Struct Multidisc Optim (2018) 57:1329–1343
DOI 10.1007/s00158-017-1815-0

RESEARCH PAPER

Employing partial metamodels for optimization with scarce samples

Di Wu[1] · Kambiz H. Hajikolaei[1] · G. Gary Wang[1]

Received: 2 September 2016 / Revised: 3 August 2017 / Accepted: 14 September 2017 / Published online: 30 September 2017

Abstract To deal with high-dimensional, computationally expensive and black-box optimization (HEB) problems, a Partial Metamodel-based Optimization (PMO) method using Radial Basis Function-High Dimensional Model Representation (RBF-HDMR) along with a moving cut-center strategy is developed. To reduce the exponentially increasing cost of building an accurate metamodel for high dimensional problems, partial RBF-HDMR models of selected design variables are constructed at every iteration in the proposed strategy based on sensitivity analysis. After every iteration, the cut center of RBF-HDMR is moved to the most recent optimum point in order to pursue the optimum. Numerical tests show that the PMO method in general performs better than optimization with a complete RBF-HDMR for high-dimensional problems in terms of both effectiveness and efficiency. To improve the performance of the PMO method, a trust region based PMO (TR-PMO) is developed. When the allowed number of function calls is scarce, TR-PMO has advantages over compared metamodel-based optimization methods. The proposed method was then successfully applied to an airfoil design problem. The use of a partial metamodel for the purpose of optimization shows promises and may lead to development of other novel algorithms.

Keywords High dimension · HDMR · Metamodeling · Sensitivity analysis · Optimization

✉ Kambiz H. Hajikolaei
khajihaj@sfu.ca

1 Product Design and Optimization Laboratory (PDOL), Simon Fraser University, Surrey, BC, Canada

1 Introduction

High dimensionality, high computational cost and being black-box are three main challenges in simulation-based design optimization (Shan and Wang 2010a), and their combination makes a problem very difficult to optimize (Bates et al. 1996; Srivastava et al. 2004). Many science and engineering problems are of high dimension, with different interpretations for what constitutes "high dimension" in different fields. In the design engineering context, if the objective/constraint function is computationally expensive, problems with more than ten variables are considered high dimensional (Shan and Wang 2010b). When the dimensionality increases, the optimization search space grows exponentially, that makes the systematic searching intractable. Computational tools such as Finite Element Analysis (FEA) and Computational Fluid Dynamics (CFD) are widely used for modeling complex engineering problems. The simulation-based models are considered black-box functions because their key information such as functional form, (non)linearity of the function with respect to each variable, and variable correlations are not known to a user. The computational costs of simulation models are different, depending on the complexity of the model and computer power. The authors' team provided a detailed review of techniques solving these so-called High Dimensional, Expensive, Black-box (HEB) problems (Shan and Wang 2010a).

Since gradients of black-box functions are not readily available or often not reliable with numerical analysis, gradient-based global optimizations are commonly not used for HEB problems. Genetic algorithm (GA) (Goldberg 1989), Simulated Annealing (SA) (Kirkpatrick et al. 1983), DIviding RECTangles (DIRECT) (Jones and Law 1993), and Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995) are well-known non-gradient based optimization methods used for optimization of black-box functions. The main difficulty of using these non-gradient optimization methods for HEB problems is the excessive number of needed function calls, which is often not feasible for expensive problems. Various approaches are used for dealing with high dimensionality. Parallel computing, increasing computational power, design space reduction (Wang and Shan 2004), mapping (Somorjai et al. 2004; Rassokhin et al. 2000), visualization (Winer and Bloebaum 2002) and decomposition (Bloebaum et al. 1992) are some of the methods used for dealing with high dimensionality. Among all mentioned methods, decomposition is identified as one of the most promising methods. The main challenge in decomposition-based methods is how to decompose a problem, considering the fact that the objective function is black-box. Analytical target cascading (Kim et al. 2003), hierarchical overlapping coordination (Michelena et al. 1999), simultaneous partitioning and coordination (Allison et al. 2009), and Cooperative Coevolution (CC) methods (Liu et al. 2001; Potter and De Jong 1994; Shi et al. 2005; Yang et al. 2008) are among decomposition-based strategies for large-scale optimization. The number of function calls before reaching an acceptable sub-problems scheme is usually too high for computationally expensive problems, especially when the problem has complicated structure.

Recently, the authors' team introduced the combination of metamodeling and decomposition, as a means to tackle HEB problems (Hajikolaei et al. 2014; Hajikolaei et al. 2016). In the new approach, metamodels are used to "uncover" some of the knowledge of a black-box function and the correlation between the variables, applied for decomposing the original function into smaller sub-problems. Kriging (Cressie 1988), radial basis function (Fang and Horstemeyer 2006), neural network (Papadrakakis et al. 1998), and Multivariate Adaptive Regression Splines (MARS) (Friedman 1991) are examples of well-known metamodeling techniques. However, their weakness is found with high dimensionality, which demands exponentially increasing number of sample points. High Dimensional Model Representation (HDMR), first introduced by Sobol (Sobol′ 1993), is identified as a promising metamodeling approach for HEB problem and is used in previous works (Hajikolaei et al. 2014; Hajikolaei et al. 2016). There are two main types of HDMR (Rabitz et al. 1999): (1) Analysis of variation-HDMR (i.e., ANOVA-HDMR), which was used for sensitivity analysis and identifying key variables (Kaya et al. 2004); and (2) cut-HDMR, which is more computationally efficient than ANOVA-HDMR and is usually used to generate approximation for black-box functions. The main disadvantage of using ANOVA-HDMR is the large number of function evaluations, which come from the Monte Carlo summations that replace integrals in evaluating ANOVA-HDMR terms (Rabitz et al. 1999; Hajikolaei and Wang 2013). Cut-HDMR does not have the disadvantage of Monte Carlo summations because it does not have any integral (Rabitz et al. 1999). But the need of having control on sampling is a major drawback of using cut-HDMR, especially for optimization (Hajikolaei and Wang 2013). Sample points in cut-HDMR should be on specific lines, planes, and hyperplanes based on the cut center, and this type of sampling is called structured sampling (Hajikolaei and Wang 2013), as opposed to random sampling. But during an optimization process, an optimization algorithm does not follow the structure and may sample anywhere in the search space.

In the previous work (Hajikolaei et al. 2014), a type of cut-HDMR called Radial Basis Function-HDMR (RBF-HDMR) (Shan and Wang 2010b; Shan and Wang 2009) is used as metamodeling approach and then sensitivity analysis has been performed on the metamodel to quantify the intensities of the correlations between variables. A brief review of RBF-HDMR is given in the next section. RBF-HDMR is an efficient metamodeling approach for high-dimensional problems, but it is rarely used for optimization purposes. First, the number of sample points used to construct a complete RBF-HDMR model is large for high-dimensional optimization problems. Second, RBF-HDMR requires structured sample points, which makes updating the metamodel difficult. If one uses RBF-HDMR in optimization, often a new model should be built at every iteration. The samples used to construct the previous models cannot be used in constructing the new model, which is a waste of computational resources.

By keeping only the strong correlations and omitting the weak ones, a decomposition scheme was introduced (Hajikolaei et al. 2016). In (Hajikolaei et al. 2016), the decomposition and optimization steps are incorporated together in a loop in a way that the sample points used in the optimization step are reused for decomposition. However, reusing the optimization sample points that were non-uniformly scattered in the search space needed a new metamodeling method. Principle Component Analysis-HDMR (PCA-HDMR) (Hajikolaei and Wang 2013), that was a modified version of ANOVA-HDMR, is used for this purpose. However, PCA-HDMR was not as efficient as RBF-HDMR.

In this work, we tried to combine the pros of the methods proposed in (Hajikolaei et al. 2014) and (Hajikolaei et al. 2016), by developing a partial metamodel based optimization approach in order to push the limit further, i.e., to obtain the best optimal solution with scarce samples. The developed metamodel has a moving cut center and its components are built during the optimization process. In this way, we leverage the advantage of RBF-HDMR, along with the idea of incorporating decomposition and optimization steps together in a loop. The rest of this paper is organized as follows. First, a brief review of basic RBF-HDMR is given. Then the proposed decomposition-based optimization strategy, called Partial Metamodel-based Optimization (PMO), is described with a step-by-step example and description of its properties. Section 4 provides the result of testing the proposed method on different benchmark functions. A trust region strategy is added to the PMO method. Description of the trust region based PMO (TR-PMO) and its comparison with others are in Section 5. Section 6 discusses its application to an airfoil design problem. Finally conclusions are given in Section 7.

