Why Not Run the Efficient Global Optimization Algorithm with Multiple Surrogates?

Felipe A. C. Viana, Raphael T. Haftka
University of Florida, Gainesville, FL, USA, 32611

Layne T. Watson
Virginia Polytechnic Institute & State University, Blacksburg, VA, USA, 24061

Surrogate-based optimization has become popular in the design of complex engineering systems. Each optimization cycle consists of analyzing a number of designs, fitting a surrogate, performing optimization based on the surrogate, and finally performing exact simulation at the design obtained by the optimization. Adaptive sampling algorithms that add one point per cycle are readily available in the literature. They use uncertainty estimators to guide the selection of the next sampling point(s). The addition of one point at a time may not be efficient when it is possible to run simulations in parallel. So we propose an algorithm for adding several points per optimization cycle based on the simultaneous use of multiple surrogates. The need for uncertainty estimates usually limits adaptive sampling algorithms to surrogates such as kriging and polynomial response surface because of the lack of uncertainty estimates in the implementation of other surrogates. We import uncertainty estimates from surrogates having such estimates to use with other surrogates such as support vector regression models. The approach was tested on two analytic examples for nine basic surrogates including kriging, radial basis neural networks, linear Shepard and support vector regression. For these examples we compare our approach with traditional sequential optimization based on kriging. We found that our approach was able to deliver better results in a fraction of the optimization cycles needed by the traditional kriging implementation.

Nomenclature

\[ e_{RMS} \] = root mean square error
\[ \mathbf{x} \] = vector of impute variables (point in the design space)
\[ y(\mathbf{x}) \] = actual response at \( \mathbf{x} \)
\[ \hat{y}(\mathbf{x}) \] = surrogate model of \( y(\mathbf{x}) \)
\[ s^2(\mathbf{x}) \] = prediction variance of a kriging model at \( \mathbf{x} \)
\[ E[I(\mathbf{x})] \] = expected improvement at \( \mathbf{x} \)
\[ PRESS_{RMS} \] = square root of the mean prediction sum of errors

I. Introduction

SURROGATE modeling reduces the costs associated with optimization of complex engineering systems\textsuperscript{1-7}. Thirty years ago, polynomial response surfaces\textsuperscript{8,9} were almost exclusively used as surrogates for engineering design. Part of the reason was that fitting them requires only the solution of a system of linear algebraic equations. With the increase in computational power, surrogates that require the solution of an optimization problem such as

* Research Assistant, Department of Mechanical and Aerospace Engineering, fchegury@ufl.edu, Student member.
† Distinguished Professor, Department of Mechanical and Aerospace Engineering, haftka@ufl.edu, Fellow AIAA.
‡ Professor, Departments of Computer Science and Mathematics, ltw@cs.vt.edu.

Copyright © 2010 by Felipe A. C. Viana. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.
kriging, radial basis neural networks, Shepard interpolation, and support vector regression are increasingly gaining popularity. A surrogate-based optimization cycle begins by fitting a surrogate to data from a number of expensive simulations. Modern surrogate-based optimization strategies use both the prediction and the error estimates offered by surrogates to perform adaptive sampling directed by the optimization. For example, the efficient global optimization (EGO) and the sequential kriging optimization (SKO) algorithms use the kriging prediction and prediction variance to seek the point of maximum expected improvement as the next simulation point for the optimization process. The optimization task is repeated for many cycles adding one point at a time until convergence criteria are achieved.

Traditional implementations of EGO-like algorithms add a single simulation point per cycle. However, opportunities for parallel computing and the human effort associated with setting up complex simulations drive complex applications towards running as many simulations as possible per cycle. The fragility of many complex simulations (i.e., they can abort) also encourages a large number of simulations per cycle, which is then less sensitive to a few failed simulations. In addition, in many engineering applications, it may take weeks to complete simulations, and only very few cycles are undertaken. Hence, there is a strong incentive to overcome the limitation of adding a single data point per cycle. The strategy we propose is motivated by reported success in the use of multiple surrogates for optimization. These approaches usually involve: (i) fitting multiple surrogates (e.g., kriging, polynomial response surface, radial basis neural network, and support vector regression models) and picking one based on a figure of merit (e.g., estimators of the root mean square error), and (ii) performing optimization multiple times with multiple surrogates.

