Preprint ANL/MCS-P1808-1110

Gradient-Enhanced Universal Kriging for Uncertainty Propagation

Brian A. Lockwood * and Mihai Anitescu †

November 14, 2010

Abstract

In this work, we investigate the issue of providing a statistical model for the response of a computer model-described nuclear engineering system, for use in uncertainty propagation. The motivation behind our approach is the need for providing an uncertainty assessment even in the circumstances where only a few samples are available. Building on our recent work in using a regression approach with derivative information for approximating the system response, we investigate the ability of a universal gradient-enhanced Kriging model to provide a means for inexpensive uncertainty quantification. The universal Kriging model can be viewed as a hybrid of polynomial regression and Gaussian process regression. For this model, the mean behavior of the surrogate is determined by a polynomial regression, and deviations from this mean are represented as a Gaussian process. Tests with explicit functions and nuclear engineering models show that the universal gradient-enhanced Kriging model provides a more accurate surrogate model when compared to either regression or ordinary Kriging models. In addition we investigate the ability of the Kriging model to provide error predictions and bounds for regression models.

Keywords uncertainty quantification; Gaussian process; derivative; universal Kriging, nuclear engineering

Mathematical Subject Classification 60G15,62P30.

1 Introduction

In the field of computational modeling, uncertainty quantification is increasingly used to enhance the predictive skill and applicability of simulation to the design and analysis of engineering systems [5, 6, 20]. In this work, we are interested primarily in uncertainty propagation [12, 17]. That is, given a probability distribution on an input parameter $\vec{x} \in K_x \subset \mathbb{R}^{n_x}$, determine the distribution of an output function $J(s, \vec{x}) : \mathbb{R}^{n_s} \times \mathbb{R}^{n_x} \rightarrow \mathbb{R}$, where $s \in \mathbb{R}^{n_s}$, the state, is obtained, for fixed $\vec{x}$, by solving the well-posed, possibly nonlinear system $f(s, \vec{x}) = 0$. The resolution of the latter system is carried out by a numerical code that returns $s(\vec{x})$ and, subsequently, $\widehat{J}(\vec{x}) = J(s(\vec{x}), \vec{x})$.

In principle, any statistic of $\widehat{J}(\vec{x})$, can be obtained by exhaustive sampling of the input uncertainty space K_x. Nevertheless, despite progresses in hardware, traditional uncertainty quantification techniques based on exhaustive sampling are still prohibitively expensive for high-fidelity simulations of complex engineering systems. In order to help alleviate this limitation, parametric surrogate models $\widetilde{J}(\vec{x}; a) \approx \widehat{J}(\vec{x})$ that inexpensively predict simulation outputs can be used. Here a are the parameters of the surrogate model, such as the coefficients of an approximation that is polynomial in $\vec{x}$. These surrogate models are constructed from a limited number of simulations and are used to predict simulation outputs for desired input parameters. These simulation outputs typically take the form of solution functionals and the surrogate model can be thought of as approximating a function within a region of the multi-dimensional space $\mathbb{R}^{n_x}$, refered to as the design space. In aerospace engineering, a surrogate model for lift as a function of freestream parameters may be constructed in order to predict lift for a variety of flight conditions [14]. For nuclear engineering

*Graduate Student, Department of Mechanical Engineering, University of Wyoming, Laramie, WY

†Corresponding author. Address: Argonne National Laboratory, Mathematics and Computer Science Division, 9700 S Cass Avenue, Argonne, IL, 60439. *E-mail:anitescu@mcs.anl.gov*

simulations, a safety metric such as peak fuel temperature can be represented by a surrogate model approximating the relationship between the safety metric and input physical parameters, such as crosssections, thermal conductivities, or heat transfer coefficients [22]. This relationship can then be used to estimate the uncertainty in the safety metric due to uncertain physical parameters. Because the surrogate model evaluation is inexpensive, a variety of uncertainty quantification strategies based on exhaustive sampling, such as Monte Carlo, may be used.

The utility of these surrogate models is determined by both the error of the approximations it produces and the number of baseline simulations required to train the model, both of which we would like to be as small as possible. These requirements are contradictory, as less error comes at the cost of more system simulations. Recently, we proposed using information about the gradient $\nabla_x \widehat{J}(\vec{x})$ in the construction of the surrogate in order to reduce the number of baseline samples needed. Specifically, we constructed a surrogate by using a least squares polynomial regression approach that used both function and gradient values from $\widehat{J}$ [22, 23]. All the n_x components of the gradient can be obtained by adjoint approaches at a cost that is at most five times [8], and, in many circumstances, substantially fewer times [22], the cost of one evaluation of $\widehat{J}(\vec{x})$. As a result, one gradient evaluation will produce $n_x/5$ more information for the same cost. Similar ideas have been proposed and demonstrated in other application areas [13, 16]. In nuclear engineering applications, we have demonstrated that problems with 12–40 uncertainty parameters can be computed to a substantial accuracy with information from only 12–20 samples [22, 23]. This is also made possible by using a basis pruning approach based on sensitivity information [22], which controls the complexity of the basis used.

On the other hand, using the surrogate will clearly introduce a bias, since it is highly unlikely that $\widehat{J}(\vec{x})$ belongs to the space spanned by the considered basis. In [22] we have addressed this by using the surrogate as a control variate and thus reducing the number of samples needed for estimating a mean to a certain accuracy. Nevertheless, this approach still needs additional samples from $\widehat{J}(\vec{x})$. In principle, we could use a statistical model derived from regression for $\widehat{J}(\vec{x}) - \widetilde{J}(\vec{x})$. Indeed, unweighted least-squares regression can be thought of as the following statistical model:

$$\begin{pmatrix} \widehat{J}(\vec{x}) \\ \nabla_x \widehat{J}(\vec{x}) \end{pmatrix} = \begin{pmatrix} \widetilde{J}(\vec{x}) \\ \nabla_x \widetilde{J}(\vec{x}) \end{pmatrix} + \epsilon_x. \tag{1.1}$$

Here ϵ_x is a n_x+1-dimensional random variable with independent identically distributed normal components and mean 0. Moreover, for $\vec{x}_1 \neq \vec{x}_2$ standard regression assumes that ϵ_x is independent from ϵ_x. Either of these independence assumptions is unlikely to lead to a good statistical model of the discrepancy $\widehat{J}(\vec{x}) - \widetilde{J}(\vec{x}; a)$. For example, independence for different $\vec{x}$ would result in the right-hand side of (1.1) being (almost surely) discontinuous in the first component and thus almost surely not differentiable. In turn, this feature is inconsistent with the assumption of using gradient information. Certainly, this problem is evident only in the limit of very large number of samples, so regression could still be a reasonable model for limited information. But we expect that more consistency in the model will result in better uncertainty estimation for the uncertainty propagation.

To address this issue, we combine the least-square regression approach with a Gaussian process (GP) approach, in an attempt to harness the strengths of both approaches. In the latter approach, a GP is fit to a data set, and the mean and variance conditional to the observed values can be used for prediction and uncertainty quantification [21, 26]. In its most standard form, the GP approach is used with mean zero, in which case the conditional estimates are effectively an interpolation. Such approaches, like nonparametric statistical approaches, are vulnerable to the curse of dimensionality: the rapid reduction of the sampling density with n_x, the dimension of the input [9]. We will thus import the mean function from the regression approach, on the presumption that it is a good approximation of the system response $\widehat{J}(\vec{x})$, and impose it as a parametric constraint on the GP. The conditional prediction with a specified mean function is called universal Kriging. On the other hand, the GP approach allows us to naturally insert the additional structure, such as derivative information, in a consistent manner. Given our emphasis on the use gradient information, we call this approach gradient-enhanced universal Kriging (GEUK).

The main contribution of this work is the investigation of the potential of GEUK for providing good models for uncertainty quantification of nuclear applications from very few runs of the computer model.

2 Model Overview and Implementation Details

Following the notations from §1, our aim is to model $J(s(\vec{x}), \vec{x}) = \widehat{J}(\vec{x})$ as a statistical system response $\mathcal{J}(\vec{x})$. Here $\mathcal{J}$ is a random process, that is, a mapping from the probability space into the space of continuously differentiable functions over $\vec{x}$. Each of its realizations is thus a differentiable function of $\vec{x}$. Clearly, $\widehat{J}(\vec{x})$ is a deterministic response, which becomes statistical in nature only after considering a random variable $X(\omega)$ over a probability space whose distribution is given and investigating the mapped random variable $\widehat{J}(X(\omega))$. On the other hand, the fact that in the end we are interested in distributions offers the opportunity to look at the system response $\widehat{J}(\vec{x})$ as a stochastic response, where the stochastic response is an expression of the lack of information in exploring $J(\vec{x})$ over $\vec{x} \in K_x$. This lack of information is a consequence of the reality that we can afford only a very limited number of evaluation of $s(\vec{x})$ and, implicitly, of $\widehat{J}(\vec{x})$. Since the essence of the process $\mathcal{J}(\vec{x})$ is an approximation of $\widehat{J}(\vec{x})$ over K_x, and thus not related to the probabilistic structure of $X(\omega)$, we can assume that it is independent of the random variable $X(\omega)$. Therefore, the distribution of $J(X(\omega))$ will be approximated by the distribution of $\mathcal{J}(X(\omega))$. Sampling from this random variable can be accomplished by sampling from $X(\omega)$ and returning the value $\vec{x}_1$ and then sampling from the process $\mathcal{J}$ evaluated at $\vec{x}_1$.

The model we will seek for $\mathcal{J}(\vec{x})$ is a Gaussian process model. Specifically, we consider

$$\mathcal{J}(\vec{x}) = \tilde{J}(\vec{x}; a) + \epsilon(\vec{x}; \theta). \tag{2.1}$$

Here $\epsilon(\vec{x}; \theta)$ is a mean zero GP, whose covariance is controlled by the parameters θ. The mean function $\tilde{J}(\vec{x}; a)$ is considered here to be a polynomial in $\vec{x}$ with coefficient a. GPs are often used with a zero mean function $\tilde{J}$, an approach that serves well in the machine learning applications in a relatively low-dimension, data-rich environment [21]. In our case, we want to employ a small number of samples and use the information that many nuclear applications tend to exhibit fairly smooth responses to include a polynomial mean function. Additional motivation was provided by our recent inquiry [22] indicating that explicit mean models in a regression approach show substantial promise in creating a good approximant of the system response; we thus expect that the use of a more flexible error model in the guise of a GP to further improve on that result. Moreover, an important feature of our investigation is an insistence on understanding the consequences for using gradient information in the model calibration. Our overarching hypothesis is that the extra information brought about by the smoothness in the mean and the gradient information will in the end require fewer samples to calibrate the model for the same levels of error.

GPs have been used extensively to create statistical models of functional responses, as $\widehat{J}(\vec{x})$ is in our case [11, 15, 7, 10, 2, 24, 21]. GPs tend to be a preferred choice in such circumstances owing to their flexibility and the ability to obtain an embedded uncertainty model in the approximation [15]. Derivative information can be included in the approach [21, 25], though this is far less intensively studied topic because GPs are many times used on experimental data for which gradient information is not available. The model calibration and interogation approach can be either Bayesian [11, 15, 7, 2] or maximum likelihood/Kriging [21, 24]. Bayesian approaches have substantial flexibility, allowing easy composition of multicomponent models that could have fairly different uncertainty structures. On the other hand, model calibration and investigation can have substantial difficulties in scaling up, because of difficulties in implementing efficient Markov-chain Monte Carlo (MCMC) sampling of the Bayesian posterior [2]. While this initial investigation is limited to a moderate number of dimensions in the uncertainty space (12) for which MCMC could conceivably work well, our intent to eventually progress to larger dimensions has led to our choice to pursue the maximum likelihood/Kriging approach.

2.1 Setup of Universal Kriging

We now state our model assumption (2.1) in the language of Gaussian processes [21]. That is, we state the distribution of the error term $\epsilon(\vec{x}, \theta)$ in terms of a covariance function $K(\vec{x}, \vec{x}; \theta)$ to obtain the model

$$\mathcal{J}(\vec{x}) = \hat{y}(\vec{x}) = N(m(\vec{x}), K(\vec{x}, \vec{x}; \theta)). \tag{2.2}$$

To emphasize the mean function meaning of $\tilde{J}(\vec{x}; a)$ we denote it by $m(\vec{x})$, by means of the equivalence identification $m(\vec{x}) \equiv \tilde{J}(\vec{x}; a)$. Also, we denote the output function $\mathcal{J}(\vec{x})$ as the "data" $\hat{y}(\vec{x})$, by means of the identification $\mathcal{J}(\vec{x}) \equiv \hat{y}(\vec{x})$.

For a mean function based on a regression model, the functional form becomes

$$\hat{y}(\vec{x}) = N(h(\vec{x})\beta, K(\vec{x}, \vec{x}; \theta)), \tag{2.3}$$

where $h(\vec{x})$ is a column vector containing the basis functions of the regression evaluated at the point $\vec{x}$ and β are the regression parameters. Using the covariance between points in the domain, we determine model predictions throughout the domain by sampling from the conditional distribution $y_*|\vec{X}, Y$, where $\vec{X}, Y$ are the training data. The posterior mean predictions are given by the formula

$$y_*|\vec{X}, Y = h(\vec{x}_*)\beta + k_*^T K^{-1}(Y - H\beta), \tag{2.4}$$

where H is the collocation matrix of the regression, Y is the vector of training function values, K is the covariance matrix between the training points, and k_* is the vector of covariances between the training points and the test point (x_*). Since the regression is built from a limited number of training points, it is prudent to assume that the regression parameters belong to a distribution of parameters. In the limit of zero knowledge of this distribution (vague prior assumption), the regression parameters are given by

$$\hat{\beta} = (H^T K^{-1} H)^{-1} H^T K^{-1} Y = A^{-1} H^T K^{-1} Y, \tag{2.5}$$

where A is the regression matrix defined as $H^T K^{-1} H$. Using this definition of regression parameters, we can make mean predictions using the following single formula:

$$y_*|\vec{X}, Y = k_*^T K^{-1} Y + (h(\vec{x}_*) - k_*^T K^{-1} H)\hat{\beta} = k_*^T K^{-1} Y + R(\vec{x}_*)\hat{\beta}. \tag{2.6}$$

In addition to predicting the mean behavior, the variance associated with the prediction can be computed based on the posterior distribution of the data. The variance of the prediction is given by

$$V[y_*] = cov(\vec{x}_*, \vec{x}_*) - k_*^T K^{-1} k_* + R(\vec{x}_*) A^{-1} R(\vec{x}_*)^T. \tag{2.7}$$

A more detailed explanation of the model basis can be found in Chapter 2 of Rasmussen [21]. The model outlined above is a description of a universal Kriging model (also known as Gaussian process regression). In the case where $h \equiv 0$ and $\dim \hat{\beta} = 0$ (that is, no inversion is carried out in (2.5)) the approach is called simple Kriging in the GP literature. When the mean function is treated as an unknown constant, corresponding to zeroth-order regression, the approach is referred to as ordinary Kriging. Also note that our description is abstract enough that it can accommodate heterogeneous data (such as temperature and pressure or function and derivative information of any order). The actual physical significance of the data is used only when the covariance matrix K and the mean function h are generated.

