Expert Systems with Applications 42 (2015) 2703–2709

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Effect of error metrics on optimum weight factor selection for ensemble of metamodels

Erdem Acar *

Department of Mechanical Engineering, TOBB University of Economics and Technology, Sogutozu, Ankara 06560, Turkey

ARTICLE INFO

Article history:
Available online 20 November 2014

Keywords:
Ensemble
Error metrics
Metamodeling
Surrogate modeling

ABSTRACT

Optimization of complex engineering systems is performed using computationally expensive high fidelity computer simulations (e.g., finite element analysis). During optimization these high-fidelity simulations are performed many times, so the computational cost becomes excessive. To alleviate the computational burden, metamodels are used to mimic the behavior of these computationally expensive simulations. The prediction capability of metamodeling can be improved by combining various types of models in the form of a weighted average ensemble. The contribution of each models is usually determined such that the root mean square cross validation error (RMSE-CV) is minimized in an aim to minimize the actual root mean square error (RMSE). However, for some applications, other error metrics such as the maximum absolute error (MAXE) may be the error metric of interest. It can be argued, intuitively, that when MAXE is more important than RMSE, the weight factors in ensemble should be determined by minimizing the maximum absolute cross validation error (MAXE-CV). Interestingly, it is found that the ensemble model based on MAXE-CV minimization is less accurate than the ensemble model based on RMSE-CV minimization even if the MAXE is the metric of interest. The reason is found to be that MAXE-CV is mostly related with the geography of the DOE rather than the prediction ability of metamodels.

1. Introduction

Optimization of complex engineering systems is performed using computationally expensive high fidelity computer simulations (e.g., finite element analysis). During optimization these high-fidelity simulations are performed many times, so the computational cost becomes excessive. To alleviate the computational burden, metamodels are used to mimic the behavior of these computationally expensive simulations.

There exists a vast of metamodeling methods developed in literature. The commonly used metamodel types include but not limited to the polynomial response surface approximations, PRS (Box, Hunter, & Hunter, 1978; Myers & Montgomery, 2002), Kriging, KR (Sacks, Welch, Mitchell, & Wynn, 1989; Simpson, Mauery, Korte, & Mistree, 2001), radial basis functions, RBF (Buhmann, 2003; Dyn, Levin, & Rippa, 1986), Gaussian process, GP (MacKay, 1998; Rasmussen & Williams, 2006), neural networks (Bishop, 1995; Smith, 1993), and support vector regression, SVR (Clarke, Griebsch, & Simpson, 2005; Gunn, 1997). A good review of metamodeling methods can be found in Queipo et al. (2005), Wang and Shan (2007) and Forrester and Keane (2009).

Even though most research on metamodels focus on determining the most accurate metamodel for the problem at hand, there exist other studies that focus on merging multiple metamodels into a weighted average ensemble model (Acar & Rais-Rohani, 2009; Acar, 2010; Goel, Haftka, Shyy, & Queipo, 2007; Hamza & Saitou, 2012; Muller & Piche, 2011; Sanchez, Pintos, & Queipo, 2008; Zhou, Ma, Tub, & Feng, 2012). It is observed in these studies that the generated ensemble model has a better prediction ability than the individual metamodels that contribute to the ensemble.

The weight factors in an ensemble are chosen such that an error metric is optimized. The error metric can be a local error metric (Acar, 2010; Sanchez et al., 2008) or a global error metric (Acar & Rais-Rohani, 2009; Goel et al., 2007; Hamza & Saitou, 2012; Muller & Piche, 2011; Zhou et al., 2012). In this paper, we consider global error metrics. The most popular error metric used for selecting the weight factors in an ensemble is the root mean square cross validation error (RMSE-CV). Selecting the weight factors based on RMSE-CV aims at constructing the ensemble such that the mean square error over design space is minimized. However, for some applications, other error metrics may be of interest. For instance, in design of safety critical components, minimization of MAXE may be more important than minimization of RMSE. For these problems, the weight factor selection based on RMSE-CV minimization may not be appropriate and weight factor selection should

