SANDIA REPORT

SAND2013-7022
Unlimited Release
Printed August 2013

Efficient and Robust Gradient Enhanced Kriging Emulators

Keith R. Dalbey

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

SAND2013-7022
Unlimited Release
Printed August 2013

Efficient and Robust Gradient Enhanced Kriging Emulators

Keith R. Dalbey
Optimization and Uncertainty Quantification Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1318
kdalbey@sandia.gov

Abstract

"Naive" or straight-forward Kriging implementations can often perform poorly in practice. The relevant features of the robustly accurate and efficient Kriging and Gradient Enhanced Kriging (GEK) implementations in the DAKOTA software package are detailed herein. The principal contribution is a novel, effective, and efficient approach to handle ill-conditioning of GEK's "correlation" matrix, $\underline{\underline{R_\nabla}}$, based on a pivoted Cholesky factorization of *Kriging's* (not GEK's) correlation matrix, $\underline{\underline{R}}$, which is a small sub-matrix within GEK's $\underline{\underline{R_\nabla}}$ matrix. The approach discards sample points/equations that contribute the least "new" information to $\underline{\underline{R_\nabla}}$. Since these points contain the least new information, they are the ones which when discarded are both the easiest to predict and provide maximum improvement of $\underline{\underline{R_\nabla}}$'s conditioning. Prior to this work, handling ill-conditioned correlation matrices was a major, perhaps the principal, unsolved challenge necessary for robust and efficient GEK emulators.

Numerical results demonstrate that GEK predictions can be significantly more accurate when GEK is allowed to discard points by the presented method. Numerical results also indicate that GEK can be used to break the curse of dimensionality by exploiting inexpensive derivatives (such as those provided by automatic differentiation or adjoint techniques), smoothness in the response being modeled, and adaptive sampling. Development of a suitable adaptive sampling algorithm was beyond the scope of this work; instead adaptive sampling was approximated by omitting the cost of samples discarded by the presented pivoted Cholesky approach.

Acknowledgment

The author wishes to thank Laura Swiler and Brian Adams for help editing this report; David Day and Mark Hoemmen for advice on rank revealing pivoted Cholesky algorithms; Anthony O'Hagan for advice on how to appropriately utilize rejected samples as a validation set; and several individuals who have shared their experiences with gradient enhanced Kriging with this researcher in private communication.

Contents

Figures

Tables

1 Introduction

Computational models, or simulators, are frequently used to make predictions about the performance of physical systems. A simulator provides a mapping of inputs to outputs that may be used for design optimization, uncertainty quantification, or other purposes.

For typical engineering systems, the simulator, $f(\underline{x})$, has a large number of input variables or dimensions, $\underline{x}$, and a single evaluation of the simulator can expend a substantial amount of computational resources. The high computational cost of the simulator means that only a small number ($\mathscr{O}(10)$ to $\mathscr{O}(1000)$) of simulations can be performed. The so called "Curse of Dimensionality" [2] states that the number of samples or simulations, N, required for a given fidelity of representation of an unknown function is exponential in the number of dimensions, M,

$$N \propto b^M.$$

Uncertainty Quantification (UQ) techniques frequently make use of sampling. Evaluating a simulator at a set of random samples drawn from the distributions for the inputs is one of the oldest [13], most robust, and universally applicable UQ methods. Likewise, an optimization method can be described as systematic strategy to guess and check. That strategy may be as simple as evaluating a simulator at the N points in a random sample design and picking the best one. It could also be significantly more complex and involving local derivative information and/or global sampling. Furthermore, it is becoming increasingly common to optimize within the context of uncertainty, such as uncertain operating conditions and small variations in manufacturing.

The error in a mean computed by random sampling, for example Monte Carlo (MC) [22] or Latin Hypercube Sampling (LHS) [21, 15, 25], is nominally independent of the number of dimensions, M, and therefore not subject to the curse of dimensionality. In the context of either UQ or optimization of an integrated quantity, it is highly desirable for computational cost being independent of dimension. However, the error in a sample mean computed by MC (or LHS) scales as the true function's standard deviation divided by the square root of the number of samples, N,

$$\sigma_{error_{MC}} = \mathscr{O}\left(\frac{\sigma_f}{\sqrt{N}}\right).$$

So in general, one million simulations would be required for three significant figures of accuracy. This is usually far too expensive. Naturally, substantial work has been done to develop sampling methods with faster convergence in the error, such as Quasi Monte Carlo (QMC) [29, 23], and space-filling LHS [30, 26, 32, 4, 6, 7].

An alternative and complementary approach to sampling the simulator directly is to construct a surrogate model, also referred to as an emulator, from a small number of simulator evaluations and use the surrogate in place of the simulator to predict the physical system. The utility of a surrogate

is determined by the number of samples required to obtain a desired level of accuracy. Kriging models, also known as Gaussian process emulators [27, 24], are a popular type of surrogate model because they:

- Interpolate the data that they are built from as long as their correlation matrix, $\underline{\underline{R}}$, is real, symmetric, and positive definite
- Provide a spatially varying estimate of the variance of the error in their predictions, and
- Do not require a specific type of sample design.

However, the "small" number of simulations required to build the surrogate is, at least in principle, still subject to the curse of dimensionality. Some approaches to breaking the curse include:

1. **Exploiting derivative information** (e.g. gradients, Hessians) to lower the computational cost per piece of information. If evaluating the simulator produces function values plus gradients, then the number of samples required by the curse of dimensionality is reduced to
$$N \propto \frac{b^M}{1+M}.$$
2. **Exploiting assumed smoothness** of the unknown true function to accurately interpolate/extrapolate available information into regions with no sample points.
3. **Adaptive sampling**, e.g. exploiting knowledge gained from previous simulator runs to choose the next sample point or set of sample points so as to maximize the improvement in the emulator's predictive ability.

In this paper, the author details an implementation of a Gradient Enhanced (universal) Kriging (GEK) emulator [17] which combines these three approaches in an effort to break the Curse of Dimensionality. Special attention has been devoted to developing computationally efficient general purpose software to construct GEK emulators whose predictions are both accurate and robust. This implementation is freely available in version 5.1+ of the DAKOTA [1] software package with documentation forthcoming in the DAKOTA 5.2 user manual.

The first of these approaches, **exploiting derivative information**, is only attractive when the function value plus its derivatives can be evaluated at a computational cost that is comparable to the function value by itself. This rules out approximate gradients obtained via finite differences; also clusters of closely spaced samples tend to provide much less useful information than an equal number of well spaced points. Fortunately, accurate derivatives can be obtained cheaply through either automatic differentiation (AD) or adjoint techniques.

In this work, the term adjoint (also known as the "continuous adjoint") is used to refer to the numerical solution of the dual problem as opposed to reverse sensitivity analysis via AD (also

known as the "discrete adjoint"). A well known theoretical result in AD is that any number of first derivatives can be obtained for at most five times the cost of the function value [12], and in practice the cost is typically within a factor of two to three. The adjoint methodology can often provide derivatives at a substantially cheaper cost. Usually, evaluating the simulator requires the forward solution of a non-linear system of equations. With that forward solution in hand, the adjoint methodology can produce gradients for the additional cost of a single backward solve of the linearized dual equations.

The second option, **exploiting assumed smoothness**, is closely related. In Kriging models, the degree of assumed smoothness, s, is determined by the choice of correlation function, $r(\underline{x}, \underline{x}')$, between sample points $\underline{x}$ and $\underline{x}'$. Here s is the "regularity", i.e. the response being modeled is assumed to be s times differentiable. Employing gradient enhanced Kriging naturally restricts the choice of correlation functions to those that produce outputs which are one or more times differentiable, $s \geq 1$. Of these, the Matern $\nu = 3/2$ and squared-exponential correlation functions are among the more popular choices. The Matern $\nu = 3/2$ correlation function assumes that the output is once differentiable, $s = 1$. The squared-exponential, also known as Gaussian, correlation function assumes the output is infinitely differentiable, $s = \infty$. This tends to make the squared-exponential correlation function a good choice when there are few and well spaced data points. To break the curse of dimensionality, one must operate in the regime of sparse and well spaced data. However, the squared-exponential correlation function can cause the correlation matrix, $\underline{\underline{R}}$, to be numerically singular when the number of points, N, is very large, or the sample points are poorly spaced. $\underline{\underline{R}}$ contain evaluations of the correlation function at all pairwise combination of points in the sample design. Note that the Matern function converges to the squared exponential as ν approaches ∞; knowledge about the smoothness of the true response can be used to pick an appropriate intermediate value of ν.

For Kriging without gradient enhancement, an example of poorly spaced points is an otherwise sparse and uniformly distributed sample design that has tight clusters of points appropriate to computing finite differences. From this, one might correctly surmise that using analytical gradients

- Can have a similar detrimental effect on the condition number of the correlation matrix in gradient enhanced Kriging, and
- Greatly exasperates ill-conditioning problems due to poorly spaced samples.

One solution to these problems, albeit sometimes a poor one, is to decrease the correlation lengths, $\underline{L}$, until $\underline{\underline{R}}$ is non-singular. For example, one could explicitly constrain the condition number of $\underline{\underline{R}}$ (or LAPACK's fast estimate of its reciprocal) during the optimization process that selects correlation lengths.

Unfortunately, the sample design can be so poorly spaced that correlation lengths short enough to prevent ill-conditioning will render the Kriging model inept at extrapolating and even interpolating between distant points. This is particularly problematic for gradient enhanced Kriging. One

could switch to another less differentiable correlation function, such as Matern $\nu = 3/2$. However, the root cause of the problem is a poorly spaced sample design, and that can be addressed directly.

The third option for breaking the curse of dimensionality, **adaptive sampling**, is usually thought of as adding new simulations at points where the emulator predicts that it's most uncertain. This is indeed important and useful. The other side of the adaptive sampling coin is to discard sample points that make $\underline{\underline{R}}$ ill-conditioned, and therefore also restrict it to short correlation lengths, which in turn prevents the Kriging or gradient enhanced Kriging model from extrapolating or interpolating well over large distances. A key contribution of this work is an efficient and effective way to select which points to discard.

This paper proposes an approach incorporating all three techniques and discusses implementation details that improve robustness and efficiency. It also presents the results of computer experiments which show that the approach ameliorates the curse of dimensionality.

2 Background

2.1 Notation

In this work, vectors are indicated by a single underline and matrices are indicated by a double underline. Capital letters indicate data, or functions of data, that is used to construct an emulator. Lower case letters indicate arbitrary points, i.e. points where the simulator may or may not have been evaluated, and functions of arbitrary points.

Let $y = f(\underline{x})$ be a function (i.e. simulator) with one output (of interest) and M input dimensions. Here $\underline{x}$ is an arbitrary point in the M dimensional input space and y is the output of the function at $\underline{x}$; the true value of y is only known if the potentially costly function has been evaluated at $\underline{x}$. Let $\underline{\underline{X}}$ be a N by M sample design matrix, that contains N input points with one point per row. Then $\underline{X_i}$ represents the i-th point in the sample design and the transpose of the i-th row of $\underline{\underline{X}}$. The vector $\underline{X_{:,k}}$ has length N and is the k-th column or input dimension of $\underline{\underline{X}}$. Let $\underline{Y}$ be the vector containing the evaluations of the function at the N points in the sample design, $Y_i = f\left(\underline{X_i}\right)$. The N by M matrix $\underline{\underline{\frac{\partial Y}{\partial X}}}$ contains the partial first derivatives of $\underline{Y}$ with respect to $\underline{\underline{X}}$ such that $\underline{\underline{\frac{\partial Y}{\partial X}}}_{i,k} = \frac{\partial Y_i}{\partial X_{i,k}}$. Then the length M vector $\underline{\frac{\partial Y}{\partial X}}_i$ is the transpose of row i of $\underline{\underline{\frac{\partial Y}{\partial X}}}$ which is the gradient of f evaluated at $\underline{X_i}$. The length N vector $\underline{\frac{\partial Y}{\partial X}}_{:,k}$ is the k-th column of $\underline{\underline{\frac{\partial Y}{\partial X}}}$.

The following additional conventions are used in this work. Where appropriate, vectors and matrices are also treated as being notationally equivalent to sets. For example, the sample design matrix $\underline{\underline{X}}$ is also used to indicate the set of points in the sample design. Estimates, approximations, and models are indicated by hats. For instance, $\hat{f}(\underline{x})$ is a model/emulator of $f(\underline{x})$ and $\hat{y}$ is the emulator's prediction or estimate of the true response y. A tilde indicates a reordering of points/equations, with the possible omission of some but not all points. For example, $\underline{\tilde{Y}} \subseteq \underline{Y} : \underline{\tilde{Y}} \neq \emptyset$ and $\underline{\underline{\tilde{X}}} \subseteq \underline{\underline{X}} : \underline{\underline{\tilde{X}}} \neq \emptyset$ are **reordered**, non-empty, and possibly improper subsets of $\underline{Y}$ and $\underline{\underline{X}}$ such that $\forall j : 1 \leq j \leq \tilde{N} \exists! \, i : 1 \leq i \leq N, \, \tilde{Y}_j = Y_i, \, \underline{\tilde{X}_j} = \underline{X_i}$ where $\tilde{N} : \; 1 \leq \tilde{N} \leq N$ is the number of rows/points in $\underline{\tilde{Y}}$ and $\underline{\underline{\tilde{X}}}$.

2.2 Kriging Emulators

The set of interpolation techniques known as Kriging, also referred to as Gaussian processes, were originally developed in the geostatistics and spatial statistics communities to produce maps of underground geologic deposits based on a set of widely and irregularly spaced borehole sites[5]. Building a Kriging model typically involves the

1. Choice of a trend function,

2. Choice of a correlation function, and

3. Estimation of correlation parameters.

A Kriging emulator, $\hat{f}(\underline{x})$, consists of a trend function (frequently a least squares fit to the data, $\underline{g}(\underline{x})^T \underline{\beta}$) plus a Gaussian process error model, $\varepsilon(\underline{x})$, that is used to correct the trend function.

$$\hat{f}(\underline{x}) = \underline{g}(\underline{x})^T \underline{\beta} + \varepsilon(\underline{x})$$

This represents an estimated distribution for the unknown true surface, $f(\underline{x})$. The error model, $\varepsilon(\underline{x})$, makes an adjustment to the trend function so that the emulator will interpolate, and have zero uncertainty at, the data points it was built from. The covariance between the error at two arbitrary points, $\underline{x}$ and $\underline{x}'$, is modeled as

$$\mathrm{Cov}\left(y(\underline{x}), y\left(\underline{x}'\right)\right) = \mathrm{Cov}\left(\varepsilon(\underline{x}), \varepsilon\left(\underline{x}'\right)\right) = \sigma^2 r\left(\underline{x}, \underline{x}'\right).$$

Here σ^2 is known as the unadjusted variance and $r(\underline{x}, \underline{x}')$ is a correlation function. Measurement error can be modeled explicitly by modifying this to

$$\mathrm{Cov}\left(\varepsilon(\underline{x}), \varepsilon\left(\underline{x}'\right)\right) = \sigma^2 r\left(\underline{x}, \underline{x}'\right) + \Delta^2 \delta\left(\underline{x} - \underline{x}'\right)$$

where

$$\delta\left(\underline{x} - \underline{x}'\right) = \begin{cases} 1 & \text{if } \underline{x} - \underline{x}' = \underline{0} \\ 0 & \text{otherwise} \end{cases}$$

and Δ^2 is the variance of the measurement error. In this work, the term "nugget" refers to the ratio $\eta = \frac{\Delta^2}{\sigma^2}$.

By convention, the terms simple Kriging, ordinary Kriging, and universal Kriging are used to indicate the three most common choices for the trend function. In simple Kriging, the trend is treated as a known constant, usually zero, $g(\underline{x})\beta \equiv 0$. Universal Kriging [20] uses a general polynomial trend model $\underline{g}(\underline{x})^T \underline{\beta}$ whose coefficients are determined by least squares regression. Ordinary Kriging is essentially universal Kriging with a trend order of zero, i.e. the trend function is treated as an unknown constant and $g(\underline{x}) = 1$. N_β is the number of basis functions in $\underline{g}(\underline{x})$ and therefore number of elements in the vector $\underline{\beta}$.

For a finite number of sample points, N, there will be uncertainty about the most appropriate value of the vector, $\underline{\beta}$, which can therefore be described as having a distribution of possible values. If one assumes zero prior knowledge about this distribution, which is referred to as the "vague prior" assumption, then the maximum likelihood value of $\underline{\beta}$ can be computed via least squares generalized by the inverse of the error model's correlation matrix, $\underline{\underline{R}}$

$$\underline{\hat{\beta}} = \left(\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{\underline{G}}\right)^{-1} \left(\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{Y}\right).$$

Here $\underline{\underline{G}}$ is a N by N_β matrix that contains the evaluation of the least squares basis functions at all points in $\underline{\underline{X}}$, such that $G_{i,l} = g_l\left(\underline{X}_i\right)$. The real, symmetric, positive-definite correlation matrix, $\underline{\underline{R}}$,

of the error model contains evaluations of the correlation function, r, at all pairwise combination of points (rows) in the sample design, $\underline{\underline{X}}$.

$$R_{i,j} = R_{j,i} = r\left(\underline{X_i}, \underline{X_j}\right) = r\left(\underline{X_j}, \underline{X_i}\right)$$

There are many options for r, among them are the following families of correlation functions:

- **Powered-Exponential**

$$r\left(\underline{X_i}, \underline{X_j}\right) = \exp\left(-\sum_{k=1}^{M} \theta_k \left|X_{i,k} - X_{j,k}\right|^{\gamma}\right) \tag{1}$$

  where $0 < \gamma \leq 2$ and $0 < \theta_k$.

- **Matern**

$$r\left(\underline{X_i}, \underline{X_j}\right) = \prod_{k=1}^{M} \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\theta_k \left|X_{i,k} - X_{j,k}\right|\right)^{\nu} \mathscr{K}_{\nu}\left(\theta_k \left|X_{i,k} - X_{j,k}\right|\right)$$

  where $0 < \nu$, $0 < \theta_k$, and $\mathscr{K}_{\nu}(\cdot)$ is the modified Bessel function of order ν; $\nu = s + \frac{1}{2}$ is the smallest value of ν which results in a Kriging model that is s times differentiable.