2 Review of RBF-HDMR

The general form of HDMR (Rabitz et al. 1999) is:

$$\begin{aligned} f(\boldsymbol{x}) = f_0 &+ \textstyle\sum_{i=1}^{d} f_i(x_i) + \sum_{1\le i<j\le d} f_{ij}(x_i, x_j) \\ &+ \sum_{1\le i<j<k\le d} f_{ijk}(x_i, x_j, x_k) + \cdots \\ &+ \sum_{1\le i_1<\cdots<i_l\le d} f_{i_1 i_2\ldots,i_l}(x_{i_1}, x_{i_2}, \ldots, x_{i_l}) + \cdots \\ &+ f_{12\ldots d}(x_1, x_2, \ldots, x_d) \end{aligned} \quad (1)$$

where, f_0 is a constant representing the zero-order effect on $f(\boldsymbol{x})$; the first order component function, i.e., $f_i(x_i)$, gives the effect of the variable x_i acting independently on the output $f(\boldsymbol{x})$, which can be either linear or nonlinear; $f_{ij}(x_i, x_j)$, the second order component function, describes the correlated contribution of variable x_i and x_j upon $f(\boldsymbol{x})$. As mentioned before, there are two main types of well-known technologies of HDMR: ANOVA-HDMR and cut-HDMR; and RBF-HDMR is of the latter type.

In RBF-HDMR (Shan and Wang 2010b), a Radial Basis Function (RBF) model with a sum of thin plate spline plus a linear polynomial is employed to approximate the component functions. The RBF model is shown as follows (Shan and Wang 2010b):

$$\begin{aligned} &\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \beta_i \left|x - x_i^s\right|^2 \log\left|x - x_i^s\right| + P(x) \\ &\sum_{i=1}^{N} \beta_i p(x) = 0 \\ &P(\boldsymbol{x}) = \boldsymbol{pa} = \left[p_1, p_2 \cdots p_q\right] \left[\alpha_1, \alpha_2, \cdots \alpha_q\right]^T \end{aligned} \quad (2)$$

Where x_i^s is the sampled point of input variables; $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_N]$ and $\boldsymbol{\alpha}$ are parameters to be found. N is the number of sample points. $P(x)$ is a polynomial function and p is the vector of basis of polynomial, chosen as $(1, x_1, x_2, \cdots x_d)$, so $q = d + 1$. The function $\sum_{i=1}^{N} \beta_i p(\boldsymbol{x}) = 0$ is imposed on $\boldsymbol{\beta}$ to avoid the singularity of distance matrix.

The modeling process is described as follows.

(1) Randomly choose a point $\boldsymbol{x}_0$ in the design space as the cut center. Evaluate $f(x)$ at $\boldsymbol{x}_0$ to obtain the zeroth-order component function f_0.

(2) To approximate the first-order component function $f_i(x_i)$, first generate samples in the close neighborhood of the upper bound and lower bound of x_i. Evaluate those two ends and model the component function as $\hat{f}_i(x_i)$ by a one-dimensional RBF for variable x_i using those two points.

(3) Check the linearity of $\hat{f}_i(x_i)$. If the cut center is on the line formed by the approximation model $\hat{f}_i(x_i)$, then consider $\hat{f}_i(x_i)$ as linear and terminate the modeling process for $f_i(x_i)$. Otherwise, rebuild the RBF model $\hat{f}_i(x_i)$ by using the cut center and the two end points. Generate a random point along x_i to test the accuracy of the newly built $\hat{f}_i(x_i)$. If the relative error between the actual value and the approximation one is larger than a given criterion (e.g., 0.01), the test point and all the existing points will be used to rebuild $\hat{f}_i(x_i)$ until sufficient accuracy is obtained.

(4) Check the accuracy of the first-order HDMR model. Form a new point through randomly combining the sample values for each input variable. Then, compare the value predicted by the approximation model with the value obtained from the original expensive function. If these two values are sufficiently close, it indicates that no higher-order components exist in the model, the modeling process terminates. Otherwise, go to Step 5.

(5) Combine the value of x_i and x_j $(j \neq i)$ in the existing samples with the rest of the elements $x_k (k \neq i, j)$ at $\boldsymbol{x}_0$ to create new points in two-dimensional planes. One of the new points is randomly chosen to test the first-order RBF-HDMR model. If the approximation model goes through the new point, x_i and x_j are deemed not correlated and continue to test the next pair of input variables. Otherwise, use the new point as well as the aforementioned evaluated points to construct the second order component function, $\hat{f}_{ij}(x_i, x_j)$. This sampling-remodeling process continues iteratively for all two-variable correlation until convergence. The higher order component functions can be constructed in the same manner of Step 5.

The above process of building a RBF-HDMR model adaptively models a problem and leads to high model accuracy for high-dimensional problems. The construction process is simple. Moreover, RBF-HDMR can significantly reduce the number of expensive function evaluations in approximating high-dimensional problems.

Besides the original RBF-HDMR, there are several modifications of RBF-HDMR in the literature. Cai et al. (2016) proposed an enhanced RBF-HDMR (ERBF-HDMR) that uses enhanced RBF model based on ensemble model to increase the accuracy of HDMR. Other types of metamodel were employed, instead of RBF, to construct the component functions. Huang et al. (2015) and Wang et al. (2011) employed Support Vector Regression (SVR) and Moving

Least Square (MLS) to replace RBF model in RBF-HDMR respectively to obtain more accurate metamodels. The mentioned modifications all focus on how to improve the accuracy of RBF-HDMR. Although original RBF-HDMR is used in this paper to construct the partial metamodel-based optimization, the user may use other variations as well.

3 Description of partial Metamodel-based optimization (PMO)

Although RBF-HDMR is an efficient metamodeling method, the cost of building a complete RBF-HDMR can still be very high for high-dimensional problems. Also RBF-HDMR, as rooted in cut-HDMR, requires structured samples. This is essentially in conflict with the fact that the optimization process may lead the search anywhere in a design space. One approach is to build a new RBF-HDMR in a smaller area, such as a trust region. The cost of doing so is also too high as almost none of the existing points can be inherited for the new model owing to its demand for structured samples.

This work is based on our fundamental belief that optimization can be performed on an imperfect or incomplete metamodel. Instead of building a costly complete metamodel, we propose to use partial metamodels in the optimization process, in order to gain efficiency without sacrificing, or even gain, search quality.

3.1 Algorithm description

With the goal of optimizing HEB problems with reduced number of expensive function calls, in our proposed Partial Metamodel-based Optimization (PMO) method, a partial RBF-HDMR is built at every iteration according to the importance of variables, found by sensitivity analysis. The cut center of RBF-HDMR model is moving after every iteration to the newest optimum point. The flow chart of the PMO method is shown in Fig. 1. To better understand of the procedure, an n-dimensional optimization problem is employed for the ease of description of the proposed method. Steps of the algorithm are described as follows.

Step 1. Construct a first-order RBF-HDMR and use this metamodel for optimization. A random cut center in the design space (i.e., $\boldsymbol{x}_0$) is selected. Then, the first-order RBF-HDMR model (i.e., (3)) is built based on this cut center.

$$\tilde{f}(\boldsymbol{x}) = f_0(\boldsymbol{x}_0) + \sum_{i=1}^{n} f_i(x_i) \tag{3}$$

Then, the first-order HDMR model is optimized to obtain the optimal point $\boldsymbol{x}_{opt}$. The cut center $\boldsymbol{x}_0^{new}$ is moved to this newly found optimum point.

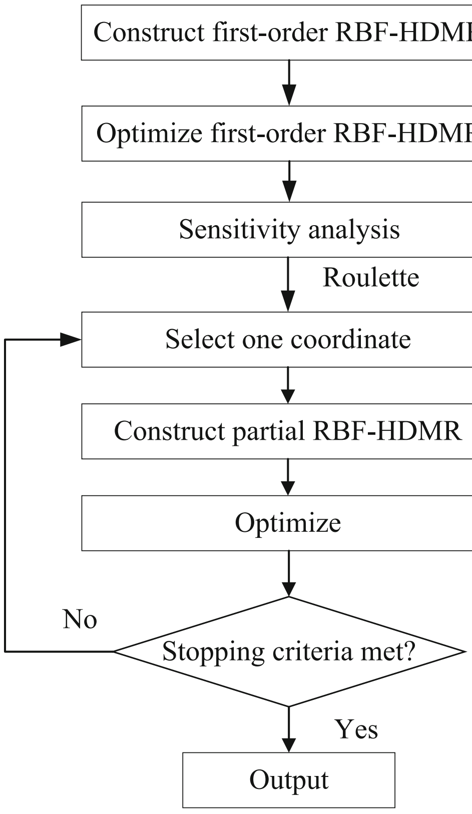

Fig. 1 Flow chart of PMO

Step 2. Select one dimension. First, sensitivity analysis is done on the constructed first order RBF-HDMR model and the normalized sensitivity indices of the variables are used for quantifying the importance of each variable. Next, the sensitivity indices are sorted in descending order to obtain the sensitivity set $\boldsymbol{S} = [s_1, s_2, \ldots, s_n]$, where s_1 is the sensitivity index of the most important variable (highest index) and s_n is the sensitivity index of the least important variable (lowest index). Next, use the sensitivity set $\boldsymbol{S}$ to construct the probability density set $\boldsymbol{G} = [g_1, g_2, \ldots, g_n]$, where $g_i = s_1 + s_2 + \ldots + s_i$, $i = 1, 2, \ldots, n$. Hence, $g_1 = s_1$ and $g_n = 1$. When determining which dimension is selected in the PMO approach, the larger the value of sensitivity index, the higher the chance of being selected for optimization. However, in most cases, the probability densities of the dimensions are close to each other. To ensure the most sensitive dimension is picked up, a speed control factor used in Mode Pursuing Sampling method (Wang et al. 2004) is also used in PMO to adjust the sampling aggressiveness. With the adjustment, the probability density set $\boldsymbol{G}$ is changed to $\hat{\boldsymbol{G}} = \left[g_1^{1/r}, g_2^{1/r}, \ldots, g_n^{1/r}\right]$, where r is the speed control factor. To avoid being trapped into the same solution and balance between exploration and exploitation, a roulette wheel selection operator is used next to randomly select one variable, x_{k_1}, according to the set $\hat{\boldsymbol{G}}$, where the subscript k_1 is the index of the variable, $k_1 \in [1, n]$ and k_1 is an integer. The index is stored in the selected index set $\boldsymbol{K} = [k_1]$.