* Tel.: +90 312 292 4257; fax: +90 312 292 4091.
E-mail address: acar@etu.edu.tr

http://dx.doi.org/10.1016/j.eswa.2014.11.020
0957-4174/

Nomenclature

$\mathbf{C}$	mean square error matrix
E_i	root mean square cross validation error of the ith metamodel
EN_{MAXE}	ensemble model obtained through MAXE-CV minimization
EN_{RMSE}	ensemble model obtained through RMSE-CV minimization
KR0, KR1	Kriging models obtained by using zeroth-order and first-order trend models, respectively
MAXE	maximum absolute error (computed at a large number of test points)
MAXE-CV	maximum absolute cross validation error
N_M	number of models of the ensemble
PRS2	polynomial response surface of the second-order
RBF	radial basis functions
RMSE	root mean square error (computed at a large number of test points)
RMSE-CV	root mean square cross validation error (computed at training points)
w_i	contribution of the ith model in the ensemble
$\hat{y}_{ens}$	prediction of response obtained from the ensemble model
$\hat{y}_i$	prediction of response obtained from the ith model of the ensemble

be revised. The main objective of this paper is to explore the effects of error metrics on weight factor selection in an ensemble of metamodels.

The paper is organized as follows. The formulation for weighted average ensemble along with determination of the contribution of metamodels is explained in the next section. Section 3 presents the error metrics considered in this study. The mathematical and engineering example problems used in this study is presented in Section 4. Details of ensemble model generation is provided in Section 5. The results obtained from the example problems are discussed in Section 6. Finally, the paper culminates with a list of important conclusions presented in Section 7.

2. Ensemble of metamodels

In metamodel based optimization studies, first many different types of metamodels are constructed, and then the most accurate metamodel is selected to be used further whereas the other constructed metamodels are discarded. There are two major drawbacks of this practice. First, information obtained through building various different metamodels is not fully acknowledged. Second, the accuracies of the constructed metamodels depend on the current training data set, and a different metamodel than the selected one may become the most accurate with a new data set. These shortcomings can be addressed by using ensemble of metamodels.

Suppose that there exists a data set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ that consists of N observations of a D-dimensional variable $\mathbf{x}$, together with the corresponding observations of the response of interest $\{y_1, y_2, \ldots, y_N\}$. The predictions of the response corresponding to different types of stand-alone metamodels can be combined in the form of an ensemble method. The most commonly used ensemble method is the weighted average ensemble, where various different metamodels are combined as

$$\hat{y}_{ens}(\mathbf{x}) = \sum_{i=1}^{N_M} w_i \hat{y}_i(\mathbf{x}) \tag{1}$$

where $\hat{y}_{ens}$ is the response prediction obtained from the ensemble model, N_M is the number of different models in the ensemble, w_i is the contribution (or weight factor) of the ith model in the ensemble and $\hat{y}_i$ is the response prediction obtained from the ith model of the ensemble. To have an unbiased response estimation, the following equation must be satisfied by the weight factors:

$$\sum_{i=1}^{N_M} w_i = 1 \tag{2}$$

The weight factors, w_i, for the metamodels are usually chosen such that the root mean square cross validation error (RMSE-CV) is minimized in an aim to minimize the actual root mean square error (RMSE). However, for some applications, minimization of other error metrics may be more important. In that case, one may intuitively argue that the cross validation versions of these metrics should be minimized while selecting the weight factors. In this paper, the validity of this argument is questioned.

3. Error metrics

Prediction accuracy of metamodels can be measured using different metrics, and these metrics can be used for multiple purposes including (i) assessing the goodness of the approximation to be used for analysis and optimization studies, (ii) identifying the regions of high uncertainty in design space and performing additional sampling (adaptive sampling) at these regions, (iii) selecting the best metamodel among alternative models, and (iv) determining the weight factors of stand-alone metamodels in an ensemble of metamodels (Acar & Rais-Rohani, 2009; Goel et al., 2007). The most commonly used metrics are (i) root mean square error (RMSE), (ii) mean absolute error (MAE), (iii) coefficient of multiple determination (R^2), (iv) maximum absolute error (MAXE). The relative, normalized or adjusted versions of these metrics are also frequently used.

The most popular error metric is the RMSE, which measures the square root of the average value of the squared deviations of the predictions from the observed values. RMSE can be computed from

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_v} (y_i - \hat{y}_i)^2}{n_v}} \tag{3}$$

where n_v is the number of out-of-sample validation points. In design of safety critical components, MAXE may be more important. MAXE measures the absolute value of the maximum deviation of the predictions from the observed values. MAXE can be computed from

$$MAXE = \max_{n_v} |y_i - \hat{y}_i| \tag{4}$$

4. Example problems

Overall nine example problems are considered. The first seven example problems are well-known mathematical benchmark problems used in optimization studies. These are followed by two structural mechanics problems.

4.1. Mathematical benchmark problems

4.1.1. Branin–Hoo function

$$y(x_1, x_2) = \left(x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \tag{5}$$

where $x_1 \in [-5, 10]$, and $x_2 \in [0, 15]$.