- **Cauchy**

$$r\left(\underline{X_i}, \underline{X_j}\right) = \prod_{k=1}^{M} \left(1 + \theta_k \left|X_{i,k} - X_{j,k}\right|^{\gamma}\right)^{-\nu}$$

  where $0 < \gamma \leq 2$, $0 < \nu$, and $0 < \theta_k$.

Gneiting et al. [11] provide a more thorough discussion of the properties of and relationships between these three families. Some additional correlation functions include the Dagum family [3] and cubic splines.

The author selected the squared exponential or Gaussian correlation function (Equation 1 with $\gamma = 2$) on the basis that its infinite smoothness or differentiability should aid in leveraging the anticipated and hopefully sparse data. For the Gaussian correlation function, the correlation parameters, $\underline{\theta}$, are related to the correlation lengths, $\underline{L}$, by

$$\theta_k = \frac{1}{2\, L_k^2}. \tag{2}$$

Here, the correlation lengths, $\underline{L}$, are analogous to standard deviations in the Gaussian or normal distribution and often have physical meaning. The adjusted (by data) mean of the emulator is a best linear unbiased estimator of the unknown true function,

$$\hat{y} = \mathrm{E}\left(\hat{f}\left(\underline{x}\right) | \underline{f}\left(\underline{\underline{X}}\right)\right) = \underline{g}\left(\underline{x}\right)^T \underline{\hat{\beta}} + \underline{r}\left(\underline{x}\right)^T \underline{\underline{R}}^{-1} \underline{\varepsilon}. \tag{3}$$

Here, $\underline{\varepsilon} = \left(\underline{Y} - \underline{\underline{G}}\,\underline{\hat{\beta}}\right)$ is the known vector of differences between the true outputs and trend function at all points in $\underline{\underline{X}}$ and the vector $\underline{r}(\underline{x})$ is defined such that $r_i(\underline{x}) = r\left(\underline{x}, \underline{X_i}\right)$. This adjustment can be interpreted as the projection of prior belief (the least squares fit) into the span of the data. The adjusted mean of the emulator will interpolate the data that the Kriging model was built from as long as its correlation matrix, $\underline{\underline{R}}$, is numerically non-singular.

Ill-conditioning of $\underline{\underline{R}}$ and other matrices is recognized as a significant challenge for Kriging. Davis and Morris [8] gave a thorough review of six factors affecting the condition number of matrices associated with Kriging (from the perspective of semivariograms rather than correlation functions). They concluded that "Perhaps the best advice we can give is to be mindful of the condition number when building and solving kriging systems." In the context of estimating the optimal $\underline{\theta}$, Martin [19] stated that Kriging's "three most prevalent issues are (1) ill-conditioned correlation matrices,(2) multiple local optimum, and (3) long ridges of near optimal values." Martin used constrained optimization to address ill-conditioning of $\underline{\underline{R}}$. Rennen [28] advocated that ill-conditioning be handled by building Kriging models from a uniform subset of available sample points. That option has been available in DAKOTA's "Gaussian process" model (a separate implementation from DAKOTA's "Kriging" model) since version 4.1 [10].

Adding a nugget, η, to the diagonal entries of $\underline{\underline{R}}$ is a popular approach for both accounting for measurement error in the data and alleviating ill-conditioning. However, doing so will cause the Kriging model to smooth or approximate rather than interpolate the data. Methods for choosing a nugget include:

- Choosing a nugget based on the variance of measurement error (if any); this will be an iterative process if σ^2 is not known in advance.
- Iteratively adding a successively larger nugget until $\underline{\underline{R}} + \eta \underline{\underline{I}}$ is no longer ill-conditioned.
- Exactly calculating the minimum nugget needed for a target condition number from $\underline{\underline{R}}$'s maximum λ_{max} and minimum λ_{min} eigenvalues. The condition number of $\underline{\underline{R}} + \eta \underline{\underline{I}}$ is $\frac{\lambda_{max}+\eta}{\lambda_{min}+\eta}$. However, calculating eigenvalues is computationally expensive. Since Kriging's $\underline{\underline{R}}$ matrix has all ones on the diagonal, its trace and therefore sum of eigenvalues is N. Consequently, a nugget value of $\eta = \frac{N}{\text{target condition number} - 1}$ will always alleviate ill-conditioning. A smaller nugget that is also guaranteed to alleviate ill-conditioning can be calculated from LAPACK's fast estimate of the reciprocal of $\underline{\underline{R}}$'s condition number, rcond $\left(\underline{\underline{R}}\right)$.
- Treating η as another parameter to be selected by the same process used to choose $\underline{\theta}$. Two such approaches are discussed below.

The Kriging model's adjusted variance is commonly used as a spatially varying measure of uncertainty. Knowing where, and by how much, the model "doubts" its own predictions helps build user confidence in the predictions and can be utilized to guide the selection of new sample

points during optimization or to otherwise improve the surrogate. The adjusted variance is

$$\begin{aligned}\mathrm{Var}\left(\hat{y}\right) &= \mathrm{Var}\left(\hat{f}\left(\underline{x}\right)|\underline{f}\left(\underline{\underline{X}}\right)\right) \\ &= \hat{\sigma}^2\left(1-\underline{r}\left(\underline{x}\right)^T\ \underline{\underline{R}}^{-1}\underline{r}\left(\underline{x}\right)+\ldots\right. \\ &\quad \left.\left(\underline{g}\left(\underline{x}\right)^T-\underline{r}\left(\underline{x}\right)^T\ \underline{\underline{R}}^{-1}\underline{\underline{G}}\right)\left(\underline{\underline{G}}^T\underline{\underline{R}}^{-1}\underline{\underline{G}}\right)^{-1}\left(\underline{g}\left(\underline{x}\right)^T-\underline{r}\left(\underline{x}\right)^T\ \underline{\underline{R}}^{-1}\underline{\underline{G}}\right)^T\right)\end{aligned}$$

where the maximum likelihood estimate of the unadjusted variance is

$$\hat{\sigma}^2=\frac{\underline{\varepsilon}^T\underline{\underline{R}}^{-1}\underline{\varepsilon}}{N-N_\beta}.$$

There are at least two types of numerical approaches for choosing $\underline{\theta}$. One of these is to use Bayesian techniques such as Markov Chain Monte Carlo to obtain a distribution represented by an ensemble of vectors $\underline{\theta}$. In this case, evaluating the emulator's mean involves taking a weighted average of Equation 3 over the ensemble of $\underline{\theta}$ vectors.

The other, more common, approach to constructing a Kriging model involves using optimization to find the set of correlation parameters $\underline{\theta}$ that maximizes the likelihood of the model given the data. It is equivalent, and more convenient to maximize the natural logarithm of the likelihood, which assuming a vague prior is,

$$\begin{aligned}\log\left(\mathrm{lik}\left(\underline{\theta}\right)\right) &= -\frac{1}{2}\left(\left(N-N_\beta\right)\left(\frac{\hat{\sigma}^2}{\sigma^2}+\log\left(\sigma^2\right)+\log(2\pi)\right)+\ldots\right. \\ &\quad \left.\log\left(\det\left(\underline{\underline{R}}\right)\right)+\log\left(\det\left(\underline{\underline{G}}^T\underline{\underline{R}}^{-1}\underline{\underline{G}}\right)\right)\right).\end{aligned}$$

And, if one substitutes the maximum likelihood estimate $\hat{\sigma}^2$ in for σ^2, then it is equivalent to minimize the following objective function

$$\mathrm{obj}\left(\underline{\theta}\right)=\log\left(\hat{\sigma}^2\right)+\frac{\log\left(\det\left(\underline{\underline{R}}\right)\right)+\log\left(\det\left(\underline{\underline{G}}^T\underline{\underline{R}}^{-1}\underline{\underline{G}}\right)\right)}{N-N_\beta}.$$

Because of the division by $N-N_\beta$, this "per-equation" objective function is mostly independent of the number of sample points, N. It is therefore useful for comparing the (estimated) "goodness" of Kriging models that have different numbers of sample points, which will be important later.

2.3 Kriging Implementation Details

It is fairly simple to implement a "naive" or straight-forward Kriging code, however, such an implementation will often perform poorly. For this reason, significant care and effort was invested

to develop the Kriging implementation in version 5.1 of the DAKOTA software package [1]. The gradient enhanced universal Kriging implementation presented in this paper is a modification of the DAKOTA 5.1 universal Kriging model. The following features of the DAKOTA 5.1 Kriging implementation are relevant to the efficient and robust construction of accurate gradient enhanced Kriging models.

- An extremely lightweight templated C++ dense matrix class with column major memory storage (to facilitate use with BLAS and LAPACK) and dynamic memory allocation. The underlying arrays are automatically deallocated by the class destructor and as necessary allocated/reallocated by the BLAS and LAPACK wrapper functions.

- For the sake of robustness, the matrix operations functions that wrap BLAS and LAPACK will dynamically reallocate the size of matrices as necessary. However, for the sake of efficiency, the Kriging implementation avoids dynamic memory allocation wherever it is possible to do so. This is accomplished by preallocating the known required space for not only quantities that need to be retained but also for temporary intermediate vectors and matrices of known size. Separate space is allocated for temporary vectors and matrices of different sizes.

- Since $\underline{\underline{R}}$ and $\left(\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{\underline{G}}\right)$ are theoretically real, symmetric, and positive definite, LAPACK's very efficient (level 3 BLAS) Cholesky factorization is used. It requires about half the number of operations as an LU factorization. The Cholesky wrapper function also computes the reciprocal of the condition number, rcond. The *additional* cost is negligible compared to the cost of the already required Cholesky factorization.

- The determinant of $\underline{\underline{R}}$ (and $\left(\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{\underline{G}}\right)$) can be efficiently calculated as the square of the product of the diagonals of its Cholesky factorization. However, this will often underflow, i.e. go to zero, making its log incorrectly go to $-\infty$. A more accurate and robust calculation of $\log\left(\det\left(\underline{\underline{R}}\right)\right)$ can be achieved by taking twice the sum of the log of the diagonals of $\underline{\underline{R}}$'s Cholesky factorization.

- In combination with a precompute, store, and reuse strategy, matrix operations are ordered to minimize the flops (floating point operations) cost of emulator construction and evaluation.

- The only option for the correlation function in the DAKOTA 5.1 Kriging is the squared-exponential or Gaussian correlation function, Equation 1 with $\gamma = 2$. The basis for implementing this correlation function first is that the most likely reason to build an emulator is that the simulator is expensive and hence the data will be sparse. For efficiency, the $\underline{\underline{R}}$ matrix is evaluated as follows. A $\frac{N\,(N-1)}{2}$ by M matrix $\underline{\underline{Z}}$ is precomputed and stored, such that $Z_{ij,k} = -\left(X_{i,k} - X_{j,k}\right)^2$. The single index ij is defined such that $ij = i + (j-1)N - (j+1)j/2$ where $1 \leq j < N$, $j < i \leq N$, and $1 \leq ij \leq \frac{N\,(N-1)}{2}$. The lower triangular part of $\underline{\underline{R}}$, not including the diagonal whose elements are identically equal to 1, can be computed by the element by element exponential of the matrix vector product $\left(\underline{\underline{Z}}\ \underline{\theta}\right)$. This is beneficial because a large number of $\underline{\theta}$ vectors will be tried during the maximum likelihood optimization. This strategy

is easily extensible to the powered-exponential family of correlation functions; Matern and similar correlation functions will require additional operations after the exponential.

- By default, the size of the input space is defined as the smallest hyper-rectangle that contains the sample design. The user has the option to define a larger input space that includes a region where they wish to extrapolate. Note that the emulator can be evaluated at points outside the defined input space, but this definition helps the Kriging model determine the region in correlation length space that it should search during the maximum likelihood optimization.

- The input space is normalized to the unit hypercube centered at the origin. This can greatly improve the condition number of $\left(\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{\underline{G}}\right)$, and simplifies the definition of an appropriate range of correlation lengths $\underline{L}$ to search during the maximum likelihood optimization.

- In the normalized input space, i.e. the unit hypercube, the average distance between nearest neighboring points is

$$d = \left(\frac{1}{N}\right)^{1/M}.$$

  The range of correlation lengths, $\underline{L}$, searched for the Gaussian exponential function is

$$\frac{d}{4} \leq L_k \leq 8d.$$

  This range was chosen based on the correlation lengths being analogous to the standard deviation in the Gaussian or Normal distribution. If the correlation lengths are set to $L_k = d/4$, then nearest neighboring points "should be" roughly four "standard deviations" away making them almost completely uncorrelated with each other. $\underline{\underline{R}}$ would then be a good approximation of the identity matrix and have a condition number close to one. Therefore, in the absence of a pathological spacing of points, this range of $\underline{L}$ should contain some non-singular $\underline{\underline{R}}$. $L_k = 8d$ implies approximately 32% trust in what points 8 neighbors away have to say and 5% trust in what points 16 neighbors away have to say. It is possible that the optimal correlation lengths are larger than $8d$; but if so, then either almost all of the same information will be contained in more nearby neighbors, or it was not appropriate to use the squared-exponential/Gaussian correlation function. When other correlation functions are added to the DAKOTA Kriging implementation, each will be associated with its own range of appropriate correlation lengths chosen by similar reasoning. A different definition of d could be used for non-hypercube input domains.

- The user has other options, but by default the DIRECT (DIvision of RECTangles) global optimizer is used to maximize the likelihood of the Kriging model given the data under the constraint that $\underline{\underline{R}}$ is non-singular. The search is actually performed in log of correlation length space. The author chose to implement the non-singular $\underline{\underline{R}}$ constraint as a requirement that $2^{-40} \leq \text{rcond}\left(\underline{\underline{R}}\right)$. Recall that the additional cost of computing rcond is negligible compared to the cost of the Cholesky factorization. Based on empirical results, the author believes this is sufficient[1] to make a global maximization of likelihood a robust method of choosing

[1] Disclaimer: there may be NO "good" set of correlation lengths when the sample design is pathologically spaced.

the correlation lengths, $\underline{L}$, or equivalently the correlation parameters, $\underline{\theta}$. In the absence of constraints, maximizing the likelihood would result in singular $\underline{\underline{R}}$ which makes the emulator incapable of reproducing the data from which it was built. This is because a singular $\underline{\underline{R}}$ makes $\log\left(\det\left(\underline{\underline{R}}\right)\right) = -\infty$ and the *estimate* of likelihood infinite.

- Used through the DAKOTA interface, the default trend function is a "reduced" or main effects quadratic polynomial

$$\underline{g}\left(\underline{x}\right)^T \underline{\beta} = \beta_1 + \sum_{k=1}^{M} x_k \beta_{k+1} + \sum_{k=1}^{M} x_k^2 \beta_{k+M+1}$$

in the normalized input space. However, the user can specify an arbitrary order polynomial with or without cross terms.

3 Gradient Enhanced Kriging

This section focuses on the incorporation of derivative information into Kriging models and challenges in their implementation. Special attention is paid to conditioning issues.

There are at least three basic approaches for incorporating derivative information into Kriging. These are

1. **Indirect**: The sample design is augmented with fictitious points which are nearby actual sample points and predicted from derivative information. A Kriging model is then built from the augmented design.
2. **Co-Kriging**: The derivatives with respect to each input variables are treated as separate but correlated output variables and a Co-Kriging model is built for the set of output variables. This would use $\begin{pmatrix} M+2 \\ 2 \end{pmatrix}$ $\underline{\theta}$ vectors.
3. **Direct**: The relationship between the response value and its derivatives is leveraged to use a single $\underline{\theta}$ by assuming

$$\mathrm{Cov}\left(y\left(\underline{x}^1\right), \frac{\partial y\left(\underline{x}^2\right)}{\partial x_k^2} \right) = \frac{\partial}{\partial x_k^2} \left(\mathrm{Cov}\left(y\left(\underline{x}^1\right), y\left(\underline{x}^2\right) \right) \right). \tag{4}$$

The direct approach is the topic of this work and herein it is referred to simply as Gradient Enhanced (universal) Kriging (GEK). The equations for GEK can be derived by assuming Equation 4 and then taking the same steps used to derive function value only Kriging. The superscript on $\underline{x}$ in Equation 4 and below indicates whether it's the 1st or 2nd input to $r\left(\underline{x}^1, \underline{x}^2\right)$. Note that when the first and second arguments are the same, the derivative of $r(\ ,\ )$ with respect to the first argument is equal in magnitude but opposite in sign compared to the derivative with respect to the second argument. The GEK equations can also be obtained by starting from a Kriging model and making the following substitutions $\underline{Y} \to \underline{Y_\nabla}$, $\underline{\underline{G}} \to \underline{\underline{G_\nabla}}$, $\underline{r} \to \underline{r_\nabla}$, $\underline{\underline{R}} \to \underline{\underline{R_\nabla}}$, and $N \to N_\nabla = N\,(1+M)$, where N_∇ is the number of equations rather than the number of points,

$$\underline{Y_\nabla} = \begin{bmatrix} \underline{Y} \\ \frac{\partial \underline{Y}}{\partial X_{:,1}} \\ \frac{\partial \underline{Y}}{\partial X_{:,2}} \\ \vdots \\ \frac{\partial \underline{Y}}{\partial X_{:,M}} \end{bmatrix}, \quad \underline{\underline{G_\nabla}} = \begin{bmatrix} \underline{\underline{G}} \\ \frac{\partial \underline{\underline{G}}}{\partial X_{:,1}} \\ \frac{\partial \underline{\underline{G}}}{\partial X_{:,2}} \\ \vdots \\ \frac{\partial \underline{\underline{G}}}{\partial X_{:,M}} \end{bmatrix}, \quad \underline{r_\nabla} = \begin{bmatrix} \underline{r} \\ \frac{\partial \underline{r}}{\partial X_{:,1}} \\ \frac{\partial \underline{r}}{\partial X_{:,2}} \\ \vdots \\ \frac{\partial \underline{r}}{\partial X_{:,M}} \end{bmatrix}$$

$$\underline{\underline{R_\nabla}} = \begin{bmatrix} \underline{\underline{R}} & \frac{\partial \underline{\underline{R}}}{\partial X^2_{:,1}} & \frac{\partial \underline{\underline{R}}}{\partial X^2_{:,2}} & \cdots & \frac{\partial \underline{\underline{R}}}{\partial X^2_{:,M}} \\ \frac{\partial \underline{\underline{R}}}{\partial X^1_{:,1}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,1} \partial X^2_{:,1}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,1} \partial X^2_{:,2}} & \cdots & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,1} \partial X^2_{:,M}} \\ \frac{\partial \underline{\underline{R}}}{\partial X^1_{:,2}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,2} \partial X^2_{:,1}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,2} \partial X^2_{:,2}} & \cdots & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,2} \partial X^2_{:,M}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \underline{\underline{R}}}{\partial X^1_{:,M}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,M} \partial X^2_{:,1}} & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,M} \partial X^2_{:,2}} & \cdots & \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,M} \partial X^2_{:,M}} \end{bmatrix}$$

$$\frac{\partial \underline{\underline{R}}}{\partial X^1_{:,I}} = -\left(\frac{\partial \underline{\underline{R}}}{\partial X^1_{:,I}}\right)^T = -\frac{\partial \underline{\underline{R}}}{\partial X^2_{:,I}} = \left(\frac{\partial \underline{\underline{R}}}{\partial X^2_{:,I}}\right)^T$$

$$\frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,I} \partial X^2_{:,J}} = \left(\frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,I} \partial X^2_{:,J}}\right)^T = \frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,J} \partial X^2_{:,I}} = \left(\frac{\partial^2 \underline{\underline{R}}}{\partial X^1_{:,J} \partial X^2_{:,I}}\right)^T$$

Here capital I and J are scalar indices for the input dimension (column) of the sample design, $\underline{\underline{X}}$. Note that for the Gaussian correlation function

$$\frac{\partial^2 R_{j,j}}{\partial X^1_{j,I} \partial X^2_{j,I}} = 2\theta_I$$

and has units of length^{-2}. Two of the conditions necessary for a matrix to qualify as a correlation matrix are that all of its elements must be dimensionless and all of its diagonal elements must identically equal one. Since $\underline{\underline{R_\nabla}}$ does not satisfy these requirements, it technically does not qualify as a "correlation matrix." However, referring to $\underline{\underline{R_\nabla}}$ as such is a small abuse of terminology and allows GEK to use the same naming conventions as Kriging.