Step 3. Construct a partial RBF-HDMR of x_{k_1} and use the partial metamodel for optimization. Once the variable is selected, a partial RBF-HDMR model with only one variable is constructed based on the new cut center.

$$\hat{f}(x_{k_1}) = f_0 + \hat{f}_{k_1}(x_{k_1}) \tag{4}$$

Thus, the partial HDMR model is a one-dimensional function of only x_{k_1} with the rest of variables taking the corresponding values of $\boldsymbol{x}_0$. Then, optimize the partial HDMR model to obtain the optimum $x_{k_1}^*$. The cut center is moved to $\boldsymbol{x}_0^{new} = \left(x_1, \ldots, x_{k_1}^*, \ldots, x_n\right)^T$, and the function value f_0 at the new cut center $\boldsymbol{x}_0^{new}$, which is the current optimum value, is calculated.

Step 4. Select the d-th variable. Assume before this step, (d-1) variables have been picked from all the variables ($d \geq 2$), and the selected index set is $\boldsymbol{K} = [k_1, k_2, \ldots, k_{d-1}]$. The new cut center $\boldsymbol{x}_0^{new}$ equals to $\left(x_1, \ldots, x_{k_1}^*, \ldots, x_{k_2}^*, \ldots, x_{k_{d-1}}^*, \ldots, x_n\right)^T$, and the function value at $\boldsymbol{x}_0^{new}$ is selected as the new f_0. After removing the selected variables, the left-out sensitivity set is expressed as $\boldsymbol{S} = \{s_i\}$, $i \notin \boldsymbol{K}$, and the transferred probability density set can be represented as $\hat{\boldsymbol{G}} = \left\{g_i^{1/r}\right\}$, $i \notin \boldsymbol{K}$. The d-th variable is then selected through the roulette wheel selection operator from the rest of un-picked variables. The index of that variable k_d is then added to the index set $\boldsymbol{K}$.

Step 5. Construct a new partial RBF-HDMR model and use the new partial metamodel for optimization. Once the d-th variable is selected, the partial RBF-HDMR model can be constructed as follows.

$$\hat{f}(\boldsymbol{x}) = f_0 + \sum_{i=1}^{d-1} \hat{f}_{k_i}(x_{k_i}) + \hat{f}_{k_d}(x_{k_d}) + \sum_{1 \leq i \leq j \leq d-1} \hat{f}_{k_i k_j}\left(x_{k_i}, x_{k_j}\right) + \sum_{i=1}^{d-1} \hat{f}_{k_i k_d}(x_{k_i}, x_{k_d}) \tag{5}$$

As shown in (5), the samples used to construct components $\hat{f}_{k_i}(x_{k_i})$ ($i = 1, 2, \ldots, d-1$) and $\hat{f}_{k_i k_j}\left(x_{k_i}, x_{k_j}\right)$ ($1 \leq i \leq j \leq d-1$) are all located in the partial design space, $\boldsymbol{x} = (x_{k_1}, x_{k_2}, \ldots, x_{k_{d-1}})^T$, $\boldsymbol{x} \in [\boldsymbol{x}_{lb}, \boldsymbol{x}_{ub}]$, where $\boldsymbol{x}_{lb}$ and $\boldsymbol{x}_{ub}$ are respectively the lower and upper bound of the design space. To reduce the number of function evaluations, function values of most samples used to construct those components can be predicted by the RBF-HDMR model built in the last iteration, which is represented as

$$\hat{f}(\boldsymbol{x}) = f_0 + \sum_{i=1}^{d-1} \hat{f}_{k_i}(x_{k_i}) + \sum_{1 \leq i \leq j \leq d-1} \hat{f}_{k_i k_j}\left(x_{k_i}, x_{k_j}\right) \tag{6}$$

(6) is a function of $\boldsymbol{x} = (x_{k_1}, x_{k_2}, \ldots, x_{k_{d-1}})^T$. Therefore, the function values of the component function $\hat{f}_{k_i}(x_{k_i})$ ($i = 1, 2, \ldots, d-1$) and $\hat{f}_{k_i k_j}\left(x_{k_i}, x_{k_j}\right)$ ($1 \leq i \leq j \leq d-1$) in (5) can be calculated via (6). Thus, to construct the new partial HDMR model, only the points used to construct component functions $\hat{f}_{k_d}(x_{k_d})$ and $\hat{f}_{k_i k_d}(x_{k_i}, x_{k_d})$ ($i = 1, 2, \ldots, d-1$) need to be calculated by calling the actual function. Next, optimize the d-dimensional partial HDMR and obtain the optimum solution $\boldsymbol{x}^* = \left(x_{k_1}^*, x_{k_2}^*, \ldots, .x_{k_d}^*\right)^T$. Combining with other fixed variables, the new cut center moves to $\boldsymbol{x}_0^{new} = \left(x_1, \ldots, x_{k_1}^*, \ldots, x_{k_d}^* \ldots, x_n\right)^T$. The function value f_0 at the new cut center is set to be the current optimum value.

Step 6. Repeat Steps 4 and 5 until reaching the termination criterion. In PMO, the maximum number of iterations is chosen as the termination criterion. If the maximum number of iterations is reached, the process is stopped and output the current cut center $\boldsymbol{x}_0^{new}$ as the optimum solution and the function value f_0 at that cut center as the optimum value; otherwise, go to Step 4 and repeat the procedure. The maximum number of iterations gives the number of variables selected to perform PMO. Selecting more design variables can improve the optimization results. However, selecting more design variables means more second-order component functions need to be constructed and higher sampling costs. In practice, four or five selected design variables give a good balance between optimization effectiveness and efficiency as measured by the number of function calls.

3.2 Example of PMO

A 3-dimensional problem (Adorio and Diliman 2005) shown in (7) is selected as an example to explain the process of the PMO method step-by-step.

$$f(\boldsymbol{x}) = -\sum_{i=1}^{4} \alpha_i \exp\left[-\sum_{j=1}^{3} A_{ij}\left(x_j - P_{ij}\right)^2\right] \qquad x_{1,2,3} = [0, 1] \tag{7}$$

Table 1 Optimization results with numerical benchmark problems

	dim	Actual optimum	PMO		Optimizing complete RBF-HDMR	
			f^*	NFE	f^*	NFE
SUR-T1–14	10	0	21.38	156.6	74.33	161.3
Rosenbrock	10	0	107.08	153.5	187.61	194.5
Trid	10	−210	151.7	150.4	618.11	161.7
F16	16	25.88	25.93	151.0	26.76	397.2
Griewank	20	0	3.19	167.0	6.10	194.2
Ackley	20	0	10.31	236.6	21.07	1547.1
Rastrigin	20	0	196.85	158.0	234.27	111.2
SUR-T1–16	20	0	837.61	231.0	2060.3	430.1
Powell	20	0	596.96	215.4	7222.8	434.6
Perm	20	0	5.69e51	238.0	2.45e52	1625.0

Where, $\alpha = [1, 1.2, 3, 3.2]^T$, $\boldsymbol{A} = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}$,

$$\boldsymbol{P} = 10^{-4} \begin{bmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8808 \end{bmatrix}.$$

The theoretical optimum point is $\boldsymbol{x}^* = [0.114, 0.556, 0.852]^T$; the optimum value is −3.86.

