USE OF ADAPTIVE METAMODELING FOR DESIGN OPTIMIZATION

Jay D. Martin*  
Applied Research Laboratory  
State College, PA, USA 16805

Timothy W. Simpson†  
The Pennsylvania State University  
University Park, PA, USA 16802

ABSTRACT

This paper describes a method to implement an adaptive metamodeling procedure during simulation-based design. Metamodels can be used for design space visualization and design optimization applications when model evaluation performance is critical. The proposed method uses a sequential technique to update a kriging metamodel. This sequential technique will determine the point of the metamodel’s design space with the maximum mean square error and select this as the next point to use to update the metamodel. At each iteration the quality of the metamodel is assessed using a leave-$k$-out cross-validation technique with three different values for $k$. The method is intended to permit continuous updating of the metamodel to investigate the entire design space without concern of finding an optimal value in the metamodel or model.

Keywords: metamodel, kriging, design of experiments, conceptual design

INTRODUCTION

Researchers at the Applied Research Laboratory at Pennsylvania State University are developing an advanced computing environment to support Simulation Based Design (SBD) of undersea vehicles, focusing on their conceptual design.1, 2 The approach involves composing a vehicle from a variety of subsystems, considering multiple technologies for each possible subsystem (see Figure 1), such as a maximum size and speed of operation. Our SBD method will iterate through possible subsystem designs in order to achieve the system-level requirements while satisfying all of the system constraints. These constraints can be characterized as either constraints placed on the subsystem by high-level requirements or constraints that result from the coupling of subsystems into a single system. A typical example of this is the selection of a propulsion system that has sufficient thrust to overcome the drag the vehicle experiences at its desired operating conditions.

---

* Research Assistant, Applied Research Laboratory. Phone: (814) 865-5930. Email: jdm111@psu.edu.
† Assistant Professor, Departments of Mechanical & Nuclear Engineering and Industrial & Manufacturing Engineering. Member AIAA. Corresponding Author. Phone/fax: (814) 863-7136/4745. Email: tws8@psu.edu.
feasible designs instead and then visualize all of the possible designs to understand the tradeoffs in the system design space. The procedure adds significantly to the number of calculations needed. In order to make this a computationally efficient process, significant improvements in the time to determine subsystem performance is required. Consequently, we employ metamodels in our work to achieve this goal.

A metamodel is a “model of a model” that provides an approximation to the physics-based model that is much faster to execute. The creation of metamodels often uses Design of Experiments (DOE) to select a set of points in the design space in order to create an adequate model describing the behavior of the subsystems.

There are many different techniques that have been developed for DOE. The focus in this work is on an automated approach to the Design of Experiments and the creation of the most appropriate metamodels to approximate the physics-based model. As shown in Figure 2, the proposed method is used to a) automatically design experiments on the different subsystems, b) evaluate these experiments, c) fit a metamodel, d) test the fit of the model, e) decide if and where to perform another experiment on the model, and f) update the existing model with the new information.

Using the kriging metamodel, the Mean Square Error (MSE) at any point in the space can be determined. The point in the space with the maximum MSE is taken as the next experimental point. The maximum MSE is significant as it correlates both the lack of knowledge in the input space and the lack of knowledge about the output. The selected point is executed by the physics-based model, and a new metamodel is fit to the data.

The times to calculate the maximum MSE and to refit the kriging metamodel have proven to be significant. Therefore we hope to determine if it is necessary to refit the metamodel at each iteration. If not, a significant saving in computing time will be possible. A review of previous work related to this work follows.

**SEQUENTIAL METAMODELING TECHNIQUES**

Many researchers advocate the use of a sequential metamodeling approach using move limits or trust regions. For instance, the Concurrent SubSpace Optimization procedure uses data generated during concurrent subspace optimization to develop response surface approximations of the design space which form the basis of the subspace coordination procedure. More recently, Perez, et al. present an adaptive experimental strategy for reducing the number of sample points needed to maintain accuracy of second-order response surface approximations during optimization. Similarly, Wang proposes an adaptive response surface methodology that inherits sample points from one iteration to the next to reduce the number of samples need to maintain an accurate approximation. Mathematically rigorous techniques and trust regions are also being combined to develop several other metamodel management frameworks to manage the use of approximation models in optimization. Osio and Amon develop a multi-stage kriging strategy to sequentially update and improve the accuracy of surrogate approximations as additional sample points are obtained. The Hierarchical and Interactive Decision Refinement methodology uses statistical regression and other metamodeling techniques to recursively

![](image)

**Figure 2. Functional Diagram of Method**

This is a continuous process, meant to occur even before a need for the metamodel is recognized, taking advantage of idle computing time to autonomously investigate the design space with minimal user interaction. The method will (1) make a preliminary version of the metamodel available once the initial version has been fitted and (2) continuously update the metamodel with new data as it becomes available.