We propose a strategy for adding several points per optimization cycle based on the simultaneous use of multiple surrogates. As suggested by Viana and Haftka, we import uncertainty estimates from one surrogate to another to enable running EGO with the surrogates that do not furnish error estimates (such as support vector regression models). Then, we use the pool of surrogates to provide multiple points per cycle for the EGO algorithm. We do not aim to outperform kriging, instead we expect to reduce the number of cycles for convergence and yet deliver comparable results.

The remainder of the paper is organized as follows. Section II presents the background necessary for the development of the proposed approach. Section III introduces our method for running EGO with multiple surrogates. Section IV presents the results of numerical experiments and some discussion, followed by concluding remarks.

## II. Background

### A. Surrogate Modeling

When the response of interest \( y(x) \) is expensive to evaluate, we may approximate it by a cheaper model \( \hat{y}(x) \) based on (i) assumptions on the nature of \( y(x) \), and (ii) on the observed values of \( y(x) \) at a set of \( p \) data points called experimental design. More explicitly,

\[
y(x) = \hat{y}(x) + \varepsilon(x),
\]

where \( x = [x_1, \ldots, x_d]^T \) is a real \( d \)-dimensional vector and \( \varepsilon(x) \) represents both the error of approximation and measurement (random) errors.

The accuracy of a surrogate is measure by the root mean square error, \( e_{RMS} \):

\[
e_{RMS} = \sqrt{\frac{1}{V} \int_V e^2(x) \, dx} = \sqrt{\frac{1}{V} \int_V [\hat{y}(x) - y(x)]^2 \, dx}
\]

Here, \( e_{RMS} \) is computed by Monte-Carlo integration of the error at \( p_{test} = 10,000 \) test points (obtained using a Latin hypercube design with MATLAB function \textit{lhsdesign} set with the “maxmin” option with 10 iterations):
\[ e_{\text{RMS}} = \sqrt{\frac{1}{p_{\text{test}}} \sum_{i=1}^{p_{\text{test}}} e_i^2} = \sqrt{\frac{1}{p_{\text{test}}} \sum_{i=1}^{p_{\text{test}}} \left[ \hat{y}_i - y_i \right]^2}. \]  

Due to the computational cost of estimating \( e_{\text{RMS}} \), cross validation is often used as an alternative for both assessing accuracy and surrogate selection. It is attractive because it does not depend on the statistical assumptions of a particular surrogate technique and it does not require extra test points\(^\text{31}\). Nevertheless, cross validation should be used with caution, since the literature has reported problems such as bias in error estimation\(^\text{32}\). Cross validation is a process of estimating errors by constructing the surrogate without some of the points and calculating the errors at these omitted points. The process proceeds by dividing the set of \( p \) data points into \( k \) subsets. The surrogate is fit to all subsets except one, and error is computed for the omitted subset that was left out. This process is repeated for all subsets to produce a vector of cross-validation errors \( e_{XV} \) (also known as the PRESS vector, where PRESS stands for prediction sum of squares). Figure 1 illustrates the cross-validation errors for a kriging surrogate. The \( e_{\text{RMS}} \) is estimated from \( e_{XV} \):

\[ \text{PRESS}_{\text{RMS}} = \sqrt{\frac{1}{p} e_{XV}^T e_{XV}}. \]  

Figure 1: Cross-validation error \( e_{XV} \) at the second point of the experimental design exemplified by fitting a kriging model (KRG) to \( p = 5 \) data points.