The center of the design space, $\boldsymbol{x}_0 = [0.5, 0.5, 0.5]^T$ is selected as the initial cut center, and the function value at this cut center is evaluated as $f_0 = -0.628$. Then, a first-order RBF-HDMR model is constructed based on this cut center. Genetic Algorithm (GA) from Matlab is employed to optimize the first-order RBF-HDMR, and the optimum point $\boldsymbol{x}^* = [0.132, 0.816, 0.774]^T$ is found with the function value equal to −2.143. The new cut center $\boldsymbol{x}_0^{new}$ moves to the optimum point. Then, sensitivity analysis is performed based on the first-order RBF-HDMR model. The normalized sensitivity index of the variables are 0.590, 0.289 and 0.121, respectively, for x_1, x_2, and x_3. The sensitivity indices are sorted in descending order to obtain the sensitivity indices set $\boldsymbol{S} = [0.590, 0.289, 0.121]$. The probability density set is then obtained as $\boldsymbol{G} = [0.590, 0.879, 1.000]$. After adjusting the speed control factor $r = 2$, the transferred probability density set $\hat{\boldsymbol{G}}$ is [0.768,0.938,1.000]. The roulette wheel selection is performed to determine the variable constructed in the first iteration. Thus, a partial RBF-HDMR, which only contains zeroth-order component function (i.e., f_0) and first-order component function of x_1 (i.e., $f_1(x_1)$), will be constructed as follows:

$$\hat{f}(\boldsymbol{x}) = \hat{f}_0 + \hat{f}_1(x_1) \tag{8}$$

One-dimensional optimization is performed on the partial RBF-HDMR model, and the optimum point is found $x_1^* = 0.131$. The cut center moves to the new point $\boldsymbol{x}_0^{new} = [0.131, 0.816, 0.774]^T$ along x_1. The function value at the new cut center is −2.143. Next, excluding the first variable, the new sensitivity indices value set is $S_2 = [0.705, 0.295]$ and the transferred probability density set $\hat{\boldsymbol{G}}$ is [0.840,1]. After a roulette wheel selection process, variable x_2 is selected as the next constructed variable. Therefore, a partial RBF-HDMR (as shown in (9)) with zeroth-order component function, first-order component function and second-order component of (x_1, x_2) is constructed based on the new cut center.

$$\hat{f}(\boldsymbol{x}) = \hat{f}_0 + \hat{f}_1(x_1) + \hat{f}_2(x_2) + \hat{f}_{1,2}(x_1, x_2) \tag{9}$$

At this time, this two-dimensional partial HDMR model is optimized and the optimum point $\boldsymbol{x}_{12}^* = [0.134, 0.575]^T$ is obtained. Replace the first and second value of the cut center, the optimum point in this iteration is generated, $\boldsymbol{x}_0^{new} = [0.134, 0.575, 0.774]^T$, and the function value at this optimum point is −3.3654, which is much closer to the theoretical optimum point. Finally, only x_3 is left, and the RBF-HDMR with one zeroth-order, three first-order and three second-order components are built. The optimum point $\boldsymbol{x}_{12}^* = [0.132, 0.568, 0.863]^T$ with optimum value −3.847 can be found. In this example, 3 × 5 = 15 points are used to construct the initial first-order RBF-HDMR model; five new points are used to construct the partial HDMR model shown in (8), and 5 + 8 = 13 points are used to construct the model of (9). Adding the initial cut center and two optimum points obtained in two iterations, in total 36 new points are involved to finish the second iteration. On the other hand, 1 + 3 × 5 + 3 × 8 + 1 = 41 sample points are needed to construct and optimize a complete second-order RBF-HDMR model. Using GA to optimize the complete RBF-HDMR with the same initial cut center, the optimum value is −3.013. Hence, the PMO method can find a better optimum with higher efficiency than optimizing on a complete RBF-HDMR model.

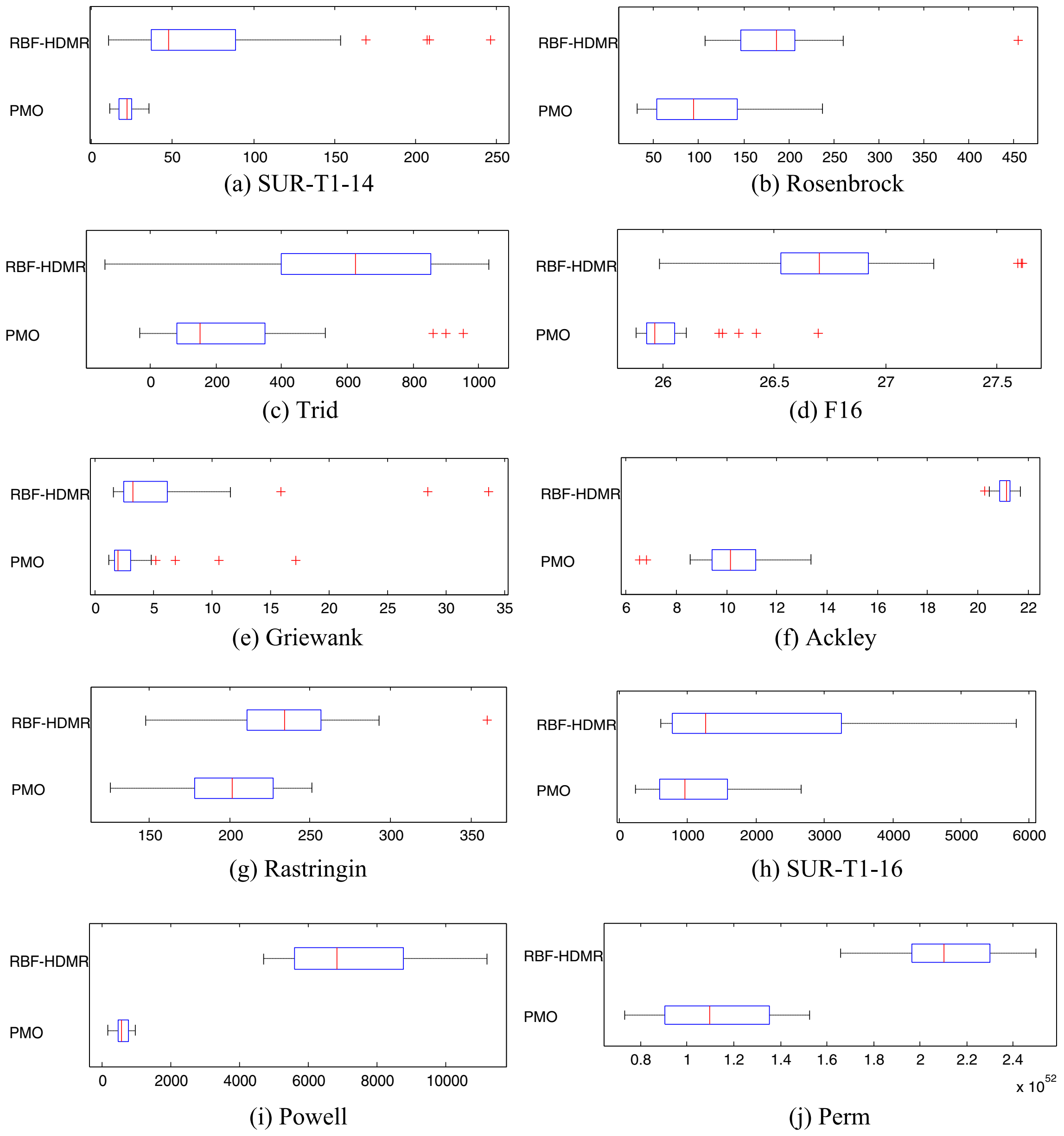

Fig. 2 Box-plots of optimized values

3.3 Properties of PMO

There are two key strategies in PMO. The first one is that a partial HDMR model is used in optimization. Since not all of the variables are involved in the partial HDMR model, the number of sample points used to construct the HDMR model is much less than building the complete model. For instance, for a 10-dimensional problem, assuming that five points are used to construct each first-order component function and eight points are needed to construct each second-order component function, constructing a full second-order HDMR needs $1 + 10 \times 5 + 45 \times 8 = 411$ sample points, where 45 is the number of second-order component functions. On the other hand, assuming a second-order partial HDMR is built with five iterations (i.e., five variables are involved in the partial HDMR with 10 possible second-order component functions), one only needs to generate $5 \times 5 + 10 \times 8 = 105$ expensive sample points during the iterations, adding the initial $1 + 10 \times 5 = 51$ samples for the first order model at the start of PMO, the total number of sample points is only 156, about one third of the cost of the complete model approach.

Another important strategy used in PMO is the moving cut center. In PMO process, the cut center is moving at every iteration to the current optimum point, and a new partial RBF-HDMR is constructed based on the new cut center. That means PMO does not focus on the global accuracy of

Table 2 Optimized results with benchmark functions in different dimensions

	dim	Actual optimum	PMO		Optimizing complete RBF-HDMR	
			f^*	NFE	f^*	NFE
SUR-T1–14	10	0	21.38	156.6	74.33	161.3
	20	0	214.00	171.5	1117.5	434.1
	30	0	863.29	221.7	3971.4	836.3
Griewank	10	0	1.19	93.5	1.28	138.2
	20	0	3.19	167.0	6.10	194.2
	30	0	28.10	204.2	37.03	223.9
Ackley	10	0	5.55	165.9	20.55	407.4
	20	0	10.31	236.6	21.07	1547.1
	30	0	11.24	264.9	21.37	3387.0

the HDMR model but pay more attention on the accuracy around the interesting area (i.e., the area around the current optimum point). With the moving cut center, a HDMR model will be built in a more interesting area at every iteration. Moreover, when a new variable is selected to be added to the partial HDMR, we can use the former partial HDMR model to predict the values at the new samples, rather than invoking the actual expensive function. Although there is a risk in using the former partial HDMR due to the moving cut-center and inaccuracy of the models themselves, such a risk is mitigated by the PMO process, as evidenced from the test results in the next section. It is easy to see that no matter how many iterations PMO takes, the total number of function calls in PMO equals to the number of sample points used to construct the final partial HDMR, plus those used for constructing the first-order HMDR model at the beginning.