The work presented here is a preliminary investigation into the implementation of this method. Given a physics-based model, a preliminary set of experiments is executed against it. A kriging metamodel is fit to the data. After each fit of the metamodel, three different tests are completed to assess the quality of the current metamodel. As discussed later in this paper, cross-validation techniques are used rather than calculating the true error in the metamodel, an expensive process.
decompose the design space into sub-regions and fit each region with a separate model during design space refinement. Finally, Schonlau, et al. describe a sequential algorithm to balance local and global searches using approximations during constrained optimization.

Most of these approaches are based on the desire to find the optimal design. Schonlau, et al. and Sasena, et al. have described the algorithm, Efficient Global Optimization (EGO), to select a new experiment location based on a combination of a measurement of the model’s lack of information at a location and the chance that the location may be an extreme point. Cox and John described a similar method that uses the lower confidence bound of the metamodel to drive the selection of the next experimental point. In the work by Perez, et al., a typical augmented Lagrangian optimization technique is employed, but the lack of knowledge of the metamodel is used to determine the step size. Model evaluation is completed once the optimization algorithm has selected a new point and the metamodels are updated with the new point. All of these processes are repeated until some stopping condition is met.

Osio and Amon use maximum entropy when selecting their new experiment points rather than results from optimization. In their approach, a preliminary set of experiments is selected without any knowledge of the outputs; they are chosen using a space filling technique. A metamodel is fit to the data, and an assessment of the next set of points to take that will best improve the capability of the model to predict the response is made. After all of these experiments have been completed, the metamodel is fit to the data once again.

The method we propose is a combination of these methods. We desire to use a continuous process, but we do not want to only look for extreme values of the outputs; they are chosen using a space filling technique. A metamodel is fit to the data, and an assessment of the next set of points to take that will best improve the capability of the model to predict the response is made. After all of these experiments have been completed, the metamodel is fit to the data once again.

Kriging Metamodels

Originating in geostatistics, kriging models have been used successfully by many researchers to approximate deterministic computer simulations. A kriging model is a combination of a polynomial model plus departures of the form:

$$\hat{y}(x) = f(x) + Z(x)$$  \hspace{1cm} (1)$$

where $$\hat{y}(x)$$ is the unknown function of interest, $$f(x)$$ is a known (usually polynomial) function of $$x$$, and $$Z(x)$$ is the realization of a stochastic process with mean zero, variance $$\sigma^2$$, and non-zero covariance. The $$f(x)$$ term in Eqn. (1) provides a “global” model of the design space and is similar to the polynomial model in a response surface. In this work, we take $$f(x)$$ to be a constant term based on previous studies.

While $$f(x)$$ “globally” approximates the design space, $$Z(x)$$ creates “localized” deviations so that the kriging model interpolates the $$n_s$$ sampled data points; however, non-interpolating kriging models can also be created to smooth noisy data. The covariance matrix of $$Z(x)$$ is:

$$\text{Cov}[Z(x'),Z(x'')] = \sigma^2 R([R(x',x')]]$$  \hspace{1cm} (2)$$

In Eq. (2), $$R$$ is the correlation matrix, and $$R(x',x'')$$ is the correlation function between any two of the $$n_s$$ sampled data points $$x'$$ and $$x''$$. $$R$$ is a ($$n_s x n_s$$) symmetric matrix with ones along the diagonal. The correlation function $$R(x',x'')$$ is specified by the user, and a variety of correlation functions exist. In this work, we utilize the popular Gaussian correlation function:

$$R(x',x'') = \exp\left[-\sum_{k=1}^{n_c} \theta_k \|x'_k - x''_k\|^2 \right]$$  \hspace{1cm} (3)$$

where $$n_c$$ is the number of design variables, $$\theta_k$$ are the unknown correlation parameters used to fit the model, and $$x'_k$$ and $$x''_k$$ are the $$k$$th components of sample points $$x'$$ and $$x''$$. In some cases using a single correlation parameter gives sufficiently good results; however, we use a different $$\theta$$ for each design variable to maximize the flexibility of the approximation.