A straight-forward implementation of GEK tends be significantly more accurate than Kriging given the same sample design provided that the

- Derivatives are accurate
- Derivatives are not infinite (or nearly so)
- Function is sufficiently smooth, and
- $\underline{\underline{R_\nabla}}$ is not ill-conditioned (this can be problematic).

If gradients can be obtained cheaply (e.g. by automatic differentiation or adjoint techniques) and the previous conditions are met, GEK also tends to outperform Kriging for the same computational budget. Previous works, such as Dwight and Han [9], state that the direct approach to GEK

is significantly better conditioned than the indirect approach. While this is true, (direct) GEK's $\underline{\underline{R_\nabla}}$ matrix can still be, and often is, horribly ill-conditioned compared to Kriging's $\underline{\underline{R}}$ for the same $\underline{\theta}$ and $\underline{X}$

The author has explored a variety of approaches to deal with the ill-conditioning of $\underline{\underline{R_\nabla}}$. These include but are not limited to:

- **Reducing the correlation lengths to make $\underline{\underline{R_\nabla}}$ non-singular.** Provided that there are no duplicate points, the correlation lengths $\underline{L}$ can always be reduced enough to make $\underline{\underline{R_\nabla}}$ invertible, i.e. $\underline{L} = \underline{0}$. However, sufficiently small $\underline{L}$ often means that the GEK model will be inept at extrapolating or even interpolating over long distances.

- **Equilibrating $\underline{\underline{R_\nabla}}$.** Equilibration improves the accuracy of solving linear systems, especially when the matrix is poorly scaled. Theorem 4.1 of van der Sluis [31] states that if $\underline{\underline{a}}$ is a real, symmetric, positive definite n by n matrix and the diagonal matrix $\underline{\underline{\alpha}}$ contains the square roots of the diagonals of $\underline{\underline{a}}$, then the equilibration

  $$\breve{\underline{\underline{a}}} = \underline{\underline{\alpha}}^{-1} \underline{\underline{a}} \, \underline{\underline{\alpha}}^{-1},$$

  minimizes the 2-norm condition number of $\breve{\underline{\underline{a}}}$ (with respect to solving linear systems) over all such symmetric scalings, to within a factor of n. The equilibrated matrix $\breve{\underline{\underline{a}}}$ will have all ones on the diagonal; this means the *equilibrated* version of $\underline{\underline{R_\nabla}}$ does qualify as a correlation matrix. However, rounding the elements of $\underline{\underline{\alpha}}$ to the nearest power of 2 is more common in numerical linear algebra since it prevents the loss of precision due to round off error during the equilibration. Both of these options were explored; neither was sufficient to reliably mitigate the effect of an ill-conditioned $\underline{\underline{R_\nabla}}$. The wrapper function for the LAPACK Cholesky factorization in the current Kriging implementation has been updated to always equilibrate; this affects the Cholesky factorization of $\underline{\underline{G}}^T \underline{\underline{R}}^{-1} \underline{\underline{G}}$.

- **Using a $\underline{\underline{R_\nabla}} = \underline{\underline{U}}^T \underline{\underline{D}} \, \underline{\underline{U}}$ factorization with pivoting in place of Cholesky.** Here $\underline{\underline{U}}$ is an upper triangular matrix and $\underline{\underline{D}}$ is a block diagonal matrix. $\underline{\underline{U}}$ and $\underline{\underline{D}}$ are computed with the LAPACK symmetric triangular factorization. This does not require the matrix to be positive definite but is remarkably slower than the (level 3) LAPACK Cholesky and frequently insufficient to mitigate an ill-conditioned $\underline{\underline{R_\nabla}}$ matrix even after the optimal equilibration has been applied.

- **Using the Moore Penrose pseudo inverse in place of the Cholesky factorization.** This researcher was not satisfied with this approach's computational cost or the accuracy of its predictions at validation points.

- **Adding a nugget, η, (a small noise term) to the diagonal of $\underline{\underline{R_\nabla}}$.** This researcher was not satisfied with the prediction quality of this approach.

- **Thresholding small off-diagonal elements to zero.** This researcher was not satisfied with the prediction quality of this approach. Correlation functions with compact support, such

as cubic splines, may perform better; but private communication from other researchers working on GEK indicated dissatisfaction with these.

The author has added the powered exponential and Matern families to provide other choices for the correlation function. These correlation functions are implemented in Dakota 5.3 and Surfpack 1.1. Note that these functions were added after this technical report was written but before it was published. Although a different correlation function may alleviate the ill-conditioning for some test problems, the root cause of the ill-conditioning is a poorly spaced sample design. Furthermore, a sufficiently bad sample design could make any interpolatory Kriging model, gradient enhanced or otherwise, ill-conditioned, regardless of the choice of correlation function. For the sake of robustness, this root cause will be directly addressed.

3.1 Effective and Efficient Handling of Ill-Conditioned Correlation Matrices

A matrix being ill-conditioned means that its rows or columns contain a significant amount of duplicate information. For a real, symmetric, positive definite matrix the condition number equals the ratio of its maximum to minimum eigen or singular values. In this context, mitigating ill-conditioning involves increasing the smallest and/or decreasing the largest eigen or singular value.

A pseudo inverse handles ill-conditioning by discarding small singular values, which can be interpreted as throwing away the information that is least present while keeping all of what is most frequently duplicated. This causes a Kriging model to not interpolate any of the data points used to construct it while using some information from all rows. An alternate strategy, one that the author finds more conceptually appealing, is to discard additional copies of the information that is most duplicated and keep more of the barely present information. In the context of eigenvalues, this can be described as decreasing the maximum eigenvalue and increasing the minimum eigenvalue by a smaller amount than a pseudo inverse. The result is that the GEK model will exactly fit all of the retained information. This can be achieved using a pivoted Cholesky factorization, such as the one developed by Lucas [18] to determine a reordering $\tilde{\underline{\underline{R}}}_{\nabla}$ and dropping equations off its end until it tightly meets the constraint on rcond. However, a straight-forward implementation of this is neither efficient nor robust.

In benchmarking tests, Lucas' level 3 pivoted Cholesky implementation was not competitive with the level 3 LAPACK non-pivoted Cholesky in terms of computational efficiency. In some cases, it was an order of magnitude slower. Note that Lucas' level 3 implementation can default to his level 2 implementation and this may explain some of the loss in performance.

More importantly, applying pivoted Cholesky to $\underline{\underline{R}}_{\nabla}$ tends to sort derivative equations to the top/front and function value equations to the end. This occurred even when $\underline{\underline{R}}_{\nabla}$ was equilibrated

to have ones for all of its diagonal elements. The result was that for many points at least some of the derivative equations were retained while the function values at the same points were discarded. This can (and often did) significantly degrade the accuracy of the GEK predictions. The implication is that derivative equations contain more information than, but are not as reliable as, function value equations.

To address computational efficiency and robustness, the approach was modified to:

- Equilibrate $\underline{\underline{R_\nabla}}$ to have all ones on the diagonal, making it a correlation matrix.
- Perform pivoted Cholesky on $\underline{\underline{R}}$, instead of $\underline{\underline{R_\nabla}}$, to rank points according to how much new information they contain. This ranking was reflected by the ordering of points in $\underline{\underline{\tilde{R}}}$.
- Apply the ordering of points in $\underline{\underline{\tilde{R}}}$ to whole points in $\underline{\underline{R_\nabla}}$ to produce $\underline{\underline{\tilde{R}_\nabla}}$. Here a whole point means the function value at a point immediately followed by the derivatives at the same point.
- Perform a LAPACK non-pivoted Cholesky on the equilibrated $\underline{\underline{\tilde{R}_\nabla}}$ and drop equations off the end until it satisfies the constraint on rcond. LAPACK's rcond estimate requires the 1-norm of the original (reordered) matrix as input so the 1-norms for all possible sizes of $\underline{\underline{\tilde{R}_\nabla}}$ are precomputed and stored prior to the Cholesky factorization. A bisection search is used to efficiently determine the number of equations that need to be retained/discarded; it is described in the implementation Section 3.2.

This algorithm is visually depicted in Figure 1. Because inverting/factorizing a matrix with n rows and columns requires $\mathcal{O}\left(n^3\right)$ flops, the cost to perform pivoted Cholesky on $\underline{\underline{R}}$ will be much less than, i.e. $\mathcal{O}\left((1+M)^{-3}\right)$, that of $\underline{\underline{R_\nabla}}$ when the number of dimensions M is large. It will also likely be negligible compared to the cost of performing LAPACK's non-pivoted Cholesky on $\underline{\underline{\tilde{R}_\nabla}}$.

Note that the equations (read as rows of $\underline{\underline{R_\nabla}}$) that are dropped are the ones that contain little new information. A consequence of this is that the GEK model should well predict the dropped points. However, this is not guaranteed. Because $\underline{\underline{R_\nabla}}$ does not depend on the outputs, $\underline{Y_\nabla}$, but only the inputs $\underline{\underline{X}}$ (and $\underline{\theta}$), when there are nearby points on opposite sides of a sharp discontinuity in the output, one of them can and is likely to be discarded. Fortunately, this case can be readily checked for by comparing the difference between emulator's mean prediction and the true value to the adjusted variance at the dropped points. If a sharp discontinuity is detected, this indicates that a different strategy should be used to construct the response surface. For example, one could use adaptive sampling to characterize the shape of the discontinuity and then fit a separate response surface on either side of it.

Note that the author's warning concerning the case with nearby points on opposite sides of a discontinuity is speculative and possibly unwarranted since the author not yet encountered this

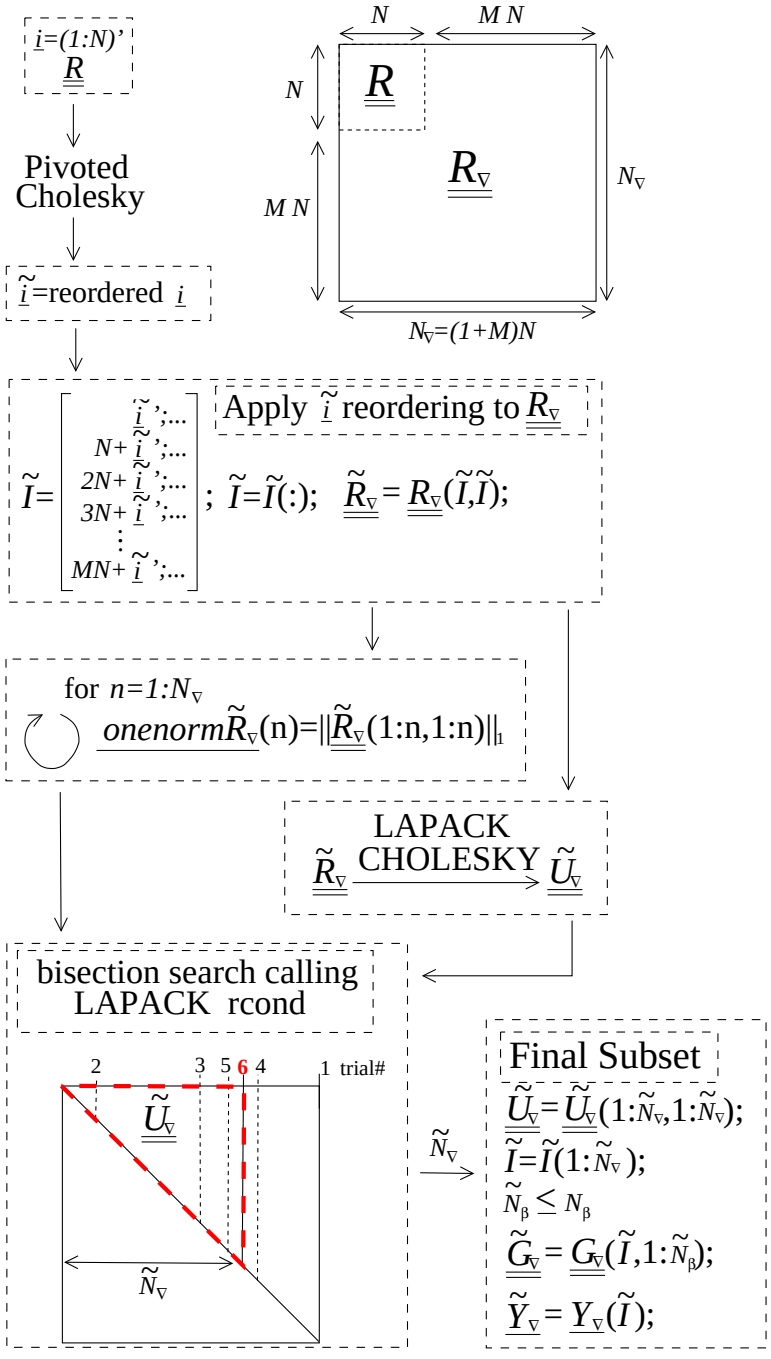

Figure 1. A diagram with pseudo code for the pivoted Cholesky algorithm used to select the subset of equations to retain when $\underline{\underline{R}}_\nabla$ is ill-conditioned. Although it is part of the algorithm, the equilibration of $\underline{\underline{R}}_\nabla$ is not shown in this figure. The pseudo code uses MATLAB notation.

situation. For test problems, the prediction quality of this pivoted Cholesky approach was significantly improved relative to when ill-conditioning was handled by either taking the pseudo inverse of $\underline{\underline{R_\nabla}}$ or adding a nugget to the diagonal of $\underline{\underline{R_\nabla}}$.

It should also be noted that the pivoted Cholesky approach presented here differs significantly from the sample point subset selection strategies advocated by Rennen [28] and used in DAKOTA's Gaussian process implementation since version 4.1. In prior works, the subset of sample points was selected using an uniformity criteria and $\underline{\theta}$ was chosen after subset selection was completed. In this work, a different subset of sample points could be used for each $\underline{\theta}$ vector examined during the maximization of the per-equation log likelihood and the subset that is chosen is the one associated with the most likely $\underline{\theta}$.

3.2 GEK Implementation Details

The following features were essential to the efficiency and/or robustness of the newly implemented Gradient Enhanced Kriging model (which is available in the most recent "stable" release of DAKOTA).

- When the number of equations retained by the pivoted Cholesky approach is insufficient to determine the unknowns in the GEK model, the polynomial order of the trend function is decreased until the number of equations that is required for the trend, is less than the number that was retained by the pivoted Cholesky approach. The number of trend functions actually used is then $\tilde{N}_\beta : \ 1 \leq \tilde{N}_\beta \leq N_\beta$.

- Minimizing a per-equation objective function (which is equivalent to maximizing the per-equation log likelihood):

$$\text{obj}\left(\underline{\theta}\right) = \log\left(\hat{\sigma}^2\right) + \frac{\log\left(\det\left(\underline{\underline{\tilde{R}_\nabla}}\right)\right) + \log\left(\det\left(\underline{\underline{\tilde{G}_\nabla}}^T \underline{\underline{\tilde{R}_\nabla}}^{-1} \underline{\underline{\tilde{G}_\nabla}}\right)\right)}{\tilde{N}_\nabla - \tilde{N}_\beta},$$

  where

$$\hat{\sigma}^2 = \frac{\left(\underline{\tilde{Y}_\nabla} - \underline{\underline{\tilde{G}_\nabla}}\,\underline{\hat{\beta}}\right)^T \underline{\underline{\tilde{R}_\nabla}}^{-1} \left(\underline{\tilde{Y}_\nabla} - \underline{\underline{\tilde{G}_\nabla}}\,\underline{\hat{\beta}}\right)}{\tilde{N}_\nabla - \tilde{N}_\beta}$$

  and the phrase "per-equation" refers to the division of the log likelihood by $\tilde{N}_\nabla - \tilde{N}_\beta$. If the likelihood of candidate emulators are not compared on a per-equation basis, then the optimization will be biased toward emulators that have a larger number of equations. That tends to favor shorter correlation lengths which generally do not extrapolate, or interpolate over large distances, as well as longer correlation lengths.