For the sake of terminology, the model presented in this section will be referred to as the Kriging model when it is based exclusively on function observations. For this work, the above framework is extended to include derivative observations within the covariance and regression relations. When derivative observations are included, the model will be referred to as a gradient-enhanced Kriging model or GEK model for short.

2.2 Covariance with Derivatives

In order to provide a higher fidelity surrogate model, gradient information may be included in the Gaussian process model. In this case, the covariance matrix becomes a block matrix that includes the covariance between derivative observations in addition to the covariance between function observations. This block matrix can be represented as [28]

$$\underline{K} = \begin{bmatrix} cov(Y, Y) & cov(Y, \nabla Y) \\ cov(\nabla Y, Y) & cov(\nabla Y, \nabla Y) \end{bmatrix}, \tag{2.8}$$

where $cov(Y, Y)$ represents a covariance matrix between the function values at the training points, $cov(\nabla Y, Y)$ is a covariance matrix between gradient components and the function values at the training points, and $cov(\nabla Y, \nabla Y)$ is a covariance matrix between gradient components. For clarity, if n is the number of training points and d is the dimension of the problem, then $cov(Y, Y)$ is an $n \times n$ matrix, $cov(\nabla Y, Y)$ is size $nd \times n$,

and $cov(\nabla Y, \nabla Y)$ is $nd \times nd$. The total size of $\underline{K}$ is thus $(d+1)n \times (d+1)n$. The components of the matrix $cov(Y, Y)$ are given by [21]

$$cov(y, y') = k(\vec{x}, \vec{x}'). \tag{2.9}$$

The covariance between function and gradient components is found by differentiating the covariance function [25]:

$$cov(\frac{\partial y}{\partial x_k}, y') = \frac{\partial}{\partial x_k} k(\vec{x}, \vec{x}'). \tag{2.10}$$

Differentiating once more (now w.r.t to the second argument of the covariance function) gives the covariance between gradient components:

$$cov(\frac{\partial y}{\partial x_k}, \frac{\partial y'}{\partial x'_l}) = \frac{\partial^2}{\partial x_k \partial x'_l} k(\vec{x}, \vec{x}'). \tag{2.11}$$

The gradient vector and Hessian matrix resulting from equations 2.10 and 2.11 can then be arranged into the matrices $cov(\nabla Y, Y)$ and $cov(\nabla Y, \nabla Y)$, respectively. In order to decrease the computational work associated with working with this matrix, the block matrix is factorized. For convenience, we redefine the blocks of $\underline{K}$ as

$$\underline{K} = \begin{bmatrix} P & R^T \\ R & S \end{bmatrix}. \tag{2.12}$$

In order to aid with the computation of the determinant and inverse of $\underline{K}$, a block LU decomposition can be performed on the matrix [21]:

$$\underline{K} = \begin{bmatrix} I & 0 \\ RP^{-1} & I \end{bmatrix} \begin{bmatrix} P & R^T \\ 0 & M \end{bmatrix}, \tag{2.13}$$

where $M = S - RP^{-1}R^T$. With the matrix factored as above, the inverse and determinant of $\underline{K}$ can be found by manipulating the blocks of $\underline{K}$. The block inverse of $\underline{K}$ can be defined as [21]

$$\underline{K}^{-1} = \begin{bmatrix} \tilde{P} & \tilde{R}^T \\ \tilde{R} & \tilde{S} \end{bmatrix} = \begin{bmatrix} P^{-1} + P^{-1}R^T M^{-1} RP^{-1} & -P^{-1}R^T M^{-1} \\ -M^{-1}RP^{-1} & M^{-1} \end{bmatrix}. \tag{2.14}$$

The determinant of $\underline{K}$ is given by multiplying the determinants of the diagonal block matrices of the LU factored block matrix (noting that the determinant of the lower part of the factorization is 1):

$$|\underline{K}| = |P| \cdot |M|. \tag{2.15}$$

Using these block definitions of $\underline{K}$, we can update the formulas for mean predictions (2.6) and variance (2.7) to include derivative observations.

2.3 Universal Kriging Model with Derivative Values

In order to incorporate derivative values into the universal Kriging model, the elements of the model must be recast by using block definitions. The block definition of the covariance matrix was outlined in §2.2. For the training data, the blockwise definition includes function values followed by gradient observations. In the case of the collocation matrix, the block definition includes the basis functions evaluated at either the training or test points as well as the derivatives of the basis functions evaluated at the training or test points. First, the definition for the regression parameters must be updated to include derivatives. This is done by using the blockwise definition of the covariance and collocation matrices. The formula for the regression parameters is given by

$$\hat{\beta} = \left(\overbrace{[H^T G^T]\underline{K}^{-1} \begin{bmatrix} H \\ G \end{bmatrix}}^{\hat{A}} \right)^{-1} [H^T G^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix} = \hat{A}^{-1}[H^T G^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix}, \tag{2.16}$$

where G is a matrix containing the derivatives of the basis functions and δY is the vector of derivative observations.

Using this new definition of regression parameters, one can predict the mean value of the model as [28]

$$y_*|\vec{X}, Y, \delta Y = [k_*^T w_*^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix} + \overbrace{\left(h(\vec{x}_*) - [k_*^T w_*^T]\underline{K}^{-1} \begin{bmatrix} H \\ G \end{bmatrix} \right)}^{\hat{R}(\vec{x}_*)} \hat{\beta} \tag{2.17}$$

$$y_*|\vec{X}, Y, \delta Y = [k_*^T w_*^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix} + \hat{R}(\vec{x}_*)\hat{\beta}, \tag{2.18}$$

where w_* is the vector of covariances between the test point function values and the gradient at each training point. The variance associated with this prediction can now be calculated as:

$$V[y_*] = cov(\vec{x}_*, \vec{x}_*) - [k_*^T w_*^T]\underline{K}^{-1} \begin{bmatrix} k_* \\ w_* \end{bmatrix} + \hat{R}(\vec{x}_*)\hat{A}^{-1}\hat{R}(\vec{x}_*)^T \tag{2.19}$$

where $\hat{A}$ and $\hat{R}$ are now defined in the block sense presented previously.

Once derivative observations have been included in the covariance matrix, the gradient may be predicted with little additional work. The gradient at a given test point can be predicted by using the formula [28]

$$\nabla y_*|\vec{X}, Y, \delta Y = [l_*^T m_*^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix} + \overbrace{\left(g(x_*) - [l_*^T m_*^T]\underline{K}^{-1} \begin{bmatrix} H \\ G \end{bmatrix} \right)}^{S(\vec{x}_*)} \hat{\beta} \tag{2.20}$$

$$\nabla y_*|\vec{X}, Y, \delta Y = [l_*^T m_*^T]\underline{K}^{-1} \begin{bmatrix} Y \\ \delta Y \end{bmatrix} + S(\vec{x}_*)\hat{\beta}, \tag{2.21}$$

where l_* is a matrix of covariance values between the gradient at the test point and the function values at the training points and m_* is the matrix of covariance values between the gradient at the test point and the gradient values at the training points.

The covariance between the components of the gradient prediction at a given test point is given by

$$cov(\frac{\partial y_*}{\partial \vec{x}}, \frac{\partial y_*}{\partial \vec{x}}) = \frac{\partial^2}{\partial \vec{x} \partial \vec{x}} cov(\vec{x}_*, \vec{x}_*) - [l_*^T m_*^T]\underline{K}^{-1} \begin{bmatrix} l_* \\ m_* \end{bmatrix} + S(\vec{x}_*)A^{-1}S(\vec{x}_*)^T. \tag{2.22}$$

The diagonal of this matrix represents the variance in that component of the gradient and can be used to provide a confidence interval for each derivative value.

2.4 Covariance Details

Within the model outlined in the previous sections, the only item left undetermined is the details of the covariance function. These details consist of the functional form of the covariance as well as the parameters governing the function (referred to as hyperparameters). The choice of covariance function reflects underlying beliefs in the data dependence and smoothness present in the design space. For this work, a stationary covariance function is assumed. This assumption expresses the belief that the covariance is simply a function of the distance between data points [21]:

$$k(\vec{x}, \vec{x}') = f(\vec{x} - \vec{x}') = f(\vec{r}). \tag{2.23}$$

We note that there is no basis to assume that this will a priori result in a good fit. Nevertheless, fitting a stationary process is an important step toward determining the suitable covariance functions. In effect, model fitting can be seen as an endeavor of fitting increasingly more complex models, starting with an independent identically distributed assumption of noise effectively, a regression approach. Then more complex models are considered, such as stationary Gaussian, nonstationary Gaussian, and non-Gaussian. Which model achieves the correct information/complexity balance is a major component of model selection in statistical endeavors, which needs insight from how each model performs [3]; the latter is our contribution here for the class of stationary Gaussian process models.

In this work, we use multidimensional covariance functions that are products of one-dimensional covariance functions, as this form produces better conditioned covariance matrices [28].

$$k(\vec{x}, \vec{x}'; \theta) = \sigma^2 \prod_{i=1}^{d} k_i(x_i - x_i'; \theta_i) = \sigma^2 \prod_{i=1}^{d} k_i(r_i; \theta_i) \tag{2.24}$$

Because derivative observations are included in the model, a twice differentiable covariance is required. Several covariance functions with a variety of properties are considered for this work. These functions are [21, 28] as follows.

- Squared Exponential:

$$k_i(x_i - x_i') = e^{-\left(\frac{x_i - x_i'}{\theta_i}\right)^2} \tag{2.25}$$

- Matern Function $\nu = \frac{3}{2}$:

$$k_i(x_i - x_i') = \left(1 + \sqrt{3}\left|\frac{x_i - x_i'}{\theta_i}\right|\right) e^{-\sqrt{3}\left|\frac{x_i - x_i'}{\theta_i}\right|} \tag{2.26}$$

- Matern Function $\nu = \frac{5}{2}$:

$$k_i(x_i - x_i') = \left(1 + \sqrt{5}\left|\frac{x_i - x_i'}{\theta_i}\right| + \frac{5}{3}\left|\frac{x_i - x_i'}{\theta_i}\right|^2\right) e^{-\sqrt{5}\left|\frac{x_i - x_i'}{\theta_i}\right|} \tag{2.27}$$

- Cubic Spline 1:

$$k_i(x_i - x_i') = \begin{cases} 1 - 15\left|\frac{x_i - x_i'}{\theta_i}\right|^2 + 30\left|\frac{x_i - x_i'}{\theta_i}\right|^3 & \text{for } 0 \le \left|\frac{x_i - x_i'}{\theta_i}\right| \le 0.2 \\ 1.25\left(1 - \left|\frac{x_i - x_i'}{\theta_i}\right|\right)^3 & \text{for } 0.2 \le \left|\frac{x_i - x_i'}{\theta_i}\right| \le 1 \\ 0 & \text{for } \left|\frac{x_i - x_i'}{\theta_i}\right| \ge 1 \end{cases} \tag{2.28}$$

- Cubic Spline 2:

$$k_i(x_i - x_i') = \begin{cases} 1 - 6\left|\frac{x_i - x_i'}{\theta_i}\right|^2 + 6\left|\frac{x_i - x_i'}{\theta_i}\right|^3 & \text{for } 0 \le \left|\frac{x_i - x_i'}{\theta_i}\right| \le 0.5 \\ 2\left(1 - \left|\frac{x_i - x_i'}{\theta_i}\right|\right)^3 & \text{for } 0.5 \le \left|\frac{x_i - x_i'}{\theta_i}\right| \le 1 \\ 0 & \text{for } \left|\frac{x_i - x_i'}{\theta_i}\right| \ge 1 \end{cases} \tag{2.29}$$

We note that, despite the definition of all these covariance functions involving absolute values of the distance between coordinates, which is not differentiable, the covariance is still twice differentiable. This property can be best observed by noting that these stationary covariance functions can be expressed as $K(x_i, x_i') = f(r)$, where $r = \left|\frac{x_i - x_i'}{\theta_i}\right|$. The functions $f(r)$ are twice continuously differentiable on $[0, \infty]$ and satisfy $f'(0) = 0$. It can then immediately be verified that the directional derivatives of K at x_i fixed and x_i' variable agree up to order 2, even if the absolute value appears in the definition of the covariance.

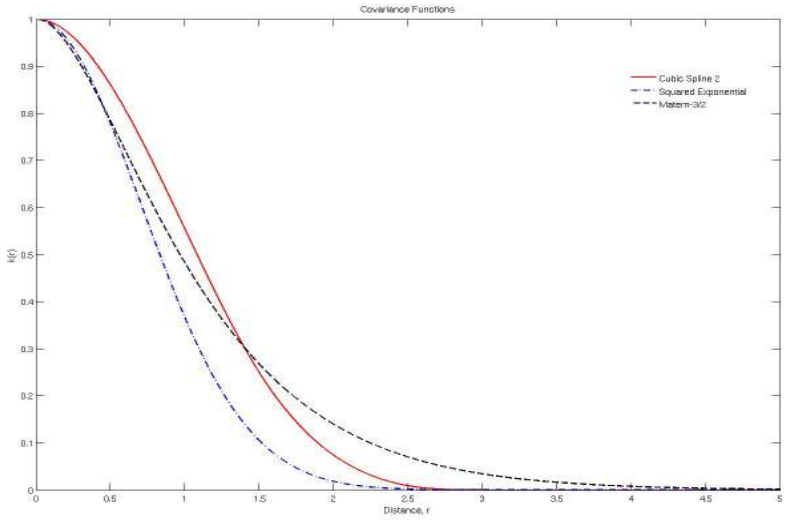

Figure 2.1: Plots of cubic spline 2, squared exponential and Matern function with $\nu = 3/2$

Of the covariance functions considered in this work, the Matern function with $\nu = 3/2$ assumes the least amount of smoothness in underlying design space, giving the most conservative estimates with respect to the variance. For the derivative-free case, this feature makes the Matern class, particularly its $\nu = \frac{1}{2}$ instance, one of the most indicated to try first [26]. Based on this theoretical analysis, we would expect the $\nu = 3/2$ Matern function to give conservative estimates of the variance for the case of the differentiable functions assumed here.