Predicted estimates, $$\hat{y}(x)$$, of the response $$y(x)$$ at untried values of $$x$$ are computed using:

$$\hat{y} = \hat{\beta} + r^T (x)R^{-1}(y - f\hat{\beta})$$  \hspace{1cm} (4)$$

where $$y$$ is the column vector of length $$n_s$$ which contains the sample values of the response, and $$f$$ is a column vector of length $$n_s$$ which is filled with ones when $$f(x)$$ is taken as a constant. The correlation vector, $$r^T(x)$$, between an untried $$x$$ and the sampled data points $$\{x_1, ..., x_{n_s}\}$$ is given by:

$$r^T(x) = [R(x,x_1), R(x,x_2), ..., R(x,x_{n_s})]^T.$$  \hspace{1cm} (5)$$

The estimate for $$\hat{\beta}$$ is given by the generalized least squares estimate of $$\beta$$:

$$\hat{\beta} = (f^T R^{-1} f)^{-1} f^T R^{-1} y$$  \hspace{1cm} (6)$$

American Institute of Aeronautics and Astronautics
and the estimate of the variance, $\sigma^2$, of the sample data, denoted as $y$, from the underlying global model (not the variance in the observed data itself) is given by:

$$\hat{\sigma}^2 = \frac{(y - \hat{f}(\hat{\beta}))^T R^{-1} (y - \hat{f}(\hat{\beta}))}{n_s}$$  \hspace{1cm} (7)

where $f(x)$ is assumed to be the constant $\hat{\beta}$. The maximum likelihood estimates (i.e., “best guesses”) for the $\theta_k$ used to fit the model are found by maximizing:

$$\max_{\theta \in \mathbb{R}^n} \frac{-[n \ln(\hat{\sigma}^2) + \ln |R|]}{2}$$  \hspace{1cm} (8)

where $\hat{\sigma}^2$ and $|R|$ are both functions of $\theta$. While any values for the $\theta_k$ create an interpolating kriging model, the “best” kriging model is found by solving this k-dimensional unconstrained non-linear optimization problem. Currently, we use FindMinimum in Mathematica® Version 4.0.1 running on Windows 2000 to perform this optimization. FindMinimum uses a modification of Powell’s search method in several dimensions. To help bound the optimization problem, the inputs to the model are scaled to be $[0..1]$, and the values of $\theta_k$ are limited to values between 0.01 and 100.

One advantage of using a kriging metamodel over other metamodels is the ease in which we can calculate the Mean Square Error (MSE) of $\hat{y}(x)$. This is given by:

$$\text{MSE}[\hat{y}(x)] = \sigma^2 \left[ 1 - [f^T(x), r^T(x)] \begin{bmatrix} 0 & f^T \end{bmatrix}^{-1} f(x) \right]$$  \hspace{1cm} (9)

The MSE can be used to calculate a confidence interval on the predicted values. In our work, we find the point in the design space with the maximum MSE. This point is taken as the next place to complete an experiment on the model to update or adapt the metamodel to maintain its accuracy. Assessment strategies for validating metamodels during this process are described next.

**METAMODEL ASSESSMENT**

There are many ways to assess metamodel accuracy. A popular measure is the Root Mean Square Error (RMSE) over the design space, but computing RMSE requires sampling the model at a large number of points that have not been used to create the metamodel, which can become prohibitive for a model that is computationally intensive. As a result, alternative approaches such as cross-validation have been developed to reduce the computational expense of metamodel assessment. A popular cross-validation method is a leave-$k$-out strategy. In this strategy, $k$ points are left out of the fitted metamodel, and the resulting error is calculated based on the omitted points. This can result in a large number of possible cases, depending on the choice of $k$ and the sample size. Mitchell and Morris and Osio and Amon describe using this technique for $k=1$, and Meckesheimer, et al. present guidelines for selecting $k$ for different types of metamodels, including kriging. They recommend using $k=0.1n_s$ or $\sqrt{n_s}$ when validating kriging metamodels.