In addition, sensitivity analysis is employed in the PMO process to help selecting the most important variables to optimize, rather than randomly selecting variables. The roulette wheel selection process helps to balance the exploration and exploitation phases to avoid being trapped in a local optimum.

4 Testing of PMO

A number of numerical benchmark functions are selected to test the performance of the PMO algorithm. In this test, the proposed PMO algorithm is directly compared to the approach of optimizing a complete RBF-HDMR. A RBF-HDMR model is deemed "complete" if the modeling process is terminated according to the modeling process as described in Section 2. In other words, it means the construction process of RBF-HDMR is completed. In RBF-HDMR construction, before constructing the second-order component functions, the accuracy of the first-order RBF-HDMR is checked. If the first-

Table 3 Dimensions selected in PMO on SUR-T1–14 for five independent runs

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	Optimum
1	Index	0.116	0.114	0.108	0.107	0.102	0.099	0.091	0.090	0.088	0.083	16.87
	Rank	1	2	3	4	5	6	7	8	9	10	
	No. of iterations	1	3	5	4			2				
2	Index	0.117	0.112	0.107	0.107	0.102	0.096	0.096	0.093	0.086	0.083	15.44
	Rank	1	2	3	4	5	6	7	8	9	10	
	No. of iterations	2	1		3	4			5			
3	Index	0.112	0.114	0.112	0.107	0.100	0.098	0.095	0.091	0.088	0.081	14.92
	Rank	2	1	3	4	5	6	7	8	9	10	
	No. of iterations	3	1	2	5					4		
4	Index	0.112	0.117	0.107	0.107	0.102	0.101	0.095	0.087	0.086	0.082	19.43
	Rank	2	1	3	4	5	6	7	8	9	10	
	No. of iterations	4	3	2		1			5			
5	Index	0.116	0.114	0.108	0.105	0.102	0.103	0.095	0.091	0.088	0.081	22.89
	Rank	1	2	3	4	6	5	7	8	9	10	
	No. of iterations	3		2		5		1		4		

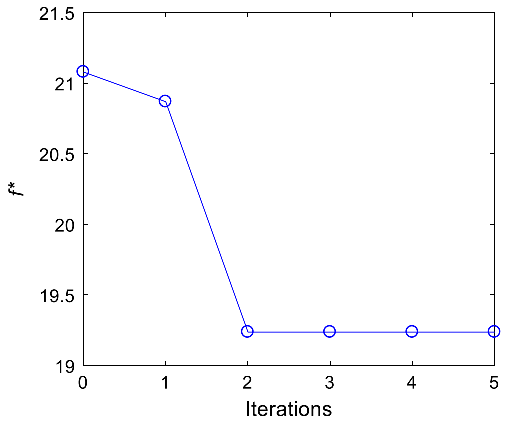

Fig. 3 Convergence plot of PMO in SUR-T1–14 problem

order RBF-HDMR is accurate enough, no second-order component functions are built and the constructing process will be terminated. Additionally, before constructing each second-order component function, whether the two variables, x_i and x_j, are correlated or not is checked. If x_i and x_j are not correlated, the component function $\hat{f}_{ij}(x_i, x_j)$ will not be built in the RBF-HDMR model. Hence, in the complete RBF-HDMR model, not all of the component functions are constructed in some cases. A full RBF-HDMR, however, indicates that all first-order and second-order component functions have been constructed. A complete RBF-HDMR may have skipped modeling of some of the second-order component functions, and thus costs less than a full RBF-HDMR.

SUR-T1–14 (Schittkowski 1987), Rosenbrock (Duan et al. 2009), Trid (Duan et al. 2009), F16 (Adorio and Diliman 2005), Griewank (Duan et al. 2009), Ackley (Duan et al. 2009), Rastrigin (Duan et al. 2009), SUR-T1–16 (Adorio and Diliman 2005), Powell (Adorio and Diliman 2005) and Perm (Duan et al. 2009) problems are chosen as the benchmark problems, which are listed in Appendix. In the test, the maximum number of points used to construct a first-order and second order component function in both PMO algorithm and RBF-HDMR are set to six and eight, respectively. The maximum number of iterations of PMO is set to be five for the test problems. Additionally, GA from MATLAB global optimization toolbox is employed as the optimizer in PMO and RBF-HDMR optimization, and the settings of GA are all as default. Each problem is run 30 times independently. The initial cut centers of both methods are randomly chosen in the 30 runs. The average of the found optimum values (f^*) and number of function evaluations (NFE) are recorded to illustrate the effectiveness and efficiency of PMO. The results are summarized in Table 1 Note that the NEF values in the table are the average results of 30 runs, so the decimal values appear. The box-plots of f^* are shown in Fig. 2.

As shown in Table 1, for all ten problems, the proposed method obtained a smaller optimum value than directly optimizing RBF-HDMR model. Figure 2 gives the box-plots of the optimum values of each problem. It can be found that for almost all the problems, the ranges of f^* in the 30 runs of PMO are smaller than the ranges of optimizing a complete RBF-HDMR, except for Rosenbrock and Ackley. This means PMO is more robust in optimization. From the perspective of the cost (NFEs), PMO clearly costs less than RBF-HDMR except for Rastrigin. Such an advantage is more distinct for higher scale problems. For twenty-dimensional functions such as Ackley and Perm, due to the structure of the function, the cost to construct a second-order metamodel becomes high because it has more second-order pairs. For PMO, because the maximum number of iterations is fixed as five, the maximum number of second-order component functions need to be constructed is 10, which is much smaller than building a full 20-dimensioanl HDMR function (i.e., 190). For Rastrigin, when using the process described in Section 2 to construct RBF-HDMR, the metamodeling process is terminated after construction of all first-order component functions. Therefore, the number of sample points used to construct the complete RBF-HDMR is very small. However, due to the lower accuracy of partial RBF-HDMR, some second-order components are constructed in PMO. Hence in the case that the problem can be accurately described by a first-order RBF-HDMR, the advantage of PMO is not fully realized, even though PMO lead to a better optimum.

Next, three benchmark functions, including SUR-T1–14, Griewank and Ackley, with three different dimensions (10, 20 and 30) are randomly chosen in order to see the performance

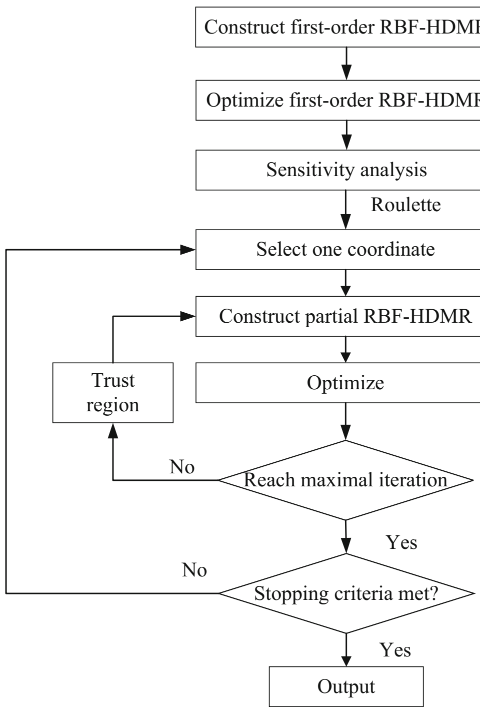

Fig. 4 Flowchart of TR-PMO

Table 4 TRMPS parameter settings

R_{min}	R_{max}	Stall iterations	$k_{reduction}$	$R_{s\,,\,initial}$	$R_{B\,,\,initial}$
0.01	1	5	0.7	0.25	1

change of PMO as the problem dimensionality increases. 30 optimization runs are performed for each problems in each dimension. The results are shown in Table 2.

As shown in Table 2, with the increase of problem dimensions, the advantages of PMO method over RBF-HDMR become larger. For SUR-T1–14 problem, in 10 dimensions, the average f^* of PMO is 21.38, which is 29% of the RBF-HDMR's result. When the dimension is increased to 30, the optimum result of PMO reduces to 21% of the RBF-HDMR's result. The advantage of PMO over higher dimensional problems becomes clearer in terms of NFEs. As mentioned before, most samples are used to construct the second-order component functions in high-dimensional problems. For Ackley function, because every variable pair has strong correlations, with the increase of dimension, the number of second-order functions becomes very large. Thus, the number of sample points increases significantly to build a complete RBF-HDMR. For SUR-T1–14 and Griewank, the variables have mixed weak and strong correlations. Hence, the PMO savings in terms of NFE is milder for SUR-T1–14 and Griewank than for Ackley when the dimensionality arises.