On the other hand, the squared exponential assumes an extremely smooth underlying function space, which can be beneficial in the limit of small sample sizes. The cubic spline 1 and 2 covariance functions are compact stencil kernels. This property improves the condition number of the covariance matrix when the number of training points is large. Additionally, with the proper limits on hyperparameters, these covariance functions can give rise to sparse covariance matrices, greatly reducing the cost associated with factoring and inverting the matrix. Three of the covariance functions considered in this work are plotted in Figure 2.1.

As shown previously, the covariance function must be differentiated to provide the covariance between function and gradient values. Additionally, the covariance must be differentiated twice to give the covariance between derivative observations. For the product form of the covariances utilized in this work, the gradient and Hessian of the covariance are given by

$$\frac{\partial}{\partial x_i} k(\vec{x}, \vec{x}') = \frac{d}{dx_i} k_i(x_i - x_i'; \theta_i) \prod_{l \neq i}^{d} k_l(x_l - x_l'; \theta_l) \tag{2.30}$$

$$\frac{\partial^2}{\partial x_i \partial x_j'} k(\vec{x}, \vec{x}') = \begin{cases} \frac{d}{dx_i} k_i(x_i - x_i'; \theta_i) \frac{d}{dx_j'} k_j(x_j - r_j'; \theta_j) \prod_{l \neq i \neq j}^{d} k_l(x_l - x_l'; \theta_l) & \text{for } i \neq j \\ \frac{d^2}{dx_i dx_i'} k_i(x_i - x_i'; \theta_j) \prod_{l=1, l \neq i}^{d} k_l(x_l - x_l'; \theta_l) & \text{for } i = j. \end{cases} \tag{2.31}$$

With memoization, these functional forms may be implemented efficiently by using the following definitions:

$$L_i = \prod_{l<i} k_l(x_l - x_l'; \theta_l) \tag{2.32}$$

$$U_i = \prod_{l>i} k_l(x_l - x_l'; \theta_l) \tag{2.33}$$

$$M_{i,j} = \prod_{i<l<j} k_l(x_l - x_l'; \theta_l) \text{ where } j > i. \tag{2.34}$$

The derivatives are subsequently evaluated as

$$\frac{\partial}{\partial x_i}k(\vec{x},\vec{x}') = L_i U_i \frac{d}{dx_i}k_i(x_i - x_i';\theta_i) \tag{2.35}$$

$$\frac{\partial^2}{\partial x_i \partial x_j'}k(\vec{x},\vec{x}') = \begin{cases} L_i M_{i,j} U_j \frac{d}{dx_i}k_i(x_i - x_i';\theta_i)\frac{d}{dx_j'}k_j(x_j - x_j';\theta_j) \\ L_i U_i \frac{d^2}{dx_i dx_i'}k_i(x_i - x_i';\theta_i) \qquad \text{for } i = j. \end{cases} \tag{2.36}$$

As the formulas above show, only the derivatives of the one-dimensional covariance functions are needed and can be computed and validated easily. For example, the derivatives of the squared exponential function are given below [25]:

$$\frac{d}{dx_i}k_i(x_i - x_i';\theta_i) = -2\left(\frac{x_i - x_i'}{\theta_i^2}\right)e^{-\left(\frac{x_i - x_i'}{\theta_i}\right)^2} \tag{2.37}$$

$$\frac{d^2}{dx_i dx_i'}k_i(x_i - x_i';\theta_i) = 2\left[\frac{1 - 2\left(\frac{x_i - x_i'}{\theta_i}\right)}{\theta_i^2}\right]e^{-\left(\frac{x_i - x_i'}{\theta_i}\right)^2}. \tag{2.38}$$

We again note that the second differentiation is with respect to the second argument of the covariance function. Typically, this fact manifests itself as an extra negative sign when compared with differentiation only with respect to the first argument.

2.5 Hyperparameter Fitting

With the functional form of the covariance function specified, only the hyperparameters need to be determined in order to fully specify the covariance function. For this work, the hyperparameters are determined by maximizing the marginal likelihood function for the data given the assumed functional form. In other words, we want to maximize the probability that the data we have observed follows the Gaussian process we have assumed with a specific set of hyperparameters. The log of this likelihood function for a zero mean Gaussian process is given as [21]

$$\log(p(y|X;\theta)) = -\frac{1}{2}Y^T K^{-1} Y - \frac{1}{2}log|K| - \frac{n}{2}log2\pi. \tag{2.39}$$

This formula consists of three terms: a quadratic term based on the observed data, a determinant term based on the covariance matrix and a scaling term based on the number of observations. The formula can be extended to a nonzero mean Gaussian process with vague regression parameters by rewriting the quadratic term as a function of the residual of the regression and including the determinant of the regression matrix. With some manipulation, the formula becomes [21]:

$$\log(p(y|X;\theta)) = -\frac{1}{2}Y^T K^{-1} Y + \frac{1}{2}Y^T C Y - \frac{1}{2}log|K| - \frac{1}{2}log|A| - \frac{n-s}{2}log2\pi, \tag{2.40}$$

where $C = K^{-1}HA^{-1}H^T K^{-1}$, A is the regression matrix ($A = H^T K^{-1} H$) and s is the number of terms in the regression. This formula can be extended to include gradient observations by incorporating the block definitions of the covariance matrix, training data, and collocation matrices:

$$\log(p(y|X;\theta)) = -\frac{1}{2}[Y^T \delta Y^T]\underline{K}^{-1}\begin{bmatrix} Y \\ \delta Y \end{bmatrix} + \frac{1}{2}[Y^T \delta Y^T]\underline{C}\begin{bmatrix} Y \\ \delta Y \end{bmatrix} \tag{2.41}$$

$$-\frac{1}{2}log|P| - \frac{1}{2}log|M| - \frac{1}{2}log|\underline{A}| - \frac{nd+n-s}{2}log2\pi, \tag{2.42}$$

where P and M are the factored diagonal terms of the block matrix $\underline{K}$ and $\underline{C}$ is the block matrix version of C defined as follows.

$$\underline{C} = \underline{K}^{-1} \begin{bmatrix} H \\ G \end{bmatrix} \underline{A}^{-1} [H^T G^T] \underline{K}^{-1} \tag{2.43}$$

Using this formula, one can determine the hyperparamters of the distribution by finding the parameters that maximize the likelihood function. This optimization problem has a size that scales linearly with the dimension of the vector of hyperparameters, θ. Assuming $\underline{K}$ is dense, the cost of evaluating the likelihood function is $O(n^3 d^3)$. Hence, this optimization step represents the bulk of the computational expense associated with creating the model.

We know of no GP functions for which one can guarantee that the empirical likelihood function has a unique maximum or that it is concave. Therefore, there is the risk of multiple minima of the final estimates, depending on the initial guesses. We have found that some heuristics are useful to make sure that the optimization algorithm selects a high-quality maximum. We impose constraints on the likelihood parameters after some experimentation, such as the scale parameter being bounded below by the minimum distance between two points and a maximum value found by some experimentation, and scaling the inputs and bounding the parameter σ in (2.24) between 0.01 and 1; the lower value also is determined empirically.

The optimization itself is carried out by using a combination of L-BFGS [19] and an active set algorithm [19] (the latter useful to deal with the inequality constraints obtained from bounding the parameters as in the preceding paragraph). The L-BFGS algorithm is used to get within the vicinity of an extrema. At this point, the active set algorithm is used to further hone in on the maximum. To further avoid local maxima, we test a fixed number of initial starting points (~ 5). Despite these mitigating practices, the selection of local minimum definitely represents a problem for determining hyperparameters. Nevertheless, in this work these practices were sufficient to determine good models, as outlined in §3.

2.6 Error Estimation for Regression

Using the universal Kriging model, we can estimate the error associated with a traditional regression. This is relevant to our work, because we seek to quantify the improvement that stationary GEUK offer versus regression [22]. The error estimate is derived by first assuming a functional form for the error in the regression. We assume the error in the original regression obeys the following relationship:

$$y = r(\vec{x}) + e(\vec{x}), \tag{2.44}$$

where $r(\vec{x})$ is the result of the original regression and $e(\vec{x})$ is some yet-to-be determined function. This additive relation combined with the original assumption of the Kriging model (that the data y is distributed according to equation 2.2) implies that the function $e(\vec{x})$ is also distributed according to a Gaussian process with mean and covariance given as

$$e(\vec{x}) = N(m(\vec{x}) - r(\vec{x}), K(\vec{x}, \vec{x}; \theta)), \tag{2.45}$$

where $m(x)$ is the mean and K is the covariance associated with the assumed distribution of the training data, $\hat{y}$. With the form of the distribution known, mean and variance predictions based on this mean can be calculated based on the observations of the error (which is merely the difference between the function observation and regression at the training points, denoted by the vector R):

$$e_* | \vec{X} Y R = h(\vec{x}_*)\beta - r(\vec{x}_*) + k_*^T K^{-1} \left[Y - R - (H\beta - R) \right] \tag{2.46}$$

$$e_* | \vec{X} Y R = h(\vec{x}_*)\beta + k_*^T K^{-1} \left[Y - H\beta \right] = y_* - r(\vec{x}_*). \tag{2.47}$$

Because the original regression is assumed to have no variance associated with it (i.e. it is deterministic or $\hat{r}(x) = N(r(x), 0)$), the mean error prediction is simply the difference between the mean function prediction of the universal Kriging model and the regression model. The variance in the prediction is equal to the variance associated with the Kriging function prediction ($V[y_*]$). Using this variance, one can place confidence bounds on the error prediction. Assuming a 99% confidence interval, a worst-case estimate of the magnitude of the error at a test point is given by

$$e(\vec{x}_*)_{max} = |y_* - r(\vec{x}_*)| + 3\sqrt{V\left[y_*\right]}. \tag{2.48}$$

This method for error estimation is applied to both the Kriging and GEK models. Although this error estimate could be applied to any deterministic surrogate model, models based on regressions allow for the reuse of the collocation matrix and basis functions between the Kriging and regression models.

2.7 Hypotheses about Gradient Enhanced Universal Kriging

In investigating the benefits of our GEUK approach, we are guided by several hypotheses. We enumerate these hypotheses and our rationale that has led to them.

[H1] GEUK results in less error compared with L_2 regression. We believe this hypothesis to be the case since GEUK uses a more suitable error model. In particular, GEUK naturally allows for differentiable realizations and nontrivial correlation patterns, which regression in its standard format doesn't.

[H2] GEUK results in less error compared with universal Kriging without derivative information. This insight follows from the observation that the gradient adds n_x times more information for a small and uniformly bounded relative cost [8]. We will investigate this hypothesis in absolute terms as well as relative to the computational cost for either approach. The first endeavor quantifies the absolute benefit of gradient information if available.

[H3] GEUK results in less error for the same number of sample values when compared with ordinary Kriging. Our rationale here is based on the fact that GEUK fitting can be seen as regression, and if the mean is a good approximation of the system response — which may be the case when the response is smooth — then we would expect GEUK to perform better than ordinary Kriging. The improvement should be more pronounced in the small sample limit where the number of observations is insufficent for the ordinary Kriging model to capture mean behavior.

[H4] GEUK approximates well the statistics of $J(s(x), x)$, and its predicted covariance is a good or conservative estimate of the error. The basis for this hypothesis is the reported good experience with Gaussian process approximations in other application areas [21, 15, 25].

[H5] The choice of covariance function matters. It will affect the predictions and usability of the model. Here, we follow the rationale from [26] but applied to the case where gradient information is also used. That is, it is probably best to use a covariance function that makes conservative assumptions about the differentiability of the process with which we are approximating $\widehat{J}(u)$, particularly in the limit of dense sample points. This suggests that the Matern $\nu = 3/2$ model, which allows for functions that are only once differentiable, is a good but conservative approach.

These hypotheses will be tested by numerical experiments in §3.

3 Numerical Simulations

In order to provide evidence for the previously stated hypotheses, a number of tests were carried out with explicit functions, §3.1, and data derived from numerical simulations of nuclear reactor models, §3.2. Initial tests were performed with explicit functions in one and two dimensions to allow for visualization of the design space. In §3.1, the benefits of using gradient observations in training the surrogate are demonstrated, and effects of various aspects of the GEUK model, such as choice of covariance function and regression order, are examined. With the beneficial aspects of GEUK demonstrated for simple functions, the performance of the model in representing the design space associated with numerical simulations (both in MATLAB and in Fortran, the latter from the package MATWS) is presented in §3.2. In reporting our findings in graphs and tables, we use the format Experiment – Covariance Function – Details. Here Experiment is one of "Explicit", "MATLAB", or "MATWS", Covariance Function is one of the functions defined in §2.4, if only one such function was used to generate the plots. For a given computational experiment and unless otherwise mentioned in the text, a specific covariance function was chosen for reporting the results because we found

the quality of predictions with the other covariance functions to be of similar quality. In the latter case, we typically chose a compact kernel covariance function, such as the cubic spline function, since its sparse covariance matrix resulted in smaller computational times.

In our numerical experiments we will track whether our hypotheses described in §2.7 are satisfied. When discussing a conclusion that represents evidence that one of our hypotheses is verified, we will add "(H?)" where "?" is replaced by the number of the hypotheses verified at the end of the sentence, as labeled in §2.7. If the evidence points to the hypothesis being false or weak for that case, we will explicitly describe it in the text.