B. Efficient Global Optimization Algorithm

We just give an overview of the efficient global optimization (EGO) algorithm by Jones et al.\(^\text{22}\). EGO starts by constructing a kriging (KRG) model interpolating the initial set of data points. Kriging estimates the value of the unknown function \( y(x) \) as a combination of basis functions \( f_i(x) \) (e.g., a polynomial basis) and departures (representing low and high frequency variation components, respectively) by

\[ \hat{y}(x) = \sum_{i=1}^{m} \beta_i f_i(x) + z(x), \]  

where \( z(x) \) satisfies \( z(x_k) = y(x_k) - \sum_{i=1}^{m} \beta_i f_i(x_k) \) for all sample points \( (x_k) \) and is assumed to be a realization of a stochastic process \( Z(x) \) with mean zero, process variance \( \sigma^2 \), and spatial covariance function given by

\[ \text{cov}(Z(x_i),Z(x_j)) = \sigma^2 R(x_i,x_j), \]  

\[ \sigma^2 = \frac{1}{p} (y - Xb)^T R^{-1} (y - Xb), \]  

American Institute of Aeronautics and Astronautics
where \( R(x_i, x_j) \) is the correlation between \( Z(x_i) \) and \( Z(x_j) \), \( y \) is the value of the actual responses at the sampled points, \( X \) is the Gramian design matrix constructed using the basis functions at the sampled points, \( R \) is the matrix of correlations \( R(x_i, x_j) \) among sample points, and \( b \) is an approximation of the vector of coefficients \( \beta_i \) of Eq. (5). See refs. 10 to 12 for details on how to compute \( b \).

We can estimate the uncertainty in \( \hat{y}(x) \) using the KRG prediction variance (also known as mean squared error of the predictor)

\[
s^2(x) = \sigma^2 \left( 1 + u^T \left( X^T R^{-1} X \right)^{-1} u - r^T X^{-1} r \right),
\]

where \( u = X^T R^{-1} r - f \), \( r \) is the vector of correlations between the point \( x \) and the design points, \( f \) is the vector of basis functions at the point \( x \). For more details about the kriging prediction variance, see refs. 10 to 12.

Figure 2 depicts the concepts presented so far showing both the prediction and the error estimates of kriging. We can see the implications of the kriging statistical assumptions; since the kriging model is an interpolator, the error vanishes at data points.

\[ \text{Figure 2: Kriging model } \hat{y}_{KRG}(x) \text{ of an arbitrary set of five points. The uncertainty (whose amplitude is } s(x) \text{) associated with } \hat{y}_{KRG}(x) \text{ is shown in gray.} \]

After constructing the kriging model, the algorithm iteratively adds points to the data set in an effort to improve upon the present best sample, \( y_{PBS} \). In each cycle, the next point to be sampled is the one that maximizes the expected improvement

\[
E[I(x)] = s(x) \left[ a \Phi(u) + \phi(u) \right],
\]

\[
u = \left[ y_{PBS} - \hat{y}(x) / s(x) \right],
\]

where \( \Phi(\cdot) \) and \( \phi(\cdot) \) are the cumulative density function (CDF) and probability density function (PDF) of a normal distribution, and \( s(x) \) is the prediction standard deviation (here estimated as the square root of the prediction variance).

After adding the new point to the existing data set, the kriging model is updated (usually without the costly optimization of the correlation parameters). Figure 3 illustrates the first cycle of the EGO algorithm. Figure 3-(a) shows the initial kriging model and the corresponding expected improvement. The maximization of \( E[I(x)] \) adds \( x = 0.19 \) to the data set. In the next cycle, EGO uses the updated kriging model shown in Figure 3-(b). We can see that regions of high error estimates push exploration.
Figure 3: Cycle of the efficient global optimization (EGO) algorithm. Figure 3-(a) shows that the maximization of the expected improvement $E[I(x)]$ suggests adding $x = 0.19$ to the data set. Figure 3-(b) illustrates the updated kriging (KRG) model after $x = 0.19$ is added to the data set.

EGO iterates until the stopping criterion is met. Due to high computational cost of actual simulations, it is common to use the maximum number of function evaluations as the stopping criterion. Another alternative is to set a target value for the expected improvement. The original EGO, proposed in ref. 22, is limited to surrogates such as kriging that have error estimates. In the next section we describe our approach for running EGO with multiple surrogates, including surrogates without error estimates.