- The matrix class was modified so that matrices can have different apparent (i.e. for mathematical purposes) and actual (memory footprint) sizes. Unless "forced" to work on the actual size, the dynamic memory allocation functions now affect the **apparent** size and element

layout of the matrix in the same way that they actually affected the DAKOTA 5.1 Kriging matrix class. However, the memory footprint of the modified matrix class does **not** change unless the requested apparent size of the matrix is larger than the current actual size or the function call specifies that the change is forced. The computational benefit of this modification is that if the actual sizes of matrices are preallocated for the full set of N_∇ equations, then subsequently discarding equations via the pivoted Cholesky approach described above will not require dynamic memory re-allocation. This avoids a dramatic loss in computational efficiency that would otherwise occur.

- The sequence of rcond estimates needed to find the last row of $\tilde{\underline{\underline{R}}}_\nabla$ which satisfies the constraint on rcond can be computed efficiently. To do so, one can use a rank one update approach to efficiently precompute the 1-norm for all numbers, n, of rows and columns of $\tilde{\underline{\underline{R}}}_\nabla$ such that $1 \leq n \leq N_\nabla$, and then use bisection, i.e. at most ceil$(\log 2\,(N_\nabla))$ calls, for LAPACK's fast estimate of the 1-norm of $\tilde{\underline{\underline{R}}}_\nabla^{-1}$. The 1-norm estimates of $\tilde{\underline{\underline{R}}}_\nabla^{-1}$ are all based on the same Cholesky factorization of $\tilde{\underline{\underline{R}}}_\nabla$ but use the number of rows of the Cholesky factorization called for by the bisection search.

Also, although it was not strictly *essential* for an efficient GEK implementation, the number of operations required to compute $\underline{\underline{R}}_\nabla$ was reduced by reusing $\underline{\underline{R}}$ to compute the derivatives of $\underline{\underline{R}}$. This store and reuse strategy eliminated most of the evaluations of the somewhat expensive exponential function which would otherwise be required.

Note that as of Dakota 5.3 we do provide an alternate approach to the pivoted Cholesky approach for handling ill-conditioning in GEK. The alternate approach is based on a nugget. However, the nugget is not just added to the diagonal of the full covariance matrix of GEK because the derivative portion of the R matrix doesn't have ones on the diagonal. Instead we multiply the diagonal by (1+*nugget*) and we choose a value for the nugget needed to fix the worst case (largest nugget) matrix possible given LAPACK's fast estimate of rcond$(\mathrm{R})_{L_1}$; note that this is an estimate of the reciprocal of the one-norm condition number which determines numerical ill-conditioning but is not amenable to analytic manipulation. The two-norm of the condition number is ratio of minimum and maximum eigenvalues of a matrix; it is amenable to analytic manipulation. The one-norm and two-norm condition numbers are related via $\frac{\mathrm{CondNum}_{L_1}}{\sqrt{n}} \leq \mathrm{CondNum}_{L_2} \leq \mathrm{CondNum}_{L_1} \cdot \sqrt{n}$ where $n = N$ for regular Kriging and $n = N_\nabla$ for GEK. We assume that the sum of the eigenvalues is equal to n; this is true for regular Kriging's $\underline{\underline{R}}$ matrix which has all ones on the diagonal, and we can equilibrate GEK's $\underline{\underline{R}}_\nabla$ matrix make this true (and multiplying the diagonal by 1+*nugget* sidesteps the need to actually equilibrate $\underline{\underline{R}}_\nabla$). In this situation, the worst case two-norm condition number occurs when one eigenvalue equals the maximum eigenvalue and $(n-1)$ eigenvalues equal the smallest eigenvalue. If we

- use $maxEigWorst + (n-1) \cdot minEigWorst = n$ and $minEigWorst = \mathrm{rcond}(\mathrm{R})_{L_2} \cdot maxEigWorst$ (and thus $maxEigWorst = \frac{n}{1.0+(n-1)\cdot \mathrm{rcond}(\mathrm{R})_{L_2}}$),
- assume that the reciprocal of the L_2 condition number, rcond$(\mathrm{R})_{L_2}$, equals the reciprocal of the L_1 condition number, rcond$(\mathrm{R})_{L_1}$ divided by $\sqrt{n}$, and

- pick a minimum allowed rcond(R)$_{L_2}$ that is a factor of $\sqrt{n}$ larger than the minimum allowed rcond(R)$_{L_1}$,

then we are able to determine the "smallest" nugget that will fix the worst case ill-conditioning from the following equation

$$nugget = \frac{(minAllowedRcondR_{L_2} \cdot maxEigWorst - minEigWorst)}{(1.0 - minAllowedRcondR_{L_2})}.$$

This "guarantee" assumes that the LAPACK fast estimate of rcond(R)$_{L_1}$ is accurate. However, since the confluence of all worst case assumptions used here is likely not realizable, this may still a be guaranteed safe value for the nugget. This approach to determining an small but "guaranteed" safe nugget requires at most two Cholesky decompositions; it should be compared to iterative approaches which could conceivably determine an even smaller nugget but have significantly greater computational cost. One could also assume that rcond(R)$_{L_2}$=0 and preemptively apply the resultant nugget which would require only a single Cholesky decomposition.

4 Experimental Procedure

This section describes the computer experiments used to assess the performance of the new GEK model presented in the preceding section relative to that of the Kriging model detailed in Section 2.3. The models were tested on four functions with differing degrees of smoothness. Two of the test functions were two dimensional; two were arbitrary dimensional and were examined in two, four, and eight dimensions. The models were compared on the basis of the error in their predictions. In the 2D case, this was done visually with plots. Root Mean Square Error (RMSE) and also Mean Absolute Error (MAE) were compared for the 2D, 4D, and 8D problems. The MAE results are not shown because they did not differ significantly from the RMSE.

4.1 Test Functions

The four test functions utilized in this work are Rosenbrock, Shubert, Herbie [16], and a smoothed version of Herbie. They are defined over the $[-2,2]^M$ hypercube. These are $M = 2$ dimensional functions but the Herbie and Smoothed Herbie functions are easily generalized to an arbitrary number of dimensions. The generalized Herbie functions have been examined for the 2, 4, and 8 dimensional cases. All four of the 2D test functions are plotted in Figure 2. These test functions were selected on the basis that they each have an explicit algebraic formula which makes them quick to evaluate and a different amount of smoothness. Here "smoothness" is used loosely; the functions are all differentiable but have different frequency content. More importantly, these functions have features that make them challenging optimization problems. Rosenbrock's analytical formula is

$$\mathrm{ros}(x_1,x_2) = 100 \cdot \left(x_2 - x_1^2\right)^2 + (1-x_1)^2.$$

Shubert's formula is

$$\mathrm{shu}(x_1,x_2) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1+i)\right)\left(\sum_{j=1}^{5} j\cos((j+1)x_2+j)\right).$$

The generalized (to M dimensions) Herbie function is

$$\mathrm{herb}(\underline{x}) = \prod_{k=1}^{M} w(x_k)$$

where

$$w(x_k) = \exp(-(x_k-1)^2) + \exp(-0.8(x_k+1)^2) - 0.05\sin\left(8\left(x_k+0.1\right)\right).$$

Lee et al. [16] used $M = 2$ and minimized the negative of this function. Although it is infinitely differentiable, the high frequency sine term effectively makes the Herbie function much less smooth and therefore a challenging problem for both optimization and surrogate modeling. To discern the effect of high frequency content on prediction quality, a modified/smoothed version of Herbie that omits the sine term was also examined. This is

$$\mathrm{herb}_{\mathrm{sm}}(\underline{x}) = \prod_{k=1}^{M} w_{sm}(x_k)$$

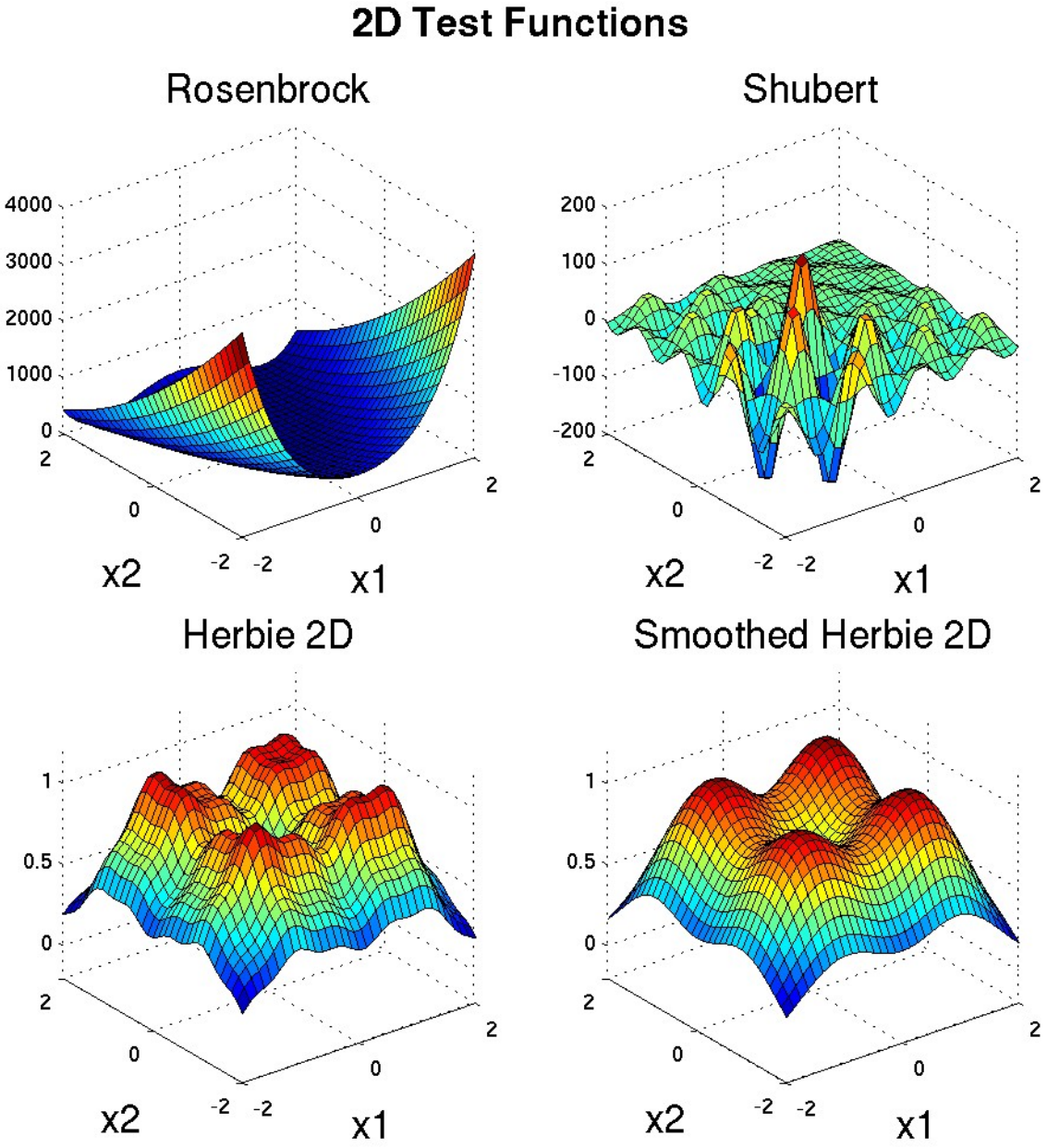

Figure 2. The 2D test functions

where

$$w_{sm}(x_k) = \exp(-(x_k-1)^2) + \exp(-0.8(x_k+1)^2).$$

Of these four 2D test functions, Rosenbrock is the smoothest followed by smoothed Herbie, Herbie, and Shubert in that order.

It was previously stated that to break the curse of dimensionality one must exploit:

1. Inexpensive derivative information
2. The smoothness of the response being modeled, and
3. Adaptive sampling.

For this paper, the first requirement is assumed to be met. Specifically it is assumed that a function value plus gradient simulation costs twice as much as a function value only simulation. The factor of two in computational cost can be significantly greater than that of a continuous adjoint implementation and is close to the typical cost of automatic differentiation. The second requirement, smoothness, was explored by selecting four test functions described above which have differing frequency content. The development of an appropriate adaptive sampling strategy is beyond the scope of this paper; this third requirement was instead mimicked by the choice of sample design and assuming that the adaptive sampling strategy generated only those points selected for the GEK emulator by the pivoted Cholesky approach described in Section 3.1.

The sample design for the four two dimensional test problems (Table 1 and Figures 3 through 12) was chosen as follows: 100 nested BOSLHS designs with 64 points were generated and the one for which GEK emulators of the Herbie function had the smallest sum of RMSE for the $N = 16,\ 32,$ and 64 cases was selected. This was done to obtain an optimized sequence of designs and thereby better mimic adaptive sampling. The term "nested" indicates a sequence of sample designs in which each member inherits/includes all points from earlier members in the sequence. BOSLHS, or Binning Optimal Symmetric Latin Hypercube Sampling, is a form of space-filling Latin Hypercube Sampling developed by Dalbey and Karystinos [6, 7]. Error for the 2D test problems was computed on a 33 by 33 grid of points separated by a distance of $1/8$ in each dimension. This is the grid shown in Figures 2 through 12.

For large N or N_V, it generally isn't feasible to build Kriging models from many different designs and pick the one with the smallest RMSE at a set of validation points. The computational cost of typical simulators also precludes such a strategy. Although the intent of using this strategy for the 2D test problems was to better mimic adaptive sampling, the author anticipates it being a point of contention for some readers. To address potential objections of this nature, a different strategy was employed to generate the 2D, 4D, and 8D sample designs that were used to study the effect of dimension on emulator prediction accuracy (Figures 13 and 14).

The prediction RMSE of a Kriging model tends to correlate well with the centered L_2 discrepancy, CD_2, (originally defined by Hickernell [14]) of the sample design that the model was built from. The discrepancy of a sample design in a hypercube input space is some norm, over all hyper-rectangle sub-volumes, of the difference between points per sub-volume and a uniform smearing of points. As such, discrepancy measures the sample design's uniformity or degree to which it is space-filling. Different choices of the norm lead to different definitions of discrepancy. CD_2 is a popular discrepancy that uses an L_2 norm over all hyper-rectangle sub-volumes with one corner at the center of the hypercube input space.

The low CD_2 nested 2D, 4D, and 8D LHS designs used to study the effect of dimension on emulator prediction RMSE were obtained in the following way:

1. N_{trial} random M dimensional LHS designs with $N = 2M$ points were generated and the design with the lowest CD_2 was selected. Candidate designs had one point placed in each of $2M$ 1D bins in each of the M dimensions and coordinates from the 1D designs were randomly permuted and paired to form M dimensional points.

2. N_{trial} random nested LHS extensions of the previous design with $2N$ points were generated and the one that resulted in the lowest CD_2 was selected. Each extension was generated by splitting each one dimensional bin in half, randomly placing a point in the half-bins without one and randomly pairing dimensions for the new points.

3. N was set to $2N$ and steps 2 and 3 were looped over until the design had the desired number of samples.

4. The points in the final design were slightly shifted to be at the center of their 1D bins.

When a design with large N and N_{trial} is generated in this way, it tends to have significantly lower CD_2, than the lowest CD_2 of N_{trial} randomly generated single trial designs with the same N. The build designs were generated with $N_{trial} = 20,000$. The 2D and 8D nested build designs had 2,048 points. The 4D build design had 4,096 points. The error was computed on validations designs with $N = 2^{14} = 16,384$ points using $N_{trial} = 100$; the smaller N_{trial} was used for the validation designs since the cost to compute CD_2 is $\mathcal{O}\left(N^2M\right)$. Step 4 in this strategy ensured that points present in the build designs could not appear in the validation designs.

The purpose of using high quality sample designs was to prevent sample design quality from confounding the results of the computational experiments without resorting to replication. The results presented below are for Kriging and GEK emulators with a reduced (main effects) quadratic trend function.

5 Results and Discussion

function	N	Kriging		Gradient Enhanced Kriging		
		$\mathrm{RM}(\mathrm{Var}(\hat{y}))$	RMSE	$\tilde{N}_\nabla/N_\nabla$	$\mathrm{RM}(\mathrm{Var}(\hat{y}))$	RMSE
Rosenbrock	16	59.24	83.55	42/48	0.6934	1.780
	32	0.6133	1.134	85/96	0.3580	0.4255
	64	0.6358	0.7477	121/192	0.1212	0.2697
Shubert	16	23.48	65.66	48/48	41.42	41.33
	32	40.65	37.22	96/96	32.29	28.69
	64	35.87	32.62	192/192	9.146	5.448
Herbie	16	0.1628	0.1232	48/48	0.09718	0.07054
	32	0.06470	0.07166	96/96	0.05494	0.05355
	64	0.04660	0.05519	192/192	0.01445	0.01155
Smoothed	16	0.1569	0.1167	48/48	0.02519	0.005703
Herbie	32	0.03355	0.05015	96/96	0.001858	0.0009920
	64	0.03886	0.002189	191/192	0.0001836	0.0002322

Table 1. Comparison of predicted and actual Root Mean Square Error (RMSE) for the Kriging and Gradient Enhanced Kriging (GEK) emulators in Figs 3, 7, 9, and 11. The columns labeled RMSE were calculated as the RMS of the difference between adjusted mean and truth at the nodes of the 33 by 33 grid plotted in the figures. The columns labeled $\mathrm{RM}(\mathrm{Var}(\hat{y}))$ were calculated as the square root of the mean of the adjusted variance on the same grid; these columns can be interpreted as the emulators' prediction of RMSE. N_∇ was the number of equations available to the GEK emulator; $\tilde{N}_\nabla$ was the number of equations that GEK actually used. For all cases examined, the RMSE of GEK (with equations discarded according to Section 3.1) was significantly less than that of Kriging with the same N.