3.1 Results for Explicit Functions

To initially analyze the performance of the GEUK model, we represented the design space associated with several explicit functions. The performance of the surrogate was assessed based on the overall level of error present in the approximation as well as the ability of the variance prediction to provide bounds for the surrogate error. The functions used for these tests were the following:

$$y(\vec{x}) = cos(\sum_{i=1}^{d} x_i) \qquad \text{A multidimensional cosine function} \tag{3.1}$$

$$y(\vec{x}) = \frac{1}{1 + \sum_{i=1}^{d} x_i^2} \qquad \text{The multidimensional Runge function} \tag{3.2}$$

$$y(\vec{x}) = 10d + \sum_{i=1}^{d} x_i^2 - 10cos(2\pi x_i) \qquad \text{The Rastrigin function} \tag{3.3}$$

These functions were chosen based on their popularity in optimization problems as well as the range of complexity they encompass. For the cosine and Runge functions, the domain for each variable was $[-2, 2]$. For the Rastrigin function, the interval $[-5.12, 5.12]$ was used for each variable.

In order to compare the GEUK model to least squares polynomial regression, the cosine and Runge functions in one dimension were approximated by using each model. The approximations produced by the two models as well as the exact functions are plotted in Figure 3.1. Both functions were approximated using three sample points using function and derivative values. The regression was second order, and the cubic spline 2 covariance function was used. As the plots demonstrate, the GEUK model (the mean prediction (2.6)) produces an approximation closer to the exact function by eliminating the discrepancy between the regression and the exact function values (H1). For the Kriging model, the surrogate produces the exact function and derivative values at the sample locations. In this manner, Kriging can more effectively assimilate additional sample observations compared with regression without overfitting the data.

To further examine the performance of GEUK model, we approximated the cosine function using different polynomial orders and numbers of training points. For the first test, the regression and Kriging model were based on seven function/gradient evaluations, and the regression order was varied from zeroth to eighth order. The results from this test, shown in Table 3.1, demonstrate that the approximation given by GEUK contains lower error than the approximations of the regression model (H1). Additionally, the error in the universal Kriging model remains flat for low-order regressions and begins to show exponential convergence for high polynomial order. This behavior helps identify the two error regimes present in regression-based models. Initially, the error in the regression model is dominated by the bias between the actual function values and the regression predictions. Once the regression order is high enough, the error becomes denominated by the truncation error in the polynomial expansion of the function. In the former regime (the regime likely to be of interest in application, since larger polynomial degrees require more information to fit), the GEUK improves function predictions by eliminating the bias found in the regression model. In the latter regime, the GEUK model matches the behavior of the regression model. Although the function predictions are not improved in this range, the universal Kriging model has the advantage of also giving the variance associated with the function prediction. So, in this regime, hypothesis H1 is weak in that the GEUK merely gives similar quality to regression, as opposed to outperforming it, as it did in the first case. The case of polynomial order 0 reported in Table 3.1 is the case of ordinary Kriging with gradient information. We see that for

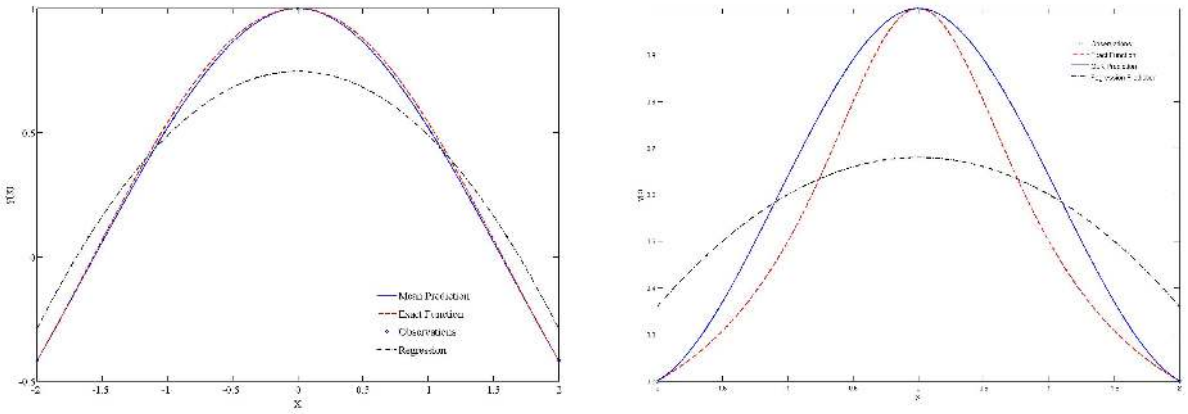

Figure 3.1: Explicit – cubic spline 2 – GEUK model vs. Regression for cosine (keft) and Runge (right) function using 3 function evaluations and second order regression

Table 3.1: Explicit – cubic spline 2– comparison of GEK model with regression for fixed number of points for Cosine Function

Polynomial Order	GEK RMS Error	Regression RMS Error
0	2.31×10^{-4}	4.67×10^{-1}
2	2.10×10^{-4}	9.73×10^{-2}
4	3.71×10^{-4}	3.71×10^{-3}
6	7.93×10^{-5}	7.93×10^{-5}
8	8.15×10^{-7}	8.15×10^{-7}

large polynomial degrees, GEUK far outperforms ordinary Kriging with gradient information (H3). For low polynomial degrees the performance of the two is comparable, so hypothesis H3 is weak here.

For numerical simulations, the cost of constructing the surrogate is dominated by the function/gradient evaluations, which are often the result of an expensive simulation. As such, the number of function evaluations is determined by the computational budget, and the highest order possible would be used. In order to further compare the GEUK model to regression under these circumstances, the cosine function was approximated by using different numbers of training points. For both models, the regression order was determined such that the number of observations (function plus derivative values) is twice the size of the basis. In addition to comparing the performance of GEUK with regression, an ordinary Kriging model based on the same number of points was constructed. Table 3.2 shows the RMS error of each approximation as the number of training points is varied. As these results show, the GEUK consistently produces a more accurate approximation than does the regression model (H1). Additionally, these results show exponential error convergence for both the universal Kriging and simple regression model. This behavior is due to the fact that the oversampling ratio was fixed for these tests, combining the effects of truncation error and bias removal into a single quantity. Additionally for this modeling scenario, the use of higher-order regression in the GEK tends to produce a surrogate that is more accurate than an ordinary GEK model, with the exception of the n=3 result (H3).

To demonstrate the effect of adding derivative evaluations, we approximated the Rastrigin function using both a Kriging and GEK model. Both models were based on 20 training points and used a zeroth-order regression (ordinary Kriging). The cubic spline 2 covariance function was used to ensure the covariance matrix was well conditioned. The mean predictions for these two models are found in Figure 3.2. As the plots show, the addition of derivative observations greatly enhances the models' ability to capture the function with relatively few sample points. Additionally, the increased accuracy resulting from the addition of derivatives causes a corresponding decrease in the variance of the prediction (H2).

In order to demonstrate the scaling of the approximation error for models with and without derivative

Table 3.2: Explicit – cubic spline 2 – comparison of GEK model with regression using maximum order for Cosine Function

Sample Points	Polynomial Order	GEK RMS Error	Regression RMS Error	Ordinary GEK
3	2	1.34×10^{-2}	1.51×10^{-1}	4.22×10^{-3}
5	4	8.21×10^{-4}	4.15×10^{-3}	1.02×10^{-3}
7	6	2.24×10^{-5}	7.93×10^{-5}	2.29×10^{-4}
9	8	1.11×10^{-6}	1.14×10^{-6}	6.54×10^{-5}

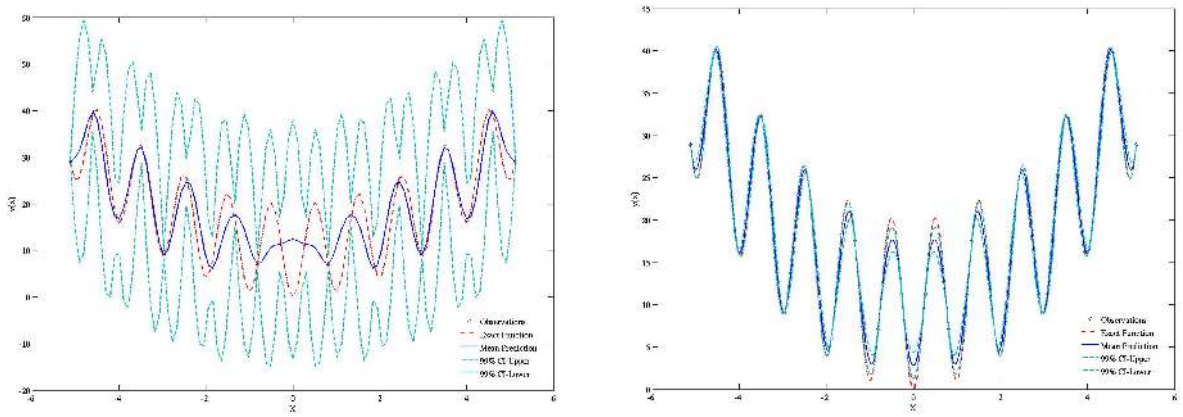

Figure 3.2: Explicit – cubic spline 2 – Kriging vs. GEK for Rastrigin function based on 20 training points

observations, the cosine function in two dimensions was approximated by using the GEK model with varying numbers of training points. The error in the associated approximation was compared with a Kriging model with the same number of training points. For this test a zeroth-order regression (ordinary Kriging) and the cubic spline 2 covariance function were again used. The error as a function of number of sample points is plotted in Figure 3.3. As the plot demonstrates, the error associated with the GEK model is consistently lower than the error of the Kriging approximation (H2).

3.2 Results for Numerical Simulations

We now investigate the performance GEUK in the context of two nuclear engineering models. For both models a large number of samples were obtained by running the corresponding codes directly. Then, surrogate models based on a limited number of outputs were created, and the approximate output produced by the surrogate models was compared with the set of directly sampled (not approximated) outputs.

3.2.1 MATLAB Model

The first is a MATLAB model of a simplified, three-dimensional steady-state reactor core with uniform fuel elements, simple heat transport description (including convection and diffusion), uniform liquid coolant flow, and no control mechanisms [22]. While our research extends to more complex systems, the idea was to work with a model that exhibits behavior typical for real-world nuclear reactors in sufficient measure to study uncertainty propagation for a significant, yet easy-to- manipulate model.

The operational parameters of the model were chosen to correspond to a sodium-cooled fast reactor with realistic temperature distribution. The simulation problem is defined by twelve parameters relating to the thermodynamic and heat transfer relations present in the reactor. The main output of interest for this simulation is the peak fuel pin temperature. In addition to this output, the code produces the derivatives of peak fuel pin temperature with respect to the twelve input parameters. The origination and processing of the uncertainty information are described in [22]. For this case, a set of 500 samples from the uncertainty

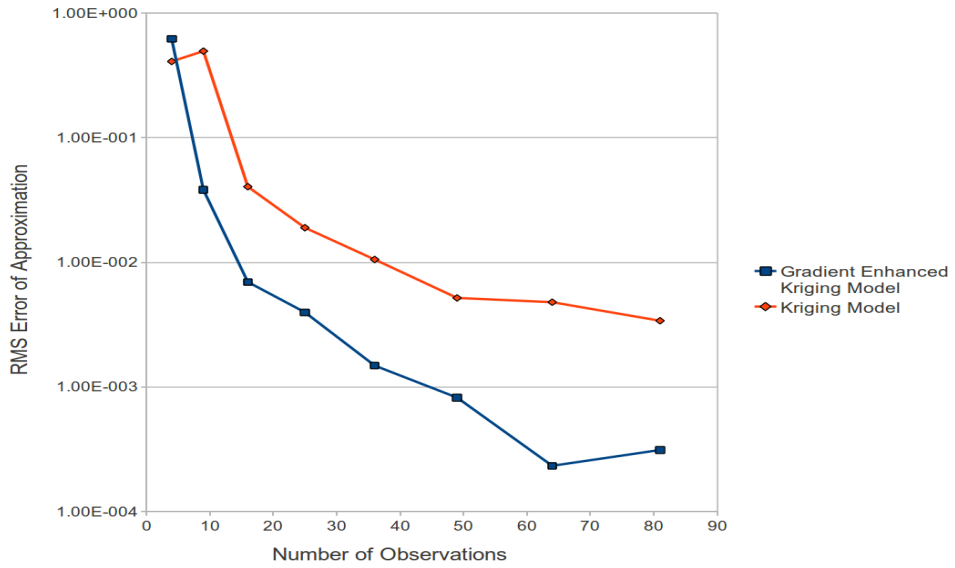

Figure 3.3: Explicit – cubic spline 2 – Error of approximation vs. number of sample points for Kriging and GEK models for two-dimensional cosine function

space was created and propagated through the code for obtaining the reference samples used in the testing and assessment (but not in model training).

3.2.2 MATWS

The second nuclear simulation test case consists of the nuclear reactor simulation code MATWS. The MATWS computer program has been compiled as part of an application library. It combines the point-kinetics module from the SAS4A/SASSYS computer code with a simplified representation of a reactor heat-removal system. A description of the underlying mathematical model is available, but the complete process of obtaining the numerical solution is documented only in the code comments, and sparsely at that. MATWS was used in combination with another simulation tool, GoldSim, to model unprotected loss-of-heat-sink, loss-of-flow and transient-overpower accidents [18].

MATWS can be used to predict the behavior of several observables of interest, such as coolant, cladding, structure, and fuel temperature. For this work, we have used only the prediction of the maximum fuel temperature. We note that if the maximum is unique, then its value is differentiable with respect to the input parameters. In our case the inputs (corresponding to physically significant sources of uncertainty) are the radial core expansion, control rod driveline expansion, fuel axial expansion, and Doppler reactivity feedback coefficients [1, 18].

The code is written in Fortran 77, and the gradient information was obtained by adjoint automatic differentiation [8], using the open-source Fortran automatic differentiation tool OpenAD [27]. In order to carry out the automatic differentiation, several changes to the code were needed to remove nonstandard, deprecated, or unsafe portions of code [1]. Obtaining the adjoint derivative information took about a week to complete by a computer science intern. The result was validated against other AD tools as well as against finite differences [1]. For this case, 1,000 samples from the uncertainty space were obtained and propagated through MATWS and used in testing and assessment (but not in training).