C. Importation of Error Estimates from One Surrogate to Another

Viana and Haftka\textsuperscript{28} developed the heuristic rationale behind the importation of uncertainty estimates. Here, we briefly illustrate the mechanism with a simple example. Suppose that a kriging (KRG) and a support vector regression (SVR) models are built from five points, as illustrated in Figure 4-(a). The KRG model has a Gaussian correlation function and a constant for the low frequency component (trend function); the SVR model uses a Gaussian kernel function with $\varepsilon = 0$ for the $\varepsilon$-insensitive loss function. As a result, both surrogates are interpolators. Viana and Haftka\textsuperscript{28} proposed combining the predictor of a model with the uncertainty estimate of another model. By comparing Figure 4-(b) and (c), we can see that kriging offers a reasonable error estimate for the support vector regression (in this example).

Figure 4: Importation of uncertainty estimates. Figure 4-(a) shows a kriging (KRG) and a support vector regression (SVR) models (both surrogates act as interpolators). Figure 4-(b) shows the KRG model and the square root of its prediction variance (in gray). Figure 4-(c) shows the SVR model with the square root of the prediction variance borrowed from KRG (in gray). Although not based on the SVR statistical assumptions, the KRG uncertainty estimate offers a reasonable replacement.
III. Implementation of the Efficient Global Optimization Algorithm with Multiple Surrogates

We assume that multiple surrogates are available – with native or imported error estimates. We propose running EGO with multiple surrogates simultaneously. In each optimization cycle, the set of surrogates potentially suggests multiple points. We hope to reduce the number of cycles EGO needs for convergence by taking advantage of parallel computation. In terms of the wall clock time, this approach is advantageous only if running one or multiple simulations takes approximately the same time (and that is why we advocate for parallel computation of the actual function).

For simplicity, we illustrate the approach using only two surrogates, kriging (KRG) and support vector regression (SVR). The prediction variance of the KRG model is imported to the SVR surrogate. Our algorithm iteratively adds two points to the data set that comes from the individual maximization of the expected improvement of both surrogates. After adding the new points to the existing data set, both surrogates are updated. Figure 5 illustrates the first cycle of the EGO algorithm running with multiple surrogates. Figure 5-(a) shows the initial kriging model and the corresponding expected improvement. The maximization of the kriging \( \mathbb{E}[I(x)] \) suggests adding \( x = 0.19 \) to the data set. Figure 5-(b) shows the initial support vector regression model and the corresponding expected improvement. Here, the suggestion is to add \( x = 0.77 \) to the data set. We add both points and in the next cycle, our algorithm uses the updated models shown in Figure 5-(c).

The stopping criterion used here is a limit on the number of cycles, assuming that we are limited by wall clock time.

![Figure 5: Cycle of the efficient global optimization (EGO) algorithm with two surrogates. Figure 5-(a) and Figure 5-(b) shows how we collect two data points for the next cycle (one from each surrogate). Figure 5-(c) illustrates the updated models after \( x = 0.19 \) and \( x = 0.77 \) are added to the data set.](image)

IV. Numerical Experiments

A. Test Set

Table 1 details the different surrogates used during this part of the investigation. The DACE toolbox of Lophaven et al.\(^{33}\), the native neural networks MATLAB toolbox\(^{30}\), and the code developed by Gunn\(^{34}\) were used for kriging, the radial basis neural network, and support vector regression algorithms, respectively. The SURROGATES toolbox of Viana\(^{35}\) was used to run the Shepard (adapted from SHEPPACK\(^{18}\)) algorithm and it was also used for easy manipulation of the surrogates.
Table 1: Information about the set of surrogates used in the surrogate-based optimization study. In the current implementation, just kriging provides uncertainty estimates.