4.1.2. Camelback function

$$y(x_1, x_2) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \tag{6}$$

where $x_1 \in [-3, 3]$, and $x_2 \in [-2, 2]$.

4.1.3. Goldstein–Price function

$$\begin{aligned} y(x_1, x_2) = {} & [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 \\ & + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 \\ & + 48x_2 - 36x_1x_2 + 27x_2^2)] \end{aligned} \tag{7}$$

where $x_1, x_2 \in [-2, 2]$.

4.1.4. Hartman function

$$y(\mathbf{x}) = -\sum_{i=1}^{m} c_i \exp\left[-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\right] \tag{8}$$

where $x_i \in [0, 1]$. In this paper, three-variable model (that is, $n = 3$) and six-variable model (that is, $n = 6$) of Hartman function are used. Note that the three-variable model and six-variable model are used as two separate test problems. The value of the function parameter m is taken four, and the other function parameters c_i, a_{ij} and p_{ij} are taken from Dixon and Szegö (1978) and given in Tables 1 and 2.

4.1.5. Extended Rosenbrock function

$$y(\mathbf{x}) = \sum_{i=1}^{m-1}\left[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2\right] \tag{9}$$

where $x_i \in [-5, 10]$. In this paper, nine-variable model of this function is used ($m = 9$ is used).

4.1.6. Dixon–Price function

$$y(\mathbf{x}) = (x_1 - 1)^2 + \sum_{i=2}^{m} i(2x_i^2 - x_{i-1})^2 \tag{10}$$

where $x_i \in [-10, 10]$. In this paper, twelve-variable model of this function is used (i.e., $m = 12$).

Table 1
Parameters used in three-variable Hartman function, $j = 1, 2, 3$.

i	a_{ij}			c_i	p_{ij}		
1	3.0	10.0	30.0	1.0	0.3689	0.1170	0.2673
2	0.1	10.0	35.0	1.2	0.4699	0.4387	0.7470
3	3.0	10.0	30.0	3.0	0.1091	0.8732	0.5547
4	0.1	10.0	35.0	3.2	0.03815	0.5743	0.8828

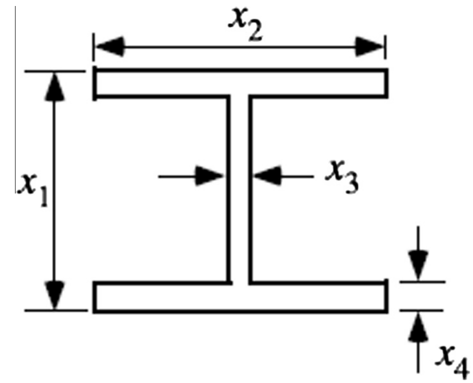


Fig. 1. The cross-section of the four variable I-beam design.

4.2. Structural mechanics problems

4.2.1. Simply supported beam

A simply supported beam is subjected to a concentrated load (see Fig. 1). The beam length is $L = 2$ m and the applied load is $P = 600$ kN. Here the maximum bending stress in the beam is the response of interest and it can be calculated from (Messac & Mullur, 2008)

$$\sigma_{\max} = \frac{\frac{P}{2}\frac{x_1}{2}}{I}, \quad I = \frac{1}{12}\left[x_2x_1^3 - (x_2 - x_3)(x_1 - 2x_4)^3\right] \tag{11}$$

The ranges of the design variables are $0.1\text{ m} \leqslant x_1, x_2 \leqslant 0.8$ m and $0.009\text{ m} \leqslant x_3, x_4 \leqslant 0.05$ m.

4.2.2. Fortini's clutch

An overrunning clutch assembly (see Fig. 2) known as Fortini's clutch is considered. In this problem, the response of interest is the contact angle y. The contact angle can be expressed in terms of the clutch geometry variables via (Lee & Kwak, 2006)

$$y = \arccos\left[\frac{x_1 + 0.5(x_2 + x_3)}{x_4 - 0.5(x_2 + x_3)}\right] \tag{12}$$

The lower and upper bounds of the clutch geometry variables are given in Table 3.

5. Details of ensemble model generation

In this section, the details of generation of the training and test points are provided first. Then, a small discussion of the individual metamodels that contribute to the ensemble is given. Finally, the ensemble model construction procedure is briefly discussed.