3.2.3 MATLAB Results

GEUK and regression models were constructed by using different numbers of training points and regression orders. Four configurations — which we call data sets due to the different number of sample points — were considered: (1) a maximum third-order regression based on 8 training points, (2) a maximum second-order regression based on 6 points, (3) a second-order regression based on 4 points and (4) a second-order regression

Table 3.3: MATLAB – square exponential – Comparison of error for Kriging and regression models

Data Set	Kriging RMS	Regression RMS	Kriging Max	Regression Max
1	0.11554	0.47118	0.70207	2.194
2	0.58351	0.76058	2.5731	3.2553
3	0.77163	1.1982	3.2202	4.8668
4	0.77163	1.289	3.2204	5.0067

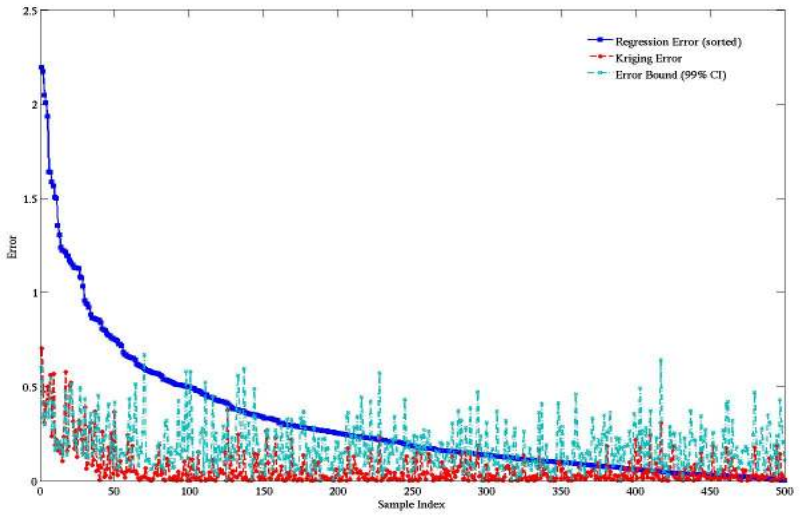

Figure 3.4: MATLAB – square exponential – pointwise error in regression compared with error in Kriging model for third order regression based on 8 training points

based on 4 training points with partially converged function values. The purpose of this final scenario was to assess the ability of the Kriging model to capture noisy data by fitting the noise parameter as an additive-independent noise component (nugget) through the maximum likelihood approach. The performance of the model was assessed based on the RMS error and maximum error associated with the surrogate approximation. In addition to accuracy, the distribution of the validation data about the mean prediction was examined to determine the validity of the variance prediction of the GEUK model. First, the GEUK model was compared with regression for the four data sets outlined previously. These results are found in Table 3.3. For these data sets, the GEUK model produces a more accurate surrogate (mean prediction (2.6)) both in terms of RMS error and maximum error (H1). Because of the small number of training points used for these models, the square exponential covariance function was used.

In order to fully assess the performance of the GEUK model, the model should be compared to regression throughout the domain. Because the data is twelve dimensional, the design space cannot be visualized directly. Hence, in order to give an idea of how the model performs throughout the domain, the pointwise error associated with the regression and the GEUK models is plotted in Figure 3.4. To add an element of order to the plot, we sort the error associated with the regression in descending order and plot it with the corresponding GEUK error. As the plot demonstrates, the error of the GEUK model is lower than that of the regression model for nearly all the validation points (H1). Plotted along with the pointwise error is a 99% error bound for the Kriging model. This bound corresponds to three standard deviations at each point. As the plot shows, the Kriging error is below this predicted bound for nearly all of the validation points (H4).

In addition to assessing the performance of the GEUK model as a function of training points, the effect of covariance function on the error of the surrogate was examined. For this examination, the first data set (8 points with third-order regression) was used. Table 3.4 shows the approximation error for a GEUK model

Table 3.4: MATLAB – square exponential – comparison of error between covariance functions

Covariance Function	RMS Error	Max Error
Cubic Spline 1	0.57759	2.2239
Cubic Spline 2	0.26542	1.5721
Squared Exponential	0.11554	0.70207
Matern-3/2	0.33149	1.7272
Matern-5/2	0.20259	1.1576

Table 3.5: MATLAB – square exponential – statistics for Kriging prediction

Data Set	$\pm 1\sigma$	$\pm 2\sigma$	$\pm 3\sigma$
1	0.690	0.882	0.950
2	0.378	0.568	0.668
3	0.362	0.626	0.720
4	0.362	0.626	0.720

using each covariance function. For the small number of training points used for this test, the accuracy of the surrogate is greatly affected by the choice of covariance function, with the smoothest covariance function producing the lowest error approximation (H5). Nevertheless, we point out that the all covariance functions performed similarly, with a ratio of the errors of 2–4 larger compared with the best case of the square exponential.

Besides producing a more accurate surrogate, the ability of the GEUK model to predict the distribution of the validation data through the variance was tested. By predicting the distribution of the validation data, confidence intervals for the GEUK prediction can be specified. The distribution predicted by the GEUK model can be validated in one of two ways. The simplistic validation assumes the validation data is independent. In this case, the statistics relating to the Kriging model can be predicted by simply measuring the fraction of validation data that falls within a specified number of standard deviations from the mean prediction. These statistics should approximately follow a normal distribution, with 99% of the data falling within three standard deviations from the mean. The results for the four MATLAB data sets are given in Table 3.5. For the first data set, the distribution of the data set roughly follows that of a normal distribution. For the other three data sets, the comparison to the normal distribution is less favorable. Although the validation data is not normally distributed about the mean prediction, the variance still bounds a large portion of the data and can be used to give a rough idea of the uncertainty present in the surrogate (H4).

In contrast to the simplistic approach, the GEUK distribution estimate can be validated by assuming correlation within the validation data. The correlation between the validation data is estimated by using the covariance matrix of the GEUK model. In order to test the distribution prediction, the validation data must be decorrelated; the resulting samples may be tested by using statistics based on multiple independent samples from the same normal distribution. In order to demonstrate this decorrelation property, the details of extracting samples from a Gaussian process must be examined. Samples from a gaussian process with mean function $m(x)$ and covariance matrix Σ can be extracted by using the following formula [21]:

$$\hat{y} = m(x) + \Sigma^{1/2} u, \tag{3.4}$$

where u is a vector of random variables sampled from the standard normal. If the validation data y, follows the proposed Gaussian process, $N(y_*, \Sigma^*)$, then the following formula should produce a vector of variables distributed by the standard normal:

$$Z = \Sigma^{-1/2}(y - m(x)). \tag{3.5}$$

The degree to which the elements of Z obey a standard normal distribution gives a measure of how well the GEUK model predicts the distribution of the validation data about the mean. For this work, the Kolmogorov-Smirnov metric is used to compare the empirical distribution of Z to the standard normal distribution. The

Table 3.6: MATLAB – square exponential – statistics for Kriging prediction using decorrelation

Training Points	KS Metric	$\pm 1\sigma^*$	$\pm 2\sigma^*$	$\pm 3\sigma^*$
4	0.39	0.1420	0.2560	0.3780
6	0.32	0.2000	0.3800	0.4880
8	0.14	0.4880	0.7220	0.8380

Table 3.7: MATLAB – comparison of data distribution between covariance functions

Covariance Function	$\pm 1\sigma$	$\pm 2\sigma$	$\pm 3\sigma$
Cubic Spline 1	0.290	0.592	0.758
Cubic Spline 2	0.776	0.878	0.930
Squared Exponential	0.690	0.882	0.95
Matern-3/2	0.676	0.874	0.932
Matern-5/2	0.704	0.884	0.928

Kolmogorov-Smirnov (KS) metric uses the supremum between the empirical distribution function of Z and the normal CDF as its test metric. As such, this norm will be used to compare the performance of the GEUK model under different scenarios. Its assymptotic behavior is such that the assumption that the underlying distribution is normal cannot be rejected at 90% confidences if the KS score is less than $\approx \frac{1.07}{\sqrt{n}}$. For example, for $n = 40$ this score is about 0.1655, whereas for $n = 10$ it is about 0.3226. Table 3.6 shows the results of the KS test for the MATLAB data using 4,6, and 8 training points. Included with the supremum results are the values of the EDF of Z at 1,2 and 3 standard deviations. These values (obtained over the entire test set) should approximately equal those of the normal distribution.

As these results show, when the correlation of the data is considered, the distribution prediction deviates significantly from the normal distribution for all cases, with only the $N = 8$ prediction giving results remotely resembling a normal distribution. Hence, the two distribution tests give conflicting results. We thus have to say that the hypothesis H4 was not satisfied here and that decorrelation actually hurt instead of helping. Determining the appropriate test for a given situation will likely require further research since both approaches have utility in different sampling regimes.

As was the case with the approximation error, the effect of covariance function on the predicted distribution was examined. Again, the first data set (8 points with third-order regression) was used for this test. Both distribution tests were used to gauge the performance of the GEUK variance prediction. Table 3.7 shows the distribution results based on the assumption of independent validation data. For this case, the choice of covariance function does not appear to greatly affect the distribution prediction with all but the cubic spline 1 function producing results resembling the standard normal (H4,H5). Table 3.8 contains the results for the decorrelated distribution test. When the validation is decorrelated, the results appear to be more sensitive to the choice of covariance function, with the squared exponential and Matern-3/2 functions performing the best and the cubic spline functions deviating the most from the theoretical results (H5).

The twelve-dimensional MATLAB model is an ideal problem to demonstrate the utility of including derivative observations in the training of the surrogate. As the dimension becomes higher, the reduced

Table 3.8: MATLAB – square exponential – statistics for Kriging prediction using decorrelation

Covariance Function	KS Metric	$\pm 1\sigma^*$	$\pm 2\sigma^*$	$\pm 3\sigma^*$
Cubic Spline 1	0.2939	0.3080	0.4520	0.5960
Cubic Spline 2	0.2766	0.2820	0.3800	0.6540
Squared Exponential	0.1412	0.4880	0.7220	0.8380
Matern-3/2	0.1974	0.4200	0.6820	0.8320
Matern-5/2	0.2383	0.3300	0.5700	0.7160

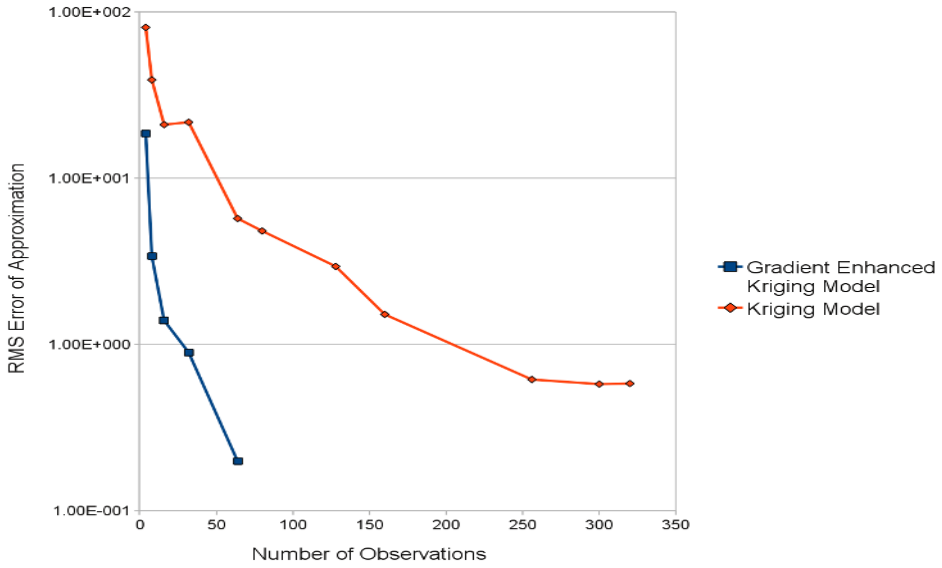

Figure 3.5: MATLAB – cubic spline 2 – error scaling with number of observations for GEK model and Kriging model.

cost possible through adjoint methods becomes more pronounced, and the additional training information is acquired for a fixed cost. Figure 3.5 shows the RMS error of the surrogate approximation as a function of training points for both the GEK model and Kriging model. For this test, the cubic spline 2 covariance function was used along with a zeroth-order regression (ordinary Kriging). As the plot shows, the error in the GEK model decreases signficantly faster than that of the Kriging model (H2). For this plot, however, the cost of acquiring the training data is not considered. For a fairer comparison, the GEK results should be scaled to account for the additional cost associated with calculating function and derivative values. If derivative values are calculated by using an adjoint-based approach, the cost is independent of the dimension of the problem and is directly proportional to the cost of a function evaluation. The constant of proportionality varies depending on the detials of the adjoint implementation. The worst-case estimate for this constant is a factor of 5, with typical values varying between 2 and 3 [8]. Assuming the worst-case scenario of the factor of 5, a function/derivative evaluation is equivalent to six function evaluations. Figure 3.6 scales the GEK results by this factor. As the plot shows, even with this conservative cost estimate, the error in the GEK model still decreases faster than in the Kriging model (H2). Moreover, the model with derivative information gets to an accuracy level that is never reached by the approach that does not use derivative information. In our experience, however, many of these derivatives take much less to compute, thus making these comparisons even more significantly tilted in favor of the gradient use.

The effect of regression order in approximating the MATLAB data was also examined. Figure 3.7 shows the approximation error of the GEUK model as a function of training points with three different regression orders (zeroth through second order). As the plot shows, increasing the regression order consistently decreased the approximation error in the surrogate model. In particular, GEUK outperforms ordinary kriging ($P = 0$) (H3). This decrease occurred for every number of training points, suggesting it is beneficial to always uses the highest possible regression order. Because of the nature of the MATLAB model, the design space is relatively smooth and slowly varying, making the MATLAB model more amenable to polynomial regression approaches.

