+(d+2)(d-d')}{2(d-d'+2)}}.$$

From Lemma A.1, $h_i^*(n) \geq \bar{h}_i^*(n)$, for $i = 1, 2, \ldots, d$, and $n > n_0$. Therefore, for $n > n_0$,

$$\frac{L_{max}(n)}{L_{min}(n)} \leq \frac{f_n}{2\bar{h}_i^*(n)} \propto f_n n^{\frac{d-d'+2}{2(d-d'+2)}} f_n^{\frac{2(d'+2)+(d+2)(d-d')}{2(d-d'+2)}} \propto n^{\frac{-(d-d')-6}{2(d+1)(d-d'+2)}}.$$

Since $n^{\frac{-(d-d')-6}{2(d+1)(d-d'+2)}} \to 0$ as $n \to \infty$, there exists an L such that $\frac{L_{max}(n)}{L_{min}(n)} \leq L$. Thus, all of the conditions in Assumption 2 are satisfied, so by Theorem 2.1 of Ruppert and Wand [1994],

Equations (2) and (3) are the conditional bias and variance of $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}^*_{l,r})$, respectively. We now show that $C_n\tilde{g}(x_0)\prod_{i=1}^d h_i^*(n) \to \infty$ as $n \to \infty$. For $n > n_0$,

$$\begin{aligned} C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d \hat{h}_i^*(n) &\geq C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d \bar{h}_i^*(n) \\ &\geq n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d \bar{h}_i^*(n) \\ &\propto n^{\frac{4-(d-d')^2}{2(d-d'+2)}} f_n^{\frac{8d'-4d-(d+2)(d-d')^2}{2(d-d'+2)}} \\ &= n^{\frac{(d-d')^2+8(d-d')+4}{2(d+1)(d-d'+2)}} \to \infty \quad \text{as} \quad n \to \infty. \end{aligned}$$

From the inequality $\prod_{i=1}^d h_i^*(n) \geq \prod_{i=1}^d \hat{h}_i^*(n)$, the limit $C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d \hat{h}_i^*(n) \to \infty$ implies that $C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d h_i^*(n) \to \infty$, as $n \to \infty$. Because $C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d h_i^*(n) \to \infty$ and $h_i^*(n) \to 0$ for $i = 1, 2, \ldots, d$, as $n \to \infty$,

$$\mathrm{MSE}\{\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}^*_{l,r}) | \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \xrightarrow{p} 0 \quad \text{as} \quad n \to \infty.$$

Therefore, the estimator $\hat{y}^{\mathrm{LOC}}(\mathbf{x}_0; \mathbf{H}^*_{l,r})$ is consistent. □

Proof of Theorem 5.4

Proof. Condition 1 ensures that MP(1) will have an optimal solution for large enough n, almost surely. Let $h_1^*(n), h_2^*(n), \ldots, h_d^*(n)$ denote the optimal solution to MP(2). Since the optimal solution is a feasible solution, it must satisfy the constraint $\dim(\Pi_1^d) + \delta \leq ng(\mathbf{x}_0)\prod_{i=1}^d h_i(n) \leq \mathrm{Mass}_{\mathrm{UB}}$. Thus,

$$C_n\tilde{g}(\mathbf{x}_0)\frac{\dim(\Pi_1^d)+\delta}{ng(\mathbf{x}_0)} \leq C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d h_i^*(n) \leq C_n\tilde{g}(\mathbf{x}_0)\frac{\mathrm{Mass}_{\mathrm{UB}}}{ng(\mathbf{x}_0)}.$$

From the last inequality, we can see that $C_n/n \to \infty$ as $n \to \infty$ is necessary and sufficient for $C_n\tilde{g}(\mathbf{x}_0)\prod_{i=1}^d h_i^*(n) \to \infty$ as $n \to \infty$. The rest of the proof is the same as the proof of Theorem 5.2. □

References

A. Adamson and M. Alexa. Anisotropic point set surfaces. In *Proceedings of AFRIGRAPH 2006 4th International Conference on Virtual Reality, Computer Graphics, Visualization and Interaction in Africa*, volume 4, pages 7–13, 2006.

N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

B. Ankenman, B. L. Nelson, and J. Staum. Stochastic kriging for simulation metamodeling. *Operations Research*, 58(2):371–382, 2010.