Table 1 compares the predicted and true RMS errors of Kriging and Gradient Enhanced Kriging for the 2D test problems. It summarizes some of the information contained in Figures 3, 7, 9, and 11 which will be discussed in more detail below. In order to fairly evaluate the merits of GEK vs. Kriging, a GEK emulator built from N points should be compared to a Kriging emulator with $2N$ points. If an unfair advantage is given to either method, comparing on this basis would favor Kriging by giving it the advantage of a possibly higher computational budget. Adaptive sampling for the GEK emulators can be approximated by only counting those simulations selected by pivoted Cholesky towards its computational cost. This approximation assumes a sequence of GEK emulators with successively larger numbers of points where the new points were chosen based on the predictions of earlier GEK emulators.

Table 1 indicates that, at least for small numbers of points, GEK with N points performs about as well as Kriging with $2N$ points for the Shubert and 2D Herbie functions. That GEK does not do worse than Kriging for the same computational budget on these non-smooth/high-frequency functions is important. For the 2D Smoothed Herbie function, GEK significantly outperforms Kriging on the same cost basis. For these small numbers of points, GEK performs about as well or better than Kriging on Rosenbrock if the points GEK discards are counted against it. Since GEK discarded a significant number of points for Rosenbrock, the comparison would be even more favorable to GEK if (to approximate adaptive sampling) those points were not counted against GEK's computational budget.

Figures 3 through 12 show the performance of Kriging and GEK for the 2D test problems in more detail. They all follow the same convention: a 3 by 3 array of subplots with a color bar. The leftmost column shows the 2D sample design. The surface plotted in the rightmost column is the GEK emulator's adjusted mean prediction at a 33 by 33 grid of points. The surface is colored by the absolute error between the true response and prediction where white is low error, dark red is high error. All plots for an individual problem, e.g. Rosenbrock, use a single color scaling but that scaling is different from problem to problem. The middle column of plots show either the prediction colored by error for some other emulator or GEK's estimate of the uncertainty in its own prediction. Kriging is the other emulator plotted in Figures 3, 7, 9 and 11. The square root of the adjusted variance is used as the uncertainty estimate in Figures 6, 8, 10, and 12. Ideally, the true error should be less than a factor of two, or at most three, larger than this estimate. If not, the emulator built from that set of samples is overconfident in its ability to predict the true response.

Figures 3, 4, 5, and 6 deal with the Rosenbrock test problem. From Figure 3, it is readily apparent that GEK outperforms Kriging for the same number of samples but as indicated above that comparison would unfairly favor GEK over Kriging. However, GEK with $N = 16$ points does about as well as Kriging with $N = 32$ points over all. In this comparison, GEK does significantly better almost everywhere except near the $\underline{x} = (-2,2)$ and $(2,-2)$ corners. As can be seen from the upper left subplot, these corners are significantly outside the convex hull of the $N = 16$ point sample design while the $N = 32$ point design (middle left subplot) has 2 additional points near each of these corners. Note that GEK with $N = 32$ points does better, even in the corners, than Kriging with $N = 64$ points despite Kriging having points even closer to the corners. The interpretation is fairly straight-forward. GEK is significantly better at both interpolation and extrapolation than Kriging, but Kriging can still outperform GEK when extrapolating a shorter distance.

One reason that GEK is better able to interpolate and extrapolate is that its ability to discard poorly spaced points allows it to remain well conditioned with substantially longer correlation lengths. In each of the subplots, the entire set of sample points are plotted as solid blue circles. In the left column's sample design plots, points that were completely omitted by GEK are covered by red squares while a downward pointing green triangle indicates a location where some or all of the gradient equations were omitted but the function value was retained.

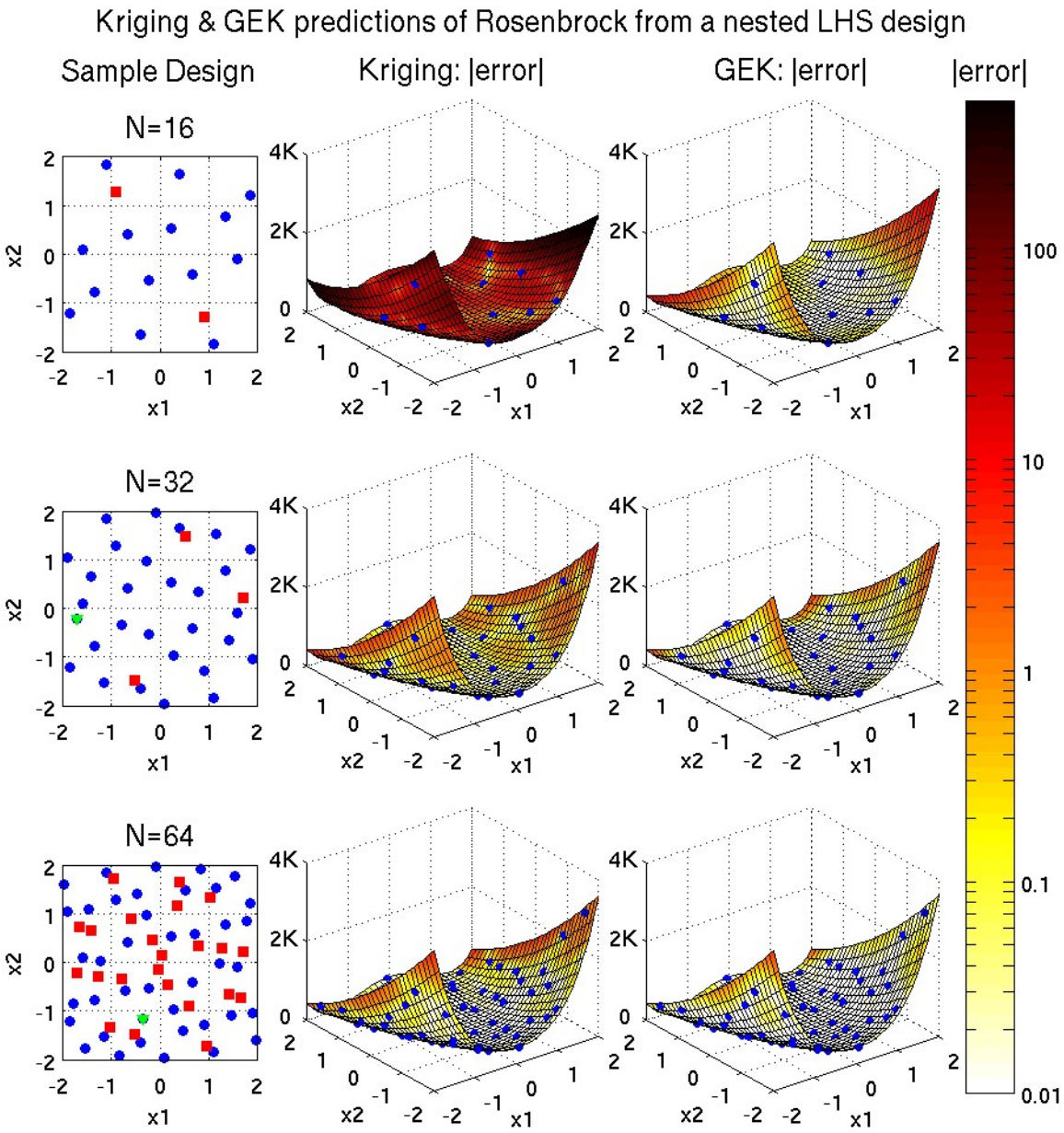

Figure 3. Comparison of the (adjusted mean) prediction of the Rosenbrock function by: (middle) Kriging that retains all points vs. (right) GEK when equations are discarded according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its pointwise error magnitude. All points are plotted as blue circles. In the left column, points are covered by a red square or green triangle. Red squares are points where the response value and its derivatives were discarded by the right column's emulator. Green triangles are points where some or all derivative equations were discarded but the response value was retained.

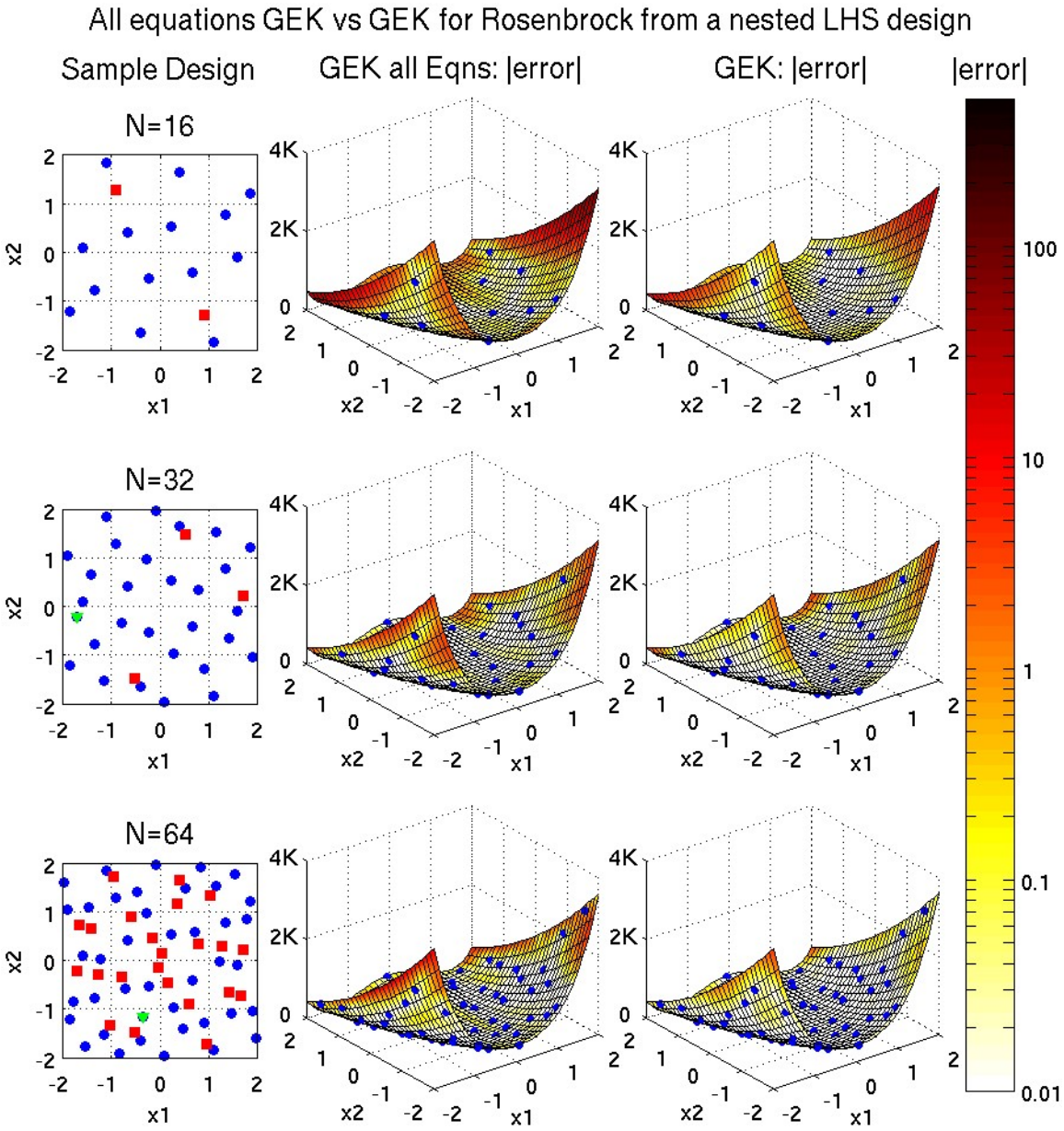

Figure 4. Comparison of the (adjusted mean) prediction of the Rosenbrock function by GEK emulators when: (middle) all equations are retained vs. (right) equations are discarded according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its point-wise error magnitude. All points are plotted as blue circles. In the left column, points are covered by a red square or green triangle. Red squares are points where the response value and its derivatives were discarded by the right column's emulator. Green triangles are points where some or all derivative equations were discarded but the response value was retained.

Figure 4 compares how well GEK predicts Rosenbrock when it is forced to retain all equations (middle column) vs. when it is allowed to discard points according to the strategy described in Section 3.1 (right column). As can be seen, GEK performs significantly better for all three sample design sizes when it is allowed to discard points, showing the benefit of using pivoted Cholesky to discard points.

Figure 5 compares an alternate strategy for discarding points (middle column) to the strategy described in Section 3.1 (right column). The alternate strategy only differs in this way: if the Kriging (not GEK) correlation matrix is ill-conditioned for a particular set of correlation lengths, then function values, but no derivative equations, will be used to build it and calculate its per equation log likelihood during the optimization over correlation lengths. The top two rows, where $N = 16$ and $N = 32$ respectively, are identical. However, the GEK discard strategies differed for the bottom row where $N = 64$. In this row, the alternate GEK in the middle column discarded four whole points (indicated by red squares in the bottom left subplot) and all gradient equations at all other locations (indicated by the downward pointing green triangles). For this $N = 64$ point case, the alternate GEK discard strategy is therefore equivalent to Kriging that is allowed to discard sample points. Note that the default GEK strategy with $N = 32$ does better than Kriging with $N = 64$ even when Kriging (the alternate GEK strategy) is allowed to discard points using the same methodology.

Figure 6 compares GEK's estimate of the uncertainty in its prediction (middle column) of Rosenbrock to the true error. This estimate is fairly accurate both qualitatively and quantitatively. It only slightly under predicts the error in the corners of the domain outside the sample design's convex hull.

Figures 7 and 8 deal with the Shubert function which is the least smooth of the four test functions. It consists entirely of high frequency content with zero trend in the mean. The amplitude of the oscillation does have a trend but that is not explicitly modeled by the Kriging or GEK implementations used here. As can be seen from the absence of red squares and green triangles in the left column of Figure 7, the pivoted Cholesky strategy described in Section 3.1 chose to retain all of the function value and derivative equations. GEK retained a different set of equations for the Shubert and Rosenbrock functions because their output differed. For Rosenbrock, the correlation lengths with the maximum per equation log likelihood required that equations be discarded to remedy the ill-conditioning of the correlation matrix. For Shubert, the correlation lengths with the maximum per equation log likelihood were short enough that no equations needed to be discarded.

In Table 1, Kriging with $N = 32$ points has slightly lower RMSE than GEK with $N = 16$ points. This summary statistic is a bit misleading regarding the relative quality of the Kriging and GEK emulators. Figure 7 indicates that GEK with $N = 16$ points (upper right subplot) has detected that Shubert has some underlying oscillatory nature that Kriging with $N = 32$ points (the middle row middle column subplot) is not yet aware of. For this case, Kriging predicted an essentially flat surface with some point-wise deviations. Kriging's slightly lower RMSE resulted from one

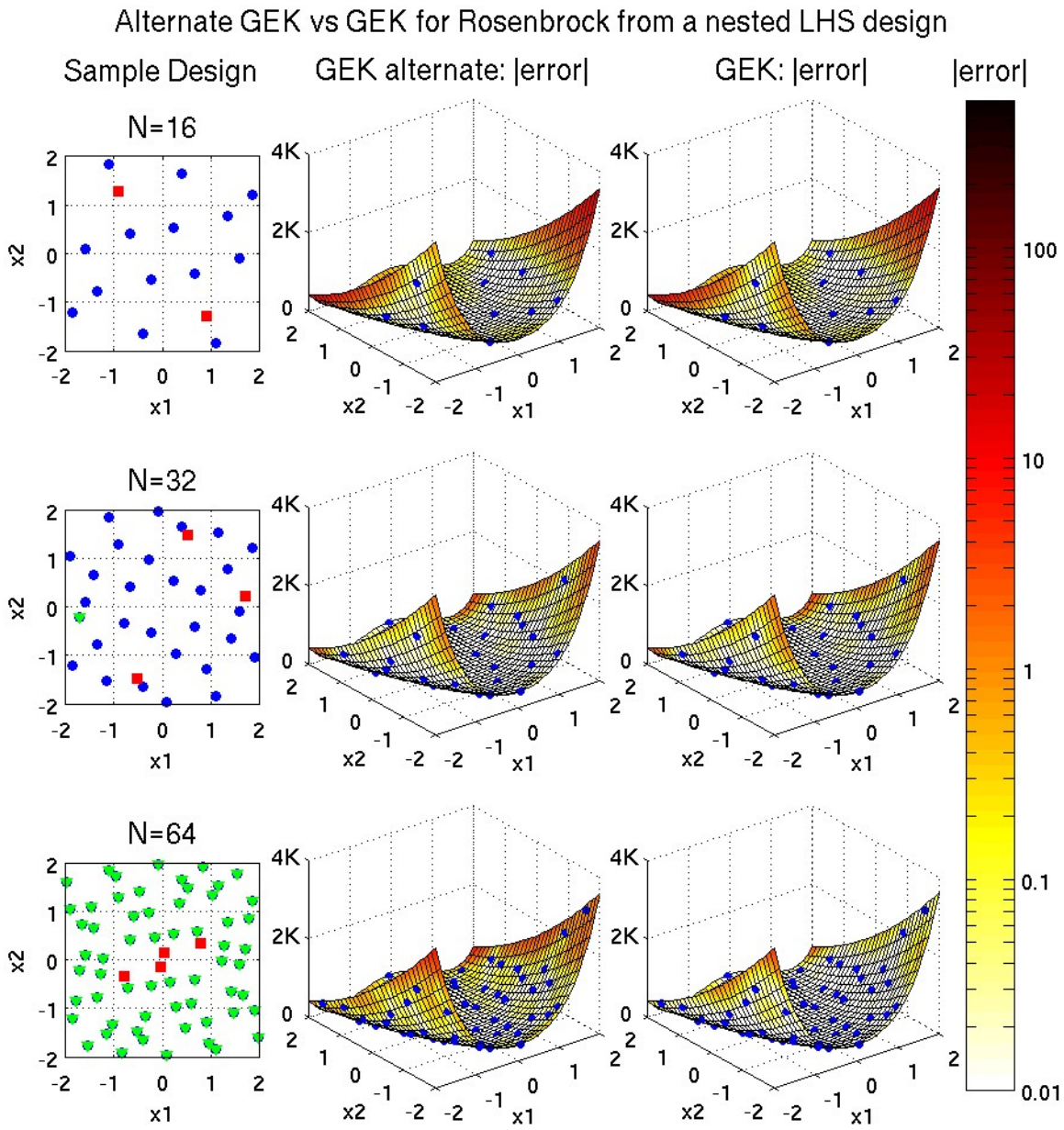

Figure 5. Comparison of the (adjusted mean) prediction of the Rosenbrock function by GEK emulators when equations are discarded: (middle) by the "alternate" method vs. (right) according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its point wise error magnitude. All points are plotted as blue circles. In the left column, points are covered by a red square or green triangle. Red squares are points where the response value and its derivatives were discarded by the middle column's emulator. Green triangles are points where some or all derivative equations were discarded but the response value was retained.