3.2.4 MATWS Results

In order to test the performance of the GEUK model on a more complicated design space, surrogates for MATWS simulation results were created. The MATWS model is four dimensional, and the surrogate was compared against 1,000 validation points. The surrogates for these tests are based on function evaluations and derivative values with respect to three of the four input parameters. For the first tests, the performance

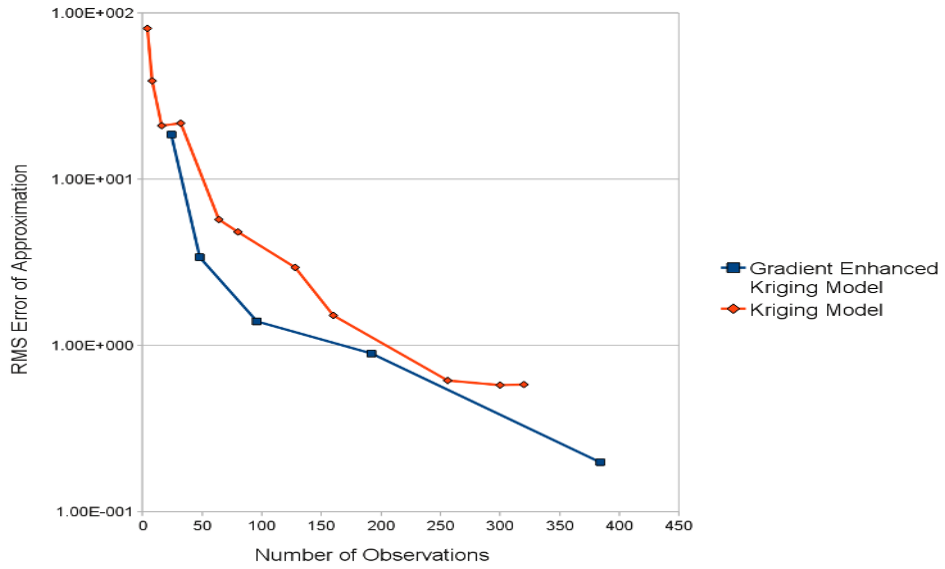

Figure 3.6: MATLAB – cubic spline 2 – error scaling with equivalent number of observations for GEK model and Kriging model

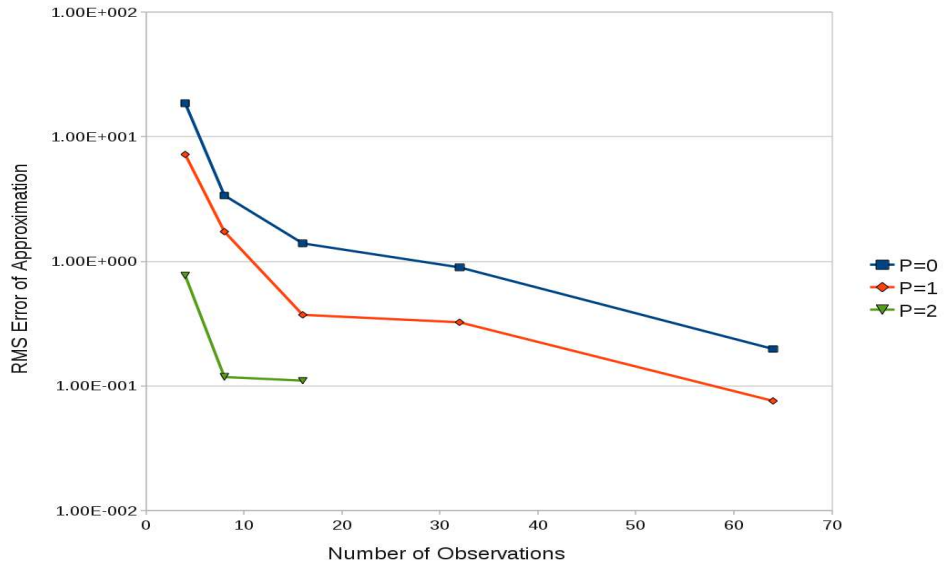

Figure 3.7: MATLAB – cubic spline 2 – comparison of error for universal kriging model using different regression orders

Table 3.9: MATWS – cubic spline 2 – comparison of error for Kriging and regression models

Sample Points	Kriging RMS	Regression RMS	Kriging Max	Regression Max
4 (p=2)	3.6433	15.2304	13.7491	56.6632
6 (p=2)	0.5260	3.2833	2.2040	14.0380
8 (p=3,trunc)	0.1841	0.5695	1.1980	3.1272
16	0.0766	0.427	0.747	2.404
24	0.0887	0.405	0.910	1.877
32	0.0995	0.309	1.118	1.959
40	0.0517	0.295	0.437	2.112
50	0.0508	0.251	0.386	1.476
100	0.0337	0.181	0.0998	1.068

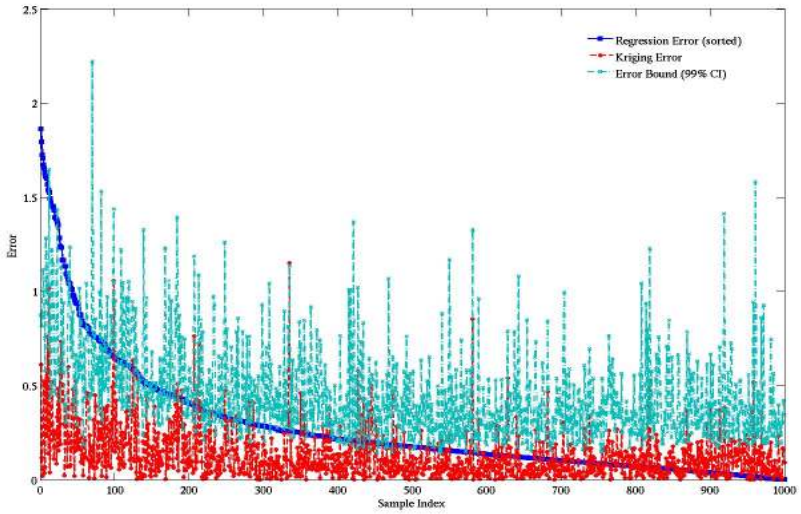

Figure 3.8: MATWS – cubic spline 2 – point-wise error in regression compared with error in Kriging model for third-order regression based on 16 training points from MATWS data

of the GEUK model was compared with the regression model for different numbers of training points. For the n=4 and n=6 results, a second-order regression with a truncated basis was used. For the n=8 results, the third-order truncated basis was used. For the rest of the results ($n > 8$), the full third-order basis was used. The comparison of the GEUK model with regression can be found in Table 3.9. The GEUK model outperforms the regression for all numbers of training points tested, particularly for large numbers of training points (H1). As the number of training points is increased, the regression system becomes more overdetermined, and the GEUK model is able to assimilate this addition information more effectively. The pointwise error comparison for the N=16 case is plotted in Figure 3.8. For the MATWS data, we see that the improvement over regression is nearly uniform and three standard deviations substantially overestimate the error (H4).

In order to gain perspective on the effect of covariance function on the approximation error for the MATWS data, the performance of the GEUK model using each covariance function was examined for the N=8 and N=50 test cases. These results are given in Tables 3.10 and 3.11. For this data, the effect of covariance function appears to be less dramatic. For the small sample case (N=8), the distribution prediction is not greatly affected by covariance choice, so the assumption H5 is weak in this case. In terms of accuracy, the smoothest covariance functions (SE and Matern-5/2) produce the most accurate approximations while the sparse kernel covariance functions perform the worst. For the large sample case (N=50), the situation is reversed, and the least smooth function (Matern-3/2) performs the best in terms of accuracy (H5). Perhaps

Table 3.10: MATWS – comparison of covariance functions for 8 pt GEUK model

Covariance Function	RMS Error	Max Error
Cubic Spline 1	0.2083	1.3567
Cubic Spline 2	0.1841	1.1980
Squared Exponential	0.1245	1.0125
Matern-3/2	0.1704	1.0645
Matern-5/2	0.1530	1.0566

Table 3.11: MATWS – comparison of covariance functions for 50 pt GEUK model

Covariance Function	RMS Error	Max Error
Cubic Spline 1	0.0567	0.3963
Cubic Spline 2	0.0528	0.4551
Squared Exponential	0.1487	2.0268
Matern-3/2	0.0398	0.2552
Matern-5/2	0.0749	0.7991

even more interesting is the failure of the square exponential kernel, whose RMS counterintuitively increases as the number of samples increases. This shows that the exceeding smoothness assumption carried by the square exponential is detrimental in this case, as was hypothesized in [26].

As was the case with the MATLAB data, the performance of the GEUK distribution prediction for the MATWS data was examined. This examination was performed assuming both independent and correlated validation data. Table 3.12 shows the distribution of the validation data about the mean prediction assuming independent validation data. As the table demonstrates, for small numbers of training points, the distribution of the data treated as independent closely follows a normal, with the 3σ bound encompassing essentially all the validation data (H4). As the number of samples is increased, the distribution of the data around the mean begins to deviate from the normal. Using the independent data distribution test, the covariance function was also varied to determine its effect on the distribution prediction. This variation was performed with 8 and 50 training points. The results are given in Tables 3.13 and 3.14. For this test, the results indicate that the Matern-3/2 covariance function gives the best distribution results for both N=8 and N=50 (H5).

In addition to testing the distribution based on independent validation data, the test was repeated using the decorrelated test. This test was performed using the cubic spline 2 covariance function and the Matern-3/2 function in order to examine the effect of covariance function for the distribution results. These two functions were chosen because their conditional distribution calculation could be carried out; the covariance matrix was negative definite to machine precision for the other cases, and particularly so for the square exponential. This is not surprising as the excessive smoothness of the square exponential results in columns in the matrix that are closer to dependency. The results in shown in 3.15 and 3.16 demonstrate two points.

Table 3.12: MATWS – cubic spline 2 – statistics for kriging prediction

Data Set	$\pm1\sigma$	$\pm2\sigma$	$\pm3\sigma$
4 (p=2)	0.847	0.977	0.999
6 (p=2)	0.513	0.964	1.000
8 (p=3,trunc)	0.599	0.968	1.000
16	0.697	0.942	1.000
24	0.533	0.857	0.971
32	0.525	0.817	0.942
40	0.475	0.800	0.937
50	0.366	0.685	0.870
100	0.221	0.424	0.600

Table 3.13: MATWS – comparison of covariance functions for MATWS data (N=8)

Covariance Function	MAX Error	RMS	$\pm 1\sigma$	$\pm 2\sigma$	$\pm 3\sigma$
Cubic Spline 1	1.3567	0.2083	0.550	0.924	1.000
Cubic Spline 2	1.1980	0.1841	0.599	0.968	1.000
Squared Exponential	1.0125	0.1245	0.524	0.887	0.982
Matern-3/2	1.0645	0.1704	0.7210	0.999	1.000
Matern-5/2	1.0566	0.153	0.455	0.880	0.997

Table 3.14: MATWS – comparison of covariance functions for MATWS data (N=50)

Covariance Function	MAX Error	RMS	$\pm 1\sigma$	$\pm 2\sigma$	$\pm 3\sigma$
Cubic Spline 1	0.3963	0.0567	0.413	0.710	0.886
Cubic Spline 2	0.4551	0.0528	0.353	0.641	0.842
Squared Exponential	2.0268	0.1487	0.333	0.601	0.787
Matern-3/2	0.2552	0.0398	0.395	0.697	0.869
Matern-5/2	0.7991	0.0749	0.453	0.746	0.880

For both covariance functions, the KS metric decreases as the number of training points increases, and the quantile values tend toward those of the normal distribution, indicating a stability of the procedure with sample size (H4). For smaller sample sizes, the Matern-3/2 covariance function tends to produce slightly better agreement with the standard normal distribution based on the KS metric and the quantile values (H5). As the number of training points is increased, the effect of covariance function becomes diminished.

In order to give a more visual representation of the decorrelated data distribution test, the resulting empirical distribution functions for Z are plotted along with the normal CDF in Figures 3.9 and 3.10. These plots correspond to GEUK models utilizing 8 and 50 training points, respectively. As the plots show, for the less accurate surrogate, the empirical distribution function for Z becomes stretched signifying that the variance in the validation data is underestimated (recall that the decorrelation requires the inverse of the covariance matrix).

Using the MATWS data, the GEUK model is compared with universal Kriging (without derivative observations) and gradient-enhanced ordinary Kriging. In order to quantify the advantage of including derivative observations, the scaling of the RMS error as a function of training points was measured for the GEUK model and universal Kriging model. For these tests, the regression order was chosen such that the size of the basis was half the available number of observations, with the regression orders varying between second order with a truncated basis and third order with the full basis. The results of this scaling test are plotted in Figure 3.11. For this plot, the results are not scaled for computational expense because of the relatively low dimension of the data. As the results show, the addition of derivative observations produces a surrogate with lower approximation error for a fixed number of training points (H2). The GEUK model

Table 3.15: MATWS – cubic spline 2 – Z scores for Kriging prediction

Data Set	KS metric	$\pm 1\sigma^*$	$\pm 2\sigma^*$	$\pm 3\sigma^*$
4 (p=2)	0.549	0.0120	0.0300	0.0360
6 (p=2)	0.315	0.2020	0.3910	0.5300
8	0.129	0.5260	0.7480	0.8380
16	0.052	0.7390	0.9410	0.9750
24	0.062	0.7590	0.9370	0.9660
32	0.098	0.5250	0.8700	0.9630
40	0.069	0.7940	0.9680	0.9900
50	0.018	0.6970	0.9500	0.9830

Table 3.16: MATWS – Matern-3/2 – Z scores for Kriging Prediction

Data Set	KS metric	$\pm 1\sigma^*$	$\pm 2\sigma^*$	$\pm 3\sigma^*$
4 (p=2)	0.5580	0.0030	0.0090	0.0150
6 (p=2)	0.2782	0.2510	0.4780	0.6030
8	0.1557	0.4640	0.7070	0.8130
16	0.0645	0.6150	0.8810	0.9510
24	0.0297	0.6890	0.9450	0.9840
32	0.0770	0.8200	0.9690	0.9840
40	0.0269	0.7150	0.9590	0.9900
50	0.0601	0.7890	0.9870	1.0000