<table>
<thead>
<tr>
<th>Surrogates</th>
<th>Details</th>
<th>Provide uncertainty estimate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>krg</td>
<td>Kriging model: Constant trend function and Gaussian correlation. $\theta_{0i} = 10$, and $10^{-2} \leq \theta_i \leq 200$, $i = 1,2,...,n_y$ were used.</td>
<td>YES</td>
</tr>
<tr>
<td>rbnn</td>
<td>Radial basis neural network: Goal = $(0.05\bar{y})^2$ and Spread = $\frac{1}{3}$.</td>
<td>NO</td>
</tr>
<tr>
<td>shep</td>
<td>Linear Shepard model: Subroutine LSHIP from SHEPPACK.</td>
<td>NO</td>
</tr>
<tr>
<td>svr-grbf-e-full</td>
<td>Support vector regression: “grbf” and “poly” indicate the kernel functions (Gaussian and second order polynomial respectively).</td>
<td>NO</td>
</tr>
<tr>
<td>svr-grbf-e-short</td>
<td>“e” and “q” indicate the loss functions (“e” for $\varepsilon$-insensitive and “q” for quadratic).</td>
<td></td>
</tr>
<tr>
<td>svr-grbf-q</td>
<td>“full” and “short” refer to different values for the regularization parameter, $C$, and for the insensitivity, $\varepsilon$. “full” adopts $C = \infty$ and $\varepsilon = 1 \times 10^{-4}$, while “short” uses $\varepsilon = \sigma_y / \sqrt{p}$ and $C = 100 \max \left(\frac{1}{</td>
<td>\bar{y} + 3\sigma_y</td>
</tr>
</tbody>
</table>

As test problems, we employed the two following analytical benchmark problems:

- Sasena function\(^\text{37}\) (two variables, see Figure 6) initially fitted using 12 data points,

$$ y(x) = 2 + 0.01 \left( x_2 - x_1^2 \right)^2 + \left( 1 - x_1 \right)^2 + 2 \left( 2 - x_2 \right)^2 + 7 \sin(0.5x_1) \sin(0.7x_1x_2), \quad 0 \leq x_1 \leq 5, \quad 0 \leq x_2 \leq 5; $$  

- Hartman3 function\(^\text{38}\) (three variables) initially fitted using 20 data points,

$$ y(x) = -\sum_{i=1}^{4} a_i \exp \left( -\sum_{j=1}^{3} B_{ij} (x_j - D_{ij})^2 \right), \quad 0 \leq x_j \leq 1, \quad j = 1, 2, 3, $$

$$ a = \begin{bmatrix} 3.0 & 10.0 & 30.0 \\ 0.1 & 10.0 & 35.0 \\ 3.0 & 10.0 & 30.0 \\ 0.1 & 10.0 & 35.0 \end{bmatrix}, \quad B = \begin{bmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}, $$

$$ D = \begin{bmatrix} 0.6068 & 0.7560 & 0.1205 \\ 0.7483 & 0.3774 & 0.2690 \\ 0.6978 & 0.2212 & 0.7614 \end{bmatrix}. $$  

(11)
The experimental designs are created by the MATLAB Latin hypercube function \texttt{lhsdesign}, set with the “maxmin” option with 1,000 iterations. To average out the influence of the initial data set, we repeat the simulations with 100 different Latin hypercube designs.

We let EGO run for six and ten cycles for the Sasena and the Hartman3 functions, respectively. We run our algorithm with two, four, and nine surrogates with one of them being kriging. Given the experimental design, we select the surrogates that will assist kriging based on $\text{RMS}_{\text{PRESS}}$. We pair kriging with the surrogates with smallest $\text{RMS}_{\text{PRESS}}$ in the set. In each cycle, we add at most the same number of points as the number of surrogates (that is two, four or nine points per cycle but avoiding repeated points) until the maximum number of optimization cycles is reached (i.e., six and ten for the Sasena and the Hartman3 functions, respectively). Full details are given in Table 2.

Table 2: Details about the EGO optimization for each of the test problems. Surrogate that will pair with kriging is chosen based on $\text{PRESS}_{\text{RMS}}$ (preference to surrogates with small $\text{RMS}_{\text{PRESS}}$).

<table>
<thead>
<tr>
<th></th>
<th>Sasena (12 initial data points)</th>
<th>Hartman3 (20 initial data points)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of optimization cycles</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Number of function evaluations running EGO with kriging (1 point per cycle)</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Maximum number of function evaluations running EGO with 2 surrogates (at most 2 points per cycle)</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>Maximum number of function evaluations running EGO with 4 surrogates (at most 4 points per cycle)</td>
<td>24</td>
<td>40</td>
</tr>
<tr>
<td>Maximum number of function evaluations running EGO with 9 surrogates (at most 9 points per cycle)</td>
<td>54</td>
<td>90</td>
</tr>
</tbody>
</table>