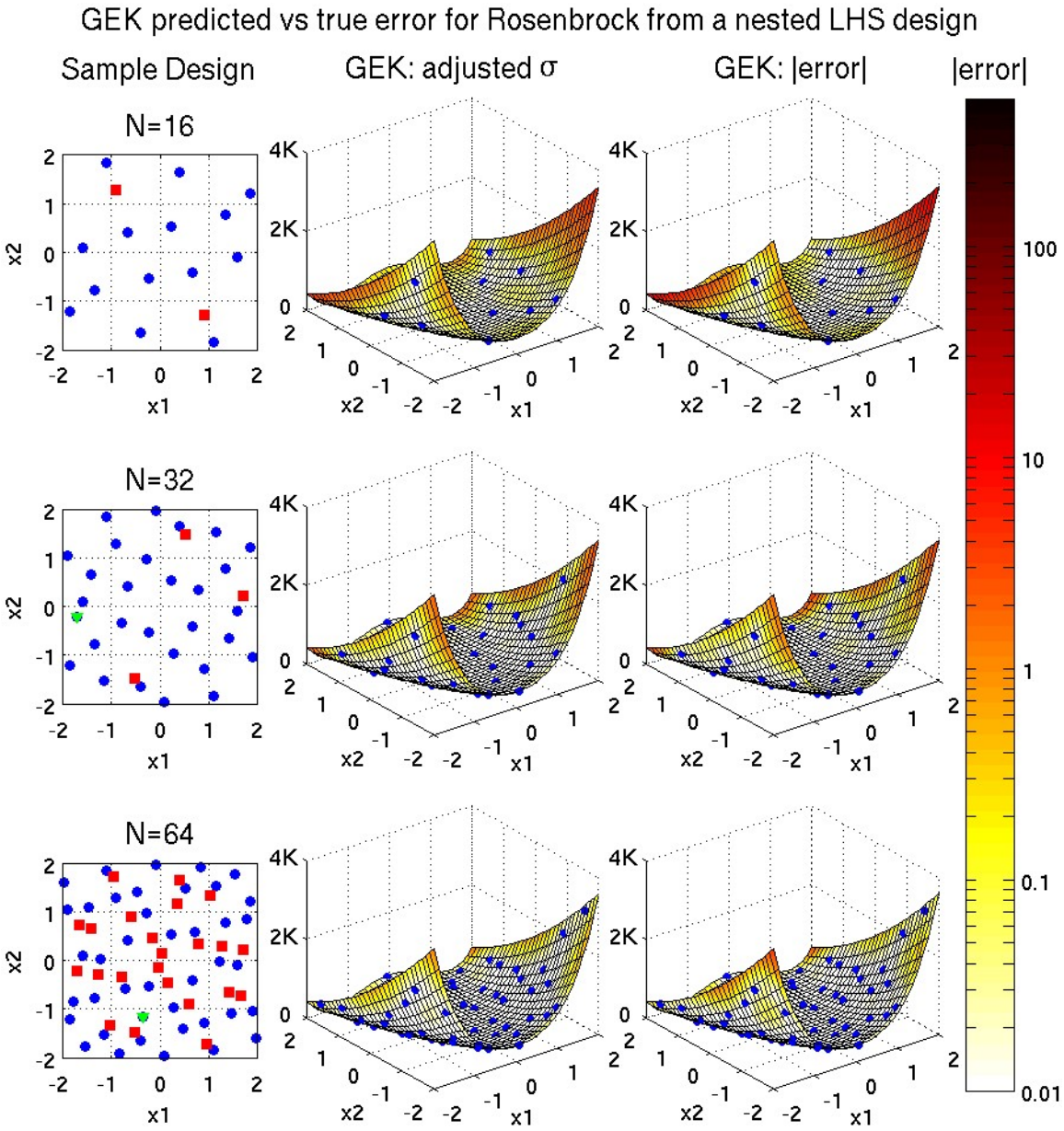

Figure 6. Comparison of (middle) error estimate (adjusted standard deviation) vs. (right) error magnitude in the (adjusted mean) prediction of the Rosenbrock function by GEK when equations are discarded according to Sec 3.1. A predicted surface's color indicates its error estimate or true error. Emulators were built from the nested BOSLHS design in the left column. All points are plotted as blue circles. On the left, points are covered by a red square or green triangle. Red squares are points where the response value and its derivatives were discarded by the right column's emulator. Green triangles are points where some or all derivative equations were discarded but the response value was retained.

of the new sample points landing near the bottom of a particularly deep downward spike. GEK with $N = 32$ points has a fairly well defined notion of Shubert's oscillatory behavior. This allowed the $N = 32$ point GEK emulator to predict Shubert with slightly lower RMSE than Kriging with $N = 64$ points; the later still relied on point-wise deviations from the mean.

At this time, it is important to make a distinction about **how** Kriging and GEK managed to obtain roughly the same RMSE for a function with high frequency content. Kriging filled the space and was at the mercy of the curse of dimensionality. GEK succeeded because its Gaussian process error model had greater ability to discover localized trends, an approach that has less dependence on the problem's dimensionality. The qualitative difference in the behavior of GEK and Kriging is not well reflected in the RMSE values listed in Table 1. This implies that there is some (for lack of a better term) "minimum threshold" number of points, which depends on the smoothness of the true response and the number of dimensions, that must be exceeded before GEK's advantage over Kriging will be reflected in summary statistics such as the RMSE.

Figure 8 compares GEK's estimate of the uncertainty in its prediction (middle column) of Shubert to the true error. For $N = 16$ and $N = 32$ this estimate is a fairly uniform shade of yellow. This results from GEK detecting that Shubert has a nearly constant least squares trend function plus substantial high frequency, that is short wave or correlation length, content. That GEK's predicted RMSE in Table 1 is fairly close to the true RMSE indicates that it has a fairly good notion of the amplitude of that high frequency content. For $N = 64$ points there is little error in GEK's prediction of Shubert function. Furthermore, GEK is able to accurately estimate the remaining error. This is indicated by fairly uniform white coloring except near a few of the boundaries where it accurately predicts a slightly greater error magnitude with a faint yellow color.

Figures 9 and 10 deal with the 2D Herbie function. Herbie has fairly substantial middle and high frequency content, making it a bit more smooth than Shubert but significantly less so than Rosenbrock (low frequency content). As was the case for Shubert, the GEK emulators in Figure 9 and 10 retained all of the available equations. Table 1 shows that for 2D Herbie, GEK with $N = 16$ and $N = 32$ points has slightly lower RMSE than Kriging with $N = 32$ and $N = 64$. For 2D Herbie, the RMSE is fairly consistent with what is shown in the plots. This was because $N = 32$ points was enough for Kriging to start to pick up 2D Herbie's middle frequency content. By comparison, the upper right subplot of 9 hints that GEK with $N = 16$ points is starting to pick up some of 2D Herbie's high frequency content (the $x_2 \approx -1$ edge ridges on the peaks); this becomes a bit more visible for GEK with $N = 32$ points (the middle right subplot).

Although it is not one of the emulators being compared, it should be noted that the $N = 16$ point Kriging emulator essentially predicts point deflections from a mean (as it did for the $N = 32$ and $N = 64$ cases for Shubert). It is noteworthy that, despite this qualitative difference in behavior over the number of sample points in the build design, the 2D Herbie RMSE for $N = 16$ point Kriging is fairly consistent with the rest of its RMSE sequence in Table 1. This re-emphasizes that RMSE is not a good metric for revealing qualitatively different behaviors of emulators with low numbers of

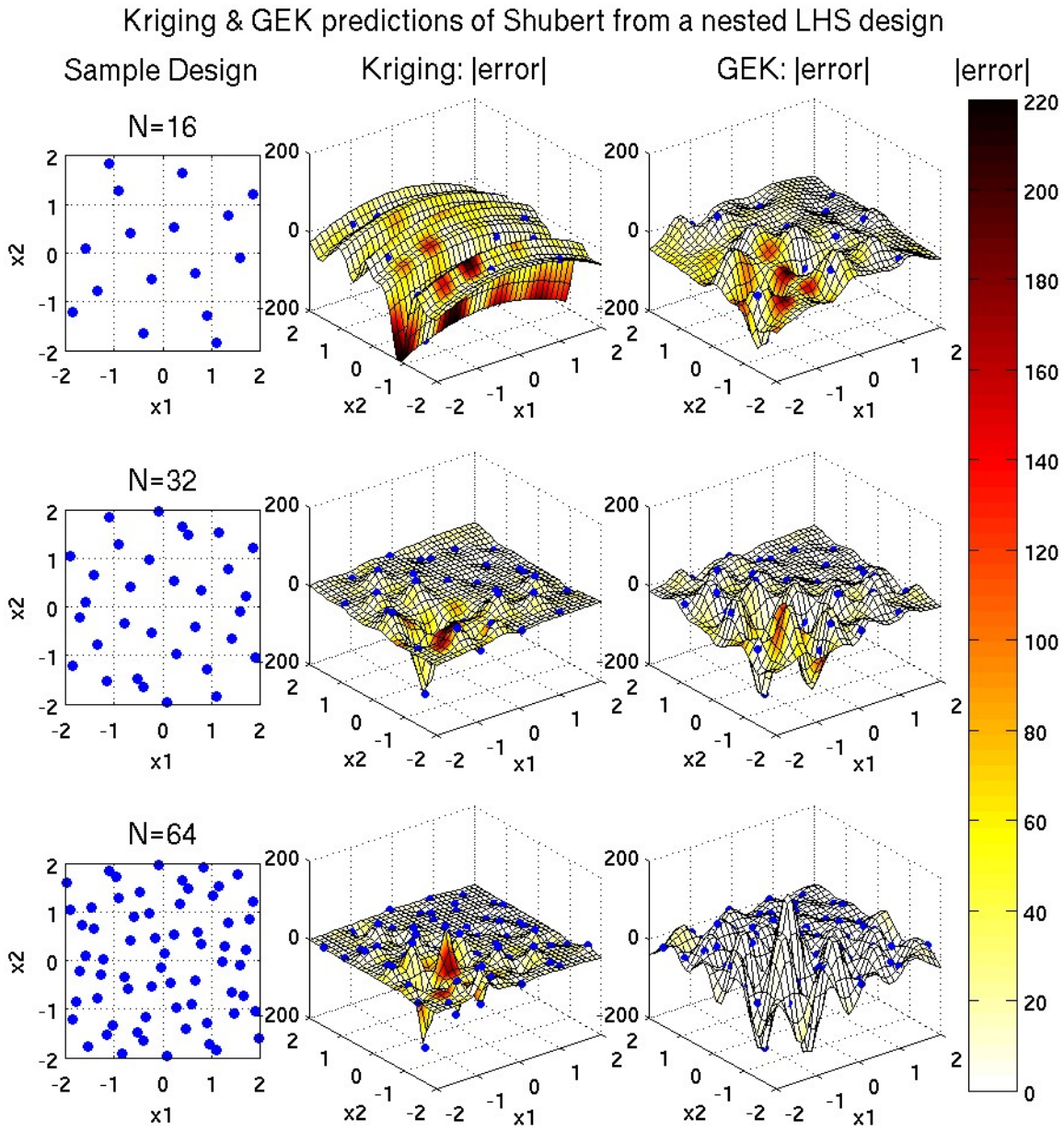

Figure 7. Comparison of the (adjusted mean) prediction of the Shubert function by: (middle) Kriging that retains all points vs. (right) GEK when equations are discarded according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its point-wise error magnitude. In this case, the method in Section 3.1 did not discard any equations.

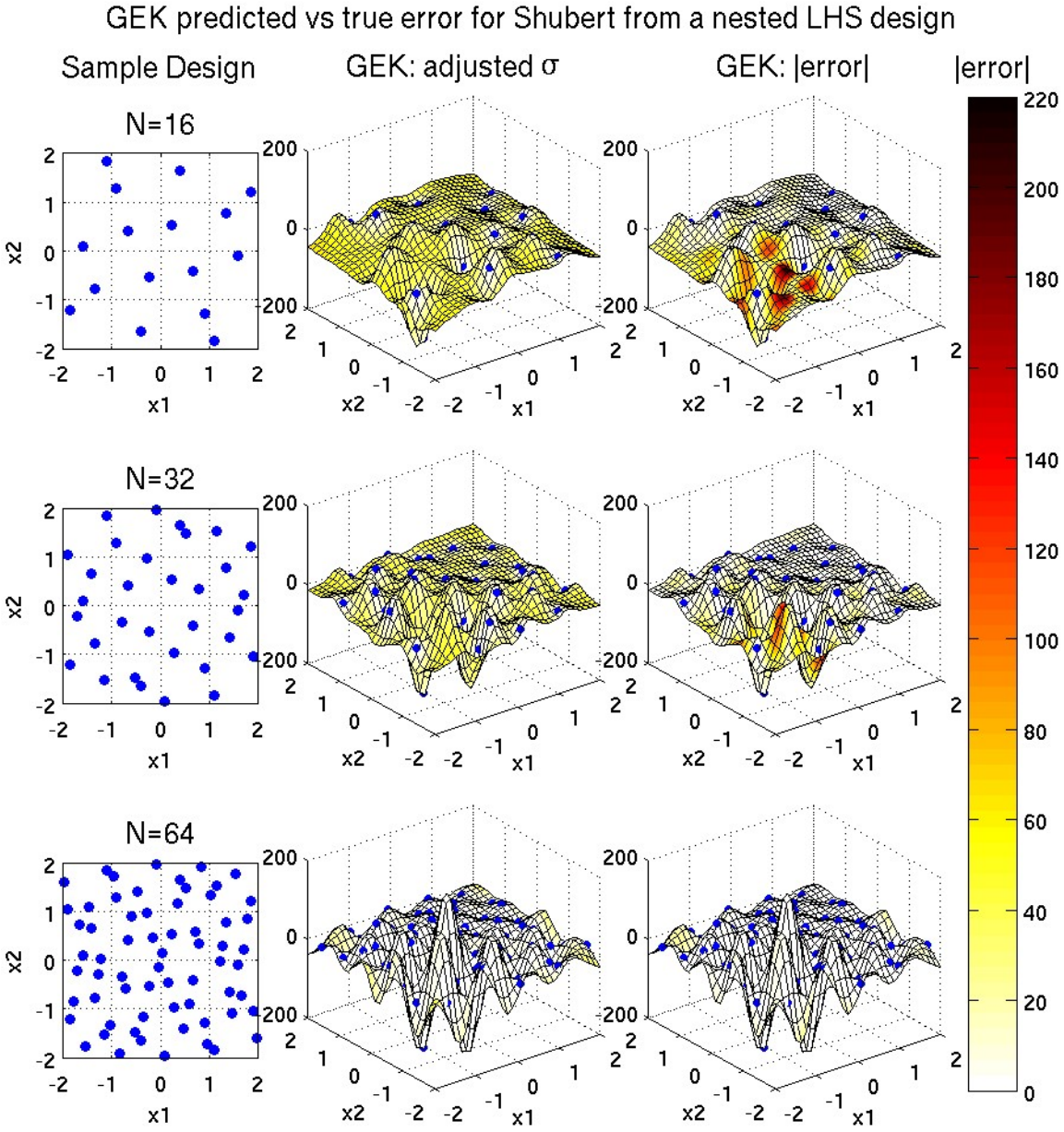

Figure 8. Comparison of the (middle) error estimate (adjusted standard deviation) vs. (right) error magnitude in the (adjusted mean) prediction of the Shubert function by GEK when equations are discarded according to Section 3.1. The point-wise error estimate or true error magnitude is indicated by the color of the predicted surface. Emulators were built from the nested BOSLHS design in the left column. In this case, the method described in Section 3.1 did not discard any equations.

points; mean absolute error fared about as well as RMSE, perhaps marginally better.

In Figure 10, GEK's estimate of the unknown true error is reasonable in most places. For $N = 16$ and $N = 32$ points GEK tends to over predict the extrapolation error and under predict interpolation error in a few larger internal voids. At that point GEK has not yet entirely distinguished between the middle and high frequency content of 2D Herbie. However, GEK with $N = 64$ points does an excellent job of both predicting the true response and estimating the uncertainty in its prediction.

Figures 11 and 12 deal with the 2D Smoothed Herbie function. This is 2D Herbie minus the high frequency sine component which leaves only the middle frequency content. This makes 2D Smoothed Herbie the second smoothest test problem considered in this work. For this problem, the RMSE listed in Table 1 is in complete agreement with Figure 11 that GEK performs dramatically better than Kriging. Figure 12 shows that the $N = 16$ points GEK's error estimate for 2D Smoothed Herbie is very conservative, significantly over-estimating the true error in its prediction. However, with $N = 32$ or more points GEK is highly accurate in its prediction of the true error.