The cross-validation RMSE, $\text{RMSE}_{CV}$, for each metamodel is computed using:

$$\text{RMSE}_{CV} (i) = \sqrt{\frac{1}{k} \sum_{j=1}^{k} (\hat{y}_j - y_j)^2}$$  \hspace{1cm} (10)

and the $\text{RMSE}_{CV}$ values for each metamodel prediction are then averaged together to form the average RMSE.

$$\text{AVE}_{\text{RMSE}_{CV}} = \frac{\sum_{i=1}^{n_s} \text{RMSE}_{CV} (i)}{n_s}$$  \hspace{1cm} (11)

This cross-validation procedure is employed in the experimental set up which is described next to reduce the computational burden of validating the metamodel.

**EXPERIMENTAL SETUP**

This section outlines the steps taken to improve the quality of a metamodel in the iterative process. The process starts with a given physics-based model.

**Preliminary Stage**

1. **Identify the input/output parameters** for the model, defining the maximum and minimum values for the inputs to the model.

2. **Perform a preliminary DOE** over the expected feasible design space of the model. Each possible set of input parameters forms an experiment. A set of 50 possible experiments are generated, each using a Latin Hypercube design, and the design that maximizes the minimum distance (maxi-min) between the points is selected in a manner similar to Mitchell and Morris.

3. **Execute the experiments** against the model and store the results of the model in the database along with the experiments.

4. **Fit a metamodel** to the results using the technique described in the kriging section and store the $\theta$ parameters in a database, making the metamodel available for use.
**Iteration Stage**

The next series of steps are repeated at each iteration to improve the quality of the metamodel.

1. **Load metamodel** from database.
2. **Assess the current metamodel.** Use the leave-$k$-out technique described in the metamodel assessment section. For this experiment, we use $k=1, 0.1n,$ and $\sqrt{n_n}$, saving the results in a database.
3. **Find the maximum MSE** location(s) in the design space. The same optimization routine used to find the MLE is used to calculate the maximum MSE. Note that there are many local maximums in the space. To reasonably ensure that a global maxima is found, the optimization will be seeded with multiple points selected at random.
   a. Select $n_s$ random points in the space bounded by the maximum and minimum values chosen in Step 1 of the preliminary stage.
   b. Calculate the maximum MSE using these random points as seeds for the gradient-based optimization.
   c. Sort the resultant points based on their MSE value.
   d. Evaluate the model at the maximum MSE location. The following steps are used to accomplish this task.
      i. **Check against known infeasible points.** This is accomplished by comparing the point with a list of known infeasible points and determining if it is proximity to a known infeasible point. If point is known to be infeasible, move to the next point in the list.
      ii. **Evaluate the candidate feasible point.** Test the point against a set of feasibility constraints to determine if the point is feasible.
         iii. If feasible, save the point marked as feasible, along with the MSE value in the database.
         iv. If infeasible, save the point marked as infeasible for tests of future candidate points, and continue with the next point in the maximum MSE list.
4. **Fit the metamodel** to the new set of feasible data points. Use the previous values of $\theta_k$ to seed the optimization.
5. **Evaluate stopping criteria.** If the criteria are not met, repeat the process. We currently stop after a predetermined number of iterations. Our current research seeks to identify suitable stopping criteria.

As an alternative to this method, the metamodel is also refit to the data after 10 iterations as a comparison to fitting every iteration.

**APPLICATION**

This method has been applied to a model being used to design a new class of underwater vehicle. The model calculates the performance of a gas generation system. It is characterized with five input and five output parameters. The model is a Steady State Steady Flow analysis of a solid fuel combustion process. It is completed in Mathematica.

**RESULTS**

A baseline case was completed for the gas generation model. An original DOE of 49 points from a maxi-min Latin hypercube was completed, and an initial kriging metamodel was fit to this data.

**Continuous Refitting of Metamodel**

The iteration process was allowed to continue for 56 cycles, and the results are plotted in Figure 3. As seen in the figure, the assessment strategies of $k=0.1n$ and $\sqrt{n_n}$ provide very similar results. The case of $k=1$ tends to underestimate the error found with the other two choices for $k$. The $\sqrt{\text{MSE}}$ is also plotted along with the other values to determine if there is any correlation among any of these values. In Figure 4 the actual error found between the metamodel and the point selected to update the model is displayed. Vertical lines have been used in both of these plots to indicate the correlation between the large prediction errors and their effect on the cross-validation assessments.