B. Performance Measure

Given the initial best solution $y_{IBS}$ and the solution found in the optimization task $y^{*}$ we can compute the actual improvement $I$ which measures the performance of the optimization algorithm

$$I = \frac{y_{IBS} - y^{*}}{|y_{IBS}|}.$$  \hspace{1cm} (12)

The actual relative improvement $I$ captures how well the optimization was performed (it measures the improvement upon the initial best solution). $I = 0$ if there is no improvement, and $I > 0$ otherwise ($I$ can be large in case of substantial improvement). However, $I$ does not tell anything about how well the improvement compares with the potential improvement. Hence consider ratio between the actual improvement $I$ and the maximum possible relative improvement.
\[ \eta_l = \frac{I}{I_{\text{max}}} \quad I_{\text{max}} = \frac{y_{\text{IBS}} - y_{\text{optm}}}{|y_{\text{IBS}}|}, \]  

where \( y_{\text{optm}} \) is the value of the function at the global optimum point.

V. Results and Discussion

We first study the estimated accuracy of the set of surrogates shown in Table 1. Figure 7 gives box plots of both \( \text{PRESS}_{\text{RMS}} \) and \( \epsilon_{\text{RMS}} \) for the set of surrogates for both test problems (Appendix A describes box plots). We can see that for both functions there is at least one surrogate that is as good as kriging. For the Sasena function, Figure 7-(a) shows that the support vector regression models with the polynomial kernel (variations of “svr-poly”) may outperform kriging in terms of the \( \epsilon_{\text{RMS}} \). For Hartman3, Figure 7-(b) illustrates that kriging is comparable to the radial basis neural network (“rbnn”). To some extent, the selection of the surrogates based on \( \text{PRESS}_{\text{RMS}} \) is almost the same as that based on \( \epsilon_{\text{RMS}} \). Table 3 shows the how the surrogates rank according to \( \text{PRESS}_{\text{RMS}} \) and \( \epsilon_{\text{RMS}} \) (overall performance). The best two and best four surrogates according to each of these criteria tend to be the same for both Sasena and Hartman3 function.

![Box plots of PRESS_{RMS} and \epsilon_{RMS} for surrogates for the test problems](image)

Figure 7: Box plots of \( \text{PRESS}_{\text{RMS}} \) and \( \epsilon_{\text{RMS}} \) of surrogates for the test problems (over 100 experimental designs). Data obtained with the initial experimental designs (12 and 20 points for Sasena and Hartman3, respectively). In terms of prediction, other surrogates might be just as good as kriging. Appendix A details box plots.
Table 3: Ranking of the surrogates according to median values (over 100 experimental designs) of $P_{RMS}$ and $e_{RMS}$. $P_{RMS}$ satisfactorily ranks the surrogates.

<table>
<thead>
<tr>
<th>Rank</th>
<th>$P_{RMS}$</th>
<th>$e_{RMS}$</th>
<th>$P_{RMS}$</th>
<th>$e_{RMS}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>svr-poly-q</td>
<td>svr-poly-e-full</td>
<td>rbnn</td>
<td>rbnn</td>
</tr>
<tr>
<td>2</td>
<td>svr-poly-e-full</td>
<td>svr-poly-q</td>
<td>krg</td>
<td>krg</td>
</tr>
<tr>
<td>3</td>
<td>svr-poly-e-short</td>
<td>rbnn</td>
<td>svr-poly-e-short</td>
<td>svr-poly-e-short</td>
</tr>
<tr>
<td>4</td>
<td>svr-grbf-q</td>
<td>svr-poly-e-short</td>
<td>svr-grbf-q</td>
<td>svr-grbf-q</td>
</tr>
<tr>
<td>5</td>
<td>svr-grbf-e-short</td>
<td>svr-grbf-e-short</td>
<td>svr-poly-q</td>
<td>svr-poly-q</td>
</tr>
<tr>
<td>6</td>
<td>krg</td>
<td>shep</td>
<td>svr-grbf-e-short</td>
<td>svr-grbf-e-short</td>
</tr>
<tr>
<td>7</td>
<td>shep</td>
<td>krg</td>
<td>shep</td>
<td>shep</td>
</tr>
<tr>
<td>8</td>
<td>rbnn</td>
<td>svr-grbf-q</td>
<td>svr-poly-e-full</td>
<td>svr-poly-e-full</td>
</tr>
</tbody>
</table>