Figures 13 and 14 examine whether GEK plus adaptive sampling can be used to break the curse of dimensionality for the M dimensional Herbie and Smoothed Herbie functions. Developing a suitable adaptive sampling strategy was beyond the scope of this work. Instead adaptive sampling was mimicked by not counting the cost of simulations discarded by GEK via the pivoted Cholesky approach described in Section 3.1 against GEK's simulation budget. To calculate the approximate simulation cost, it was also assumed that a simulation which produces a function value plus a gradient costs twice as much as a simulation that produces only the function value. On this basis, if a GEK RMSE curve is lower than than the corresponding Kriging RMSE curve, it would be considered evidence supporting that the curse of dimensionality can be broken or at least ameliorated by such an approach.

For all cases plotted in Figures 13 and 14, the RMSE for Kriging and GEK are approximately equal for low numbers of points. For all but the 8D Herbie case, the GEK RMSE begins to decrease faster than that of Kriging after a sufficient simulation budget has been expended. This can be considered evidence that it is possible to break the curse of dimensionality using Gradient Enhanced Kriging provided that inexpensive derivatives are available, the function is sufficiently smooth, and a suitable adaptive sampling strategy is employed.

The author believes that if a sufficient number of additional simulations were performed, the 8D Herbie case would exhibit the same trend as 2D and 4D Herbie and 2D, 4D, and 8D Smoothed Herbie, and the largest 8D build design tested for Herbie in Figure 13 hints that this is true. However, a larger sample design was not attempted because of the expense of constructing GEK emulators with a larger number of points.

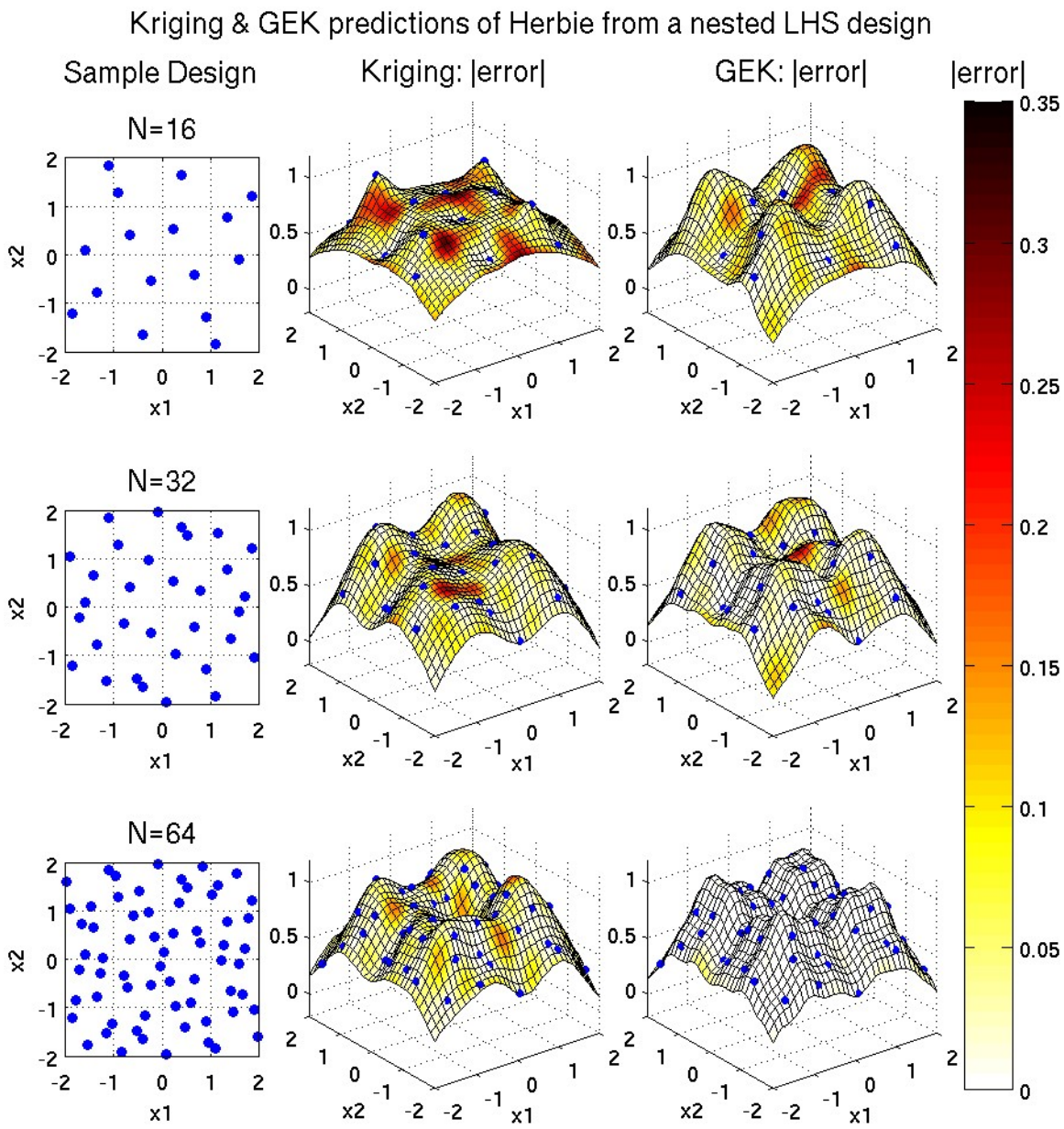

Figure 9. Comparison of the (adjusted mean) prediction of the 2D Herbie function by: (middle) Kriging that retains all points vs. (right) GEK when equations are discarded according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its point-wise error magnitude. In this case, the method described in Section 3.1 did not discard any equations.

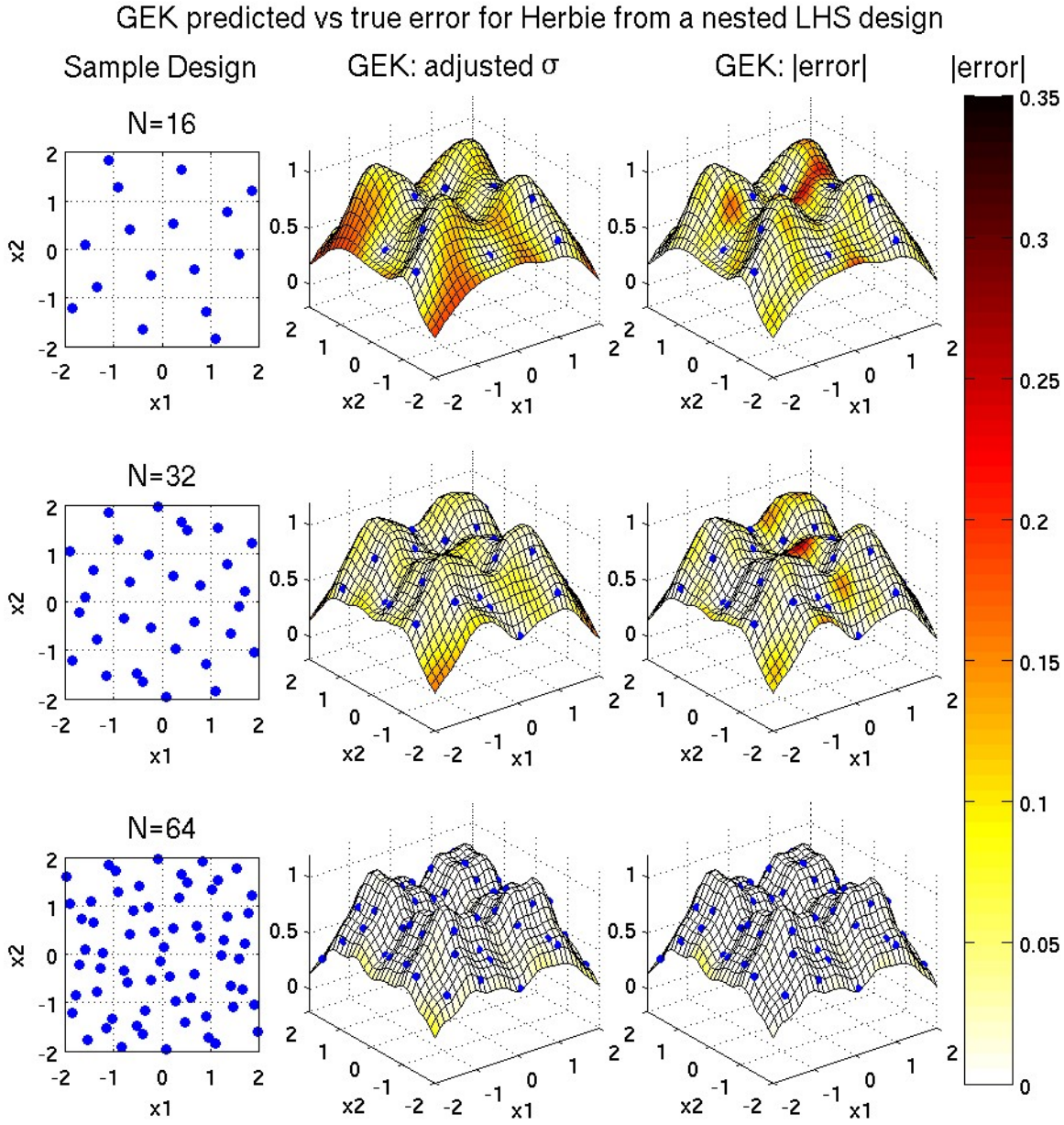

Figure 10. Comparison of the (middle) error estimate (adjusted standard deviation) vs. (right) error magnitude in the (adjusted mean) prediction of the 2D Herbie function by GEK when equations are discarded according to Section 3.1. The point-wise error estimate or true error magnitude is indicated by the color of the predicted surface. Emulators were built from the nested BOSLHS design in the left column. In this case, the method described in Section 3.1 did not discard any equations.

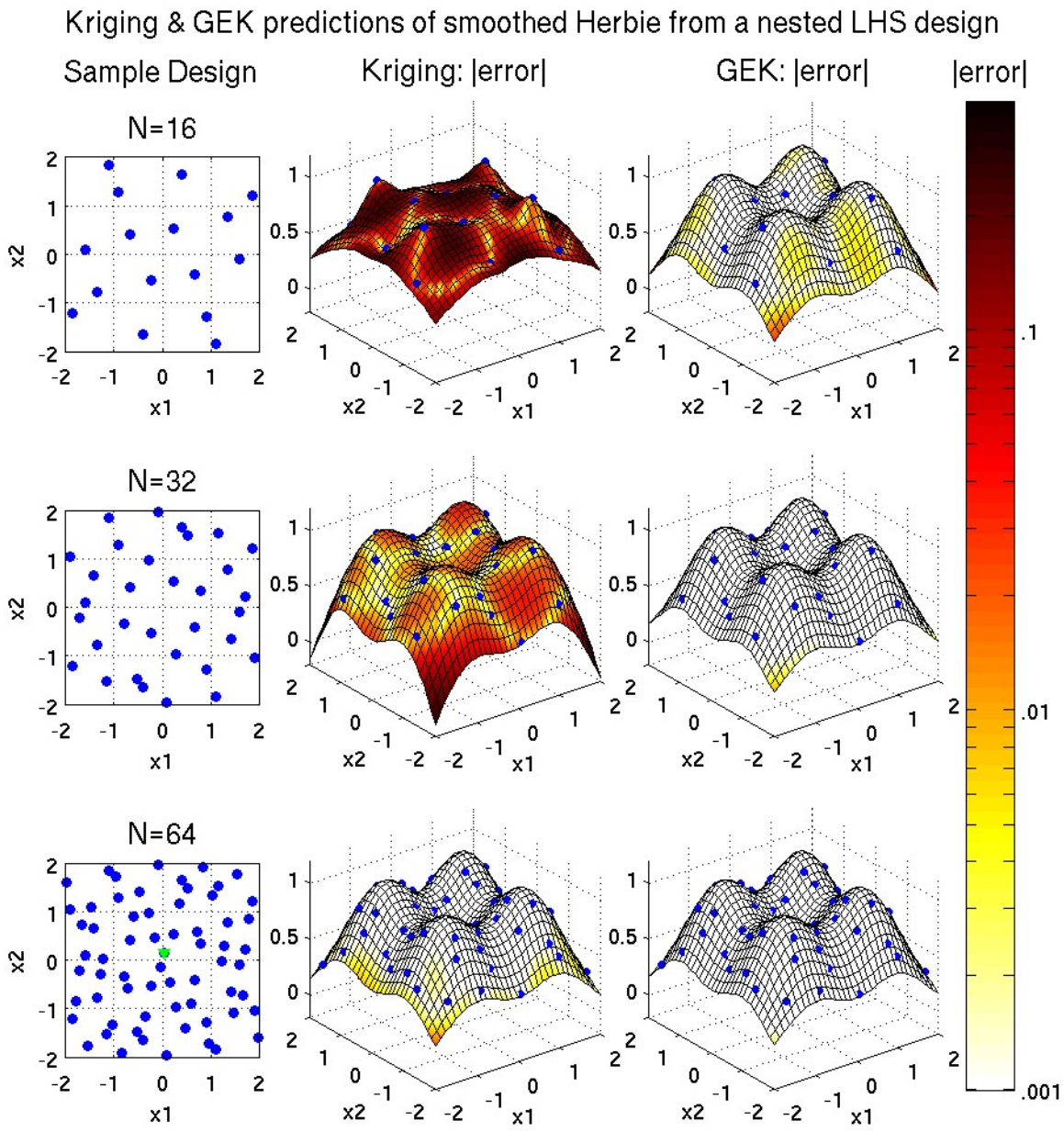

Figure 11. Comparison of the (adjusted mean) prediction of the 2D smoothed Herbie function by: (middle) Kriging that retains all points vs. (right) GEK when equations are discarded according to Section 3.1. Emulators were built from the nested BOSLHS design in the left column. The color of a predicted surface shows its point-wise error magnitude. In this case, the method described in Section 3.1 only discarded one derivative equation from one point in the 64 point sample design. This point is represented by the green triangle in the lower left subplot.

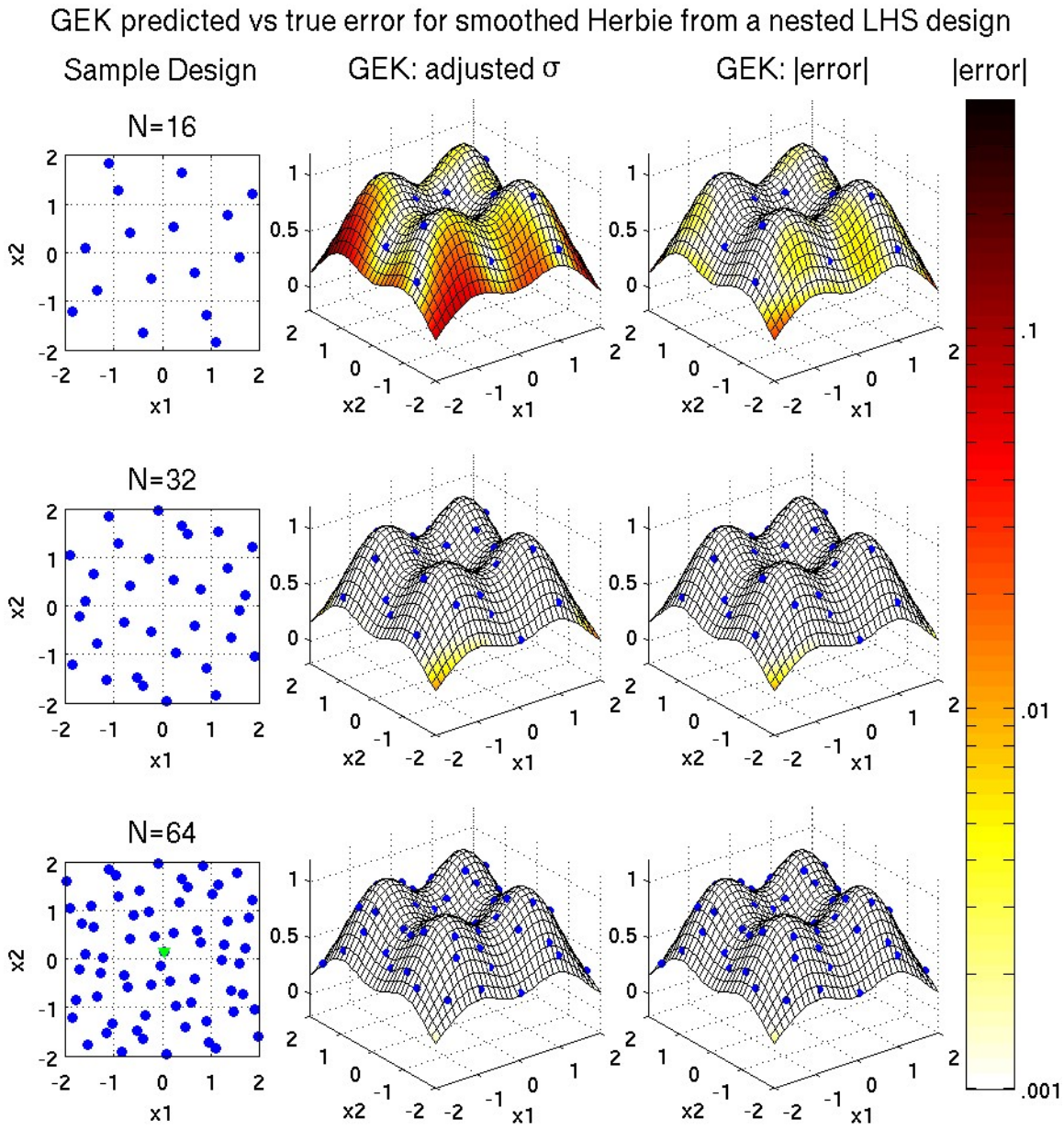

Figure 12. Comparison of the (middle) error estimate (adjusted standard deviation) vs. (right) true error magnitude in the (adjusted mean) prediction of the 2D smoothed Herbie function by GEK when equations are discarded according to Section 3.1. The point-wise error estimate or true error magnitude is indicated by the color of the predicted surface. Emulators were built from the nested BOSLHS design in the left column. In this case, the method described in Section 3.1 only discarded one derivative equation from one point in the 64 point sample design. This point is represented by the green triangle in the lower left subplot.