S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

M. Bazaraa, H. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley Interscience, 2006.

G. R. Bitran and A. C. Hax. Disaggregation and resource allocation using convex knapsack problems with bounded variables. *Management Science*, 27(4):431–441, 1981.

L. Bos and K. Salkauskas. Moving least-squares are Backus-Gilbert optimal. *Journal of Approximation Theory*, 59:267–275, 1989.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

W. S. Cleveland, S. J. Devlin, and E. Grosse. Regression by local fitting: Methods, properties, and computational algorithms. *Journal of Econometrics*, 37(1):87–114, 1988.

N. Cressie and G. Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society*, 70(1):209–226, 2008.

K. Doksum, D. Peterson, and A. Samarov. On variable bandwidth selection in local polynomial regression. *Journal of the Royal Statistical Society. Series B*, 62(3):431–448, 2000.

J. Fan and I. Gijbels. Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation. *Journal of the Royal Statistical Society. Series B*, 57(2):371–394, 1995.

R. B. Gramacy and H. K. H. Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, September 2008.

N. W. Hengartner, M. H. Wegkamp, and E Matzner-Lober. Bandwidth selection for local linear regression smoothers. *Journal of the Royal Statistical Society. Series B*, 64(4):791–804, 2002.

C. G. Kaufman, M. J. Schervish, and D. W. Nychka. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103(484): 1545–1555, 2008.

J. Lafferty and L. Wasserman. Rodeo: Sparse, greedy nonparametric regression. *The Annals of Statistics*, 36(1):28–63, 2008.

P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

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

Y. Lipman, D. Cohen-Or, and D. Levin. Error bounds and optimal neighborhoods for mls approximation. In K. Polthier and A. Sheffer, editors, *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, pages 71–80, 2006.

M. Liu and J. Staum. Stochastic kriging for efficient nested simulation of expected shortfall. *Journal of Risk*, 12(3):3–27, 2010.

C. Loader. *Local Regression and Likelihood*, volume 47 of *Statistics and Computing*. Springer, New York, 1999.

K. Prewitt and S. Lohr. Bandwidth selection in local polynomial regression using eigenvalues. *Journal of the Royal Statistical Society. Series B*, 68(1):135–154, 2006.

D. Ruppert. Empirical-bias bandwidths for local polynomial nonparametric regression and density estimation. *Journal of the American Statistical Association*, 92(439):1049–1062, September 1997.

D. Ruppert and M. P. Wand. Multivariate locally weighted least squares regression. *The Annals of Statistics*, 22(3):1346–1370, 1994.

D. Ruppert, S. J. Sheather, and M. P. Wand. An effective bandwidth selector for local least squares regression. *Journal of the American Statistical Association*, 90(432):1257–1270, December 1995a.

D. Ruppert, S. J. Sheather, and M. P. Wand. An effective bandwidth selector for local least squares regression. *Journal of the American Statistical Association*, 90(432):1257–1270, 1995b.

P. L. Salemi, B. L. Nelson, and J. Staum. Moving least squares regression for high dimensional simulation metamodeling. In C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, editors, *Proceedings of the 2012 Winter Simulation Conference*. IEEE, Piscataway, NJ, 2012.

S. Shan and G. Wang. Metamodeling for high dimensional simulation-based design problems. *Journal of Mechanical Design*, 132(5):1–11, 2010.

E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*. MIT Press, 2006.

I. M. Sobol. The distribution of points in a cube and the accurate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7:784–802, 1967.

M. H. Tongarlak, B. Ankenman, B. L. Nelson, L. Borne, and K. Wolfe. Using simulation early in the design of a fuel injector production line. *Interfaces*, 40(2):105–117, 2010.

S. Vijayakumar and S. Schaal. Locally weighted projection regression: An o(n) algorithm for incremental real-time learning in high dimensional space. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 288–293, 2000.

M. P. Wand and M. C. Jones. Comparison of smoothing parameterizations in bivariate kernel density estimation. *Journal of the American Statistical Association*, 88:520–528, 1993.

G. S. Watson. Smooth regression analysis. *The Indian Journal of Statistics, Series A*, 26(4): 359–372, 1964.

W. Whitt. Planning queueing simulations. *Management Science*, 35(11):1341–1366, 1989.

F. Yang, J. Liu, B. L. Nelson, B. E. Ankenman, and M. Tongarlak. Metamodeling for cycle time-thoughput-product mix surfaces using progressive model fitting. *Production Planning and Control*, 22(1):50–68, 2011.