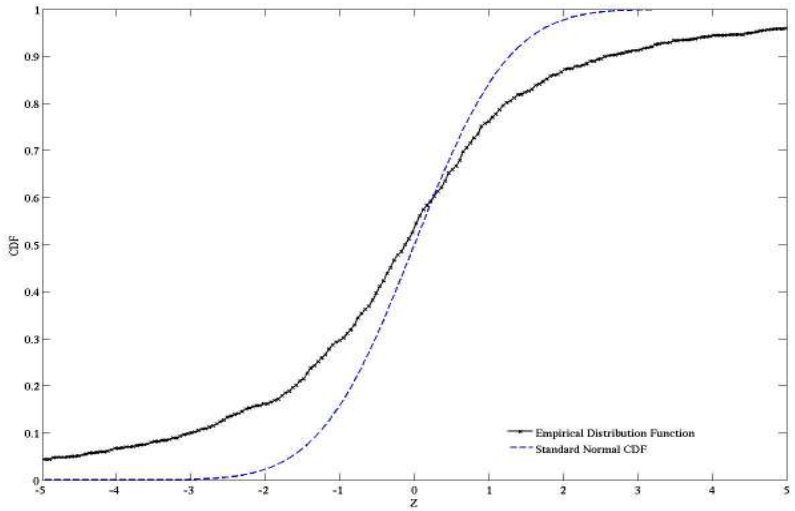

Figure 3.9: MATWS – Matern-3/2 – distribution of Z scores for N=8 case using Matern function with $\nu = 3/2$

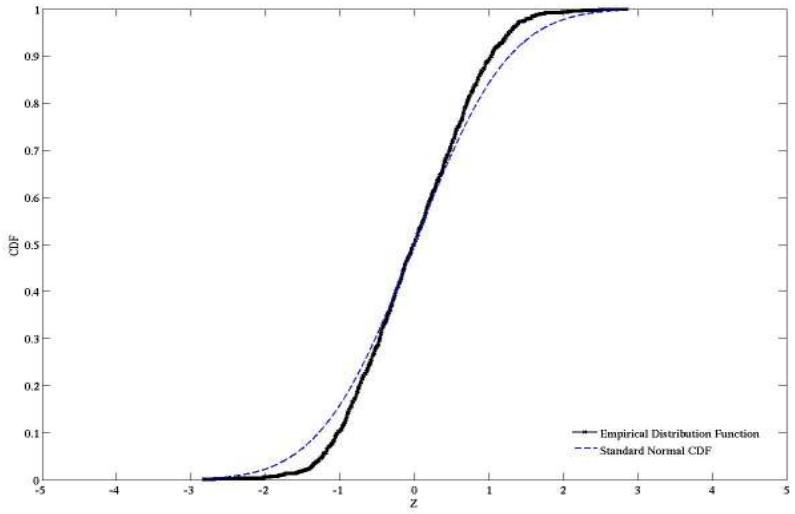

Figure 3.10: MATWS – Matern-3/2 – Distribution of Z scores for N=50 case using Matern function with $\nu = 3/2$

Table 3.17: MATWS – cubic spline 2 – comparison of GEUK model with ordinary GEK model

Training Points	GEUK RMS	OGEK RMS	GEUK Max	OGEK Max
4	3.6433	0.5729	13.749	3.9739
6	0.5260	0.2585	2.2040	1.8793
8	0.1841	0.1931	1.1980	1.7142
16	0.0766	0.0968	0.7474	1.2039
24	0.0887	0.0845	0.9104	1.0363
32	0.0995	0.0863	1.1181	1.1478
40	0.0517	0.0551	0.4368	0.7108
50	0.0508	0.0498	0.3862	0.6151

benefits over the universal Kriging model in two ways. First, the GEUK model is often able to use a higher-order regression for a given number of sample points. Second, the addition of derivative observations gives more training data for the Kriging model and allows both function and derivative values to be matched at the sample points. We nevertheless point out that the precision level of GEUK is never reached by UK for the cases considered (H3).

In addition to comparing GEUK to UK, the GEUK model was compared to ordinary GEK in terms of approximation error of the surrogate, pertaining to (H3). The results of this comparison are given in Table 3.17. For the MATWS design space, the use of a higher-order mean function gives a better approximation in terms of RMS error for $N \geq 8$, but only marginally. When the comparison is based on maximum error, the GEUK model fares slightly better, but the results are not as dramatic as the MATLAB model results (H3). This result is most likely due to the nature of the MATWS design space which contains significantly more noise when compared to the MATLAB model.

3.3 Summary Findings of the Numerical Experiments

We find that hypotheses H1 and H2 were verified well in our explicit, MATLAB, and MATWS experiments. That is, GEUK peforms better than regression, and the use of derivative information results in less error, even for similar computational costs. For the hypothesis H3, the conclusion is less strong, in that we find that the GEUK model can perform much better than ordinary Kriging (Explicit, MATLAB) but sometimes

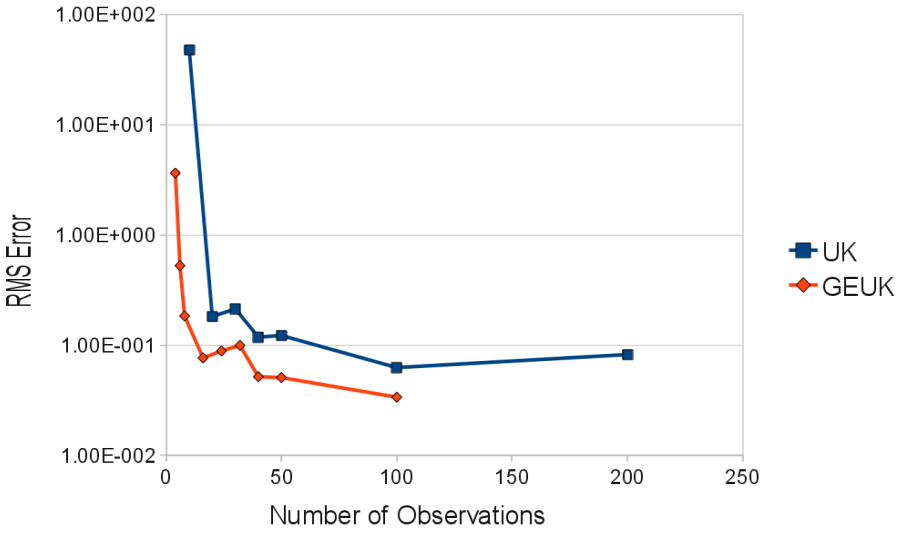

Figure 3.11: MATWS – cubic spline 2 – scaling comparison between GEUK and UK

performs comparably to it (MATWS). But in any case, the use of a mean function does not hurt and many times, it substantially helps. For the hypothesis H4, the conclusion is not that clearly cut. In general, for low sample size, ignoring correlation between samples results in correct quantile values (based on comparisons to standard normals), whereas for increasing sample size, this approach leads to the wrong values, whereas decorrelating the test values first using the inferred covariance matrix leads to the correct quantile values. In addition, the KS statistic indeed seems to decrease to zero as the sample size is increased, even when we use derivative information. The fact that the KS statistics decrease with increasing sample size and that at least one regime is correct is encouraging. The difficulty is to determine when to switch from one to the other; this topic needs further investigation. In the limit of low sample sizes, which is where we are aiming for here, the first approach to try is to use posterior variance but ignore correlation. For assumption (H5), we find that the Matern-3/2 covariance function always gives results that are at least comparable to the other covariance functions, and most times much better (have either less error in the posterior mean or distribution of the error closer to a normal distribution). This behavior can be justified by the observation that the Matern class of the tends to exhibit the most conservative covariance estimates for a given level of smoothness guaranteed in the data [26].

4 Estimating Quantiles under Input and Model Error

In §3 we investigated the system response approximation properties of GEUK but not their performance in uncertainty propagation. To some extent, the probability structure in the parameter x was not relevant. In some ways, we expect the existence of a probabilistic structure on x to simplify our task compared with the approximation investigation in §3, since occasional excursions from the mean now become less significant if they occur sufficiently rarely in the probabilistic sense induced by the distribution of x.

To demonstrate the potential of our method in propagating uncertainty, we investigate how well our approach predicts an important measure of system uncertainty, that is, quantiles of the output quantity $J(s(x), x)$ given a probability density structure on x.

4.1 Confidence Intervals for Quantiles

Suppose that we have the random variables $X_1, X_2, \ldots, X_N \overset{\text{i.i.d}}{\sim} F$. Here we denote by F the cumulative distribution of X_1 and by $X \sim F$ a variable distributed with the cumulative F and by $i.i.d$ variables independent identically distributed. We wish to estimate ζ_p the pth quantile of F, that is, $\zeta_p = F^{-1}(p)$. We

are interested in obtaining a confidence interval for ζ_p based on the N sample values. We denote by $Y_{i:N}$ the ith order statistic out of n *i.i.d* samples, that is, $Y_{i:N} = Y_{\sigma(i)}$, $i = 1, 2, \ldots, N$, where σ is a permutation such that $i \leq j \Rightarrow Y_{\sigma(i)} \leq Y_{\sigma(j)}$. In other words, $\{Y_{\sigma(i)}\}$ is the increasing ordering of $\{Y_i\}$.

We then have the following exact relationship:

$$(F(X_{1:N}), F(X_{2:N}), \ldots, F(X_{N:N})) \sim (U_{1:N}, U_{2:N}, \ldots, U_{N:N}), \tag{4.1}$$

where $U_{i:n}$ is the ith order statistics of a variable uniformly distributed in $[0, 1]$ (which we denote by $U[0, 1]$), provided F is continuous [4]. In particular,

$$P_F(X_{i:N} \leq \zeta_p) = P(U_{i:N} \leq p) = 1 - \alpha_{i,N,p}. \tag{4.2}$$

The probability above and the parameter $\alpha_{i,N,p}$ depend only on i, N, p and not on F. We note that this approach will provide estimates irrespective of how few samples N are computed. Those estimates are likely to be accurate, however, only when $N > \frac{1}{p}$.

4.2 Quantile Estimation for Mixed Uncertainty and Model Error

In our setting, we have $(\Omega^u, \mathcal{F}^u, \mathcal{P}^u)$, the probability space attached to the input uncertainty, and $x(\omega^u) : \Omega^u \to \mathbb{R}^{n_x}$ defines the random variable of the input uncertainty. In addition, we assume we have defined (by means of GEUK) a probabilistic model of the system response defined by means of the probability space $(\Omega^e, \mathcal{F}^e, \mathcal{P}^e)$. In this context the random variable is $\mathcal{J}(\omega^e, \cdot) : \Omega^e \to \mathcal{C}^1(\mathbb{R}^{n_x})$. We now have that the system response is a random variable $\widetilde{\mathcal{J}}(\omega^e, \omega^u) = \mathcal{J}(\omega^e, u(\omega^u)) : \Omega^e \times \Omega^u \to \mathbb{R}$.

With these definitions, our aim is to provide a confidence interval for the stochastic system response, the random variable $\widetilde{\mathcal{J}}(\omega^e, \omega^u)$. To provide a set of N i.i.d samples, we note that the random variables $f(\omega^e, \cdot)$ and $u(\omega^u)$ are independent. Therefore, to produce N *i.i.d* random variables with the same distribution as $\widetilde{\mathcal{J}}(\omega^e, \omega^u)$, we need only to produce N samples $x_i(\omega^u) \overset{i.i.d}{\sim} x(\omega^u)$, $i = 1, 2, \ldots, N$, and, $\mathcal{J}_i(\omega^e, \cdot) \overset{i.i.d}{\sim} \mathcal{J}(\omega^e, \cdot)$, $i = 1, 2, \ldots, N$. Then, $\mathcal{J}_i(\omega^e, u_i(\omega^u)) \overset{i.i.d}{\sim} \widetilde{\mathcal{J}}(\omega^e, \omega^u)$.

4.3 Algorithms for Estimating ζ^p

We take the following steps, assuming N, p, and the confidence level α is given. Typically, the number of samples N is given by computational considerations, whereas p and α are imposed by the design requirements. In the latter case a good example is the well-known 95/95 estimate in nuclear engineering applications, where $p = 0.95$, and $\alpha = 0.05$.

One algorithm, based on (4.1), is the following.

1. Determine k such that $\alpha_{k,p,N} \leq \alpha$ in (4.2). For example, compute k by simulation by successively extracting replications of N i.i.d samples $\sim U[0, 1]$ until the order statistics are sufficiently converged (e.g. by using a normal test for the mean estimate of the quantile correspondent of the order statistic, which holds from the law of large numbers). Then choose the largest k that is smaller than the pth quantile.

2. Extract N samples $x_i(\omega^u) \overset{i.i.d}{\sim} x(\omega^u)$, $i = 1, 2, \ldots, N$ (that is, N vectors of length u_i). In other words, use i.i.d samples from the input space.

3. Extract N samples $\mathcal{J}_i(\omega^e, \cdot) \overset{i.i.d}{\sim} \mathcal{J}(\omega^e, \cdot)$, $i = 1, 2, \ldots, N$, and evaluate at u_i. In our process model, it is sufficient to draw from the random variable $\mathcal{J}(\omega^e, x_i)$ for increasing i and denote it $\widetilde{\mathcal{J}}_i$.

4. Order $\widetilde{\mathcal{J}}_i$ increasingly, and extract the kth value, which we denote $\widetilde{\mathcal{J}}_{k:N}$. Then $[\widetilde{J}_{k:N}, \infty)$ is an at least $1 - \alpha$ confidence interval for the p-th percentile ζ^p.

We hypothesise that this approach will result in a conservative estimate because of its reliance on an exact relationship (4.1) as well as the fact that the kth value is not the exact quantile for the given p.

Another approach is the following.

Table 4.1: MATLAB – cubic spline 2 – quantile calculation for MATLAB data

Sample Points	Regression Order	Kriging Estimate	Regression Estimate	Training Estimate
4	2	2446.6	2447.2	2323.8
6	2	2448.2	2447.5	2335.2
8	3	2449.1	2448.4	2360.4

Actual Value = 2456.0

1. Sample M times from the functional UQ GP model, $\mathcal{J}_i(\omega^e, \cdot) \overset{i.i.d}{\sim} \mathcal{J}(\omega^e, \cdot)$, $i = 1, 2, \ldots, M$.
2. For each sample, report the estimate of the pth quantile, conditional on the sample functional $\mathcal{J}_i$. That is, consider the randomness only in the input space. This gives the random variable $\hat{\zeta}^p(\omega^e)$.
3. Compute the $1 - \alpha$ double sided confidence interval from the empirical distribution of $\hat{\zeta}^p(\omega^e)$.

We call this approach "conditional quantile estimate". Two difficulties arise with this approach. First, since we use empirical density estimates the M value must be large. More importantly, though, we have no theory to explain how this conditional p quantile distribution relates to the overall p quantile. Absent that, the $1 - \alpha$ confidence interval is a heuristic, albeit a fairly intuitive one. Nevertheless, the approach leverages an important feature of the GEUK approach: the existence of a reasonable (as demonstrated in §3) error model of the deviation between the GEUK prediction and the actual system response.