5.1. Generation of training and test points

Latin hypercube sampling is used to generate training sets, and Monte Carlo sampling is used to create test sets for all example problems. The number of training points is usually determined based on two approaches: (i) the number of variables multiplied by ten gives the number of training points, (ii) the number of coef-

Table 2
Parameters used in six-variable Hartman function, $j = 1, \ldots, 6$.

i	a_{ij}						c_i	p_{ij}					
1	10.0	3.0	17.0	3.5	1.7	8.0	1.0	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10.0	17.0	0.1	8.0	14.0	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3.0	3.5	1.7	10.0	17.0	8.0	3.0	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	17.0	8.0	0.05	10.0	0.1	14.0	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

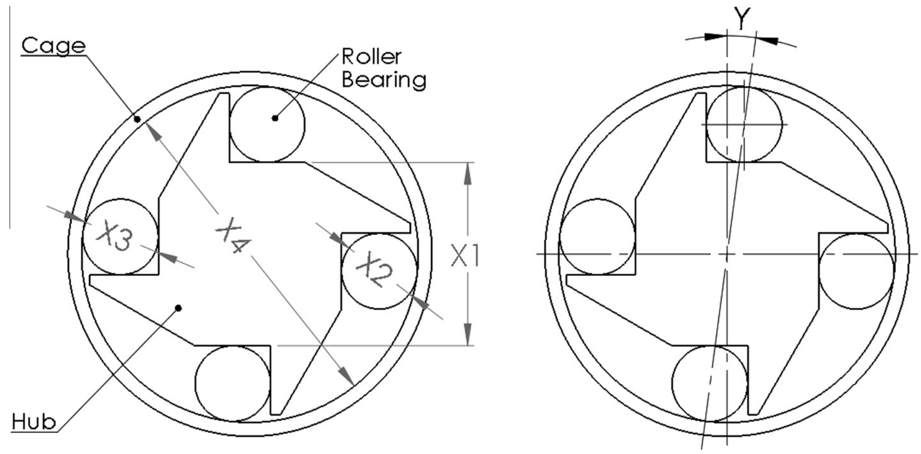


Fig. 2. The clutch assembly.

Table 3
Lower and upper bounds for the clutch geometry variables.

Variable	Lower bound	Upper bound
x_1	54.89	55.69
x_2	22.84	22.88
x_3	22.84	22.88
x_4	101.2	102.0

ficients in a quadratic response surface multiplied by two gives the number of training points. In this study, the number of points corresponding to these approaches is first computed and the larger value is used.

In order to lessen the effect of random sampling, multiple training and test sets are generated. The metamodels in the ensemble are constructed and the weight factors in the ensemble models are computed multiple times corresponding to different training and test sets. In this way, the accuracies of metamodels and the ensemble are evaluated multiple times for different training and test sets and the average accuracies over multiple training and test sets are determined. The number of training and test sets generated for the example problems are given in Table 4. To have a reasonable computational cost, the number of training sets is reduced as the number of variables is increased.

5.2. Individual metamodels

In this study, four individual metamodels contribute to the ensemble model. The first two metamodels are Kriging metamodels with a zeroth-order trend model (KR0), and a first-order trend model (KR1). For these two Kriging metamodels, correlation function is chosen as Gaussian. The Kriging metamodels are generated using the MATLAB Kriging toolbox developed by Lophaven, Nielsen, and Søndergaard (2002). The third individual metamodel is the fully quadratic PRS with all terms included. The last individual metamodel is the RBF metamodel, for which multiquadric formulation (MQ) is used. The constant in the MQ formulation is taken as $c = 1$ as suggested by Wang, Beeson, Wiggs, and Rayasam (2006). The mathematical descriptions of these metamodels can be found in the appendix of Acar and Rais-Rohani (2009).

Table 4
Summary of training and test data used in each problem.

Problem	Number of training and test sets	Number of variables	Number of points in a training set	Number of points in a test set
Branin–Hoo	1000	2	20	1000
Camelback	1000	2	20	1000
Goldstein–Price	1000	2	20	1000
Hartman3	1000	3	30	1000
Four variable beam	1000	4	40	1000
Fortini's clutch	1000	4	40	1000
Hartman6	400	6	60	1000
Extended Rosenbrock	100	9	110	1000
Dixon–Price	100	12	182	1000

5.3. Ensemble model construction

The ensemble models are constructed (that is, the optimum weight factors used in the ensemble are found) by solving the optimization problem stated in Eq. (5). For that purpose, the "fmincon" optimizer function of MATLAB is used. Since "fmincon" is a gradient-based optimizer and the objective function in Eq. (5) is not necessarily convex, a multiple starting point strategy is used, thereby the probability of converging to a global optimum solution is increased.