Figure 8 shows the mean of the relative improvement ratio, defined in Eq. (13), out of the 100 experimental designs for the traditional EGO (running only with kriging) and our EGO assisted by multiple surrogates. On average, our approach makes the ratio $I / I_{max}$ closer to 1, which means that it delivers better than the traditional EGO. This is possible because we run several simulations simultaneously after each cycle. For both Sasena and Hartman3 functions, the more points we added per cycle (i.e., ensembles with more surrogates), the better our strategy performed. Because of that, we will focus on the ensemble of nine surrogates from this point on. Additionally, we selected the surrogates that assist kriging based on the $P_{RMS}$-rank. That means that in different experimental designs kriging might be paired with different surrogates. As a consequence, the plots for the ensembles of two and four surrogates of Figure 8 show that the strategy tends to be immune to the surrogate that kriging is paired with.

Figure 8: Mean (over 100 experimental designs) of the relative improvement ratio, defined in Eq. (13), achieved by the EGO algorithm when running with kriging and when assisted by multiple surrogates. EGO with multiple surrogates offers better results than those obtained with traditional EGO.

Next, we illustrate the dispersion of the results of the traditional EGO and our approach. Figure 9 shows the box plots of the relative improvement ratio, defined in Eq. (13), out of the 100 experimental designs for the traditional EGO (running only with kriging) and our EGO assisted by nine surrogates. For both test problems, our approach outperforms kriging by the third cycle. This means that we greatly benefit from the diversity of the surrogates. Because we use parallel evaluation of the actual function, in terms of the wall clock time, our approach only takes a fraction of what the traditional EGO implementation needs.
Figure 9: Box plots of the relative improvement ratio, defined in Eq. (13), over 100 experimental designs. Figure 9-(a) and Figure 9-(b) show the results for the Sasena function with kriging and nine surrogates, respectively. Figure 9-(c) and Figure 9-(d) illustrate the results for Hartman3. Again, our approach achieves better results than the traditional EGO. See Appendix A for details about box plots.

Figure 10: Frequency in which the surrogates of Table 1 appear as the most accurate surrogate (BestRMSE has the smallest $e_{RMSE}$ of the set) out of 100 experimental designs.
VI. Concluding Remarks

We proposed an approach that enables running the efficient global optimization (EGO) algorithm with multiple surrogates simultaneously. The approach is an instance of EGO that runs in parallel by using different surrogates and at the end of each cycle share the candidate solutions. Our algorithm is advantageous if running one or multiple simulations takes about the same time (achieved by parallel computation of the actual function). Two algebraic examples were used to compare the traditional implementation of EGO (running with kriging alone) with our approach (EGO running kriging assisted by a set of surrogates). For these examples we found that the improvements in objective function offered by our approach are substantially greater than those offered by traditional EGO. They are also obtained in a fraction of the optimization cycles. It means that whenever parallel computation of the actual function is available, it pays to run EGO with multiple surrogates. We also saw that, surprisingly, running EGO with the most accurate surrogate might be less efficient than running with only kriging (meaning that global accuracy of the surrogate is not necessarily tightened with best optimization results).

Appendix

A. Box plots

In a box plot, the box is defined by lines at the lower quartile (25%), median (50%), and upper quartile (75%) values. Lines extend from each end of the box and outliers show the coverage of the rest of the data. Lines are plotted at a distance of 1.5 times the inter-quartile range in each direction or the limit of the data, if the limit of the data falls within 1.5 times the interquartile range. Outliers are data with values beyond the ends of the lines by placing a “+” sign for each point.

Acknowledgments

Authors would like to thank the AFOSR and NSF for supporting this work under the grants FA9550-09-1-0153 and CMMI-0856431. We also would like to thank Mr. David Easterling and Mr. Nick Radcliffe for their help in the implementation of the Shepard interpolator.

References