In Figure 5 and Figure 6 plots of $\theta_k$ for the first two input parameters are plotted over the iteration history. Vertical lines have been added to highlight the two events shown in Figures 3 and 4.

![Figure 3. Plot of Error Assessment per Iteration for Model 1 Using Continuous Metamodel Updating](image-url)
The error assessment for the model per iteration, when updating every 10 iterations, does not appear differ greatly from the case of refitting the metamodel every iteration. Once again there seems to be a few points with a large error that tend to distort the metamodel.

Refitting Metamodel Every 10 Iterations

In this case, the metamodel was updated only after every 10 points were added to the metamodel. Plots similar to Figure 3 and Figure 4 are given in Figure 7 and Figure 8 for comparison.

The results of refitting the metamodel for this case are given in Table 1.

<table>
<thead>
<tr>
<th>Iteration #</th>
<th>0</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta_1$</td>
<td>2.27</td>
<td>1.57</td>
<td>2.23</td>
<td>6.65</td>
<td>1.39</td>
</tr>
<tr>
<td>$\theta_2$</td>
<td>0.281</td>
<td>0.224</td>
<td>1.45</td>
<td>8.72</td>
<td>0.698</td>
</tr>
</tbody>
</table>

Table 1. Model 1 Values of $\theta_1$ and $\theta_2$ for Refitting Metamodel Every 10 Iterations

DISCUSSION

Given the results found thus far, a few conclusions can be drawn from them. The process of calculating the maximum MSE and MLE of the kriging metamodel parameters is burdensome, especially given the dimensional model considered here. As more points are added, the computation times increase considerably. Typical iteration times started out at 20 minutes per iteration but at the end approached 2 hours per iteration. Much of this time is split between the metamodel fitting process and the actual model calculations. Many of the points were still thrown out as infeasible after a calculation was made. Often this would be done for 10 or more points per iteration.
In the continuous updating case of the model, the vertical lines in the plots represent points that were added to the model that are actually infeasible and therefore should not have been added. The inclusion of these points caused difficulties with the metamodels, causing the increase in the error. Care needs to be taken when selecting points to add to the metamodel.

The continuous updating of the metamodel was clearly advantageous as compared to updating each 10th iteration. The continuous updating of the metamodel decreased the maximum MSE of the model, if not the average RMS error. The approach of periodic updating of the metamodel resulted in little or no improvement to the metrics for metamodel assessment for this example.

Inspecting the MLE values of $\theta_1$ and $\theta_2$ reveals that an asymptotic value is being approached, though the inclusion of the infeasible points disrupt this trend. In the periodic refitting of the metamodel parameters the values seem to follow the trends of the continuous refitting, albeit with larger magnitudes of error.

Another interesting effect of using maximum MSE to select points is it tends to select “corner” points first, given an initial Latin Hypercube design. In a 5 dimensional problem there are 32 corners. The procedure does not select all of them, but it does tend to select most of them first to reduce MSE. The problem with selecting corner points is that most of the infeasible points are located on the boundaries bordering the points. Therefore, using the MSE criteria to select points results in an increased likelihood of selecting infeasible points.

CLOSING REMARKS

These experiments have helped us move ahead on the details of implementing the proposed method for adaptive metamodeling. Many improvements will be added in the future. The first of these is a better method to define and identify infeasible regions in the design space. An infeasible region must be included as part of the maximum MSE calculations to keep it from selecting points in regions that are known to be infeasible. Much time was spent unnecessarily evaluating points that were infeasible.

Additional models need to be used to provide more performance metrics on using metamodel assessment to determine the quality of the metamodel and to create a suitable stopping criterion. A more efficient algorithm is also needed to decrease the computation times experienced. With some of the larger data sets, over an hour was spent per iteration. Additionally, using compiled code rather the interpreted Mathematica code will benefit the computation times required.

ACKNOWLEDGEMENT

The authors gratefully acknowledge support from Dr Kam Ng of ONR 33, Contract No. N00014-00-G-0058.

REFERENCES