6. Results and discussion

In this study, the dimensions of the example problems ranged between two and twelve. Since multiple training and test sets are considered, the mean value and the coefficient of variation of the error metrics are computed. Instead of presenting the actual values of the error metrics, the errors are normalized with respect to the most accurate stand-alone metamodel. In this section, the effects of error metrics on weight factor selection are explored first. Then, the capability of cross validation error in representing the actual error is investigated. Finally, the performances of ensembles based on different error metrics are compared.

6.1. Effects of error metrics on weight factor selection

Weight factors of stand-alone metamodels calculated through optimization of different error metrics are listed in Table 5. Note that the weight factors presented in the table are the mean values calculated over 1000 different training and test sets. Fig. 3 shows boxplots for weight factors for the Branin–Hoo problem. The boxplots provide a graphical depiction of how the weight factors vary over the range of training and test sets used. Table 5 and Fig. 3 show that the weight factors obtained from minimization of RMSE-CV are usually different than the ones obtained from minimization of MAXE-CV.

Table 5
Weight factors of stand-alone metamodels computed from optimization of two different error measures.

Measure	W_{PRS}	W_{RBF}	W_{KR0}	W_{KR1}
Branin–Hoo[a]				
RMSE	0.03	0.01	0.80	0.16
MAXE	0.05	0.02	0.70	0.23
Camelback[a]				
RMSE	0.45	0.44	0.10	0.01
MAXE	0.60	0.34	0.05	0.01
Goldstein–Price[a]				
RMSE	0.09	0.51	0.30	0.10
MAXE	0.12	0.57	0.20	0.11
Hartman3[a]				
RMSE	0.06	0.19	0.47	0.28
MAXE	0.11	0.26	0.33	0.30
Four variable beam[a]				
RMSE	0.16	0.60	0.04	0.20
MAXE	0.18	0.70	0.02	0.10
Fortini's clutch[a]				
RMSE	0.17	0.72	0.03	0.08
MAXE	0.16	0.77	0.03	0.04
Hartman6[b]				
RMSE	0.08	0.35	0.39	0.18
MAXE	0.21	0.57	0.14	0.08
Rosenbrock[c]				
RMSE	0.31	0.68	0.00	0.01
MAXE	0.46	0.47	0.02	0.05
Dixon–Price[c]				
RMSE	0.74	0.26	0.00	0.00
MAXE	0.75	0.22	0.02	0.01

[a] Mean over 1000 repetitions.
[b] Mean over 400 repetitions.
[c] Mean over 100 repetitions.

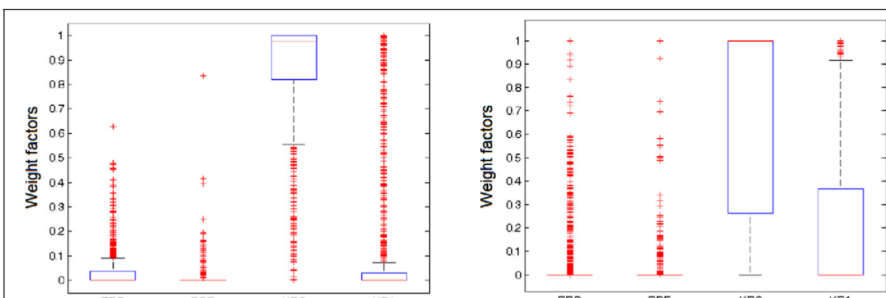
Table 6
Normalized cross validation errors for stand-alone metamodels and ensemble of metamodels.

Measure	PRS	RBF	KR0	KR1	EN_{RMSE}	EN_{MAXE}
Branin–Hoo[a]						
RMSE-CV	3.02	2.23	**1.00**	1.32	**0.97**	1.02
MAXE-CV	2.37	1.96	**1.00**	1.34	0.97	**0.93**
Camelback[a]						
RMSE-CV	1.01	**1.00**	1.21	1.37	**0.93**	0.97
MAXE-CV	**1.00**	1.07	1.29	1.48	0.99	**0.94**
Goldstein–Price[a]						
RMSE-CV	1.20	**1.00**	1.04	1.09	**0.93**	0.97
MAXE-CV	1.14	**1.00**	1.09	1.11	0.98	**0.94**
Hartman3[a]						
RMSE-CV	1.47	1.53	**1.00**	1.09	**0.86**	0.95
MAXE-CV	1.37	1.46	**1.00**	1.00	0.82	**0.73**
Four variable beam[a]						
RMSE-CV	1.43	**1.00**	1.36	1.25	**0.82**	0.89
MAXE-CV	1.36	**1.00**	1.58	1.43	0.90	**0.84**
Fortini's clutch[a]						
RMSE-CV	1.39	**1.00**	3.00	1.70	**0.70**	0.77
MAXE-CV	1.12	**1.00**	2.57	1.53	0.68	**0.62**
Hartman6[b]						
RMSE-CV	1.21	1.08	**1.00**	1.03	**0.87**	1.02
MAXE-CV	1.09	**1.00**	1.15	1.16	1.01	**0.90**
Rosenbrock[c]						
RMSE-CV	1.11	**1.00**	1.96	1.63	**0.96**	1.01
MAXE-CV	1.06	**1.00**	1.95	1.54	0.93	**0.87**
Dixon–Price[c]						
RMSE-CV	**1.00**	1.25	2.32	2.42	**0.96**	1.01
MAXE-CV	**1.00**	1.31	2.25	2.32	0.99	**0.93**