The 10-dimensional SUR-T1–14 problem is chosen to show which dimensions are selected through the roulette wheel selection method at each iteration. The SUR-T1–14 problem is optimized five times with PMO and the data are listed in Table 3. The index is the sensitivity value of each dimension and number of iterations represents at which iteration the variable is selected.

As shown in Table 3, dimensions x_1 to x_5, which have larger sensitivity values, are more likely to be picked up in the five independent runs. The selection, however, does show its stochastic nature. Different selection schemes lead to different optimum solution with variations for the test problem.

Figure 3 illustrates the current optimal value obtained from PMO in seven iterations for the SUT-T1–14 problem. As shown in Fig. 3, from the third to the fifth iteration, the optimization results do not improve.

Table 5 OMID parameter settings

N_{Init}	n_{comp}	n_{basis}	N_{as}
10 × dim	2	2	5 × dim

5 Trust region based PMO

The performance of PMO can be further improved by applying different strategies when optimizing each partial model. Trust region is often used as a strategy to achieve global convergence and balance the exploration and exploitation phases. In this section, a simple trust region strategy is added when optimizing the partial model at each iteration to generate a higher performance version of PMO.

The trust region strategy follows the description of reference (Alexandrov et al. 1998). The approximation accuracy ratio ra_t at the t-th iteration can be calculated via the following equation,

$$ra_t = \frac{f\left(\boldsymbol{x}_{0,t}\right) - f\left(\boldsymbol{x}_t^*\right)}{\hat{f}\left(\boldsymbol{x}_{0,t}\right) - \hat{f}\left(\boldsymbol{x}_t^*\right)} \tag{10}$$

Where, $\boldsymbol{x}_{0\,,\,t}$ is the center of the design space, $\boldsymbol{x}^*$ is the optimal point, $\hat{f}\left(\boldsymbol{x}_{0,t}\right)$ and $\hat{f}\left(\boldsymbol{x}_t^*\right)$ are the responses of the approximate model at $\boldsymbol{x}_{0\,,\,t}$ and $\boldsymbol{x}_t^*$, respectively. ra_t gives the accuracy of the current metamodel, and the value of ra_t determines the shrinkage or enlargement of the design space. The new size (L_{t+1}) of the trust region is defined as follows.

$$\delta_{t+1} = \begin{cases} \max(c_0 L_t, L_{min}) & ra_t < 0 \\ \max(c_1 L_t, L_{min}) & 0 \le ra_t < \tau_1 \\ L_t & \tau_1 \le ra_t < \tau_2 \\ \min(c_2 L_t, L_{max}) & ra_t \ge \tau_2 \end{cases} \tag{11}$$

Where, τ_1 and τ_2 are two positive constants to judge the accuracy of the metamodel and $\tau_1 < \tau_2 < 1$, c_0, c_1, and c_2 are positive constant ratios to shrink or enlarge the trust region where $c_0 \le c_1 < 1 < c_2$, L_t is the size of the current trust region, and L_{min} and L_{max} are the minimal size and maximal size of trust region. In this paper, the values of the parameters are set as $\tau_1 = 0.25$, $\tau_2 = 0.75$, $c_0 = 0.25$, $c_1 = 0.5$, and $c_2 = 2$. Define $L_{min} = 0.01 L_{max}$ and L_{max} is set to be the size of the original design space.

For the center of the trust region, if $ra_t < 0$, it means that the objective function value of current optimum ($\boldsymbol{x}_t^*$) is worse than the value of the center ($\boldsymbol{x}_{0\,,\,t}$). Thus, the center will not move, i.e., $\boldsymbol{x}_{0\,,\,t+1} = \boldsymbol{x}_{0\,,\,t}$. Otherwise, the center moves to the current optimum, i.e., $\boldsymbol{x}_{0,t+1} = \boldsymbol{x}_t^*$.

The flowchart of trust region based PMO (TR-PMO) algorithm is shown in Fig. 4, which is similar to the flowchart of PMO as shown in Fig. 1 with the insertion of the Trust Region box in the flow. In this algorithm, the trust region strategy is used to find a better solution in each partial metamodel. Assume that at the d-th iteration, coordinates k_1, k_2, …, and k_d are selected to construct the partial RBF-HDMR model and the current cut center and optimal solution are $\boldsymbol{x}_0$ and $\boldsymbol{x}^*$, respectively. The steps as related to the trust region are introduced as follows.

Table 6 Optimization results of using TR-PMO, TRMPS OMID and PMO

	dim	Actual optimum	NFE	f^*			NFE	f^*
				TR-PMO	TRMPS	OMID	PMO	
SUR-T1–14	10	0	276	**19.64**	20.11	91.23	279	20.01
Rosenbrock	10	0	157	**63.55**	273.23	2587.4	150	108.98
Trid	10	−210	165	**100.67**	331.7	3227.0	154	427.38
F16	16	25.88	276	26.98	**25.92**	30.97	233	27.08
Griewank	20	0	160	**2.11**	5.67	39.64	169	15.72
Ackley	20	0	287	**9.68**	15.83	17.38	280	12.29
Rastrigin	20	0	189	159.9	**124.39**	214.02	215	186.31
SUR-T1–16	20	0	255	935.87	**86.45**	2178.6	275	1079.0
Powell	20	0	288	157.05	**71.93**	2827.1	281	661.00
Perm	20	0	296	1.92e51	2.36e49	**1.39e49**	298	5.53e52

Boldfaced numbers are the best for each problem.

Step A. After optimizing on a partial RBF-HDMR, check if reaching the maximal iteration number. If the maximal iteration number is reached, the trust region loop terminates and the process goes to Step 6 as described in Section 3.1 to select a new coordinate, otherwise, continue to Step B.

Step B. Calculate the appropriate accuracy ratio ra_t via (10). The partial HDMR model of d variables is used to calculate the value $\hat{f}(\boldsymbol{x}_0)$ and $\hat{f}(\boldsymbol{x}^*)$.

Step C. Determine the new trust region. If $ra_t < 0$, the cut center remains; otherwise, the cut center will move to the current optimal point. Then, (11) is employed to shrink or enlarge the trust region. Note that only the upper and lower bounds of the selected variables are modified and the regions of other variables remain unchanged.

Step D. Generate a certain number of random sample points in the trust region (e.g., five). These new samples are used to update the partial model. If the cut center does not move, the sample points used to construct the previous partial model can be inherited. If the cut center moves, only these new samples in the updated trust region are used to build a metamodel for optimization. Then go back to Step A.

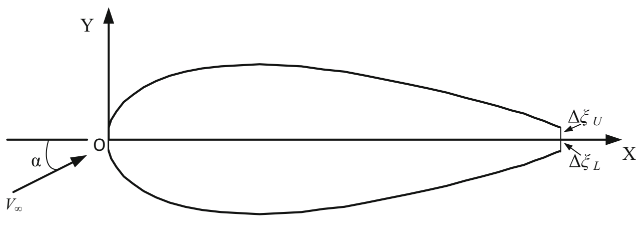

Fig. 5 Airfoil design problem

To benchmark the performance of TR-PMO, two effective optimization strategies developed for HEB problems are chosen for comparison, i.e., Trust Region based Mode Pursuing Sampling method (TRMPS) (Cheng et al. 2015) and Optimization on Metamodeling-supported Iterative Decomposition (OMID) (Haji Hajikolaei et al. 2015). PMO also participates in the comparison.

The same ten numerical problems used in Section 4 are used to perform the comparison and each problem is repeated 10 times. The numbers of points used to construct the first- and second-order component functions are five and eight respectively. In general, the NFE of TR-PMO is more likely to be larger than that of PMO when the number of selected variables and the number of sample points used to construct component functions are the same in both TR-PMO and PMO, because more points are generated when performing the trust region strategy. In this testing, the number of sample points used to construct first- and second-order components is increased for PMO to make a fair comparison with other methods with similar NFE. Note that PMO cannot terminate at a certain NFE, so the NFE is controlled to be as close as possible to the NFE used in TR-PMO. The average NFE of PMO is also listed in Table 6. Additionally, the number of selected variables is set to be four for both PMO and TR-PMO. The setting of trust region in TR-PMO is introduced earlier in this section. The parameters in TRMPS and OMID are set the same as in Refs. (Cheng et al. 2015) and (Haji Hajikolaei et al. 2015), as shown in Tables 4 and 5. The maximal number of function evaluations of TRMPS and TR-PMO is set to be the average number of function calls used by TR-PMO in each benchmark. The results are shown in Table 6.

In Table 6, the NFE data in the fourth column is the number of function evaluations used in TR-PMO, TRMPS and OMID, while the data in the eighth column is the number of function

Table 7 Parameters of NACA0012

Parameter	Au_0	Au_1	Au_2	Au_3	Au_4	Au_5
Initial value	0.1703	0.1602	0.1436	0.1664	0.1105	0.1794
parameter	Al_0	Al_1	Al_2	Al_3	Al_4	Al_5
Initial value	−0.1703	−0.1602	−0.1436	−0.1664	−0.1105	−0.1794

evaluations used in PMO. NFE's of PMO and TR-PMO only show that they are at a similar level and should not be used for efficiency comparison as NFE for PMO is artificially increased for better comparison. As shown in Table 6, with similar NFE, TR-PMO outperforms PMO in optimizing all the ten benchmark functions. In PMO, the partial HDMR model is a static metamodel at each iteration, while the trust region strategy can actively add points to find better results for each partial model optimization.