4.4 Example Results

We first investigate the estimator based on (4.1). In order to demonstrate this approach for computing quantiles, the 95th percentile for the MATLAB and MATWS data was estimated to a confidence of 95% using the GEUK model. In addition to the estimate based on the GEUK model, the same quantile was estimated by using the regression model and the training data used to construct the surrogate models. This last scenario represents the only estimate possible if no surrogate of the design space is constructed. We do not expect this to be a good estimator, we simply want to point out the value of building a surrogate by using GEUK. As the results demonstrate, the quantile estimate based on samples extracted from the surrogate are more accurate than estimates based solely on the training data. We also note that, in the case of regression, the approach treats the regression surface as if it is the exact system response, with no quantification of the error between them. If this error is small, the resulting quantity will be a good estimate of the actual quantile value.

As a first test, the 95th percentile of fuel pin temperature based on the MATLAB model was estimated with a 95% confidence level. The results of this estimate for the MATLAB model as well as the actual quantile based on the 500 validation points are given in Table 4.1. We point out that here and in the rest of this section N denotes the number of training points used to construct the model. Because the quantile estimation algorithm requires the Cholesky factorization of the covariance matrix in the case of the GEUK model, the cubic spline 2 covariance function was used for the sake of robustness. As the table shows, the estimate based on just the training data tends to greatly underestimate the quantile value. Using the GEUK or regression greatly reduces the discrepancy between quantile estimate and the actual value (as computed from a large number of sample values). For the MATLAB data, the regression and GEUK model perform similarly, with the GEUK model producing a slightly more accurate estimate when 6 or 8 training points are used.

In addition to estimating quantiles associated with the MATLAB data, the 95th percentile of the MATWS data was also estimated. Again, this estimate was determined using a 95% confidence level and the cubic spline 2 covariance function was used in the GEUK model. The results for the MATWS data and the actual quantile from the 1000 validation points is found in Table 4.2. Before examining the results, we note that the range of the data for the MATWS data is relatively small, with all values falling in the interval of $[859.15, 869.66]$. Hence, a difference of only a fraction of a degree in the quantile estimate is significant. As was the case with the MATLAB data, the quantile estimate based exclusively on training data consistently and significantly underestimates the actual quantile value. For large N, the estimate based on the regression

Table 4.2: MATWS – cubic spline 2 – quantile calculation for MATWS data

Sample Points	Kriging Estimate	Regression Estimate	Training Estimate
4 (p=2)	865.73	864.78	863.55
6 (p=2)	865.86	871.15	863.55
8	866.08	866.60	863.46
16	865.89	866.51	865.45
24	865.83	866.49	865.56
32	865.87	866.32	865.76
40	865.82	866.37	865.86
50	865.83	866.42	865.86

Actual Value = 866.16

Table 4.3: MATWS – comparison of covariance functions for quantile calculations with MATWS data N=50

Covariance Function	Kriging Estimate
Cubic Spline 1	865.8354
Cubic Spline 2	865.8666
Squared Exponential	865.8256
Matern-3/2	865.8067
Matern-5/2	865.8334

Actual Value = 866.16

surrogate model is closer to the actual estimate but consistently overestimates the actual value. For the critical region of small N, however, GEUK does substantially better in terms of distance to the actual quantile. For the MATWS data, the GEUK model performs the best because it produces a more accurate yet still conservative estimate of the quantile. We note that the GEUK quantile confidence interval (the semi-infinite interval that has as its left end the estimator) always contains the actual value. This situation substantiates our claim that this approach will result in an accurate but conservative confidence interval estimate.

In order to assess the effect of covariance function on the quantile estimate, the 95th percentile for the MATWS and MATLAB data was again estimated. For the MATWS data, the GEUK model was constructed by using each covariance function based on 50 training points. For the MATLAB data, 8 training points were used for the GEUK model. The results of this test are given in Tables 4.3 and 4.4. For both sets of data, the choice of covariance function does not greatly affect the quantile estimate. For the MATWS data, the Matern functions and cubic splines all give essentially the same estimate. For the MATLAB data, the estimates are slightly more varied, with the squared exponential giving the best estimate followed by the Matern function with $\nu = 3/2$. These results suggest a robustness in the quantile estimate algorithm to the choice of covariance function. Since the GEUK model results are treated as a Gaussian process and samples are extracted from it, deficiencies in the mean behavior of the GEUK model may be compensated by conservative variance estimates.

We now discuss the performance of our proposed conditional quantile estimate. We report the results in

Table 4.4: MATLAB – Comparison of Covariance Functions for Quantile Calculations with MATLAB N=8

Covariance Function	Kriging Estimate
Cubic Spline 1	2448.7
Cubic Spline 2	2448.7
Squared Exponential	2449.0
Matern-3/2	2448.8
Matern-5/2	2448.1

Actual Value = 2456.0

Table 4.5: Quantile – Matern-3/2 – Example results for Kriging quantile estimate with confidence interval

DATA	Training Points	Confidence(1-α)	Lower Bound	Upper Bound	Median
MATWS	8	0.95	866.17	866.45	866.28
MATWS	8	0.99	866.17	866.45	866.28
MATWS	50	0.95	866.07	866.17	866.12
MATLAB	8	0.95	2454.3	2455.6	2445.0
MATLAB	8	0.99	2454.2	2.455.8	2455.0

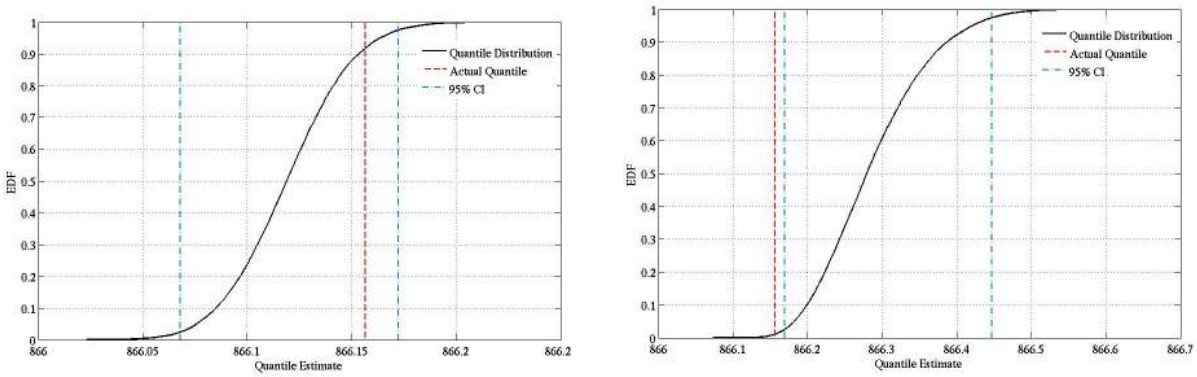

Figure 4.1: MATWS – Matern-3/2 – Distribution of quantile estimates for (left) Kriging model based on 50 points and 95% confidence interval (right) Kriging model based on 8 points and 95%.

Table 4.5 and Figure 4.1. For $N = 8$ we see that, for both MATLAB and MATHWS the actual quantile is outside both the 95% and 99% confidence interval, but only slightly. Clearly the distance to the closest interval end is much smaller than the size of the confidence interval, which itself is quite narrow. In turn, this means that the distance between the endpoints of the interval definitely gives the correct order of the error. In the case $N = 50$, the interval is substantially smaller (unsurprisingly, since we expect GEUK to have a more accurate mean prediction) and it contains the actual quantile. In all cases, the estimators (if we consider the sample median, for example) are far more accurate than in the previous approach. Absent more theoretical analysis, it is difficult to ascertain whether this behavior is simply fortunate, but at least to us its potential for producing good estimates in the case of small size training sets seems clear.

5 Conclusions

The gradient-enhanced universal kriging (GEUK) approach provides accurate representations of a function space with a limited number of function samples. The approach combines the strengths of both regression and Kriging models: it exhibits fast convergences as the polynomial order is increased and provides unbiased estimates of function behavior as the number of training points is increased. The use of gradient information demonstrably reduces the effort required to produce a certaint assessment for the same error level compared to derivative-free approaches. In addition to function predictions, the variance of the Kriging model provides reasonable confidence bounds for model error. Moreover, we demonstrate that GEUK significantly outperforms regression approaches in terms of the quality of the mean prediction, with the additional advantage of providing a statistical error estimate. As a result, quantiles of the systems response under uncertainty can be accurately and conservatively predicted with very few system model samples. Our findings were demonstrated on uncertainty propagation in several explicit function cases and two nuclear engineering models.

Acknowledgments

This work was supported in part by the U.S. Department of Energy under a CSGF fellowship (Brian Lockwood) and under Contract No. DE-AC02-06CH11357 (Mihai Anitescu). We thank our colleagues Mihai Alexe, Tom Fanning, Oleg Roderick, Jean Utke, Zhu Wang, and Wataru Yamazaki for assistance and input.

References

[1] Mihai Alexe, Oleg Roderick, Mihai Anitescu, Jean Utke, Thomas Fanning, and Paul Hovland. Using automatic differentiation in sensitivity analysis of nuclear simulation models. Transactions of the American Nuclear Society, 102:235–237, 2010.

[2] M.J. Bayarri, J.O. Berger, R. Paulo, J. Sacks, J.A. Cafeo, J. Cavendish, C.H. Lin, and J. Tu. A framework for validation of computer models. Technometrics, 49(2):138–154, 2007.

[3] K.P. Burnham and D.R. Anderson. Model selection and multimodel inference: a practical information-theoretic approach. Springer-Verlag, New York, 2002.

[4] A. DasGupta. Asymptotic theory of statistics and probability. Springer-Verlag, New York, 2008.

[5] F. D'AURIA, N. Debrecin, and G.M. Galassi. Outline of the uncertainty methodology based on accuracy extrapolation. Nuclear technology, 109(1):21–38, 1995.

[6] X. Du and W. Chen. Efficient uncertainty analysis methods for multidisciplinary robust design. AIAA Journal, 40(3):545–581, 2002.

[7] M. Goldstein and JC Rougier. Probabilistic formulations for transferring inferences from mathematical models to physical systems. SIAM Journal on Scientific Computing., 26(2):467–487, 2004.

[8] Andreas Griewank. Evaluating derivatives: principles and techniques of algorithmic differentiation. SIAM, Philadelphia, 2000.

[9] T. Hastie, R. Tibshirani, and J. Friedman. The elements of statistical learning. Springer-Verlag, Berlin, 2001.

[10] D. Higdon, J. Gattiker, B. Williams, and M. Rightley. Computer model calibration using high-dimensional output. Journal of the American Statistical Association, 103(482):570–583, 2008.

[11] Dave Higdon, Marc Kennedy, James C. Cavendish, John A. Cafeo, and Robert D. Ryne. Combining field data and computer simulations for calibration and prediction. SIAM Journal on Scientific Computing, 26(2):448–466, 2004.

[12] SS Isukapalli, A. Roy, and PG Georgopoulos. Stochastic response surface methods (SRSMs) for uncertainty propagation: Application to environmental and biological systems. Risk Analysis, 18(3):351–363, 1998.

[13] SS Isukapalli, A. Roy, and PG Georgopoulos. Efficient sensitivity/uncertainty analysis using the combined stochastic response surface method and automated differentiation: Application to environmental and biological systems. Risk Analysis, 20(5):591–602, 2000.

[14] J.C. Jouhaud, P. Sagaut, M. Montagnac, and J. Laurenceau. A surrogate-model based multidisciplinary shape optimization method with application to a 2D subsonic airfoil. Computers & Fluids, 36(3):520–529, 2007.

[15] M.C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. Journal of the Royal Statistical Society. Series B, Statistical Methodology, 63(3):425–464, 2001.

[16] N.H. Kim, H. Wang, and N.V. Queipo. Adaptive reduction of random variables using global sensitivity in reliability-based optimisation. International Journal of Reliability and Safety, 1(1):102–119, 2006.

[17] OP Le Maītre, OM Knio, HN Najm, and RG Ghanem. Uncertainty propagation using Wiener-Haar expansions. Journal of Computational Physics, 197(1):28–57, 2004.

[18] E. E. Morris. Uncertainty in unprotected loss-of-heat-sink, loss-of-flow, and transient-overpower accidents. Technical Report ANL-AFCI-205, Argonne National Laboratory, 2007.

[19] Jorge Nocedal and Stephen J. Wright. Numerical optimization. Springer Series in Operations Research. Springer, 1999.

[20] W.L. Oberkampf, J.C. Helton, C.A. Joslyn, S.F. Wojtkiewicz, and S. Ferson. Challenge problems: uncertainty in system response given uncertain parameters. Reliability Engineering & System Safety, 85(1-3):11–19, 2004.

[21] C.E. Rasmussen and C. Williams. Gaussian processes for machine learning. MIT Press, Cambridge, Massachusets., 2006.

[22] Oleg Roderick, Mihai Anitescu, and Paul Fischer. Polynomial regression approaches using derivative information for uncertainty quantification. Nuclear Science and Engineering, 164(2):122–139, 2010.

[23] Oleg Roderick, Mihai Anitescu, Paul Fischer, and Won-Sik Yang. Stochastic finite-element approach in nuclear reactor uncertainty quantification. Transactions of the American Nuclear Society, 100:317–318, 2009.

[24] J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. Statistical Science, 4(4):409–423, 1989.

[25] E. Solak, R. Murray-Smith, W.E. Leithead, D.J. Leith, and C.E. Rasmussen. Derivative observations in Gaussian process models of dynamical systems. Advances in Neural Information Processing Systems, pages 1057–1064, 2003.

[26] M.L. Stein. Interpolation of spatial data: Some theory for Kriging. Springer-Verlag, Berlin, 1999.

[27] Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch. OpenAD/F: A modular open-source tool for automatic differentiation of Fortran codes. ACM Transactions on Mathematical Software, 34(4):1–36, 2008.

[28] Stefan Görtz Zhong-Hua Han and Ralf Zimmermann. On improving efficiency and accuracy of variable-fidelity surrogate modeling in aero-data for loads context. In CEAS 2009 European Air and Space Conference, Manchester, UK, October 2009.