[a] Mean over 1000 repetitions.
[b] Mean over 400 repetitions.
[c] Mean over 100 repetitions.

6.2. Capability of cross validation error to represent the actual error

Normalized errors for stand-alone metamodels and ensemble of metamodels are listed in Table 6 (cross validation errors) and Table 7 (actual errors). In Tables 6 and 7, the ensemble model based on RMSE-CV minimization is denoted by EN_{RMSE}, and the ensemble model based on MAXE-CV minimization is denoted by EN_{MAXE}. The smallest error value in each category is shown in bold for ease of comparison. Comparing the errors in Tables 6 and 7, it is observed for the two error metrics that the stand-alone metamodel



(a) RMSE (b) MAXE

The bottom and top of each box represent the lower and upper quartile values, respectively, with the interior line representing the median. The broken line extending from each end of the box indicates 1.5 times the inter-quartile range, and the data beyond this limit (outliers) are represented by the + symbols.

Fig. 3. Weight factors boxplots for stand-alone metamodels computed from optimization of different error measures for Branin–Hoo problem.

Table 7
Normalized actual errors for stand-alone metamodels and ensemble of metamodels.

Measure	PRS	RBF	KR0	KR1	EN_{RMSE}	EN_{MAXE}
Branin–Hoo[a]						
RMSE	4.86	3.04	**1.00**	1.20	**1.09**	1.22
MAXE	2.71	2.47	**1.00**	1.17	**1.07**	1.14
Camelback[a]						
RMSE	1.17	**1.00**	1.33	1.38	**1.07**	1.13
MAXE	**1.00**	1.08	1.44	1.52	1.08	**1.06**
Goldstein–Price[a]						
RMSE	1.34	**1.00**	1.12	1.15	**1.07**	1.08
MAXE	1.19	**1.00**	1.16	1.18	1.08	**1.07**
Hartman3[a]						
RMSE	1.74	1.63	**1.00**	1.19	**1.03**	1.15
MAXE	1.63	1.85	**1.00**	1.09	**1.03**	1.15
Four variable beam[a]						
RMSE	1.61	**1.00**	1.47	1.36	**1.05**	1.08
MAXE	1.56	**1.00**	1.69	1.57	1.20	**1.16**
Fortini's clutch[a]						
RMSE	1.10	**1.00**	2.10	1.19	**0.90**	0.96
MAXE	**1.00**	1.73	2.07	1.20	**1.53**	1.61
Hartman6[b]						
RMSE	1.19	1.08	**1.00**	1.03	**0.92**	1.07
MAXE	1.05	**1.00**	1.07	1.11	**1.01**	1.03
Rosenbrock[c]						
RMSE	1.10	**1.00**	1.95	1.62	**0.98**	1.03
MAXE	1.03	**1.00**	1.96	1.50	**0.95**	1.00
Dixon–Price[c]						
RMSE	**1.00**	1.24	2.37	2.43	**0.96**	1.02
MAXE	**1.00**	1.35	2.33	2.36	**0.99**	1.06

[a] Mean over 1000 repetitions.
[b] Mean over 400 repetitions.
[c] Mean over 100 repetitions.

Table 8
Normalized maximum absolute errors for the ensembles based on RMSE and MAXE minimization.