It also can be found that in case of SUR-T1–14, Rosenbrock, TRID, Griewank and Ackley functions, TR-PMO obtained better results than TRMPS while in other problems the results of TR-PMO are worse (boldfaced numbers are the best for each problem). On the other hand, compared with OMID, TR-PMO performs better for almost all benchmark problems except for Perm function. TRMPS and OMID are two effective optimization strategies for high dimensional problems but often need much more function calls to reach a good optimal solution. When the allowed number of function calls is limited to a few hundreds, TR-PMO method has comparable or better performances than TRMPS and OMID. This is because for TRMPS and OMID, samples are generated in the entire design space. To adequately cover a high-dimensional space, a comparatively larger amount of samples are need for both methods. For TR-PMO, in 10-dimensional problems, selecting four variables seems to be enough for obtaining acceptable results with scarce samples. However, in 20-dimensional problems, four variables is only 1/5 of the total variables, which limits the optimization performance of TR-PMO. Also, it is noticed that the range of objective function values in SUR-T1–16, Powell and Perm problems are too large and a small change in the design variables causes significant changes in function values. It is likely the reason that TR-PMO did not perform as well as TRMPS for these cases.

In summary, when the number of samples is limited, the advantages of using a partial metamodel emerge and TR-PMO shows better or comparable results as other methods.

6 Application to airfoil design

After testing with benchmark functions, both PMO and TR-PMO are applied to an airfoil design problem as shown in Fig. 5. The symbol α is the attack angle and V_∞ is the flow velocity. Class function/shape function airfoil transformation representation tool (CST) (Kulfan and Bussoletti 2006), as shown in (12), is used to model the geometry of the airfoil.

$$\begin{cases} \xi_U(\psi) = \psi^{0.5}(1-\psi)^{1.0}\sum_{i=0}^{5} Au_i \dfrac{5!}{i!(5-i)!}\psi^i(1-\psi)^{5-i} + \psi\Delta\xi_U \\ \xi_L(\psi) = \psi^{0.5}(1-\psi)^{1.0}\sum_{i=0}^{5} Al_i \dfrac{5!}{i!(5-i)!}\psi^i(1-\psi)^{5-i} + \psi\Delta\xi_L \end{cases} \tag{12}$$

Where, ξ_U and ξ_L are the geometry function of the upper and lower surfaces of the airfoil, respectively; ψ is the non-dimensional horizontal coordinate; $\Delta\xi_U$ and $\Delta\xi_L$ are the thickness ratios of the trailing edge of upper and lower surfaces, which can be represented by the distance between the upper (or lower) surface and the x-axis at trailing edge; Au and Al are the coefficients of the shape function. In this example, the airfoil is a closed curve, so the trailing edge thicknesses of upper and lower surfaces are zero, i.e., $\Delta\xi_U = 0$ and $\Delta\xi_L = 0$. In this parametric function, six upper surface coefficients and six lower surface coefficients are selected as the design variables. The NACA0012 airfoil is selected as the baseline airfoil in this design problem, and the coefficients of NACA0012 are shown in Table 7. The upper and lower boundaries of the design variables are 130% and 70% of the baseline.

The objective of the airfoil design problem is to maximize the lift-to-drag ratio (*L/D*). The constraint of the problem is that the maximum thickness (t_{max}) of the new airfoil is not less than the baseline value ($t_{max}^{baseline}$). Thus, the optimization model is as shown in (13).

$$\begin{aligned} &\min \quad -\frac{L}{D} \\ &s.t. \quad t_{max}^{baseline} - t_{max} \le 0 \\ &0.7x_i^{baseline} \le x_i \le 1.3x_i^{baseline} \\ &\quad i = 1, 2, \dots, 12 \end{aligned} \tag{13}$$

Table 8 Optimization results with airfoil design problem

	dim	TR-PMO		PMO		Optimizing complete RBF-HDMR	
		f^*	NFE	f^*	NFE	f^*	NFE
Airfoil design	12	−117.94	295.1	−106.87	175.5	−51.06	584.5

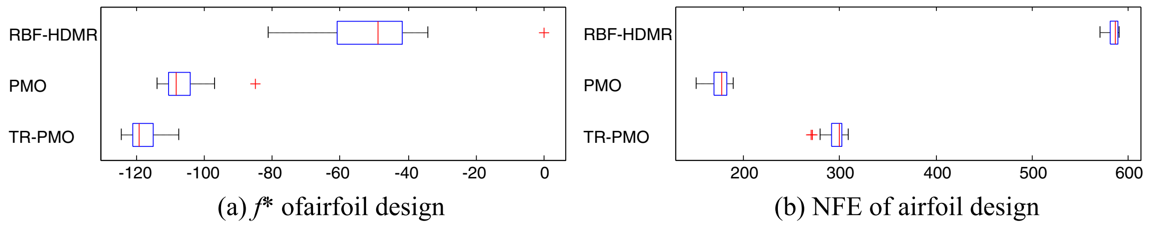

Fig. 6 Optimization results on the airfoil design problem. (**a**) *f** of airfoil design (b) NFE of airfoil design

Software XFOIL (2013) is employed to calculate the value of *L/D*. In this test, the Mach number of the flow is 0.5, and the Reynolds number is 5,000,000. The results obtained by optimizing RBF-HDMR model directly are also listed for comparison. 30 independent runs are carried out for each method. The settings of all methods are the same as in the numerical tests. The average optimization results over 30 runs are shown in Table 8. It should be noted that the constraint is considered cheap.

As shown in Table 8, the average NFE of PMO is only 30% of that used to construct a complete RBF-HDMR model, but the objective is more than twice better than the optimum value obtained from optimizing RBF-HDMR model. TR-PMO achieves a better optimum than PMO with more NFEs, which is still 50% of the cost of the RBF-HDMR approach. Figure 6 illustrates the box-plots of *f** and NFE of the three methods. It can be found that PMO and TR-PMO have similar robustness, which is better than optimizing RBF-HMDR. The variations of NFEs for the three approaches are very similar.

7 Conclusion

This work proposed a Partial Metamodel-based Optimization (PMO) algorithm to deal with High-dimensional, Expensive, and Black-box (HEB) problems. Instead of building the complete RBF-HDMR model, a series of partial RBF-HDMR models are constructed to reduce the number of function evaluations in high-dimensional optimization problems. To balance the exploration and exploitation phases of the method, a roulette wheel selection process is employed to select variables to construct the partial HDMR model, according to the sensitivity index values of all variables. The cut center of the partial HDMR model at each iteration moves to the newly found optimum point to achieve higher optimization performance. The HDMR model in previous iterations is used to predict the function values used in constructing a new partial RBF-HDMR model. The proposed method is compared with optimizing a complete RBF-HDMR using ten numerical benchmark functions. PMO obtained better optimum solutions than optimizing a complete RBF-HDMR, using less function calls in almost all the problems. A trust region strategy is combined with PMO to improve the performance of PMO, and thus the trust region based PMO method (TR-PMO) is developed. When the sample points are scarce, TR-PMO method shows comparable or better performance than both TRMPS and OMID. The proposed approaches are successfully applied to an airfoil design problem. Note that TR-PMO provides one method to improve the performance of PMO. Other space reduction-based methods and similar strategies to Efficient Global Optimization (EGO) (Jones et al. 1998) may be integrated with PMO for better performing algorithms.

PMO offers a novel approach by using partial HDMR models to guide the optimization. Such an idea is not limited to HDMR and may inspire the development of a new set of metamodel-based design optimization algorithms. Future work will apply PMO to more industrial applications.

Acknowledgments Funding from Natural Science and Engineering Research Council (NSERC) of Canada (Grant No: RGPIN 04291-2014) is gratefully appreciated.