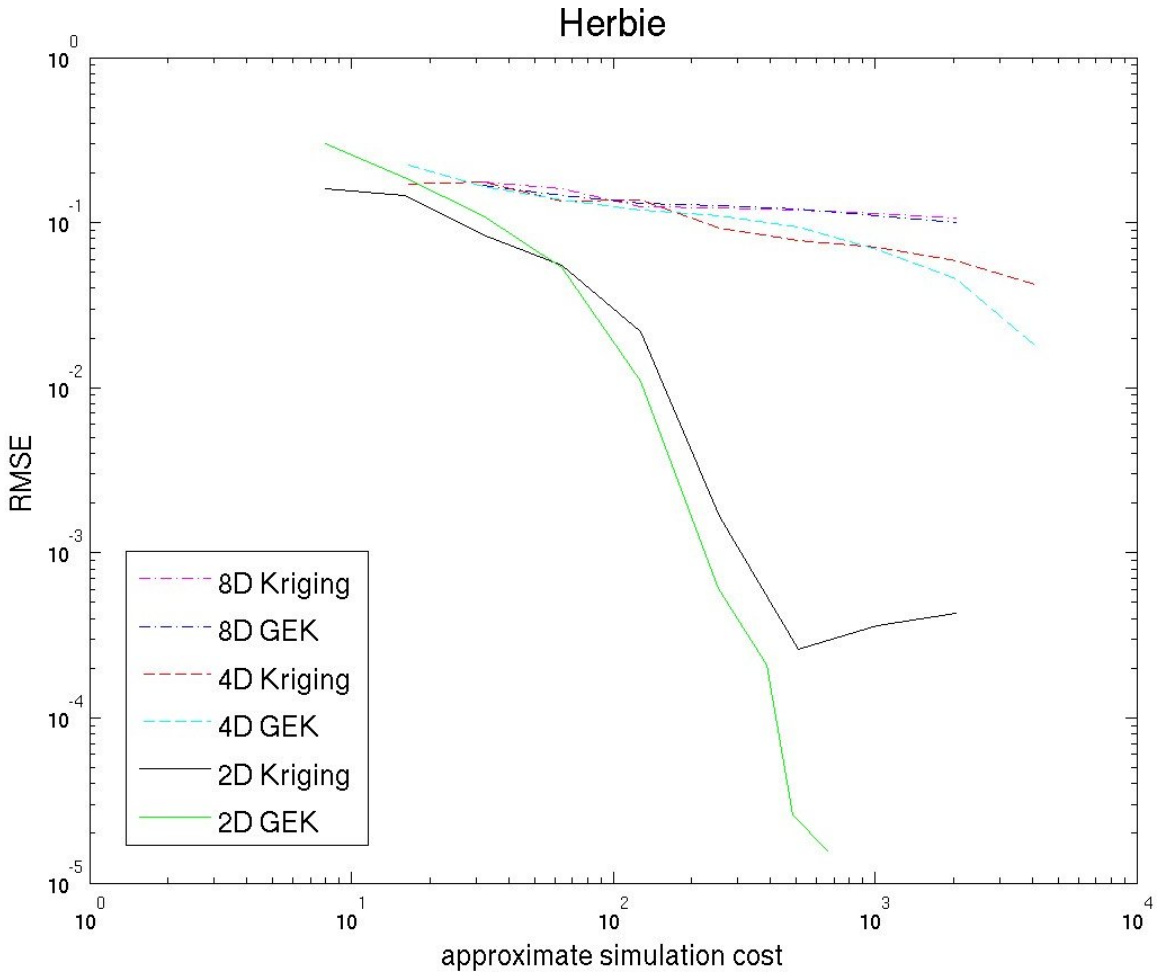

Figure 13. Root Mean Square Error in Kriging and GEK prediction of the 2D, 4D, and 8D Herbie function vs. the approximate cost of the simulations needed to build the emulators. To calculate simulation cost it was assumed that 1) a simulation that produces a function value plus gradient costs twice as much a simulation that produces only the function value, and 2) that adaptive sampling for GEK could be approximated by not counting the cost of simulations discarded by pivoted Cholesky as described in Section 3.1.

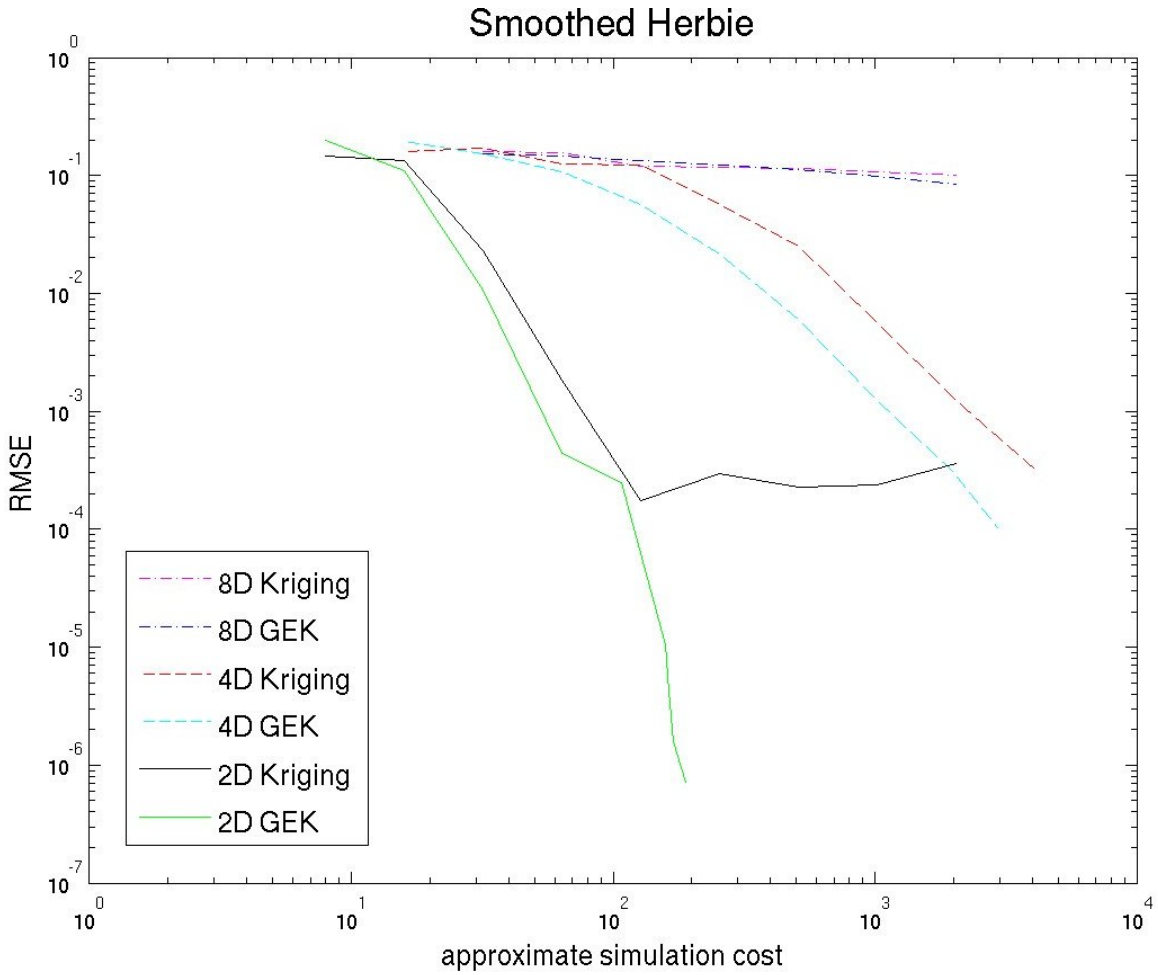

Figure 14. Root Mean Square Error in Kriging and GEK prediction of the 2D, 4D, and 8D Smoothed Herbie function vs. the approximate cost of the simulations needed to build the emulators. To calculate simulation cost it was assumed that 1) a simulation that produces a function value plus gradient costs twice as much a simulation that produces only the function value, and 2) that adaptive sampling for GEK could be approximated by not counting the cost of simulations discarded by pivoted Cholesky as described in Section 3.1.

For $M = 8$ dimensions, constructing the GEK emulator costs about 729 times as many flops as constructing a Kriging emulator (excluding the cost of evaluating the simulator) with the same number of points, N, or about 91 times more expensive than Kriging with half as many points (which is roughly the same simulation budget). This is not precise because the optimizer will choose different correlation lengths to examine for GEK and Kriging. The result is that constructing GEK emulators is cost effective when N and M are sufficiently small relative to the cost of evaluating the simulator. Using a local gradient based optimizer to select the correlation lengths would be significantly less expensive for large M but is also less robust than a global optimizer. Multi-start local optimization might be a reasonable compromise. The author expects that at least $2M + 1$ starting locations would be needed for robustness. These $2M + 1$ starting locations correspond to the end points (on the boundary of the search region) of an orthogonal set of axes that intersect at $L_k = d$ for all k, plus one starting location at the intersection.

6 Conclusions

The "curse of dimensionality" states that the number of evaluations necessary to represent an unknown function with a given level of fidelity is exponential in the number of its input dimensions. One approach to breaking, or at least ameliorating, this curse is to exploit the combination of

1. Inexpensive derivative information (e.g. from automatic differentiation or adjoint techniques)
2. Smoothness (if present) in the unknown function, and
3. Adaptive sampling.

Surrogate models, in particular Kriging emulators, are an appropriate way to combine and facilitate these three essential ingredients.

The author has detailed the implementation of the Kriging emulator in DAKOTA version 5.1 [1] and the Gradient Enhanced Kriging (GEK) emulator in DAKOTA version 5.1+. Attention was focused on the features central to the efficiency and robustness of these implementations. The principal contribution of this work is a novel, effective, and efficient approach to deal with the ill-conditioning of the GEK "correlation" matrix, $\underline{\underline{R_\nabla}}$ based on a pivoted Cholesky factorization of *Kriging's* function value only correlation matrix $\underline{\underline{R}}$ which is a small sub-matrix within GEK's $\underline{\underline{R_\nabla}}$ matrix. The approach discards sample points/equations that contribute the least new information to $\underline{\underline{R_\nabla}}$. Since these points contain the least new information, they are the ones which when discarded are both the easiest to predict and provide maximum improvement of $\underline{\underline{R_\nabla}}$'s conditioning. Prior to this work, the handling of ill-conditioned correlation matrices was a major, perhaps the principal, unsolved challenge necessary for robust and efficient GEK emulators.

Numerical results demonstrate that predictions made by GEK when some equations are discarded by this pivoted Cholesky approach can be significantly more accurate than when all equations are retained. Although adaptive sample design is beyond the scope of this work, the pivoted Cholesky approach presented here has potential application in the forward and adaptive design of computer experiments.

Numerical results also indicate that, when the effect of adaptive sampling is approximated by omitting samples discarded by the author's pivoted Cholesky approach from GEK's simulation budget, the combination of inexpensive derivative information and "adaptive sampling" is sufficient to break or at least ameliorate the curse of dimensionality. An important caveat to this statement is that there is an initial threshold number of simulations, which depends on the smoothness and dimension of the function being modeled, that must be exceeded before GEK's advantage over Kriging becomes apparent in summary statistics such as the root mean square error. However,

even in the least smooth case examined here, GEK performed about as well or better than Kriging when compared on the basis of equal computational budget for simulations used to construct the emulators.

References

[1] B.M. Adams, K.R. Dalbey, M.S. Eldred, D.M. Gay, L.P. Swiler, W.J. Bohnhoff, J.P. Eddy, and K. Haskell. Dakota users manual version 5.1. Sandia Technical Report SAND2010-2183, Sandia National Laboratories, Albuquerque, NM, 2011.

[2] R.E. Bellman. *Dynamic programming*. Rand Corporation research study. Princeton University Press, 1957.

[3] C. Berg, J. Mateu, and E. Porcu. The dagum family of isotropic correlation functions. *Bernoulli*, 14(4):1134–1149, 2008.

[4] T.M. Cioppa and T.W. Lucas. Efficient nearly orthogonal and space-filling latin hypercubes. *Technometrics*, 49(1):45–55, 2007.

[5] N.A.C. Cressie. *Statistics for spatial data*. Wiley (New York), revised edition, 1993.

[6] K.R. Dalbey and G.N. Karystinos. Fast generation of space-filling latin hypercube sample designs. In *Proceedings of the 13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages AIAA 2010–9085, September 2010.

[7] K.R. Dalbey and G.N. Karystinos. Generating a maximally spaced set of bins to fill for high dimensional space-filling latin hypercube sampling. *International Journal for Uncertainty Quantification*, 1(3), 2011.

[8] G.J. Davis and M.D. Morris. Six factors which affect the condition number of matrices associated with kriging. *Mathematical geology*, 29(5):669–683, 1997.

[9] R.P. Dwight and Z. Han. Efficient uncertainty quantification using gradient-enhanced kriging. In *11th AIAA Non-Deterministic Approaches Conference*, Reston, VA, USA, May 2009. American Institute of Aeronautics and Astronautics.

[10] M.S. Eldred, B.M. Adams, D.M. Gay, L.P. Swiler, K. Haskell, W.J. Bohnhoff, J.P. Eddy, W.E. Hart, J.P Watson, J.D. Griffin, P.D. Hough, T.G. Kolda, P.J. Williams, and M.L. Martinez-Canales. Dakota version 4.1 users manual. Sandia Technical Report SAND2006-6337, Sandia National Laboratories, Albuquerque, NM, 2007.

[11] T. Gneiting, M.G. Genton, and P. Guttorp. Geostatistical space-time models, stationarity, separability and full symmetry. In B. Finkenstadt, L. Held, and V. Isham, editors, *Statistical Methods for Spatio-Temporal Systems*, chapter 4, pages 151–172. Boca Raton: Chapman & Hall/CRC, 2007.

[12] A. Griewank. On automatic differentiation. Technical report, Argonne National Lab., IL (USA), 1988.

[13] A. Hall. On an experimental determination of pi. *Messenger of Mathematics*, 2:113–114, 1873.

[14] F.J. Hickernell. A generalized discrepancy and quadrature error bound. *Mathematics of Computation*, 67(221):299–322, 1998.

[15] R.L. Iman and WJ Conover. A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics-Simulation and Computation*, 11(3):311–334, 1982.

[16] H. Lee, R. Gramacy, C. Linkletter, and G. Gray. Optimization Subject to Hidden Constraints via Statistical Emulation. *Pacific Journal of Optimization*, to appear.

[17] B.A. Lockwood and M. Anitescu. Gradient-Enhanced Universal Kriging for Uncertainty Propagation. *Nuclear Science and Engineering*, in press. http://www.mcs.anl.gov/∼anitescu/PUBLICATIONS/2010/ lockwoodAnitescu%202010%20GEUK.pdf.

[18] C. Lucas. Lapack-style codes for level 2 and 3 pivoted cholesky factorizations. Numerical Analysis Reports 442, Manchester Centre for Computational Mathematics, Manchester, England, February 2004. LAPACK Working Note 161.

[19] J.D. Martin. Computational improvements to estimating kriging metamodel parameters. *Journal of Mechanical Design*, 131:084501, 2009.

[20] G. Matheron. *The theory of regionalized variables and its applications*. Les Cahiers du Centre de morphologie mathématique de Fontainebleau. École national supérieure des mines, 1971.

[21] M.D. McKay, R.J. Beckman, and W.J. Conover. Comparison the three methods for selecting values of input variable in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, May 1979.

[22] N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[23] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series In Applied Mathematics*. Society for Industrial Mathematics, Philadelphia, Pennsylvania, 1992.

[24] A. O'Hagan, P. Challenor, D. Cornford, H.P. Wynn, M. Goldstein, and J. Oakley. Managing uncertainty in complex models: a step change in understanding how models perform. http://www.mucm.ac.uk/index.html, accessed July 9, 2011.

[25] A.B. Owen. A central limit theorem for latin hypercube sampling. *Journal of the Royal Statistical Society. Series B (Methodological)*, 54(2):541–551, 1992.

[26] J.S. Park. Optimal latin-hypercube designs for computer experiments. *Journal of Statistical Planning and Inference*, 39(1):95–111, 1994.

[27] C.E. Rasmussen and C.K.I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006.

[28] G. Rennen. Subset selection from large datasets for kriging modeling. *Structural and Multidisciplinary Optimization*, 38(6):545–569, 2009.

[29] I.M. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

[30] B. Tang. Orthogonal array-based latin hypercubes. *Journal of the American Statistical Association*, 88(424):1392–1397, 1993.

[31] A. van der Sluis. Condition numbers and equilibration of matrices. *Numerische Mathematik*, 14(1):14–23, 1969.

[32] K.Q. Ye, W. Li, and A. Sudjianto. Algorithmic construction of optimal symmetric latin hypercube designs. *Journal of Statistical Planning and Inference*, 90(1):145–159, 2000.

DISTRIBUTION:

1	MS 0670	K. Dalbey, 05562
1	MS 0670	B. Richard, 05562
1	MS 0748	J. Helton, 01514
1	MS 0829	B. Rutherford, 00431
1	MS 1318	J. Stewart, 01441
1	MS 1318	B. Adams, 01441
1	MS 1318	M. Eldred, 01441
1	MS 1318	L. Swiler, 01441
1	MS 1318	T. Trucano, 01440
1	MS 9051	K. Chowdhary, 08351
1	MS 9051	B. Debusschere, 08351
1	MS 9051	H. Najm, 08351
1	MS 9051	K. Sargsyan, 08351
1	MS 9159	L. Bauman, 08954
1	MS 9159	P. Hough, 08954
1	MS 9159	J. Ray, 08954
1	MS 9159	C. Safta, 08954
1	MS 0899	Technical Library, 9536 (electronic copy)

Sandia National Laboratories