Problem	EN_{RMSE}	EN_{MAXE}	%Difference
Branin–Hoo	1.07	1.14	−6.1
Camelback	1.08	1.06	1.9
Goldstein–Price	1.08	1.07	0.9
Hartman3	1.03	1.15	−10.4
Four variable beam	1.20	1.16	3.4
Fortini's clutch	1.53	1.61	−5.0
Hartman6	1.01	1.03	−1.9
Rosenbrock	0.95	1.00	−5.0
Dixon–Price	0.99	1.06	−6.6
Average over all problems	---	---	**−3.2**

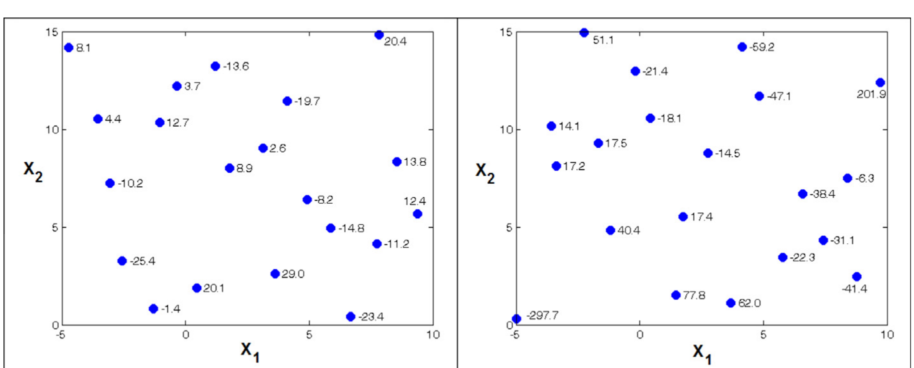
with smallest cross validation error has the smallest actual error, whereas the ensemble model with smallest cross validation error does not necessarily has the smallest actual error.

6.3. Performances of ensembles based on different error metrics

Performances of ensemble models constructed based on optimization of two different error measures are compared. Table 8 shows the normalized maximum absolute errors for the ensemble models EN_{RMSE} and EN_{MAXE}. Interestingly, it is observed that EN_{RMSE} results in 3.2% smaller maximum absolute error than EN_{MAXE}. The reason is found to be that MAXE-CV is mostly related with the geography of the DOE and the holes that are created by removing a point. Fig. 4 shows cross validation errors in two different DOEs for the Branin–Hoo problem. It is observed that cross validation errors are usually small for the training points that have very close neighbor and are large for the training points that are spread better and close to the boundaries. The training points that are spread better, however, leads to better metamodel predictions. Considering also the incapability of MAXE-CV representing the actual MAXE in ensemble models, we can conclude that the selection of weight factors in ensemble based on MAXE-CV is not a good strategy.

7. Conclusions

Metamodels can be combined in the form of a weighted average ensemble to obtain a better prediction capability. The weight factor of a metamodel in the ensemble can be determined by optimizing an error measure. The most popular error measure used in determining the weight factors is the root mean square cross validation error (RMSE-CV). However, for some applications such as design of safety critical components, minimization of the maximum absolute error (MAXE) may be more important than minimization of the root mean square error (RMSE). Hence, if the problem at hand is related to the design of a safety critical component, then the designer will intuitively aim to minimize MAXE-CV, rather than RMSE-CV, in an aim to minimize MAXE. Interestingly, it was found in this paper that the ensemble model constructed through minimization of MAXE-CV was less accurate than the ensemble model constructed through minimization of RMSE-CV even if the MAXE was the error metric of interest. The reason of this interesting finding was found in the paper that the MAXE-CV is mostly related to the geography of the DOE and it is incapable of representing the actual MAXE in the ensemble models. Another important observation of this paper was that the stand-alone metamodel with



(a) DOE #1 (b) DOE #2

Fig. 4. Cross validation errors in two different DOEs for the Branin–Hoo problem.

smallest cross validation error had the smallest actual error, whereas the ensemble model with smallest cross validation error did not necessarily have the smallest actual error. This basically indicated that the cross validation error performs very well in determining the most accurate model amongst alternative stand-alone metamodels, but does not perform that well in determining the optimum weight factors in the ensemble of metamdoels.

The main research limitation for this study was that the number of training and test sets was reduced as the number of variables was increased in order to have a reasonable computational cost. For instance, 1000 different training and test sets were used for the two-variable-problems, whereas 100 different training and test sets were used for the nine-variable-problem (or the twelve-variable-problem). Therefore, the error values reported for the nine-variable-problem (or the twelve-variable-problem) were comparatively less accurate.

In this paper, Latin hypercube sampling design of experiments was used to create the training points. In a future research, the training points can be generated in an adaptive way and the effect of design of experiments on the accuracy of ensemble and weight factor determination can be investigated. In this study, five alternative stand-alone metamodels were used to construct the ensemble of metamodels. In a future research, the number of stand-alone metamodels in the ensemble can be increased and the effect on the accuracy of ensemble and weight factor determination can be explored. In this paper, it was found that the cross validation error performs very well in determining the most accurate model amongst alternative stand-alone metamdoels, but does not perform that well in determining the optimum weight factors in the ensemble of metamdoels. In a future research, options for improving the correlation between the cross validation error and the actual error for the ensemble of metamodels can be investigated.