Appendix

Numerical benchmark functions

SUR-T1–14 function, $n = 10, 20, 30$

$$f(\boldsymbol{x}) = (x_1-1)^2 + (x_n-1)^2 + n\sum_{i=1}^{n-1}(n-i)\left(x_i^2-x_{i+1}\right)^2 \quad -3\le x_i\le 2, \quad i = 1,2,\ldots,n \tag{14}$$

Rosenbrock function, $n = 10$

$$f(\boldsymbol{x}) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1}-x_i^2\right)^2 + (x_i-1)^2\right) \quad -5\le x_i\le 5, \quad i = 1,2,\ldots,n \tag{15}$$

Trid function, $n = 10$

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}(x_i-1)^2 - \sum_{i=2}^{n}x_i x_{i-1} \quad -n^2\le x_i\le n^2, \quad i = 1,2,\ldots,n \tag{16}$$

F16 function, $n = 16$

$$f(\boldsymbol{x}) = \sum_{i=1}^{16}\sum_{j=1}^{16} a_{ij}\left(x_i^2 + x_i + 1\right)\left(x_j^2 + x_j + 1\right) \quad -1 \le x_i \le 1, \quad i = 1, 2, \ldots, n \tag{17}$$

$$a_{ij} = \begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Griewank function, $n = 10, 20, 30$

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad -300 \le x_i \le 300, \quad i = 1, 2, \ldots, n \tag{18}$$

Ackley function, $n = 10, 20, 30$

$$f(\boldsymbol{x}) = 20 + e - 20e^{-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)} \quad -30 \le x_i \le 30, \quad i = 1, 2, \ldots, n \tag{19}$$

Rastrigin function, $n = 20$

$$f(\boldsymbol{x}) = 10 \times 20 + \sum_{i=1}^{20}\left(x_i^2 - 10\cos(2\pi x_i)\right) \quad -5.12 \le x_i \le 5.12, i = 1, 2, \ldots, 20 \tag{20}$$

SUR-T1–16, $n = 20$

$$f(\boldsymbol{x}) = \sum_{i=1}^{5}\left[(x_i + 10x_{i+5})^2 + 5(x_{i+10} - x_{i+15})^2 + (x_{i+5} - 2x_{i+10})^2 + 10(x_i - x_{i+15})^4\right] \quad -2 \le x_i \le 5, \quad i = 1, 2, \ldots, n \tag{21}$$

Powell function, $n = 20$

$$f(\boldsymbol{x}) = \sum_{i=1}^{n/4}(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^2 \quad -4 \le x_j \le 5, j = 1, 2, \ldots n \tag{22}$$

Perm function, $n = 20$

$$f(\boldsymbol{x}) = \sum_{k=1}^{n}\left[\sum_{i=1}^{n}\left(i^k + \beta\right)\left((x_i/i)^k - 1\right)\right]^2 \quad -n \le x_i \le n, i = 1, 2, \ldots, n \quad \beta = 0.5 \tag{23}$$

References

Adorio E, Diliman U (2005)MVF–multivariate test functions library in c for unconstrained global optimization, Available at: http://geocities.com/eadorio/mvf.pdf

Alexandrov NM, Dennis JE, Lewis RM, Torczon V (1998) A trust-region framework for managing the use of approximation models in optimization. Struct Optim 15(1):16–23

Allison JT, Kokkolaras M, Papalambros PY (2009) Optimal Partitioning and Coordination Decisions in Decomposition-Based Design Optimization. J Mech Des 131(8):81008

Bates RA, Buck RJ, Riccomagno E, Wynn HP (1996) Experimental Design and Observation for Large Systems. J R Stat Soc Ser B 58(1):77–94

Bloebaum C, Hajela P, Sobieszczanski-Sobieski J (1992) Non-Hierarchic System Decomposition in Structural Optimization. Eng Optim 19(3):171–186

Cai X, Qiu H, Gao L, Yang P, Shao X (2016) An enhanced RBF-HDMR integrated with an adaptive sampling method for approximating high dimensional problems in engineering design. Struct Multidiscip Optim 53(6):1209–1229

Cheng GH, Younis A, Haji Hajikolaei K, Gary Wang G (2015) Trust Region Based Mode Pursuing Sampling Method for Global Optimization of High Dimensional Design Problems. J Mech Des 137(2):21407

Cressie N (1988) Spatial prediction and ordinary kriging. Math Geol 20(4):405–421

Duan X, Wang GG, Kang X, Niu Q, Naterer G, Peng Q (2009) Performance study of mode-pursuing sampling method. Eng Optim 41(1):1–21

Fang H, Horstemeyer MF (2006) Global response approximation with radial basis functions. Eng Optim 38(4):407–424

Friedman JH (1991) Multivariate Adaptive Regression Splines. Ann Stat 19(1):1–67

Goldberg DE (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston

Haji Hajikolaei K, Cheng GH, Wang GG (2015) Optimization on Metamodeling-Supported Iterative Decomposition. J Mech Des 138(2):21401

Hajikolaei KH, Wang GG (2013) High Dimensional Model Representation with Principal Component Analysis. J Mech Des 36(1):11003

Hajikolaei KH, Pirmoradi Z, Cheng GH, Wang GG (2014) Decomposition for Large Scale Global Optimization Based on Quantified Variable Correlations Uncovered by Metamodeling. Eng Optim 47(4):429–452

Hajikolaei KH, Cheng GH, Wang GG (2016) Optimization on Metamodeling- Supported Iterative Decomposition. ASME J Mech Des 138:21401

Huang Z, Qiu H, Zhao M, Cai X, Gao L (2015) An adaptive SVR-HDMR model for approximating high dimensional problems. Eng Comput Int J Comput Eng Softw 32(3):643–667

Jones DR, Law C (1993) Lipschitzian Optimization Without the Lipschitz Constant. J Optim Theory Appl 79(1):157–181

Jones DR, Schonlau M, Welch WJ (1998) Efficient Global Optimization of Expensive Black-Box Functions. J Glob Optim 13(4):455–492

Kaya H, Kaplan M, Saygın H (2004) A recursive algorithm for finding HDMR terms for sensitivity analysis. Comput Phys Commun 158(2):106–112

Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, Perth, Western Australia, pp 1942–1948

Kim HM, Michelena NF, Papalambros PY, Jiang T (2003) Target Cascading in Optimal System Design. J Mech Des 125(3):474

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by Simulated Annealing. Science 220(4598):671–680

Kulfan B, Bussoletti J (2006) 'Fundamental' parameteric geometry representations for aircraft component shapes. 11th aiaa/issmo multidiscip. Anal. Optim. Conf., 1, pp 547–591

Liu Y, Yao X, Zhao Q, Higuchi T (2001) Scaling Up Fast Evolutionary Programming with Cooperative Coevolution. In: Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, pp 1101–1108

Michelena N, Papalambros P, Park H, Kulkarni D (1999) Hierarchical Overlapping Coordination for Large-Scale Optimization by Decomposition. AIAA J 37(7):890–896

Papadrakakis M, Lagaros ND, Tsompanakis Y (1998) Structural optimization using evolution strategies and neural networks. Comput Methods Appl Mech Eng 156(1–4):309–333

Potter M, De Jong K (1994) A Cooperative Coevolutionary Approach to Function Optimization. In: Proceedings of the Third Conference on Parallel Problem Solving from Nature, Springer-Verlag, pp 249–257

Rabitz H, Alis Ö, Alış ÖF (1999) General foundations of high-dimensional model representations. J Math Chem 25(2–3):197–233

Rassokhin DN, Lobanov VS, Drive S (2000) Nonlinear Mapping of Massive Data Sets by Fuzzy Clustering and Neural Networks. J Comput Chem 22(4):373–386

Schittkowski K (1987) More Test Examples for Nonlinear Programming Codes. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, Berlin

Shan S, Wang GG (2009) Development of Adaptive RBF-HDMR Model for Approximating High Dimensional Problems. Volume 5: 35th Design Automation Conference, Parts A and B, ASME, pp 727–740

Shan S, Wang GG (2010a) Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. Struct Multidiscip Optim 41(2):219–241

Shan S, Wang GG (2010b) Metamodeling for High Dimensional Simulation-Based Design Problems. J Mech Des 132:51009

Shi Y, Teng H, Li Z (2005) Cooperative Co-evolutionary Differential Evolution for Function Optimization. In: Proceedings of the First International Conference on Natural Computation, Springer-Verlag, pp 1080–1088

Sobol′ I (1993) Sensitivity estimates for nonlinear mathematical models. Math Model Comput Exp 1(4):407–414

Somorjai RL, Dolenko B, Demko A, Mandelzweig M, Nikulin AE, Baumgartner R, Pizzi NJ (2004) Mapping high-dimensional data onto a relative distance plane–an exact method for visualizing and characterizing high-dimensional patterns. J Biomed Inform 37(5): 366–379

Srivastava A, Hacker K, Lewis K, Simpson TW (2004) A method for using legacy data for metamodel-based design of large-scale systems. Struct Multidiscip Optim 28(2):146–155

Wang GG, Shan S, (2004) Design Space Reduction for Multi-objective Optimization and Robust Design Optimization Problems. SAE Trans J Mater Manuf, pp 101–110. https://doi.org/10.4271/2004-01-0240

Wang L, Shan S, Wang GG (2004) Mode-pursuing sampling method for global optimization on expensive black-box functions. Eng Optim 36(4):419–438

Wang H, Tang L, Li GY (2011) Adaptive MLS-HDMR metamodeling techniques for high dimensional problems. Expert Syst Appl 38(11): 14117–14126

Winer EH, Bloebaum CL (2002) Development of visual design steering as an aid in large-scale multidisciplinary design optimization. Part I : method development. Struct Multidiscip Optim 23(6):412–424

XFOIL (2013) [Online]. Available at: http://web.mit.edu/drela/Public/web/xfoil/

Yang Z, Tang K, Yao X (2008) Large scale evolutionary optimization using cooperative coevolution. Inf Sci (Ny) 178(15):2985–2999