References

Acar, E. (2010). Various approaches for constructing an ensemble of metamodels using local measures. *Structural and Multidisciplinary Optimization, 42*(6), 879–896.

Acar, E., & Rais-Rohani, M. (2009). Ensemble of metamodels with optimized weight factors. *Structural and Multidisciplinary Optimization, 37*(3), 279–294.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. New York, NY: Oxford University Press Inc.

Box, G. E. P., Hunter, W. G., & Hunter, J. S. (1978). *Statistics for experimenters*. New York, NY: John Wiley & Sons.

Buhmann, M. D. (2003). *Radial basis functions: theory and implementations*. New York, NY: Cambridge University Press.

Clarke, S. M., Griebsch, J. H., & Simpson, T. W. (2005). Analysis of support vector regression for approximation of complex engineering analyses. *Journal of Mechanical Design, 127*(11), 1077–1087.

Dixon, L. C. W., & Szegö, G. P. (1978). *Towards global optimization 2*. Amsterdam, Netherlands: North Holland Publication.

Dyn, N., Levin, D., & Rippa, S. (1986). Numerical procedures for surface fitting of scattered data by radial basis functions. *SIAM Journal of Scientific and Statistical Computing, 7*(2), 639–659.

Forrester, A. I. J., & Keane, A. J. (2009). Recent advances in surrogate based optimization. *Progress in Aerospace Sciences, 45*, 50–79.

Goel, T., Haftka, R. T., Shyy, W., & Queipo, N. V. (2007). Ensemble of surrogates. *Structural and Multidisciplinary Optimization, 33*(3), 199–216.

Gunn, S.R. (1997). Support Vector Machines for classification and regression. Technical Report, Image Speech and Intelligent Systems Research Group, University of Southampton, UK.

Hamza, K., & Saitou, K. (2012). A co-evolutionary approach for design optimization via ensembles of surrogates with application to vehicle crashworthiness. *Journal of Mechanical Design, 134*(1), 011001-1–011001-10.

Lee, S. H., & Kwak, B. M. (2006). Response surface augmented moment method for efficient reliability analysis. *Structural Safety, 28*, 261–272.

Lophaven, S. N., Nielsen, H. B., & Søndergaard, J. (2002). *DACE – A MATLAB Kriging toolbox, informatics and mathematical modeling*. Lyngby, Denmark: Technical University of Denmark.

MacKay, D. J. C. (1998). Introduction to Gaussian processes. In C. M. Bishop (Ed.). *Neural networks and machine learning. NATO ASI series* (Vol. 168, pp. 133–165). Springer: Berlin.

Messac, A., & Mullur, A. A. (2008). A computationally efficient metamodeling approach for expensive multiobjective optimization. *Optimization and Engineering, 9*, 37–67.

Muller, J., & Piche, R. (2011). Mixture surrogate models based on Dempster–Shafer theory for global optimization problems. *Journal of Global Optimization, 51*, 79–104.

Myers, R. H., & Montgomery, D. C. (2002). *Response surface methodology: Process and product optimization using designed experiments*. New York, NY: Wiley.

Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., & Tucker, P. K. (2005). Surrogate-based analysis and optimization. *Progress in Aerospace Sciences, 41*, 1–28.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Cambridge: The MIT Press.

Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science, 4*(4), 409–435.

Sanchez, E., Pintos, S., & Queipo, N. V. (2008). Toward an optimal ensemble of kernel-based approximations with engineering applications. *Structural and Multidisciplinary Optimization, 36*(3), 247–261.

Simpson, T. W., Mauery, T. M., Korte, J. J., & Mistree, F. (2001). Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA Journal, 39*(12), 2233–2241.

Smith, M. (1993). *Neural networks for statistical modeling*. New York: Von Nostrand Reinhold.

Wang, L., Beeson, D., Wiggs, G., & Rayasam, M. (2006). A comparison of metamodeling methods using practical industry requirements. In *Proceedings of the 47th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference*, Newport, RI.

Wang, G. G., & Shan, S. (2007). Review of metamodeling techniques in support of engineering design optimization. *ASME Journal of Mechanical Design, 129*(4), 370–380.

Zhou, X. J., Ma, Y. Z., Tub, Y. L., & Feng, Y. (2012). Ensemble of surrogates for dual response surface modeling in robust parameter design. *Quality and Reliability Engineering International, 29*(2), 173–197.